**Algorithms for Solving Classic Chess and Puzzle Problems**

**Analysis of Existing Algorithms and Development of New Approaches**

A black and white logo

Description automatically generated with low confidence

Technical University of Sofia

German Engineering and Industrial Management Faculty

Computer Science

Logic

09.05.2023

Teodor Marinov, 201222003

Stefan Georgiev, 201222011

Deyvid Popov, 201221006

**Table of Contents**

1. Introduction3
2. Problems and Existing Solutions3

2.1 The Eight Queens Problem3

2.1.1 Problem Description3

2.1.2 Existing Solutions3

2.1.2.1 Breadth-First Search (BFS)3

2.1.2.2 Depth-First Search (DFS)4

2.1.2.3 Backtracking with Forward Checking5

2.2 The Knight's Tour6

2.2.1 Problem Description6

2.2.2 Existing Solutions6

2.2.2.1 Depth-First Search (DFS)6

2.2.2.2 Warnsdorff's Heuristic7

2.3 The Tower of Hanoi7

2.3.1 Problem Description7

2.3.2 Existing Solutions7

2.3.2.1 RecursiveAlgorithm7

1. Our Solutions8

3.1 RegalRunner Algorithm for the Eight Queens Problem8

3.2 ShadowWalker Algorithm for the Knight's Tour9

3.3 DarkAbyss Algorithm for the Tower of Hanoi10

1. Conclusion11

4.1 Additional Resources11

4.2 Summary of Results11

4.3 Outlook and Possible Extensions11

**Introduction**

In the realm of computer science and mathematics, classic chess and puzzle problems have always sparked curiosity and challenged the minds of enthusiasts. This coursework aims to explore the intricacies of three such captivating problems: the Eight Queens, the Knight's Tour, and the Tower of Hanoi. Delving into the existing algorithms that attempt to tackle these enigmatic challenges, we will not only analyze their strengths and weaknesses but also venture into uncharted territory by designing our very own algorithms. With the primary objective of creating novel solutions to these age-old conundrums, this coursework invites readers to embark on an enthralling journey through the world of algorithmic problem-solving.

**Problems and existing solutions**

**The Eight Queens Problem**

The Eight Queens problem is a classic chess puzzle in which the goal is to place eight queens on an 8x8 chessboard in such a way that no two queens threaten each other. This means that no two queens should share the same row, column, or diagonal.

**Existing Solutions:**

* **Breadth-First Search (BFS):** An approach to solving the Eight Queens problem that examines all possible queen placements level by level, expanding the search tree in breadth.

Text

Description automatically generated

Text

Description automatically generated

* **Depth-First Search (DFS):** DFS is a common method for solving the Eight Queens problem. It involves placing queens one-by-one on the chessboard, backtracking whenever a conflict arises.

Text

Description automatically generated

Text

Description automatically generated

* **Backtracking with Forward Checking:** This approach is an enhancement of the DFS method, where the algorithm not only backtracks but also checks for conflicts before placing a queen, reducing the search space.

Text

Description automatically generated

**Text

Description automatically generated**

**Knight's Tour**

The Knight's Tour problem involves moving a knight on an empty chessboard such that it visits every square exactly once. The challenge is finding a closed tour, where the knight ends up on a square that is one legal move away from its starting point.

**Existing Solutions:**

* **Depth-First Search (DFS):** DFS explores moves by diving deeper into the search tree before backtracking. It is generally faster but may not find a solution in some cases.

Text

Description automatically generated

Text

Description automatically generated

* **Warnsdorff's Heuristic:** This is a heuristic-based method that involves selecting the next move with the least number of onward moves. It is faster and more efficient than the other methods but may not always find a solution.

Text

Description automatically generated

**Tower of Hanoi**

The Tower of Hanoi is a classic puzzle that involves moving a stack of disks from one peg to another, using a third peg as an intermediary, while following certain rules: only one disk can be moved at a time, and a larger disk cannot be placed on top of a smaller disk.

**Existing Solutions:**

* **Recursive Algorithm:** The most common approach to solving the Tower of Hanoi is to use a recursive algorithm that breaks the problem down into smaller subproblems. This method guarantees a solution but can be slow for larger numbers of disks.

Text

Description automatically generated

**Our Solutions**

**RegalRunner**

The RegalRuler algorithm is a unique and innovative approach to solving the 8-queens problem. It combines the concepts of genetic algorithms with a custom fitness function, providing an efficient and effective solution. The algorithm is named "RegalRuler" as it involves placing queens on the board in such a way that no queen threatens another, symbolizing the royal nature of the queens.

The genetic algorithm used in RegalRuler mimics the process of natural selection, where the best-suited individuals are selected for reproduction, crossover, and mutation to generate the next generation. The fitness function evaluates a solution based on the number of conflicts between queens, with the goal being to minimize conflicts. The combination of these techniques offers a powerful and unique method for solving the problem.

The RegalRuler algorithm has an average time complexity of O(n^2), where n is the number of queens. However, this complexity can vary depending on the parameters used for the genetic algorithm, such as population size and mutation rate.

**Text

Description automatically generated**

**Text

Description automatically generated**

**ShadowWalker**

ShadowWalker is an original algorithm for solving the knight's tour problem. The name ShadowWalker represents the stealthy and strategic nature of a knight moving on the chessboard. The algorithm uses a depth-first search approach combined with a heuristic prioritization of distance to the center. Prioritizing distance to the center causes the knight to explore the center of the board first, increasing the likelihood of finding a solution. This is because the central squares provide the knight with more move options and thus lead to more flexible paths. The time complexity of the ShadowWalker algorithm is O(8^N), where N is the number of squares on the chessboard, and the space complexity is O(N).

Text

Description automatically generated

**DarkAbyss**

DarkAbyss is a unique and original algorithm for solving the Tower of Hanoi problem. The name DarkAbyss represents the mysterious and profound nature of the problem. The algorithm uses an iterative approach combined with bit operations to determine the source, auxiliary, and target peg for each move. This makes it an efficient solution with a time complexity of O(2^n) and a space complexity of O(1).

Text

Description automatically generated

Text

Description automatically generated

**Conclusion**

**Additional Resources**

To access the source code and files for this project, please visit our Github repository at the following link:

<https://github.com/DeyvidTheWise/Logic-Course-Assignment>

**Summary of Results**

In this project, we have explored various algorithms and methods to solve different problems, such as the Knight's Tour, Tower of Hanoi, and 8-Queens problems. It is a challenging task to create entirely new algorithms, as many solutions already exist. However, by combining different rules and algorithms, we can create innovative approaches that can potentially improve efficiency and effectiveness in solving these problems.

Our solutions, the ShadowWalker, DarkAbyss, and RegalRuler algorithms, are examples of this innovative approach. By combining existing methods and introducing new techniques, we have developed unique and efficient algorithms that tackle these classic problems. These algorithms demonstrate the power of combining multiple techniques and thinking outside the box to solve complex problems.

**Outlook and Possible Extensions**

In the future, there is a potential for further improvement and optimization of these algorithms. Additionally, new techniques and approaches can be explored to create even more efficient and effective solutions. By continually reevaluating and adapting our methods, we can contribute to the ongoing development and advancement of computer science and problem-solving techniques.