Faculty of Computers and Information
Information Systems Department
Intro. to Artificial Intelligence
Fall semester 2022-2023

Intro. to Artificial Intelligence

Team Members

	Name	ID	Level	Total grade(35)
1	**إبراهيم محمد أبو الحجاج**	**202000006**	**3**	
2	احمد محمد ابراهيم محمد	202000063	**3**	
3	اسلام اشرف عبد الغني نعيم	202000121	**3**	
4	جورج سمير جبره واصف	202000239	**3**	
5	عبدالرحمن نبيل ثابت	202000540	**3**	
6	محمود احمد محمد محمد	201900756	**3**	
7	أحمد محي عبد الحي بيومي	202000085	**3**	

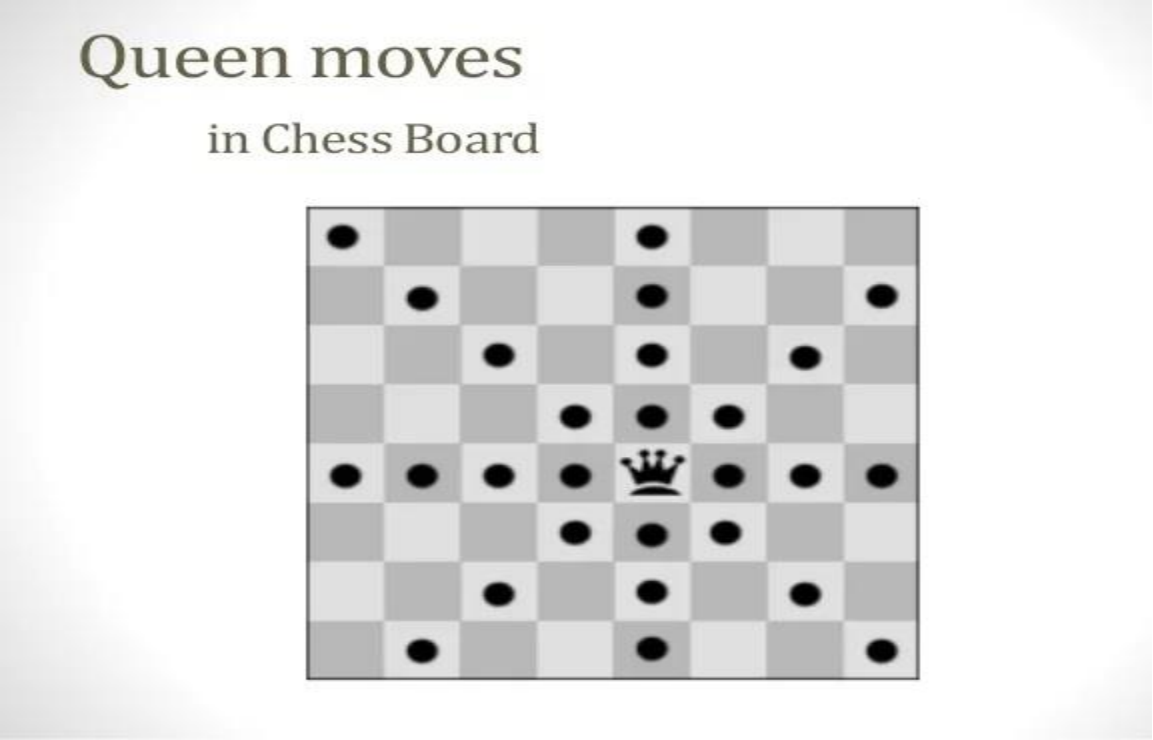
Team Grade /35

- Project idea in details

What is Queens Puzzle ?

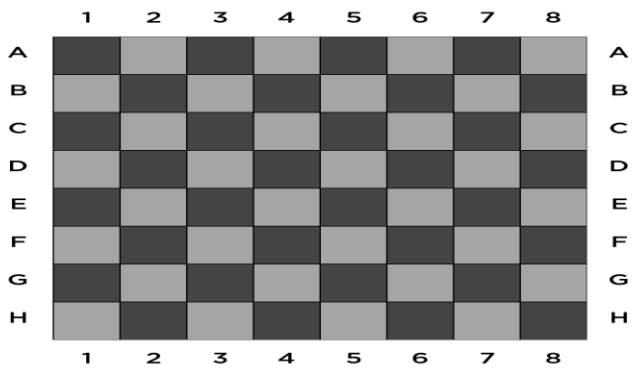
Queens puzzle(N-Queens) puzzle is a problem of placing N number of chess queens on [N*N] chessboard so that no two queens ,

So that **no two queens threaten each other** ,thus a solution requires that no Two queens share the same row , column or diagonal .

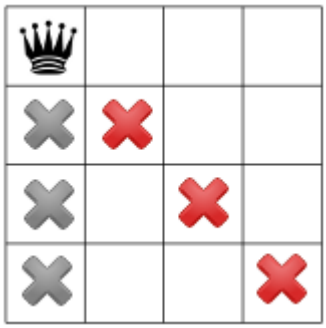


- *Information about Chessboard*

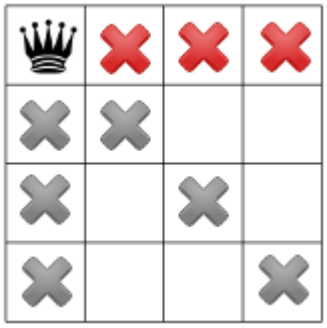
You can also choose any chessboard size between 4 and 20 because number less Than or equal 4 have no solutions and the puzzle gets difficult as the chessboard size increase A [20*20] chessboard has *4.878.666.808* solutions!



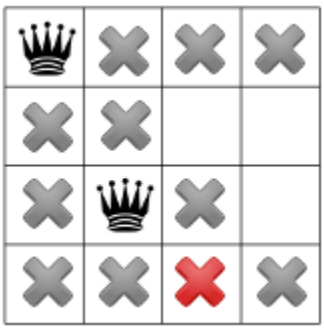
- Example



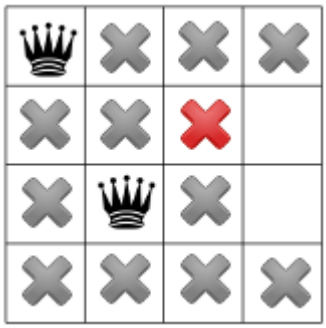
Our third constraint prohibits queens on the same row.



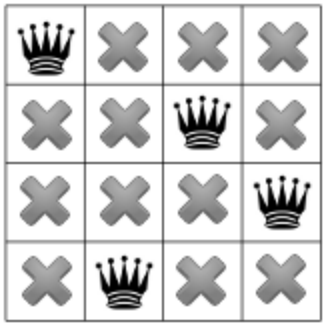
Our constraints propagated, we can test out another hypothesis, and place a second queen on one of the available remaining squares. Our solver might decide to place in it the first available .



After propagating the diagonal constraint, we can see that it leaves no available squares in either the third column or last row.

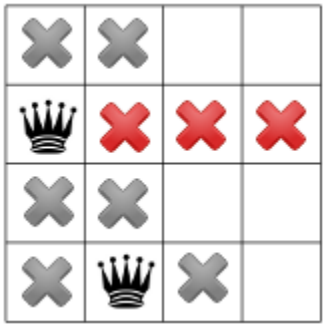


With no solutions possible at this stage, we need to backtrack. One option is for the solver to choose the other available square in the second column. However, constraint propagation then forces a queen into the second row of the third column, leaving no valid spots for the fourth queen.

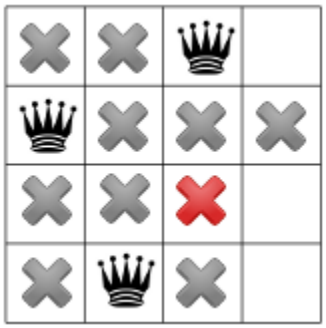


And so the solver must backtrack again, this time all the way back to the placement of the first queen. We have now shown that no solution to the queens problem will occupy a corner square.

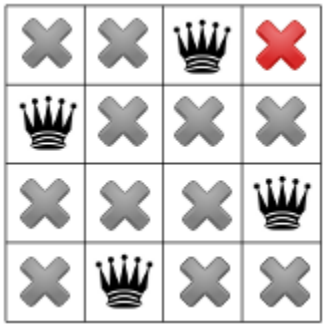
Since there can be no queen in the corner, the solver moves the first queen down by one, and propagates, leaving only one spot for the second queen.



Propagating again reveals only one spot left for the third queen.



And for the fourth and final queen.



We have our first solution! If we instructed our solver to stop after finding the first solution, it would end here. Otherwise, it would backtrack again and place the first queen in the third row of the first column.

Main functionalities

in N-Queen

1-choose board dimension.

2- create board.

3- add queen

4-read current board.

5-set and test position.

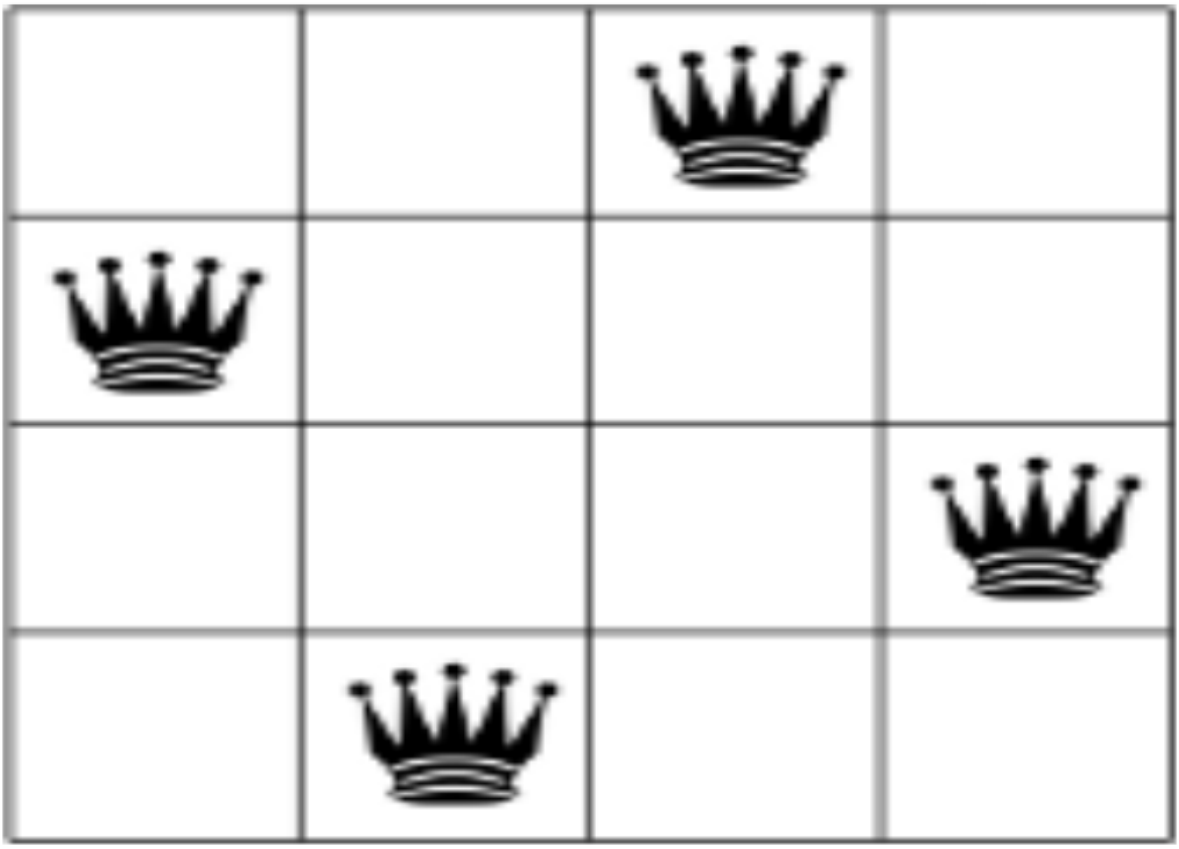
6-test check queen

6-set queen.

Similar applications to n-Queen problem in market:

- The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, the following is a solution for the 4 Queen problem.

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



There are many applications such as:

A- Travelling Salesman Problem (TSP): Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns back to the starting point.

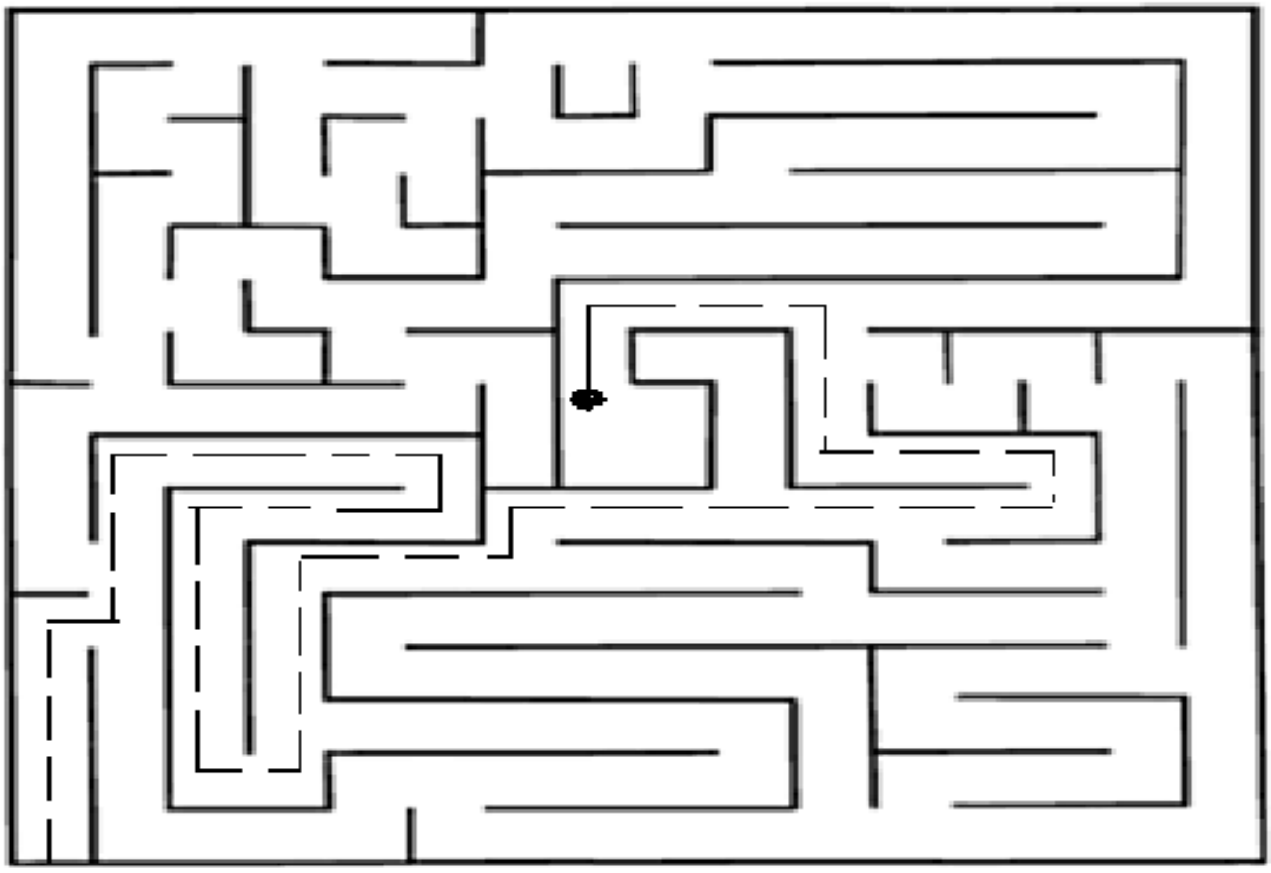
Travelling Salesman Problem states-

- A salesman should visit every city exactly once.
- He should come back to the city from where he starts his journey.
- What is the shortest possible route that the salesman must follow to complete his tour?

B- Maze-solving algorithm:

A maze-solving algorithm is an automated method for solving a maze. The random mouse, wall follower, Pledge, and Trémaux's algorithms are designed to be used inside the maze by a traveler with no prior knowledge of the maze, whereas the dead-end filling and shortest path algorithms are designed to be used by a person or computer program that can see the whole maze at once.

Where I, J are variables varying from 0 to 8, in loop.



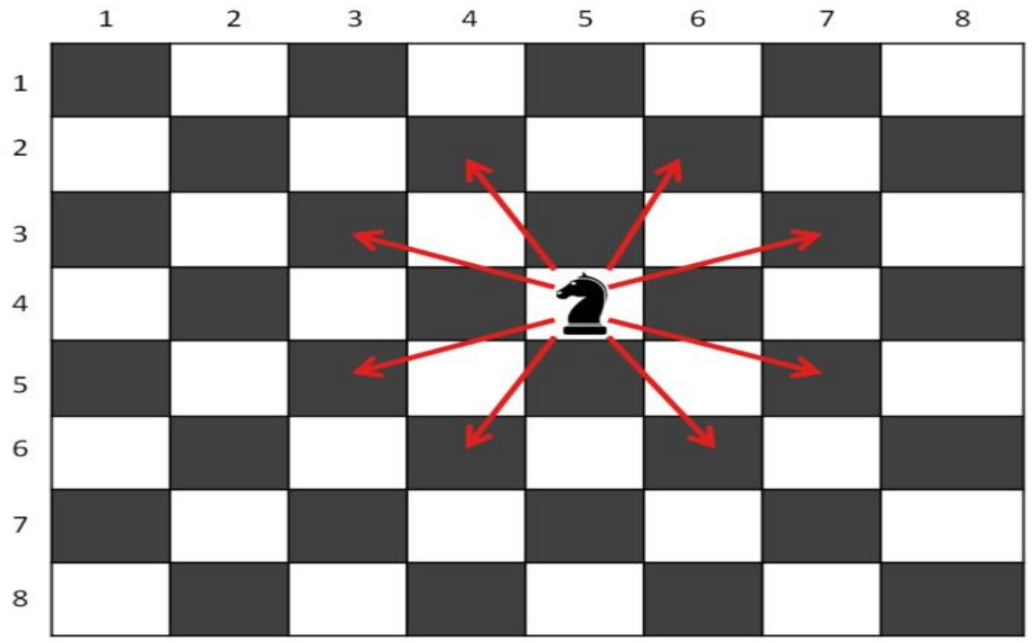
'igure 6. Maze solved through flood fill algorithm:

C- Knight's tour problem:

A knight's tour is a sequence of moves of a knight on a chessboard such that the knight visits every square exactly once. If the knight ends on a square that is one knight's move from the beginning square (so that it could tour the board again immediately, following the same path), the tour is closed (or re-entrant); otherwise, it is open

- The Another practical application such as has many practical applications, such as it is used in **VLSI testing**, **traffic control**, **parallel memory storage schemes**, and **deadlock prevention**

1. The *knight's tour* problem asks if you can start with a chess knight in the upper-left corner of a chess board and move it so that it visits every square exactly once. A knight in chess makes moves that are two squares horizontally or vertically and one square in the other direction, in an L-shape. Answer the following questions about developing a backtracking algorithm to output a knight's tour for an $n \times n$ chess board.



The knight at (4, 5) can move to (2, 4), (2, 6), (3, 7), (5, 7), (6, 6), (6, 4), (5, 3), or (3, 3).

(a) What are the states (vertices) in this problem?
(b) What are the transitions (edges) in this problem?
(c) How would you recognize when to backtrack?
(d) How would you avoid repeating the same state?

Introduction:

Queens problem is to place non attacking queens on an 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.Bezzel. We have not found any solutions to the original 8-queens problem by Bezzel himself, but two nonattacking board configurations were published as partial answer to his problem in the next edition of that serial. Starting in 1850.

Goal :

Is to place the pieces on
the board so that they do not endanger each other (which in turn means that there can be at most one queen on each horizontal, vertical, and diagonal)

There is a wide variety of possible methods for solving NQP and NQP's different formulations. comparing different approaches and publishing improved versions of existing algorithms. Published results show that genetic algorithms and their modifications based on improved genetic operations generate a large part of all possible solutions at a predetermined board size. In other scientific publications, a comparative analysis is made between different approaches and algorithms comparing the number of generated solutions and the time for their generation . Some efficient metaheuristic approaches , show that they can find solutions for NQP in a very short time. These approaches are based on combining evolutionary

algorithms with different heuristic techniques. The same heuristic approaches are successfully used in solving many classes of combinatorial optimization problems .
These studies show that combined approaches, in some cases, have significantly better computational times than basic heuristic techniques. Approaches based on neural networks Montoro et al.and Lapushkin are also used in solving NQP.

Other studies show that the N-Queens problem can be successfully solved by other techniques, such as simulated annealing, methods based on local search , heuristic and meta-heuristic approaches. In order to improve the speed of solving NQP, parallel algorithms are being developed, such as those based on multicore architecture, those based on parallel computing , and others based on specific computational models. To increase productivity, methods of using the hardware capabilities of the computer systems on which the respective algorithms are executed are used. Also, methods based on accelerated execution in solving the N-Queens problem by using the ability for communication between the cores and threads of the CPU are used. Software products that implement the NQP task in the form of a game (puzzle) have been developed.

One of the solutions is a search algorithm, what is the search algorithm?

In computer science, a search algorithm is any algorithm which solves the search problem, namely, to retrieve information stored within some data structure, or calculated in the search space of a problem domain.

Examples of such structures include but are not limited to a linked list, an array data structure, or a search tree.

As shown in the report, he delved deeper into computer science and explained the research algorithm and how this algorithm works simply as it retrieves information stored in a particular structure and gave examples of these structures. (Eight queens puzzle, n.d.)

Genetic Algorithms are inspired by the process of natural selection and utilize bio-inspired operators to generate solutions to search and optimization problems. Genetic algorithms start with a set of randomly generated states, referred to as the population. From the population, inside a loop that runs K number of times; two parents are chosen for reproduction. Offspring are created by a crossover of the two parents using random crossover points. If the two parents are very different i.e. all the Queens are on different rows and the crossover point is near the middle of the parents, the children will be vastly different from both parents. Each child is subject to random mutation, known by a given probability. In theory through the use of crossover and mutation, genetic algorithm are fast in finding a high-quality solution, however the performance of one in the N-Queens problem is debatable and highly reliant on parameters. Even then a genetic algorithm does not scale well with complexity i.e. when N is large.

In the previous paragraph, the report explains the algorithm that used to solve the N-Queens problem, it defined the algorithm and how to solve the problem, as a random case is chosen for the problem, and parents are chosen, and some operations are performed on them to know the points that must be moved to, but he explained that this

algorithm cannot deal with the problem when the number of queens is large.

Implementation:

Python is the programming language used to implement all search algorithms. State, board and configuration are used interchangeably. N and n always refer to board size. All implementation components are italicized. All parameter discussed are optimized.

Here he explains how to solve the problem using implementation as he uses the Python language and define the inputs and uses them to solve all kinds of search algorithms and explains each solution in detail in the rest of the report.

CONCLUSION:

study of the N-Queens Problem was presented. Different approaches to its solution, which are discussed in detail in the scientific literature, were analyzed. The implementation of an algorithm based on the backtracking method was also presented. The algorithm was optimized to find solutions in a specific subset of configurations among all possible ones. With this approach, the computational complexity of the algorithm was reduced from exponential to quadratic. In this way, the algorithm finds in a shorter time all possible solutions, both fundamental and their symmetrical equivalents. The definitions of the various global variables and the dynamic data structures - vectors (arrays) and matrices (two-dimensional arrays) that the algorithm uses were also described. The source codes of the implemented methods (procedures and functions) were presented and analyzed. The method for measuring the execution time of the algorithm used by the start-up

procedure takes into account the multitasking mode of the operating system. The methodology for conducting the experiments was presented. The purpose of the study, the tasks to be performed and the conditions for conducting the experiments were presented as well. As part of the methodology, 14 board sizes were presented, from 8 x 8 (standard chessboard) to 21 x 21, respectively. The ratios between the number of fundamental solutions and the number of all solutions for each of the selected board sizes were calculated. For all solutions, the incremental step between every two consecutive values was calculated. In connection with the research, an application was developed that implements the presented algorithm. Its main functions were summarized. All results obtained in this study were generated by this application. The experimental results showed that with a linear increase in the number of queens (equivalent to a quadratic increase in the number of fields on the board), the number of recursive calls made by the algorithm increased exponentially. Similarly, the number of possible solutions, as well as the execution time of the algorithm (in the different modes of the application - internal, interactive and combined), also increased exponentially. However, the execution time of the algorithm in the internal mode was significantly shorter than in the other two modes - interactive and combined. The ratio between the number of recursive calls and the number of all solutions was also calculated. This ratio varied between 22 times (with a board size of 8 x 8) to 78 times with a board size of 17 x 17.

Applications:

The n-queens problem is often studied as a "mathematical recreation", but Erbas, Tanik and Aliyazicioglu in note several applications: parallel memory storage schemes, VLSI testing, traffic control and deadlock prevention. Erbas and Tanik and Erbas, Tanik and Nair introduce a memory storage scheme for conflict free access for parallel memory

systems using n-queens solutions. Kunde and Gürtzig 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. For deadlock prevention, Tanik in shows that given a solution to the n-queens problem, one can find a set of deadlock free paths. As well, Erbas, Sarkeshik and Tanik in note that the n-queens problem has applications in neural networks and constraint satisfaction problems. Another example of an application of the n-queens problem is in image processing, discussed by Yang, Wang, Liu and Chang in. Li, Guangxi and Xiao use n-queens solutions in low-density parity-check codes. Wang, Yang, Liu and Chiang in, consider applications of n-queens solutions to motion estimation. Dean and Parisi, in, develop a statistical mechanical model of "glassy" phase transitions and note that modular n-queens solutions are the zero-energy points in their Hamiltonian. Yamamoto, Kitamura and Yoshikura give a computer analysis of the statistical secondary structure of nucleic acids, and remark that this computation is related to searching for n-queens solutions. Erdem and Lifschitz note that n-queens problems are examples of finite normal, absolutely tight *logic problems and of interest to the study of theories of negation. Colbourn and Sagols, define two weaker variants of the n-queens problem and use these to construct anti-Pasch Steiner Triple Systems in O(n2) time rather than O(n3); triple systems, themselves, have many applications. In, Taylor also observes that a weaker variant of the n-queens problem is equivalent to the problem of finding the "Florentine rows" on n symbols. uses partial n-queens solutions to construct maximal partial spreads of many sizes in PG(3,q), the three-dimensional projective space over the finite field Fq. Finally, Hsiang, Shieh and Chen note that complete mapping problems, a variant of the n-queens problem, are ideal to test propositional solvers.*

In this part, he explained the applications that are based on the N queens Problem and define the programmers who developed these programs and used these applications to solve other problems in studies and sciences.

Reference

Leon Andov

leon.andov@griffithuni.edu.au

Griffith university

Velin Kralev a,*, Radoslava Kraleva a, Dimitar Chakalov a a Department of Informatics, South-West University ,Blagoevgrad, 2700, Bulgaria
Corresponding author: * velin_kralev@swu.bg

Jordan Bell and Brett Stevens

A survey of known results and research areas for -queens

Discrete Mathematics 309(1), 1 - 31, 2009

21 March 2018

Backtracking Algorithm

Intro

Backtracking is an algorithmic technique whose goal is to use brute force to find all solutions to a problem. It entails gradually compiling a set of all possible solutions. Because a problem will have constraints, solutions that do not meet them will be removed.

Backtracking Algorithm

Backtrack(s)

ifs is not a solution

return false

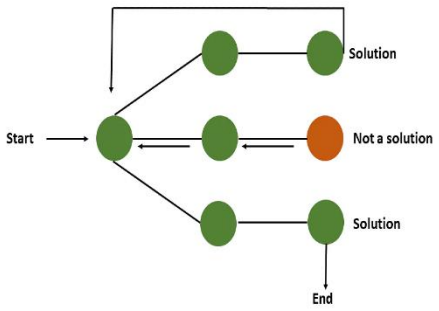
if is a new solution

add to list of solutions

backtrack(expand s)

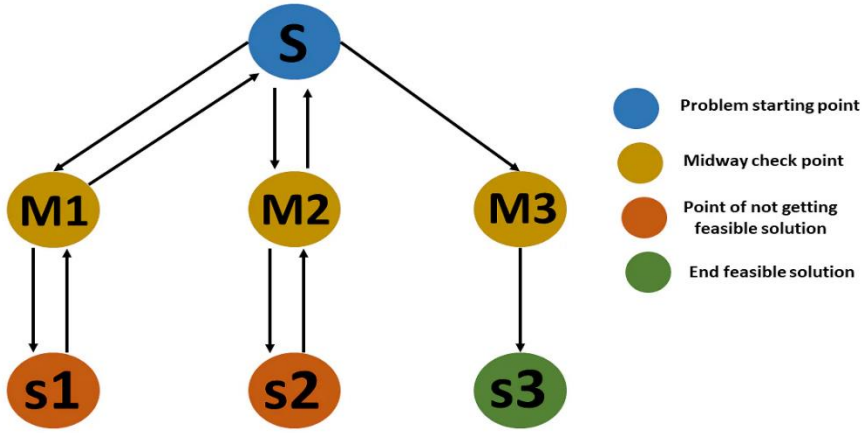
It finds a solution by building a solution step by step, increasing levels over time, using recursive calling. A search tree known as the state-space tree is used to find these solutions. Each branch in a state-space tree represents a variable, and each level represents a solution.

A backtracking algorithm uses the depth-first search method. When the algorithm begins to explore the solutions, the abounding function is applied so that the algorithm can determine whether the proposed solution satisfies the constraints. If it does, it will keep looking. If it does not, the branch is removed, and the algorithm returns to the previous level.



How Does a Backtracking Algorithm Work?

In any backtracking algorithm, the algorithm seeks a path to a feasible solution that includes some intermediate checkpoints. If the checkpoints do not lead to a viable solution, the problem can return to the checkpoints and take another path to find a solution. Consider the following scenario:



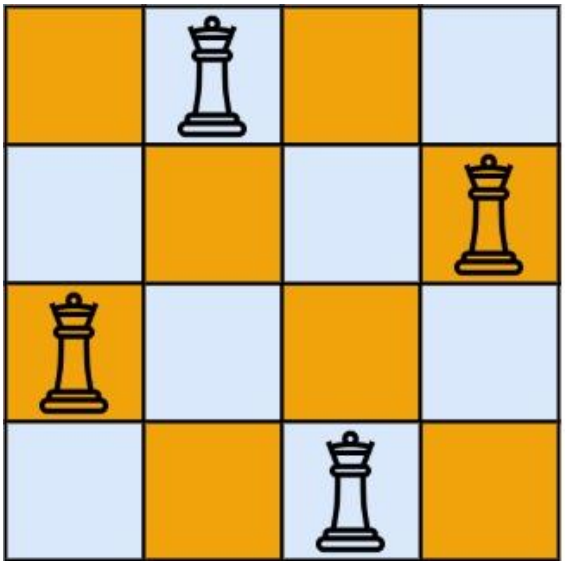
1. In this case, S represents the problem's starting point. You start at S and work your way to solution S1 via the midway point M1. However, you discovered that solution S1 is not a viable solution to our problem. As a result, you backtrack (return) from S1, return to M1, return to S, and then look for the feasible solution S2. This process is repeated until you arrive at a workable solution.
2. S1 and S2 are not viable options in this case. According to this example, only S3 is a viable solution. When you look at this example, you can see that we go through all possible combinations until you find a viable solution. As a result, you refer to backtracking as a brute-force algorithmic technique.
3. A "space state tree" is the above tree representation of a problem. It represents all possible states of a given problem (solution or non-solution).

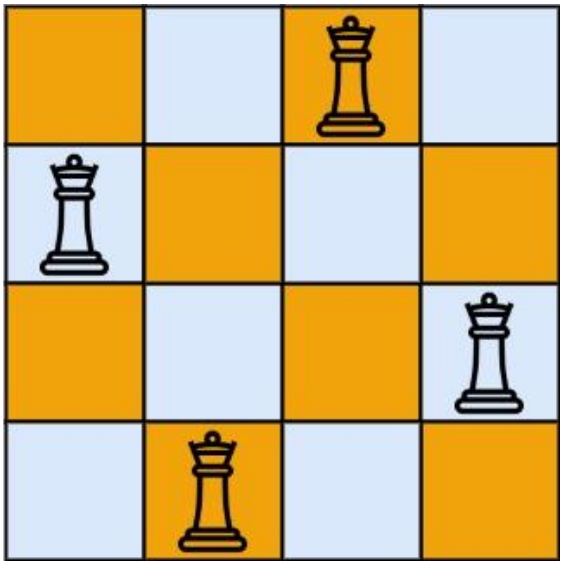
The final algorithm is as follows:

- Step 1: Return success if the current point is a viable solution.
- Step 2: Otherwise, if all paths have been exhausted (i.e., the current point is an endpoint), return failure because there is no feasible solution.
- Step 3: If the current point is not an endpoint, backtrack and explore other points, then repeat the preceding steps.

To Solve the N Queen Problem.

- The problem of placing n queens on the nxn chessboard so that no two queens attack each other is known as the n-queens puzzle.
- Return all distinct solutions to the n-queens puzzle given an integer n. You are free to return the answer in any order.
- Each solution has a unique board configuration for the placement of the n-queens, where 'Q' and. " represent a queen and a space, respectively.

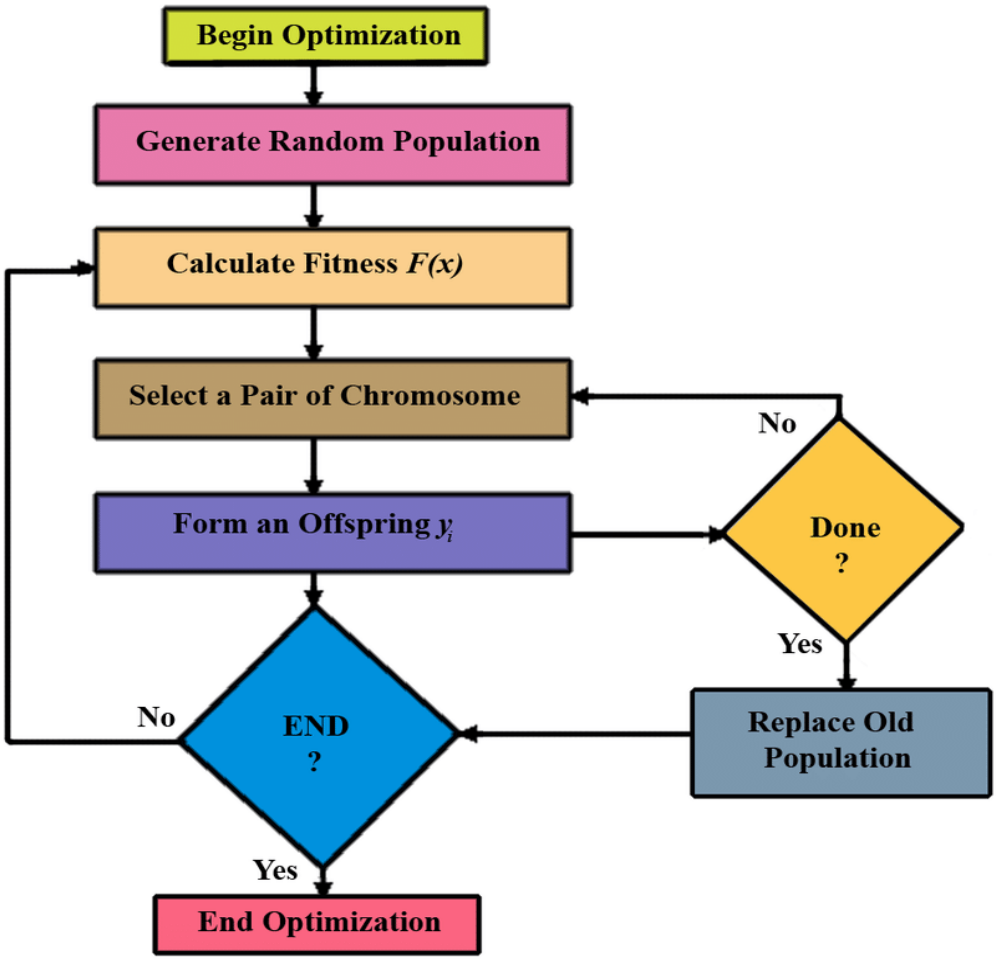




genetic algorithm

intro

are used to find optimal solutions by the method of development-induced discovery and adaptation; Generally used in problems where finding linear / brute-force is not feasible in the context of time, such as – Traveling salesmen problem, timetable fixation, neural network load, Sudoku, tree (data-structure) etc. to set. The first requirement is an encoding scheme that is suitable for representing individuals, the second requirement is an evaluation function to represent a person's fitness.



Genetic Algorithm in Artificial Intelligence:

Genetic algorithm plays the same role as **Artificial Intelligence**. But sometimes the genetic algorithm is considered stronger than AI. This is because unlike conventional AI systems, GA will adjust on the changing input and will also be able to handle noise or fuzzy input. In addition, GE is capable of handling complex problems better.

Phases in Genetic Algorithm:

1.Initial Population: The process begins with a set of individuals called the population. Everyone is the solution to the problem that you want to solve. A person is characterized by a set of parameters known as genes. To make chromosome, the gene is added to a wire.

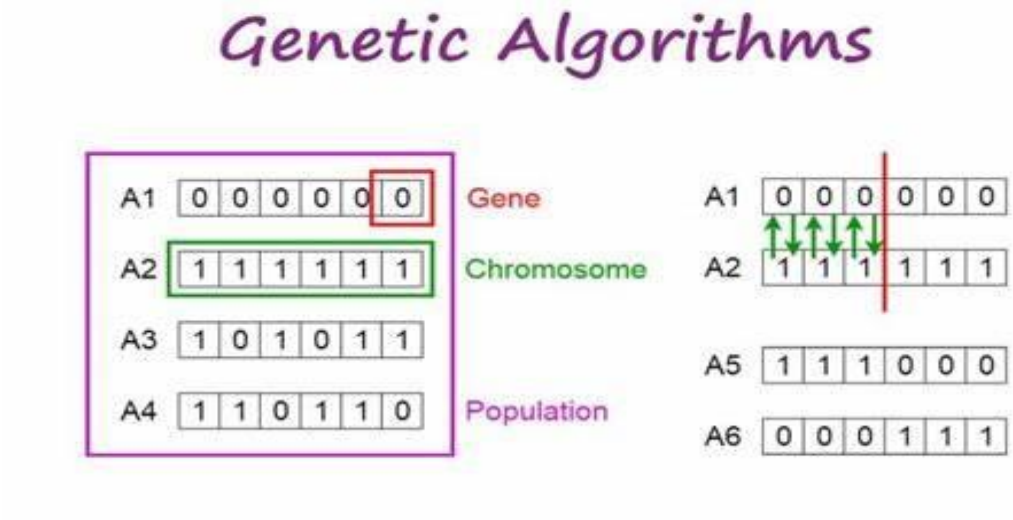
2.Fitness Function: The fitness function determines how to fit a person is (the ability to compete with someone else's person). It gives each person a fitness score The possibility that a person is selected for reproduction is based on his fitness score.

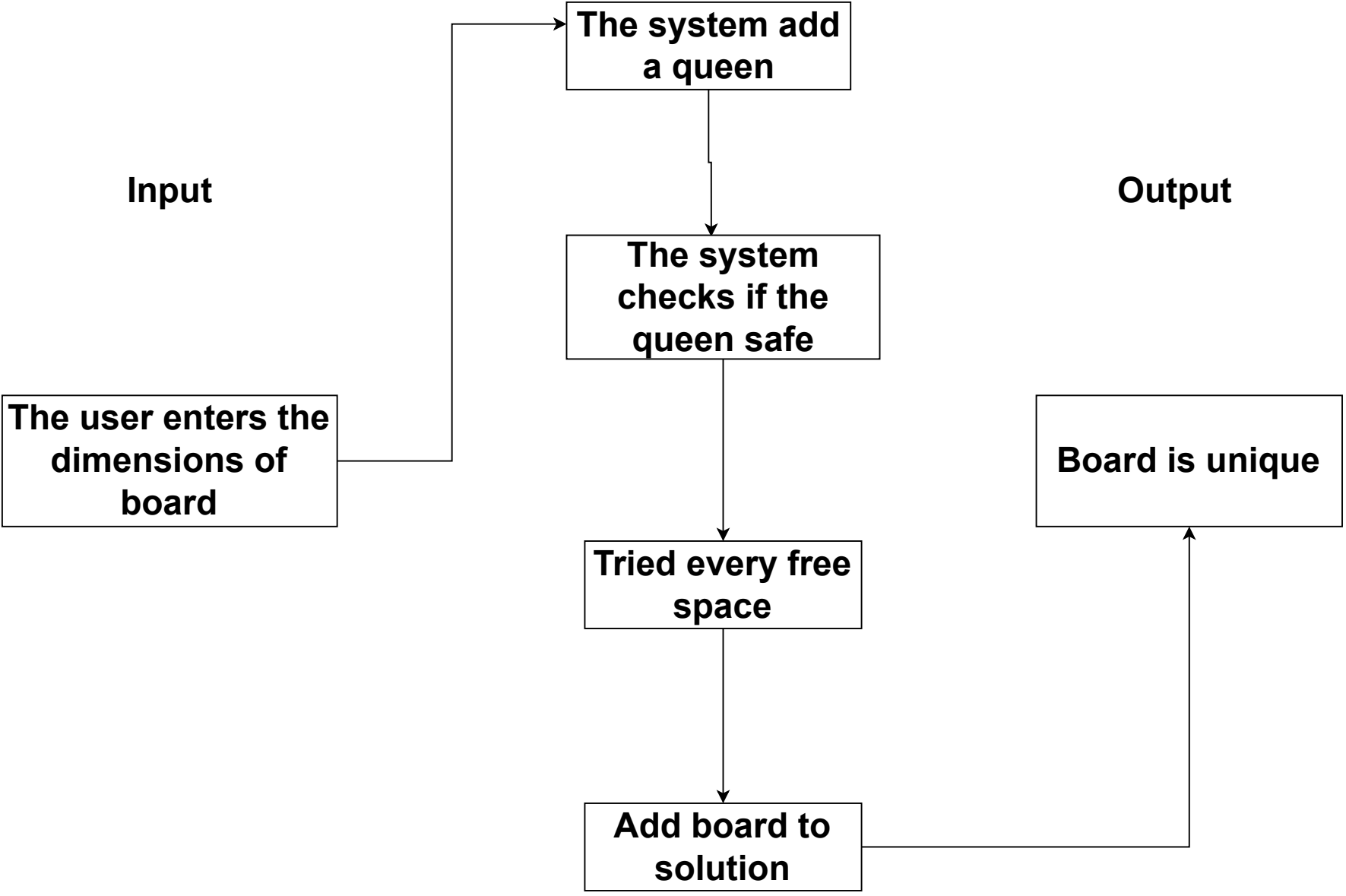
3.Selection: The idea of the election phase is to select the most suitable individuals and allow them to pass their genes to the next generation. Two pairs of parents (parents) are selected on the basis of their fitness score. People with high fitness are more likely to be selected for reproduction.

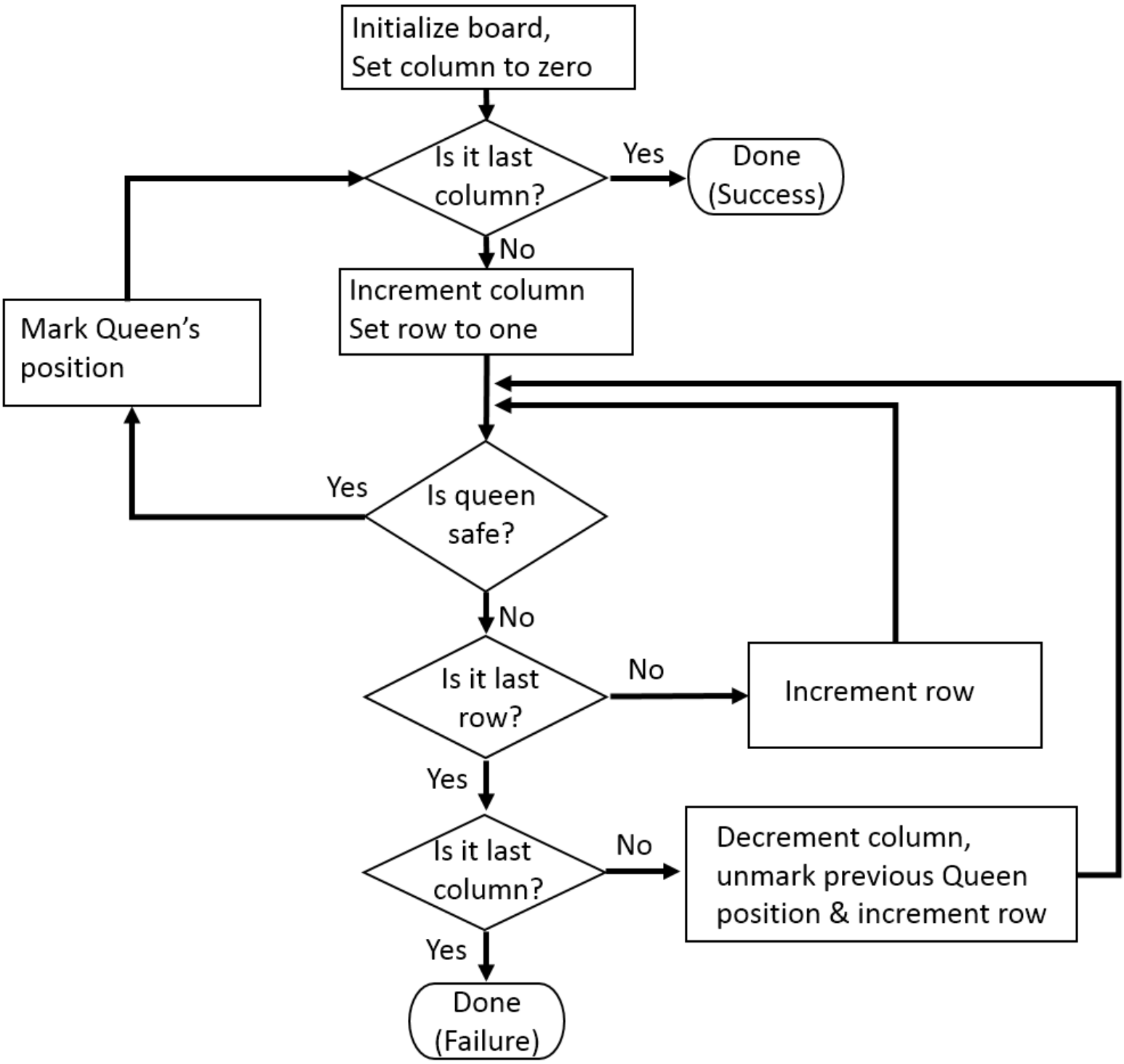
4.Crossover: Crossover is playing a vital role in genetic algorithms. For each pair of parents, a crossover point is selected from within the genes on random.

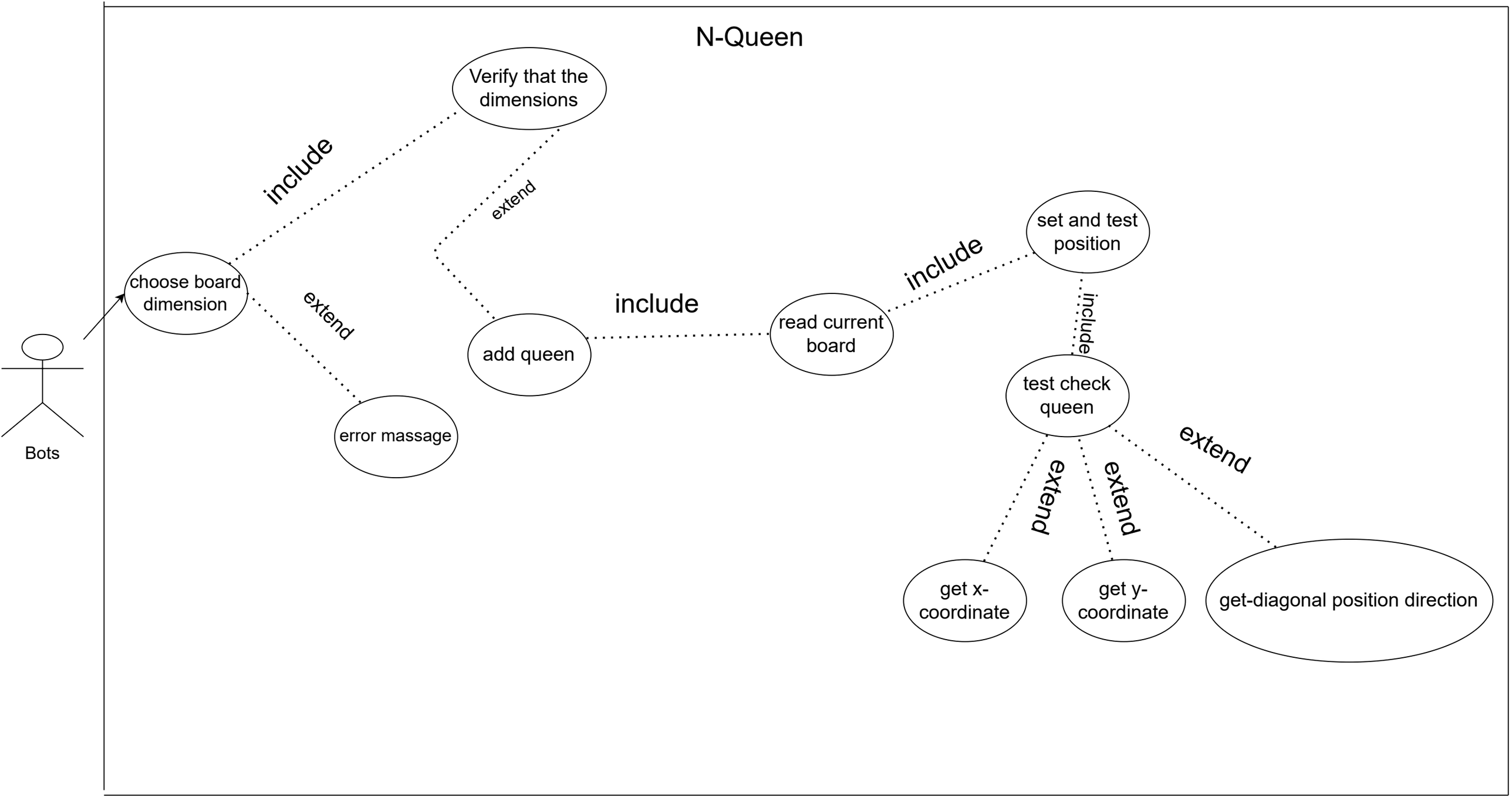
5. Mutation: In some newborns formed, some of their genes can be subjected to mutation with less random chance. This means that some bits may be flipped in the bit string.

6. Termination: The algorithm expires if the population has changed (does not produce a line which is quite different from the previous generation). Then it is said that the genetic algorithm has given a solution to our problem.




The system add a queen
Input
Output
The system checks if the queen safe
The user enters the dimensions of board
Board is unique
Tried every free space
Add board to solution


Initialize board,
Set column to zero
Is it last column?
Yes
Done (Success)
No
Increment column
Set row to one
Mark Queen's position
Yes
Is queen safe?
No
Is it last row?
No
Increment row
Yes
Is it last column?
No
Decrement column, unmark previous Queen position & increment row
Yes
Done (Failure)


N-Queen
Verify that the dimensions
include
extend
choose board dimension
extend
error massage
add queen
include
read current board
include
set and test position
include
test check queen
extend
extend
extend
get x-coordinate
get y-coordinate
get-diagonal position direction
Bots

Development platform.

Libraries:

1-Board.

2-tkinter.

3-for GUI pygame.

Algorithm:

1-backtracking algorithm.

2-genetic algorithm.

Editor:

1-Vs code.