CSE121 - Discrete Mathematics

PROJECT REPORT
KNIGHT'S TOUR PROBLEM

Abstract

The Knight's Tour problem involves finding a Knight's Tour, that is, finding a sequence of moves by a Knight on a board of m x n size such that it visits every square on the board exactly once. Depending on the nature of the tour, it can be classified as either open or closed Knight's tour. The output includes a number in each cell, representing the move number of the Knight.

Introduction

Problem Statement: Given an m x n board with the Knight placed on the first block of an empty board. Moving according to the rules of chess, a knight must visit each square exactly once. If exists, print the order of each cell in which they are visited.

Let $N \in Z^+$, $M \in Z^+$ and let P(x,y) represent the proposition that "the knight visits the cell (x,y) only once".Then the problem can be written as, show that the proposition given below is true; for a M x N grid, $\forall x \forall y P(x,y)$ where $1 \leq x \leq M$ & $1 \leq y \leq N$.

If the knight ends its tour on a cell, which is one knight's move away from the beginning cell, it is called a **Closed Knight's Tour**, otherwise, it is an **Open Knight's Tour**.

It is an instance of a Hamiltonian path problem in graph theory, which involves finding a path in an undirected or directed graph that visits each vertex exactly once.

Existence

For a knight's tour (open/closed), $\forall m, \forall n \in Z^+$, if $m \leq n$, then it exists, unless:
m = 1 or 2 **or,** m = 3 and n = 3, 5, or 6 **or,** m = 4 and n = 4

For a closed knight's tour, Schwenk proved that for a m x n board with m≤n, it is always possible, unless:

m = 1 or 2, **or,** m = 3 and n = 3, 5, or 6, **or,** m = 4 and n = 4.

Existing Literature

- In Rudrata's Kavyalankara, in 9th century AD, the knight's tour pattern has been represented on a half-board, in the form of verses of poetry- with each syllable representing a square/cell on chessboard.
- Similarly, during 14th century, poet and philosopher Vedanta Desika, composed two Sanskrit verses where second verse is derived by doing a 4 x 8 knight's tour on the first verse.
- Bhat Nilakantha also described three closed and symmetric knight's tours in his book, even before the works of Euler.
- The first rule to complete knight's tour was given by H..C. von Warnsdorf in 1823.

  Existing Approaches

  Naive Approach : This involves systematically exploring all possible moves of the knight on the chessboard until a solution is found or all possibilities have been exhausted. So, in essence, it generates all tours one by one and checks whether they are knight's tours or not, and stops when one is found.
  This is also a **Brute-force Approach**.

  Backtracking Approach: In this method, you start with an empty plan and add moves (like the way a knight moves in chess) one by one. After each move, you check if it follows the rules. If a move breaks the rules, you undo it and try another option. If there are no more options, you go back to the previous move and try a different choice. This process continues until you either find a solution or exhaust all possibilities. The time complexity of this algorithm is O(8^(n^2)), where n is the size of a chessboard because each move has a maximum of 8 possible directions, and we have to explore all possible moves until we find a solution.

Neural Network Approach: Every knight's legal move is represented in terms of a neuron signal, with output value and state value, and its output is randomly initialised as 1 (active) or 0 (inactive), and the state value is initialized as 0.
When this network runs, each neuron changes its state and output based on its neighbors' state and outputs according to some pre-defined transition rules. Eventually, the network converges, and its result is either a knight's tour or a series of connected graphs.

Warnsdorff's rule: Warnsdorff's rule is a strategy to solve the Knight's Tour puzzle by having the knight choose squares with the fewest available moves and avoiding ones it has already visited.

This is discussed in detail in the next section since this is the approach we have based our project on.

The Topic of Study

We have explored the backtracking and Warnsdorff's algorithms for the project research and prepared a code implementing both of them to solve the Knight's tour. This implementation is significantly faster than the general backtracking approach. The implementation is as follows:

1. We count the number of valid moves from the given position and store them in an array.
2. Sort the array on the basis of these valid moves using the insertion sort algorithm.
3. We choose the next position of the knight which has the least amount of valid moves as stated by Warnsdorff's rules.
4. Repeat the above steps for this new position.
5. If at any point in the recursive stack, there is no valid move and the board is not covered we backtrack from that position until there is a valid move available.

We are using coordinates to define a position on the board. For instance, an 8x8 board has 64 positions denoted by (0,0) to (7,7). For a given block if a knight's tour is possible, it displays the board with the move number at each cell, in this case between 0 to 63 for the 8x8 board.

The worst-case time complexity of this approach is O(8^(m*n)), where m and n are the number of rows and columns respectively. The space complexity of this approach is O(m*n).

We have also implemented an automated tester to test Knight's tour over the entire board, i.e. with each block as a starting position, which involves parallel computing and shared memory for inter-process communication. It creates n*m child processes which execute knight's tour solver for each point as a starting position and update the board in shared memory with the tester. This is to check possible starting positions for a successful knight's tour and return successful and failed initials blocks for knight's tour over the entire board.

Conclusion

In summary, the Knight's Tour problem is still a captivating challenge in algorithmic problem-solving. Methods like Warnsdorff's rule work well, but researchers are looking into more advanced algorithms. The future might bring solutions using artificial intelligence, like reinforcement learning and neural networks, as we discussed, to create smart strategies for different board sizes. Also, exploring parallel computing and distributed algorithms could make finding knight tours faster and more efficient.

Members:

Dhruv Prakash, 2022168
Himanshu Raj, 2022216
Tanish Verma, 2022532