An explanation of Parberry's knight's tour algorithm

This document will explain the algorithm described in Ian Parberry's 1997 paper "An Efficient Algorithm for the Knight's Tour Problem". I will explain the Knight's tour problem, the notation used in the paper, and end with a detailed discussion of the divide and conquer algorithm eluded to in the paper. A reference implementation of the algorithm in Java is provided alongside this document.

My motivation for writing this document stems from the lack of accessible material on Parberry's paper online. I found the paper itself to be very dense and often lacking in explicit notes on points of importance. Other resources available online I also found to be vague or used excessive jargon relating to graph theory. I hope, therefore, to explain the algorithm and the concepts within the paper in a manner suitable for those not familiar with the notation used.

1 The Knight's tour

Parberry describes a tour as "A *knight's tour* is a series of moves made by a knight visiting each square of an $n \times n$ chessboard exactly once". This means we have a knight, who can move in a particular way that we will describe later, and this knight is somewhere on a square board that has some width and some height that we'll call n - our goal in creating a tour is to have this knight move to every square on the board without moving to a square that he's already been to.

Parberry notes that a *knight's tour problem* is about creating a tour for a knight on some specific board size, but the distinction between a knight's tour and a knight's tour *problem* is not important for our purposes and I will use the two interchangeably.

Parberry describes "A knight's tour is called *closed* if the last square visited is also reachable from the first square by a knight's move, and *open* otherwise". For a tour to be closed the starting square must be reachable by a *single* move. We will not include this final move in our discussions or the reference implementation, but it would be valid to do so.

1.1 Tour representation