STATE SPACE TREE FOR A N-QUEENS PROBLEM

Challenging Assignments and Mini Projects (CHAMP)

submitted as part of the course
Artificial Intelligence
BCSE306L
School Of Computer Science and Engineering
VIT Chennai

FALL 2024-2025

Course Faculty : Dr. B Radhika Selvamani
Submitted By
Bharanee. B (22BCE1371)

Abstract

The N-Queens problem is a combinatorial optimization problem that places N queens on an N×N chessboard such that no two queens threaten each other. This report details the creation of a state space tree representation for the N-Queens problem, encapsulating logic for generating successor states, checking goal states, and evaluating heuristics. A user-friendly graphical interface enhances visualization and interaction.

Contents

1 Introduction **3**

2 Problem **3**

3 Scope **3**

4 Design **3**
4.1 Overall Architecture 3
4.2 Interfaces Defined 4

5 Observations **4**

1 Introduction

The N-Queens problem serves as a benchmark in artificial intelligence for evaluating backtracking algorithms and heuristic-based solutions. This project models the problem through a state space tree structure, allowing for systematic exploration of solutions. It also introduces a graphical interface to provide an engaging way to explore and solve the problem interactively.

2 Problem

The problem is to position N queens on an N×N chessboard such that:

- No two queens threaten each other horizontally, vertically, or diagonally.
- The program returns the next valid states from the current configuration.
- It evaluates configurations using a heuristic value.

3 Scope

This project demonstrates the use of state space tree exploration to solve constraint satisfaction problems. It highlights the power of algorithmic design in artificial intelligence and its applications in optimization problems. The interactive interface ensures its utility for educational purposes and algorithm analysis.

4 Design

4.1 Overall Architecture

The system's architecture is modular, ensuring separation of concerns for better maintainability:

- **Logic Module**:
  - Implements state generation, goal state verification, and heuristic computation.
  - Includes backtracking logic to explore the state space and find solutions.
- **GUI Module**:
  - Provides an interactive interface for input and visualization.
  - Displays the chessboard with real-time updates of queen placements.

4.2 Interfaces Defined

- **NQueens Class**:
  - `is_safe(row, col)`: Ensures that a queen placed at a specific position does not conflict with others.
  - `heuristic_value()`: Evaluates the number of conflicts on the board for a given state.
  - `place_queens(row)`: Uses backtracking to solve the N-Queens problem.
  - `is_goal_state()`: Checks if the current configuration satisfies the goal conditions.
- **NQueensGUI Class**:
  - `update_board_size(event)`: Allows dynamic adjustment of the chessboard size.
  - `solve_n_queens()`: Finds the solution using the logic module and updates the GUI.
  - `draw_board()`: Renders the chessboard and queens graphically.

5 Observations

- **Efficiency**:
  - The backtracking algorithm performs efficiently for smaller board sizes.
  - Computational complexity increases significantly for larger boards ($N > 12$).
- **User Experience**:
  - The GUI simplifies the problem-solving process by providing clear visual feedback.
  - Interactive elements allow users to modify board sizes and experiment with configurations.
- **Heuristic Insights**:
  - The heuristic evaluation helps analyze intermediate states.
  - It offers an additional perspective but does not directly optimize backtracking performance.
- **Scalability and Challenges**:
  - While the modular design supports scalability, performance bottlenecks emerge for large board sizes.
  - Future improvements may include heuristic-guided search or parallel processing for enhanced scalability.

Acknowledgments

I sincerely thank Dr. B Radhika Selvamani for her guidance throughout this project.

References

1. Gutiérrez-Naranjo, Miguel A., Martínez-del-Amor Miguel A., Pérez-Hurtado Ignacio, and Pérez-Jiménez Mario J. "Solving the N-Queens Puzzle with P Systems", 7th Brainstorming Week on Membrane Computing, volume 1, pp 199-210, 2009.

2. Russell, S., and Norvig, P. "Artificial Intelligence: A Modern Approach", Second Edition, pp 139-140.

3. Martinjak, I., and Golub, M. "Comparison of Heuristic Algorithms for the N-Queen Problem", Proceedings of the ITI 2007 29th Int. Conf. on Information Technology Interfaces, June 25-28, Cavtat, Croatia, 2007.

4. Letavec, C., and Ruggiero, J. "The N-Queen Problem", INFORMS Transaction on Education, volume 2, number 3, May 2002.

5. Horowitz, E., Sahni, S., and Rajasekaran, S. "Fundamentals of Computer Algorithm", pp 339-355.