# CSE422
# Artificial Intelligence
# Lab

8- Queen

8 X 8 2d board.
Queens can be allocated that no queen can be attacked by another queen horizontally, vertically or diagonally

Task

- Write a fitness function which checks the fitness of a board by checking the number of non-attacking pairs of queens. Hint: Maximum number of non-attacking pairs of queens can be (8*7)/2. Input: board, Output: a number telling the fitness of the board
- Write a Crossover function. A crossover function will take two boards, an index number as input and return two new boards.
- Write a Mutation function.
- Create a population of randomly generated boards.
- Randomly select two members of the population.
- Randomly generate an index number.
- Call crossover function using the above two as input.
- Call the fitness function for the new boards from the output.
- Call mutation function if necessary.
- Add new members to a new population set if appropriate.
- Run 5 to 10 until the new population set is large enough.
- Select a few members from the old population to add to the new population set.
- Run 1- 12 until a board with the highest fitness value is created.

# Department of Computer Science and Engineering

**Course Code:CSE422**

**Course Name: Artificial Intelligence** **Prerequisite:** CSE111, CSE221

---

## Lab 03

## Genetic Algorithm

**Lab Overview:**

The students will solve N-Queen problem using python programming and visualizing the evolution performance.

**Learning Objective:**

Introducing the 4-Queen problem

Solution of 4-Queen problem in Backtracking approach

Demerits of Backtracking approach

Introducing 8-Queen problem

Discussion on Genetic Algorithm

Solution of 8-Queen problem using GA

**Lesson Fit:**

There is pre-requisite to this lab: CSE111, CSE221. You should have intensive Programming Knowledge and capability to understand algorithms.

**Acceptance and Evaluation**

Students will show the output using different datasets and python code. They will be marked according to their lab performance. The main evaluation criteria will be based on project report and demonstration.

**Learning Outcome:**

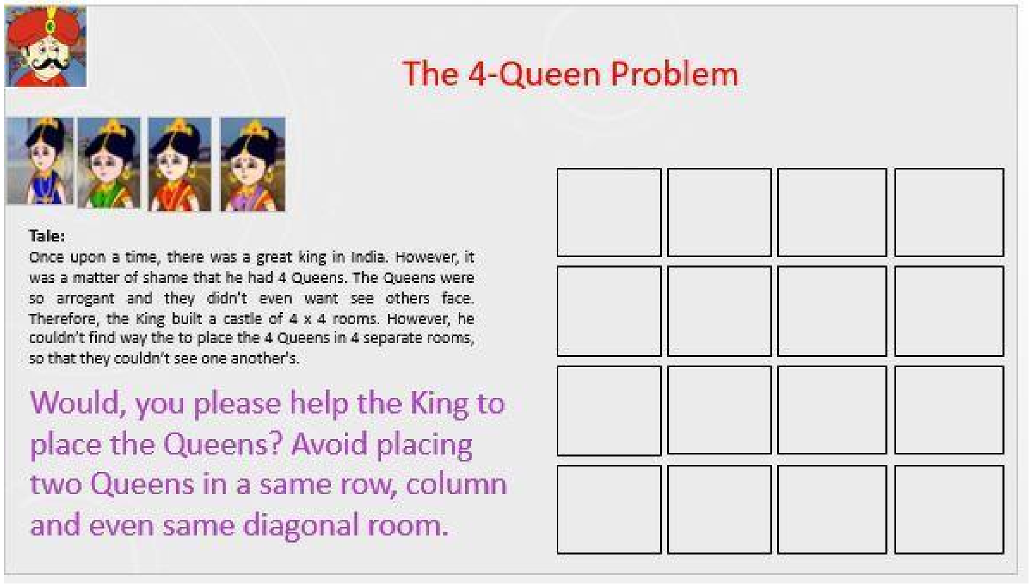
After this lab, the students will be able to:

Demerits to solve N-Queen problem using Backtracking approach.

Solve the N-Queen problem using Genetic Algorithm

**Activity Detail**

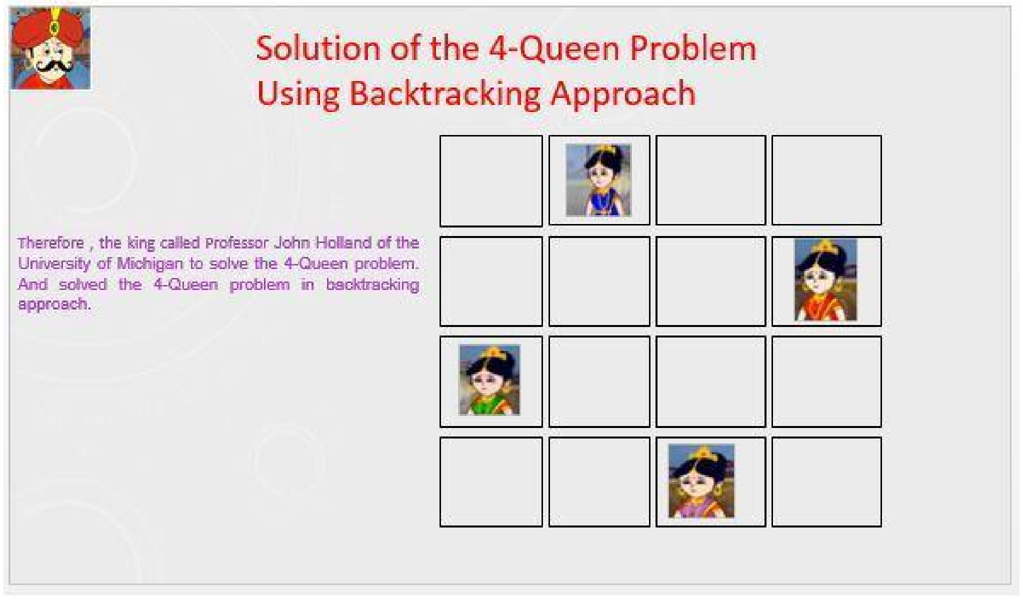
**Hour: 1.0 - 2.0**



## The 4-Queen Problem

**Tale:**
Once upon a time, there was a great king in India. However, it was a matter of shame that he had 4 Queens. The Queens were so arrogant and they didn't even want see others face. Therefore, the King built a castle of 4 x 4 rooms. However, he couldn't find way the to place the 4 Queens in 4 separate rooms, so that they couldn't see one another's.

Would, you please help the King to place the Queens? Avoid placing two Queens in a same row, column and even same diagonal room.



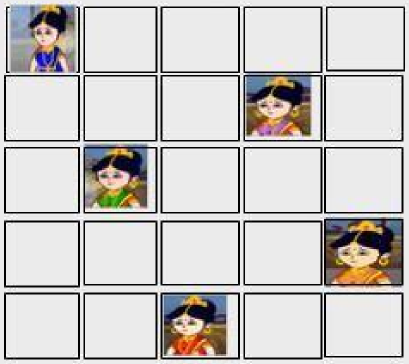
## Solution of the 4-Queen Problem Using Backtracking Approach

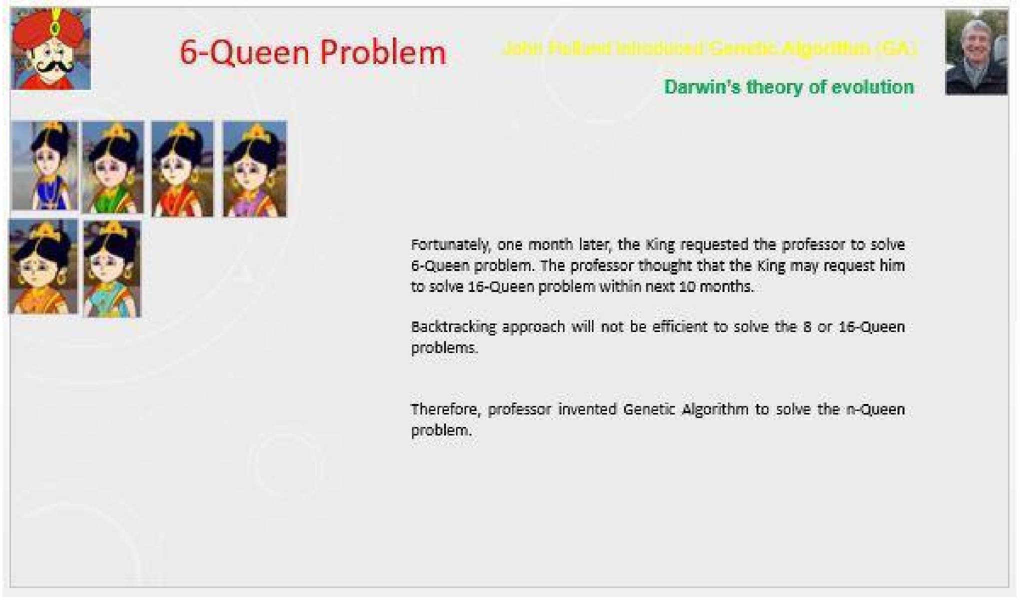
Therefore , the king called Professor John Holland of the University of Michigan to solve the 4-Queen problem. And solved the 4-Queen problem in backtracking approach.

## The 5-Queen Problem

One month later, Professor received a call from the great King to solve his 5-Queen problem. Professor, solved the 5-Queen problem in backtracking approach.

## Solution of the 5-Queen Problem Using Backtracking Approach





## 6-Queen Problem

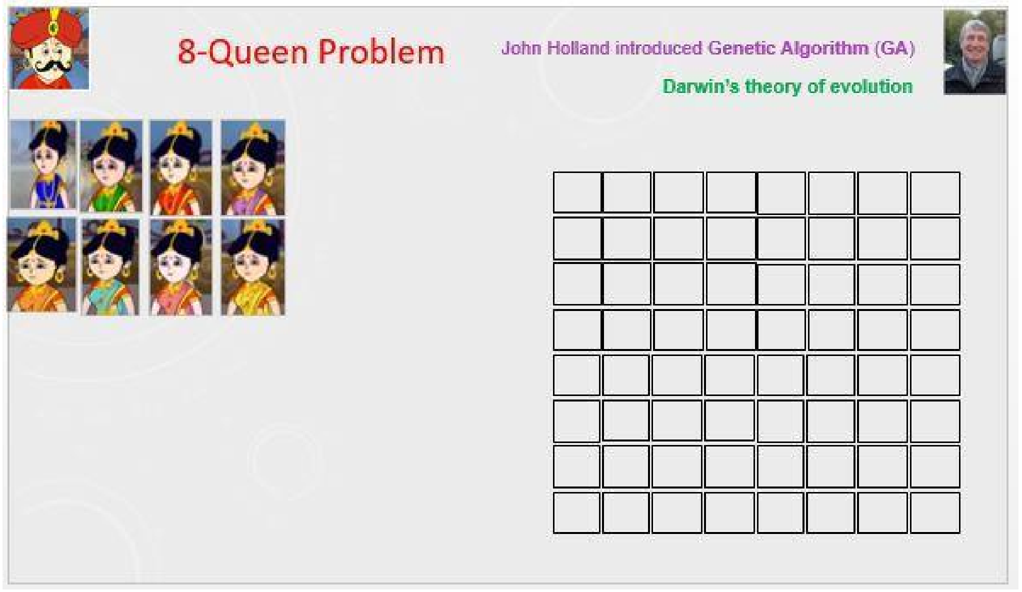
John Holland introduced Genetic Algorithm (GA)

Darwin's theory of evolution

Fortunately, one month later, the King requested the professor to solve 6-Queen problem. The professor thought that the King may request him to solve 16-Queen problem within next 10 months.

Backtracking approach will not be efficient to solve the 8 or 16-Queen problems.

Therefore, professor invented Genetic Algorithm to solve the n-Queen problem.



## 8-Queen Problem

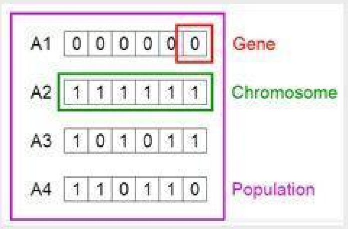
John Holland introduced Genetic Algorithm (GA)

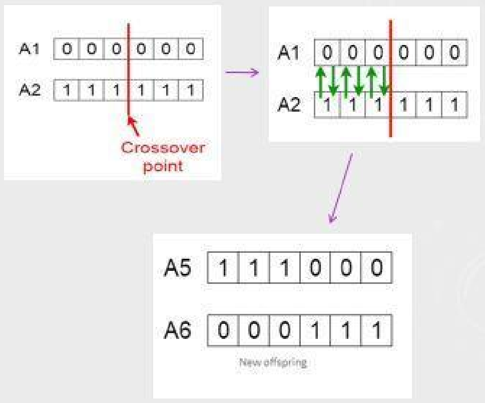
Darwin's theory of evolution

Introduced in the 1970s by John Holland at University of Michigan

- begin with *k* randomly generated states (population)
- each state (individual) is a string over some alphabet (chromosome)
- fitness function (bigger number is better)
- crossover
- mutate (evolve?)

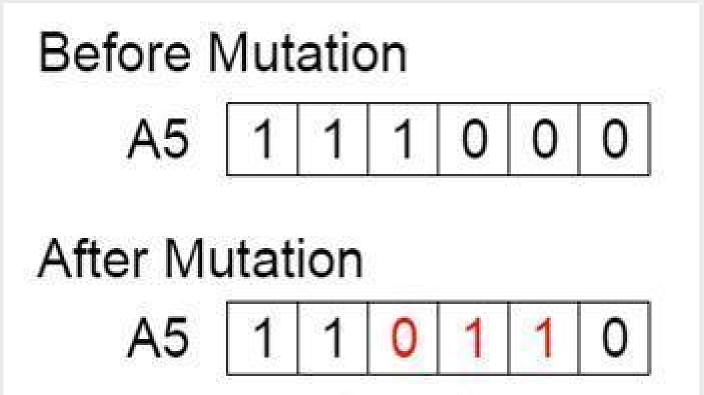
## Crossover:



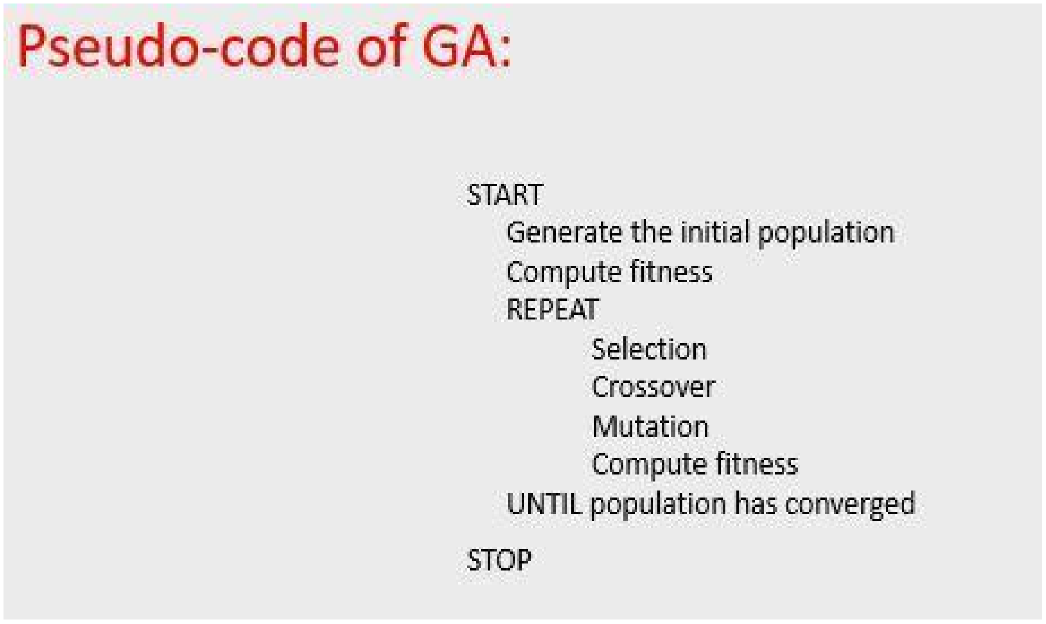


New offspring

## Mutation:



Mutation: Before and After

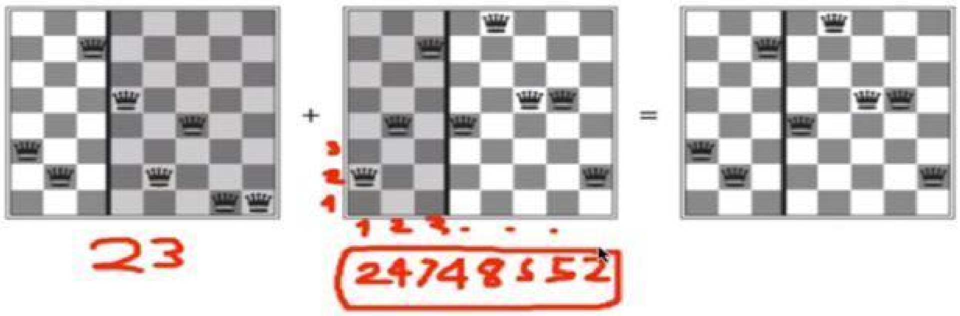


## Pseudo-code of GA:

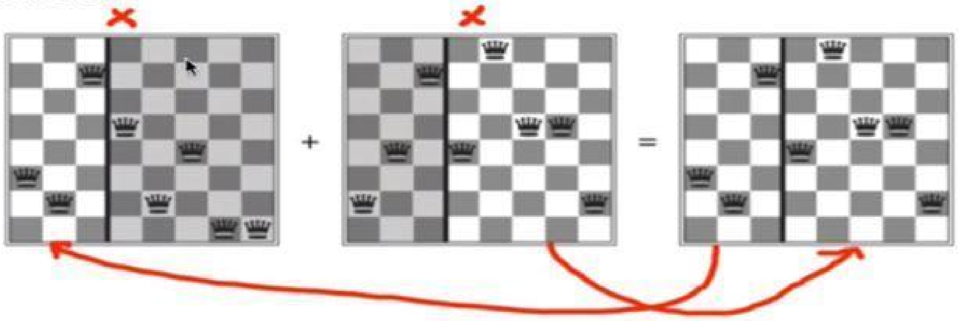
```
START
    Generate the initial population
    Compute fitness
    REPEAT
        Selection
        Crossover
        Mutation
        Compute fitness
    UNTIL population has converged
STOP
```

Fitness Function: Pairs of nonattacking queens

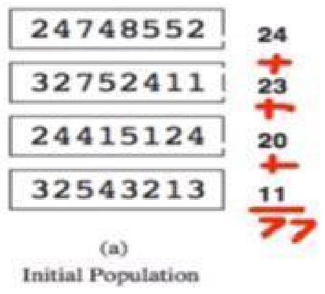
That way, higher scores are better.

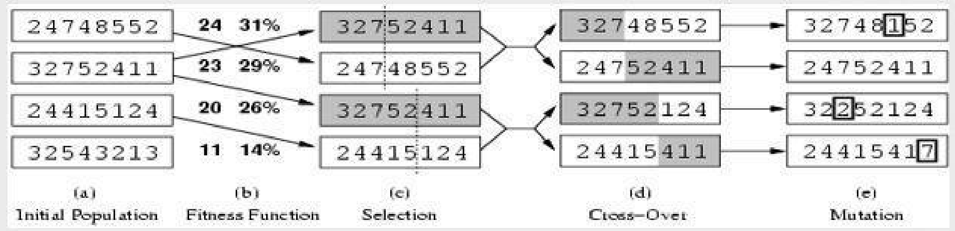


The good genes (features) of the parents are passed onto the children



## Represent states and compute fitness function.



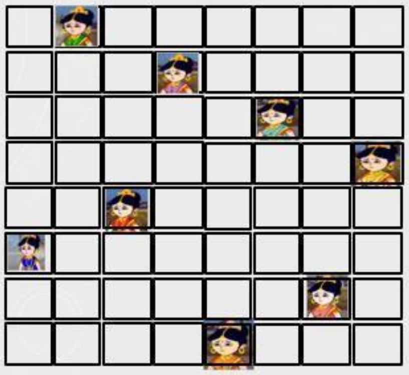


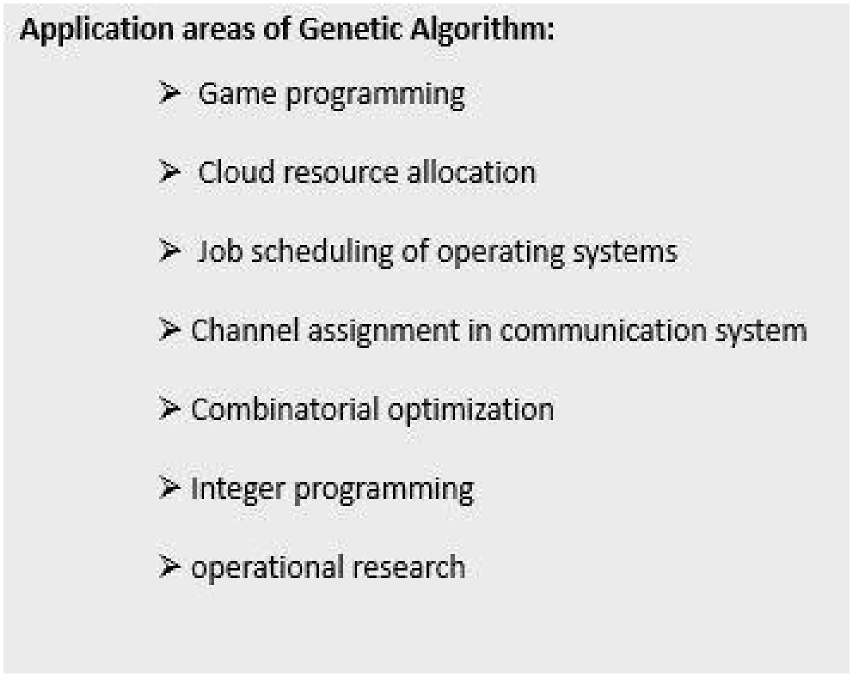
- Fitness function: number of non-attacking pairs of queens (min = 0, max = 8 × 7/2 = 28)
- 24/(24+23+20+11) = 31%
- 23/(24+23+20+11) = 29% etc

## Solution of 8-Queen Problem using Genetic Algorithm

John Holland introduced **Genetic Algorithm** (**GA**)

**Darwin's theory of evolution**




**Application areas of Genetic Algorithm:**

- Game programming
- Cloud resource allocation
- Job scheduling of operating systems
- Channel assignment in communication system
- Combinatorial optimization
- Integer programming
- operational research

**Hour: 2.0-3.0**

(It is Not a Group Task, Try Individually)

**Marks: 10** **Time: 50 minutes**

**Task 1**: Implement N-Queen problem using Genetic Algorithm in python programming.

**Task 2**: Visualize the evolution through plotting the changes of fitness values, and the variances of fitness values for convergence.

Hints: Take help from Prateek Joshi's Book chapter 8, you can follow Covariance Matrix Adaptation Evolution Strategy (CMA-ES).

Evaluation Process (VIVA and Written answers): You have to explain your program and show your work to the Lab Instructor. Instructor may ask you some questions to evaluate your knowledge and expertise level.

---End---