What is backtracking?

Backtracking is a method for solving problems incrementally, where each step depends on previous decisions. It involves exploring a path, checking if it leads to a solution, and, if not, retracing steps to try a different path.

Explain the general procedure of backtracking.

Start with a sub-solution.
Check if this sub-solution leads to the final solution.
If not, backtrack and modify the sub-solution, then repeat the process.

What is the n-Queens problem?

It involves placing n queens on an n×n chessboard so that no two queens threaten each other, meaning no two queens share the same row, column, or diagonal.

Why is the n-Queens problem a good example of backtracking?

The n-Queens problem exemplifies backtracking because it requires placing queens step-by-step, checking for conflicts at each step, and backtracking when no safe position is available.

What does it mean for a cell to be "under attack" in the context of the n-Queens problem?

A cell is under attack if placing a queen there would allow it to threaten another queen, i.e., if there is another queen in the same row, column, or diagonal.

Describe the base case in solving the n-Queens problem with backtracking.

The base case is when all queens have been successfully placed on the board without conflicts, which means a solution has been found.

What are the three types of attacks a queen can make in the n-Queens problem?

A queen can attack horizontally (same row), vertically (same column), and diagonally.

How does the algorithm handle placing the first queen?

The algorithm places the first queen in an arbitrary position and then tries to place subsequent queens in safe positions.

What happens when no safe place is found for a queen in the current row?

The algorithm backtracks, moving the previous queen to a new position to see if this allows a safe placement for the current queen.

Why can't a solution be found for the n-Queens problem with n=2 or n=3?

For n=2 and n=3, it's impossible to arrange the queens without them attacking each other due to limited positions on the board.

Explain how the backtracking algorithm for n-Queens problem operates row-wise.

The algorithm places a queen row-by-row, checking each position within a row for safety, and only moving to the next row if a safe position is found in the current row.

What is the purpose of the `isSafe` function in the n-Queens algorithm?

It checks if placing a queen in a specific cell results in an attack from another queen, ensuring safe placement.

How is recursion used in solving the n-Queens problem?

Recursion allows the algorithm to attempt placing queens one by one and backtrack if a conflict arises, exploring different configurations until a solution is found.

In the n-Queens problem, how does the algorithm detect conflicts in the diagonals?

The algorithm checks cells along the diagonals by adjusting row and column indices simultaneously to verify that no queens are present along these paths.

Give an example of backtracking in daily life.

Solving a maze is an example; if a path doesn't lead to the exit, you backtrack to a previous decision point and try a different path.

How does the backtracking algorithm know when to stop?

The algorithm stops once all queens are placed successfully on the board, meaning a valid solution has been reached.

What is the time complexity of the n-Queens problem?

The time complexity is generally $O(N!)$ due to the factorial number of ways queens can be arranged on the board.

How many solutions exist for the 4-Queens problem?

There are two unique solutions for the 4-Queens problem, excluding symmetrical variations.

What modifications can be made to the algorithm for larger chessboards?

For larger boards, optimizations like pruning irrelevant branches early and using bitwise operations for row, column, and diagonal checks can reduce computation.

Can the n-Queens problem be solved non-recursively? If so, how?

Yes, using an iterative approach with stacks to simulate recursion, though it is more complex and less intuitive than recursive backtracking.