Available online at www.sciencedirect.com
SCIENCE DIRECT®
Discrete Applied Mathematics 146 (2005) 219–232
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Optimal algorithms for constructing knight's tours on arbitrary $n \times m$ chessboards

Shun-Shii Lin[a], Chung-Liang Wei[b]

[a] *Graduate Institute of Computer Science and Information Engineering, National Taiwan Normal University, No. 88, Sector 4, Ting-Chow Road, Taipei, Taiwan, ROC*
[b] *Department of Information and Computer Education, National Taiwan Normal University, Taipei, Taiwan, ROC*

Received 6 May 2003; received in revised form 17 October 2004; accepted 5 November 2004
Available online 10 December 2004

Abstract

The knight's tour problem is an ancient puzzle whose goal is to find out how to construct a series of legal moves made by a knight so that it visits every square of a chessboard exactly once. In previous works, researchers have partially solved this problem by offering algorithms for subsets of chessboards. For example, among prior studies, Parberry proposed a divided-and-conquer algorithm that can build a closed knight's tour on an $n \times n$, an $n \times (n+1)$ or an $n \times (n+2)$ chessboard in $O(n^2)$ (i.e., linear in area) time on a sequential processor. In this paper we completely solve this problem by presenting new methods that can construct a closed knight's tour or an open knight's tour on an arbitrary $n \times m$ chessboard if such a solution exists. Our algorithms also run in linear time ($O(nm)$) on a sequential processor.

MSC: 68W05; 68Q25

Keywords: Divide-and-conquer algorithm; Knight's tour problem; Optimal algorithm

1. Introduction

There are several versions of the knight's problem, all of which use the moves of a knight to achieve some particular task. The most basic task is how to shift the knight from

E-mail address: linss@csie.ntnu.edu.tw (S.S. Lin)

0166-218X/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2004.11.002

a starting point to an ending point with the fewest moves on a given chessboard. Another revised version is how to find the shortest path on which the knight can visit all the given squares on a chessboard from a specified origin. Such kinds of knight's problems also appear in many programming contests.

A knight's tour is a series of moves made by a knight visiting every square of a chessboard exactly once. The knight's tour problem is the problem of constructing such a tour on a given chessboard. A knight's tour is called closed (or re-entrant) if the last square visited is also reachable from the first square by a single knight's move, and open, otherwise.

It is believed that the formal study of the knight's tour problem began on a standard 8×8 chessboard by Euler [9] in 1759. After Euler, there were many researchers who devoted themselves to this problem by using many different methods. In the beginning, studies were carried out on a smaller chessboards so that many skills frequently used by the artificial intelligence (AI) research community, like depth-first search, breadth-first search and heuristic methods, were adopted by most people. The advantages of these methods are intuition and ease of implementation. Unfortunately, with the use of larger chessboards, the computation time rises unacceptably rapidly. Dudeney [7,8] pointed out what kinds of rectangular chessboards have knight's tours; in particular, an $n \times n$ chessboard has a closed knight's tour if and only if n is even and greater than 5, and an open knight's tour if and only if n is greater than 4. Ball and Coxeter [2] revised the knight's tour problem by dividing the board horizontally into two rectangular compartments. The tour has to visit all the squares in one compartment before proceeding to the second; this is called the bisected knight's tour problem. They successfully found a solution to the 8×8 case due to Euler. Dudeney [7,8] made use of this idea and further refined it to divide the board into four rectangular compartments. The revision is called the quadrisected knight's tour problem. Domoryad [6] described a quadrisected open knight's tour on an 8×8 board and a closed knight's tour on a 7×7 board with its center square missed (all squares are visited except the center one). Hurd and Trautman [11] also noted an open knight's tour with one of its corners missed exists on a 4×4 board. Cannon and Dolan [3] settled the question of knight's tours on boards with an even number of squares ("even boards"), i.e., $n \times m$ is even. The basic result in their paper is that all even boards with $n, m \geqslant 6$ are tourable (namely, that there is an open knight's tour between any pair of opposite colored squares). Ralston [13] considered the question of open knight's tours on odd boards and discussed in what circumstances an odd board can be said to be odd-tourable. (That is, there is an open knight's tour between any pair of squares colored the same as the corner squares.) In 1994, Conrad et al. [4,5] proposed a linear time sequential algorithm to construct open knight's tours between any pair of squares on $n \times n$ boards for $n \geqslant 5$. In 1996, Rees [14] discussed and solved the knight's tour problem on a $3 \times n$ board.

In [12], Parberry presented a divide-and-conquer algorithm that can generate closed knight's tours on $n \times n$ or $n \times (n+2)$ boards in linear time (i.e. $O(n^2)$) for all even n and $n \geqslant 10$, and closed knight's tours missing one corner in linear time if n is odd and greater than 4. This algorithm can also construct closed knight's tours on $n \times (n+1)$ boards when $n \geqslant 6$. First, he found closed knight's tour on some smaller boards as bases. When facing a larger board, he divided it into four smaller pieces, generated a closed knight's tour for each compartment, and finally removed 4 edges and added 4 edges at the inside corners to

combine these four smaller closed knight's tours into a complete closed knight's tour for the larger board.

Note that Parberry's algorithm is only able to find closed knight's tours on $n \times n$, $n \times (n+1)$ or $n \times (n+2)$ boards when $n \geqslant 10$. His method fails to deal with boards of any other arbitrary size. In this paper, we will solve the knight's tour problem completely. We will propose new methods to work out all the unknown regions that previous researchers left unsolved. On an arbitrary $n \times m$ board, if there is a knight's tour on it, our methods can always find one in linear time.

2. Definitions and notations

The coordinates of the squares of an $n \times m$ board in the first row are $(0, 0)$, $(0, 1)$, $(0, 2)$, $\ldots$, $(0, m-1)$, and those in the second row are $(1, 0)$, $(1, 1)$, $(1, 2)$, $\ldots$, $(1, m-1)$ etc.. A knight's tour is said to be closed if the last square visited is also reachable from the first square by a single knight's move. In other words, the knight can visit every square exactly once along a closed knight's tour on the board and then go back to the starting point. A knight's tour in which every square on the board is visited exactly once but without being able to return to the origin in one move is called an open knight's tour.

The colors of the visited squares must be black and white interlaced when the knight moves on the chessboard. If the knight visits every square exactly once on the board, returning to the starting point, i.e. there exists a closed knight's tour, then the number of black squares must be equal to the number of white squares. But, on a board with both n and m odd, the difference in number of black and white squares is one, and there does not exist a closed knight's tour. In this circumstance, if we abandon one square, we may have a chance to find a closed knight's tour. The squares in the corners must belong to the group that has an extra square. By forsaking a corner square, our algorithms can find a closed knight's tour for the remaining squares on an $n \times m$ board if n, m are both odd and greater than 4. These kinds of knight's tours are referred to as corner-missed closed knight's tours.

The knight's moves shown in Fig. 1 are necessary for a structured knight's tour. If a knight's tour contains these moves, it is said to be structured [12].

Let us now define the stretched knight's tour. This is our own definition. The stretched knight's tour is a special case of the open knight's tour, but with two extra conditions added: (1) The starting and ending points must be a corner square and an adjacent square, respectively. (2) Except for the corner in which the starting point and the ending point are situated, the other three corners must satisfy the requirements of a structured knight's tour.

Fig. 2 shows a stretched knight's tour on a 6×6 board. In addition to the characteristics of the open knight's tour, the starting and ending points are located at $(0, 0)$ and $(0, 1)$, and satisfy the first condition mentioned above; the other three corners also satisfy the second condition.

Let us define the double-loop knight's tour. (This is also our own definition.) It is actually a pair of closed knight's tours where each tour forms a cycle visiting just half of the squares on the board and occupies two adjacent squares per column. The union of the pair of closed knight's tours must satisfy the requirement of a structured knight's tour.

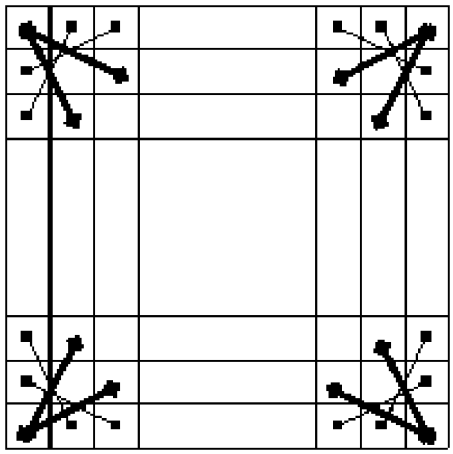

Fig. 1. Required moves for a structured knight's tour.

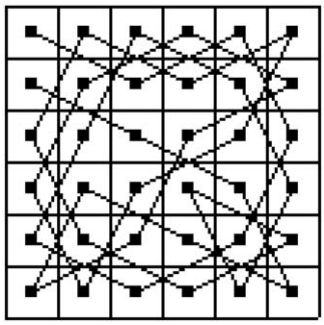

Fig. 2. A stretched knight's tour on a 6×6 board.

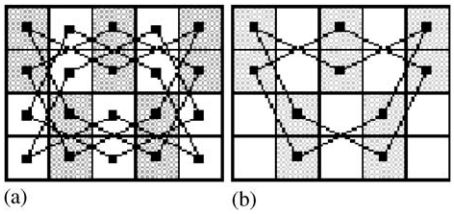

Fig. 3. An example of double-loop knight's tour: (a) a double-loop knight's tour on a 4×5 board, (b) only one loop is shown.

Fig. 3(a) shows a double-loop knight's tour on a 4×5 board, and Fig. 3(b) shows only one closed knight's tour that visits only half of the squares. It is obvious that the closed

knight's tour in Fig. 3(b) occupies two adjacent squares in each column and interlaces them on opposite sides between adjoining columns.

3. Parberry's algorithm

We discuss an innovative algorithm by Parberry [12] in this section. By knowing his method the reader can understand our new methods more easily. The first step of his method is to find some structured closed knight's tours on boards smaller than 10×10 as bases. When trying to generate a closed knight's tour on a larger board, he splits the board into four smaller pieces to construct a knight's tour for each piece and then deletes four edges to add another four edges as replacements. After that, these four independent closed knight's tours on the smaller boards are connected to create a complete closed knight's tour for the original board. This method claims that all the bases must be structured knight's tours. The reason is that some particular edges need to be removed when combining four smaller boards. If the knight's tours on them are structured, there are some fixed edges in the corners (e.g., A, B, C, D in Fig. 4(b)) that can be altered (e.g., E, F, G, H in Fig. 4(c)) and the structured characteristic of the combined larger board is guaranteed.

The principle of partitioning is to divide the board into four quadrants as evenly as possible. More precisely, each side of length $n = 4k$ for some $k \in N$ is divided into two parts of length $2k$, and each side of length $n = 4k + 2$ is divided into one part of length $2k$ and another of length $2(k + 1)$. For example, in the construction of an $n \times n$ board, where $n = 4k$, the four quadrants are each $2k \times 2k$. If $n = 4k + 2$, then the four quadrants are $2k \times 2k$, $2k \times 2(k+1)$, $2(k+1) \times 2k$, and $2(k+1) \times 2(k+1)$. Notwithstanding the length of each side is not always even, divide it into the most nearly equal two parts reserving the odd part for the first segment and the even one for the second segment. Hence, each side of length $4k + 1$ is partitioned into one part of length $2k + 1$ and the other part of length $2k$. Similarly, each side of length $4k + 3$ is divided into one part of length $2k + 1$ and the other part of length $2(k + 1)$.

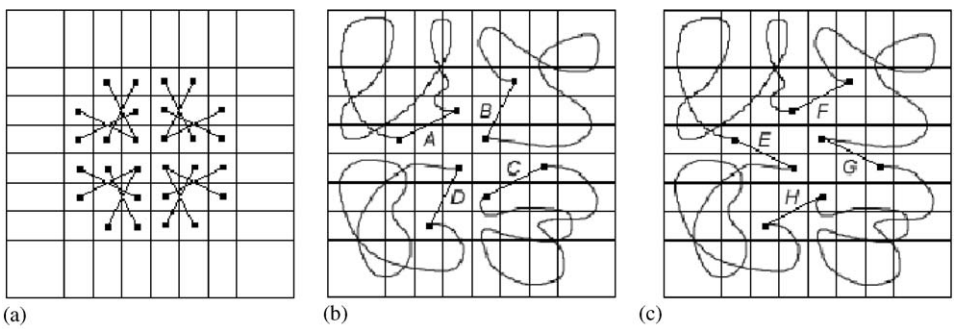

Fig. 4. The technique of merging four structured knight's tours into one: (a) the moves at the inside corners, (b) the edges A, B, C, D to be removed, and (c) the edges E, F, G, H to be added.

After partitioning and generating a closed knight's tour for each quadrant, Parberry adopts the strategy presented in Fig. 4 to integrate them with each other. Fig. 4(a) illustrates the moves at the inside corners of the quadrants. First, remove the four edges A, B, C, D shown in Fig. 4(b) and replace them with the four edges E, F, G, H shown in Fig. 4(c). Obviously, the outcome is also a structured knight's tour.

In the paper by Parberry, the boards that can be processed are all square-like boards, such as $n \times n$, $n \times (n+1)$ and $n \times (n+2)$. His algorithm can be easily implemented by using recursion. But, unfortunately, his method cannot be applied to an arbitrary rectangular $n \times m$ board.

4. The new methods

Parberry's divide-and-conquer algorithm is very clever and very efficient, but it only solves square like boards and leaves a lot of unknown regions. In this section, our main objective is to propose a novel approach to make it possible to completely solve the knight's tour problem.

4.1. Finding knight's tours on smaller boards

All the procedures discussed later need some solutions for smaller boards as bases, and later blend these bases to form complete solutions. We have written a program to find knight's tours on the boards smaller than 12×12. The method we used is fairly simple: take advantage of the backtracking trick and prune the search space that is unable to reach a solution. Though it takes a little time on some boards, it is more efficient than the brute force method. Which child node to expand first depends on the number of directions in which the knight can move. One child node is preferred if it has fewer successor moves than the other child nodes. Indeed, this strategy is known as Warnsdorff's method [1,16].

Fig. 5 shows what kinds of knight's tours exist on smaller boards, where 'X' indicates that no knight's tour can be found, 'O' indicates that there exists an open knight's tour, 'C' indicates that a close knight's tour can be found, and 'E' indicates that a corner-missed closed knight's tour exists.

We have established a website (http://www.csie.ntnu.edu.tw/∼linss/knighttours/bases.html) that includes the bases needed while generating a knight's tour on a larger board including the structured closed knight's tours, the structured corner-missed closed knight's tours, the stretched knight's tours, and the structured open knight's tours.

4.2. Closed knight's tours and corner-missed closed knight's tours

First, we introduce two previous ideas. The mathematician Allen Schwenk [15] provides an interesting characterization of those rectangular boards on which there exists a closed knight's tour. He finds that an $n \times m$ board, where $n \leqslant m$, has a knight's circuit (an alias of the closed knight's tour) unless any of the following conditions is satisfied: (1) Both n and m are odd, (2) $n = 1$, 2 or 4, (3) $n = 3$ and $m = 4$, 6 or 8.

n \ m	1	2	3	4	5	6	7	8	9	10	11	12
1	X	X	X	X	X	X	X	X	X	X	X	X
2	X	X	X	X	X	X	X	X	X	X	X	X
3	X	X	X	O	X	X	O	O	E	C	E	C
4	X	X	O	X	O	O	O	O	O	O	O	O
5	X	X	X	O	E	C	E	C	E	C	E	C
6	X	X	X	O	C	C	C	C	C	C	C	C
7	X	X	O	O	E	C	E	C	E	C	E	C
8	X	X	O	O	C	C	C	C	C	C	C	C
9	X	X	E	O	E	C	E	C	E	C	E	C
10	X	X	C	O	C	C	C	C	C	C	C	C
11	X	X	E	O	E	C	E	C	E	C	E	C
12	X	X	C	O	C	C	C	C	C	C	C	C

Fig. 5. The kinds of knight's tours that exist on small boards.

In addition, there is much information on the Internet related to the knight's tour. Ted Filler's web page [10] quotes Allen Schwenk's amazing discovery and points out that if there is a closed knight's tour on an $n \times m$ board then there must exist one on an $n \times (m+4)$ board, also on an $(n+4) \times m$ board.

These two ideas give us some cues. We found that it is possible to combine knight's tours on two boards to generate a closed knight's tour (or a corner-missed closed knight's tour). In addition to the bases provided by Parberry, we also gather a group of stretched knight's tours to be another set of bases. When combining knight's tours, both of these two sets are required. Hence, we are able to combine a stretched knight's tour and a structured knight's tour to form a larger structured knight's tour.

For dealing with the four quadrants of an $n \times m$ board, we propose a new strategy to replace Parberry's method. First, we separately and horizontally combine the two quadrants on the top and the two quadrants on the bottom and then combine these vertically. Note that during the combination process only one board is required to have a closed knight's tour (or a corner-missed closed knight's tour), e.g. board A in Fig. 6, and the other three boards simply need stretched knight's tours. Note that in Fig. 6, the stretched knight's tour on board C is flipped and rotated from the original stretched knight's tour. Now boards A and B are combined, as are boards C and D, by applying the method depicted in Fig. 6. Then combine board A with board C to integrate the four knight's tours into a complete closed knight's tour (or a corner-missed closed knight's tour). Note that the top-left quadrant of the original larger board needs a closed knight's tour (or a corner-missed closed knight's tour) while the remaining quadrants need only have stretched knight's tours.

Due to the absence of closed knight's tours and corner-missed closed knight's tours on 3×5, 3×6, 3×7 and 3×8 boards, we need to deal with $3 \times m$ boards separately. As shown in Fig. 5, we have closed knight's tours on 3×10 and 3×12 boards, corner-missed closed knight's tours on 3×9 and 3×11 boards and a stretched knight's tour on a 3×4 board. When tackling $3 \times m$ boards, we must select a basis from among the 3×9, 3×10, 3×11 or 3×12 boards and concatenate some number of 3×4 boards with the stretched knight's tours.

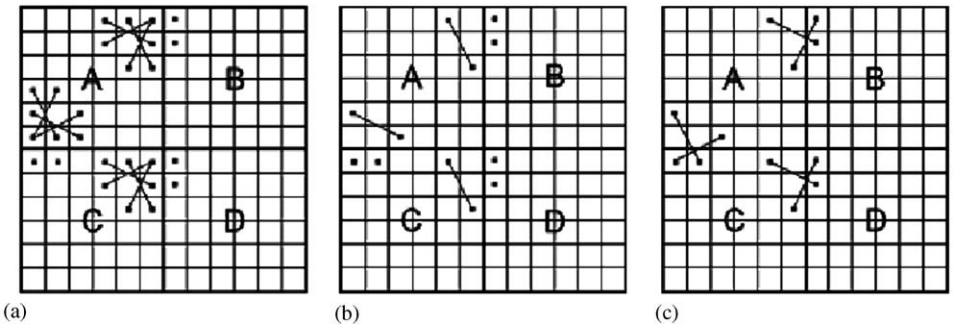

Fig. 6. Our strategy of combining four quadrants: (a) the moves at the inside corners, (b) the edges to be discarded, and (c) the edges to be added.

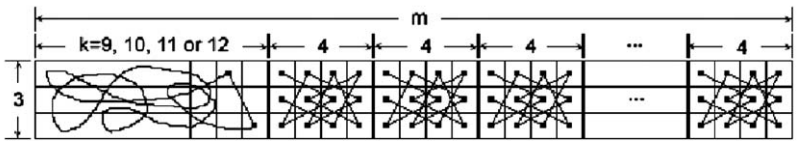

Fig. 7. The partition of a $3 \times m$ board.

The skeleton of our method of coping with an $n \times m$ board, $n \leqslant m$, is shown in Algorithm Knight(n, m).

Algorithm knight (n, m)

{

Case 1: $n \leqslant 10$ and $m \leqslant 10$. Use the available bases shown in Fig. 5. Choose a closed knight's tour (or a corner-missed knight's tour) or a stretched knight's tour depending on the requirements. If a basis can be found in Fig. 5, then it is selected as the answer, otherwise there is no solution for this board.

Case 2: $n = 3$ and $m > 10$. The board can be partitioned as Fig. 7. In this case, a $3 \times k$ board with a closed knight's tour and $(m - k)/4$ pieces of 3×4 boards with stretched knight's tours are needed, where $k = [(m - 9) \bmod 4] + 9$. Now we combine each pair of adjacent boards. Fig. 8 is an illustration of joining the partitions.

Case 3: $4 \leqslant n \leqslant 10$ and $m > 10$. The board can be divided as shown in Fig. 9. After partitioning, the two boards $n \times m_1$ and $n \times m_2$ can be combined as in Fig. 10, where $m_1 = \lfloor m/4 \rfloor \times 2 + (m \bmod 2)$ and $m_2 = m - m_1$.

Case 4: $n > 10$ and $m > 10$. The board can be partitioned as shown in Fig. 11. In this case $n_1 = \lfloor n/4 \rfloor \times 2 + (n \bmod 2)$, $n_2 = n - n_1$, $m_1 = \lfloor m/4 \rfloor \times 2 + (m \bmod 2)$ and $m_2 = m - m_1$. Fig. 12 shows how to combine the four quadrants.

}

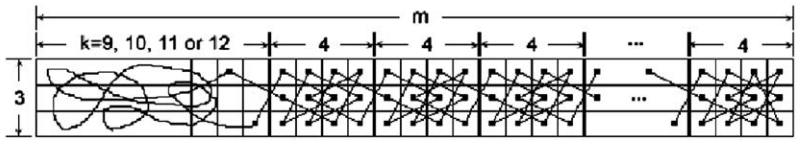

Fig. 8. The way to combine the knight's tours to form a complete tour for Fig. 7.

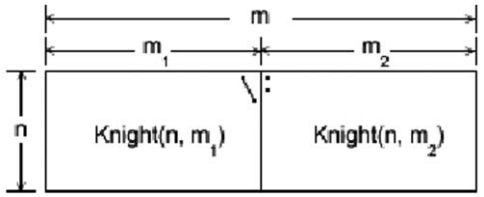

Fig. 9. The moves at the inside corners between $n \times m_1$ and $n \times m_2$ boards.

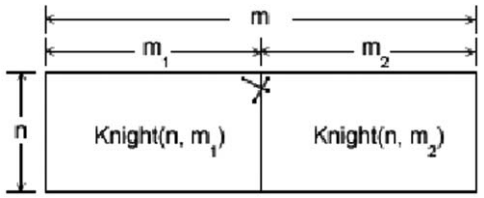

Fig. 10. The combination result for Fig. 9.

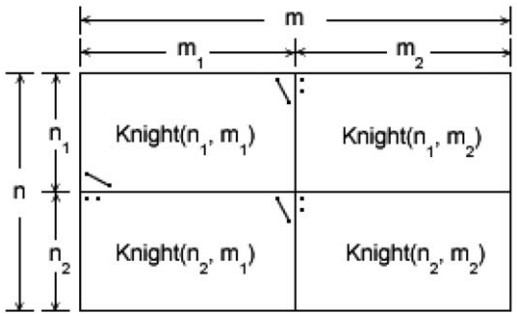

Fig. 11. The moves at the inside corners among the quadrants $n_1 \times m_1$, $n_1 \times m_2$, $n_2 \times m_1$ and $n_2 \times m_2$.

In Case 3 of the above algorithm, we want to divide m as evenly as possible. But in case we are unable to meet this requirement, we would make the first partition m_1 odd. In this situation, we choose $m_1 = \lfloor m/4 \rfloor \times 2 + (m \bmod 2)$ and $m_2 = m - m_1$. This partition rule guarantees that m_2 is always even. In Case 4, the rule of partitioning is of the same meaning as Case 3.

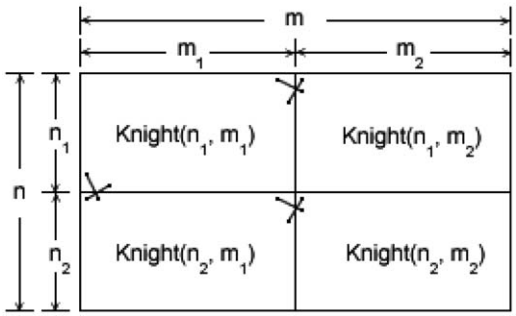

Fig. 12. The way to combine the knight's tours of four quadrants shown in Fig. 11 to form a complete knight's tour.

Now let us analyze the run time $T(n, m)$ of this algorithm. The area is $n \times m$ and partitioned into two or four disjoint subareas with a constant extra cost c. Therefore, the number of partitions is less than $n \times m$. Hence, the cost is bounded by $c \times n \times m$. Therefore it takes linear time $O(n \times m)$ to generate a closed knight's tour on an $n \times m$ board.

The following Theorem 1 is due to Allen Schwenk [15]. Here we prove it from an algorithmic point of view.

Theorem 1. *There exists a closed knight's tour on an $n \times m$ board, $n \leqslant m$, if and only if* (1) *n and m are not both odd, and* (2) *$n \geqslant 5$ or ($n = 3$ and $m \geqslant 10$).*

Proof. ($=>$) The proof of "only if" can be omitted since it was previously known.

($<=$) When $n = 3$, as Fig. 5 shows, we have closed knight's tours on 3×10 and 3×12 boards, and a stretched knight's tour is also available on a 3×4 board. So Algorithm Knight(n, m) can construct closed knight's tours on boards of dimensions 3×14, 3×16, 3×18, $3 \times 20, \ldots$, and so on.

Now let us consider the condition of $n \geqslant 5$. The claim is easily seen to be true for $5 \leqslant n \leqslant m \leqslant 10$ with at least one of n and m being even by inspecting Fig. 5, where the knight's tours were obtained using the simple algorithm described in the previous subsection "Finding knight's tours on smaller boards". Assume that $5 \leqslant n \leqslant 10$ and $m \geqslant 11$ with at least one of n and m being even. In Algorithm Knight(n, m), length m will be divided recursively according to the partition rule, and eventually it will be decomposed into lengths $m_1, m_2, m_3, \ldots, m_k$. Note that only $5 \leqslant m_1 \leqslant 10$ may possibly be odd, and the rest are even with possible values 6, 8 or 10. The combining strategy shown in Algorithm Knight(n, m) guarantees that the top-left quadrant of the originally larger board needs to have a closed knight's tour (or a corner-missed closed knight's tour) and the other partitions need stretched knight's tours. Since, (1) a closed knight's tour exists on an $n \times m_1$ board with $5 \leqslant m_1 \leqslant 10$, (2) at least one of n and m_1 is even, and (3) stretched knight's tours exists on $n \times 6$, $n \times 8$ and $n \times 10$ boards with $5 \leqslant n \leqslant 10$, so the closed knight's tour on an $n \times m$ board can be constructed by applying Algorithm Knight(n, m).

Now, suppose that $11 \leqslant n \leqslant m$ and at least one of n and m is even. After dividing n and m recursively we get $n_1, n_2, n_3, \ldots, n_i$ and $m_1, m_2, m_3, \ldots, m_j$. By the partition rule, only

one of n_1 and m_1 may be odd with $5 \leqslant n_1 \leqslant 10$ and $5 \leqslant m_1 \leqslant 10$; the rest are even and their possible values are 6, 8 or 10. Since closed knight's tours exist on 5×6, 5×8, 5×10, 6×6, 6×7, 6×8, 6×9, 6×10, 7×8, 7×10, 8×8, 8×9, 8×10, 9×10 and 10×10 boards and stretched knight's tour exists on 5×6, 5×8, 5×10, 6×6, 6×8, 6×10, 7×6, 7×8, 7×10, 8×6, 8×8, 8×10, 9×6, 9×8, 9×10, 10×6, 10×8 and 10×10 boards, a closed knight's tour on an $n \times m$ board can be constructed by applying Algorithm Knight(n, m).

The foregoing has proved that there exists a closed knight's tour on an $n \times m$ board if not both n and m are odd and $n \geqslant 5$ or ($n = 3$ and $m \geqslant 10$). □

Theorem 2. *There exists a corner-missed closed knight's tour on an $n \times m$ board, $n \leqslant m$, if and only if* (1) *both n and m are odd, and* (2) $n \geqslant 5$ *or* ($n = 3$ *and* $m \geqslant 9$).

Proof. The proof is similar to that of Theorem 1. We omit it here. □

4.3. Open knight's tours

The way to construct an open knight's tour is quite easy for some boards. If there exists a closed knight's tour on an $n \times m$ board, we can just remove an arbitrary edge from the tour and then we get an open knight's tour. On the other hand, if we have a corner-missed closed knight's tour on the board, redirecting an edge to a corner can also produce an open knight's tour.

However, the strategies shown above fail to solve some boards such as 3×7, 3×8 and so on. There is another simple way to tackle this problem. We can apply Algorithm Knight(n, m) described in the previous subsection by providing another set of open knight's tours as bases to substitute for that of closed knight's tours or corner-missed closed knight's tour. A slight modification is needed for Case 1 of Algorithm Knight(n, m): choose an open knight's tour instead of a closed knight's tour (or a corner-missed closed knight's tour). Now only the top-left smaller board of the originally larger board needs an open knight's tour, all the other partitions adopt stretched knight's tours. Unfortunately, we cannot find any stretched knight's tours that can be used to extend $4 \times k$ boards horizontally with $k < m$, so the above methods are still not sufficient to generate open tours on all $n \times m$ boards, if they exist. We have to add a special case to handle $4 \times m$ boards. Since, we have no stretched knight's tour to extend $4 \times k$ boards horizontally, we introduce the idea of a double-loop knight's tour to substitute for stretched knight's tours when dealing with $4 \times m$ boards in Case 3 of Algorithm OpenKnight(n, m). Our algorithm can also generate open knight's tours on such boards in linear time if they exist.

Algorithm OpenKnight (n, m)

{

Case 1: $n \leqslant 10$ and $m \leqslant 10$. Use the bases shown in Fig. 5. Choose an open knight's tour or a stretched knight's tour, depending on requirements. If a basis can be found in Fig. 5, then it is selected as the answer, otherwise there is no solution for this board.

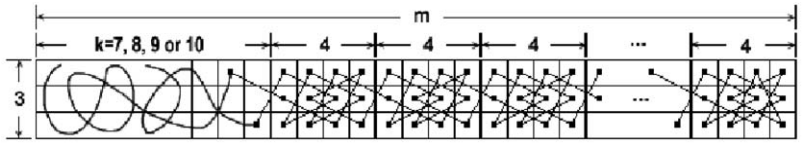

Fig. 13. The partition of a $3 \times m$ board.

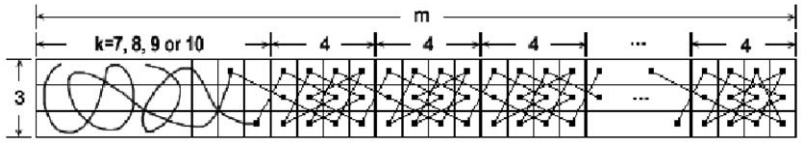

Fig. 14. The way to combine the knight's tours to form a complete tour for Fig. 13.

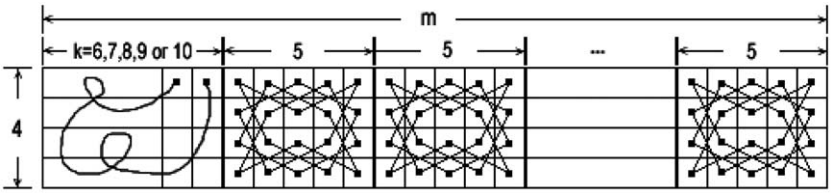

Fig. 15. The partition of a $4 \times m$ board.

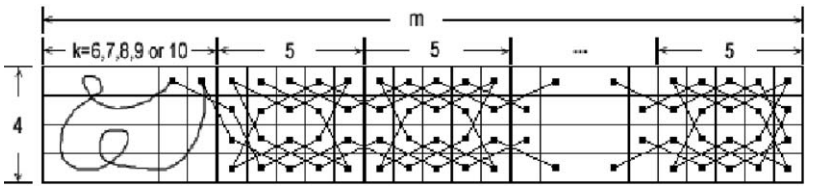

Fig. 16. The way to combine a stretched knight's tour and several double-loop knight's tours to form a complete tour for Fig. 15.

Case 2: $n = 3$ and $m > 10$. The board can be partitioned as shown in Fig. 13. In this case, a $3 \times k$ board with open knight tour and $(m - k)/4$ pieces of 3×4 boards with stretched knight's tour are needed, where $k = [(m - 7) \bmod 4] + 7$. Now we combine each pair of adjacent boards by applying the method depicted in Fig. 14.

Case 3: $n = 4$ and $m > 10$. The board can be partitioned as shown in Fig. 15. In this case, a $4 \times k$ board with a stretched knight tour and $(m - k)/5$ pieces of 4×5 boards with a double-loop knight's tour are needed, where $k = [(m - 6) \bmod 5] + 6$. Now we combine each pair of adjacent boards by applying the method depicted in Fig. 16.

Case 4: $5 \leqslant n \leqslant 10$ and $m > 10$. This is similar to Case 3 of Knight(n, m).
Case 5: $n > 10$ and $m > 10$. This is similar to Case 4 of Knight(n, m).
}

Theorem 3. *There exists an open knight's tour on an $n \times m$ board, $n \leqslant m$, if and only if* (1) $n = 3$ *and* ($m = 4$ *or* $m \geqslant 7$) *or* (2) $n \geqslant 4$ *and* $m \geqslant 5$.

Proof. The proof is similar to that of Theorem 1. We omit it here. □

5. Conclusion

In this paper, we have found new methods to conquer all the unknown areas of the knight's tour problem. So far we can use our algorithms to construct an open knight's tour, a closed knight's tour, or a corner-missed closed knight's tour on an arbitrary $n \times m$ board very quickly if a solution exists. Our algorithms run in O(nm) time (i.e. linear time) and solve the famous old knight's tour problem completely.

Follow-up work may lead to the construction of an efficient algorithm to solve the knight's tour problem from a given starting point to a given ending point, if it exists, on an $n \times m$ board. We hope that the methodologies proposed in this paper will prompt researchers to study other related problems.

Acknowledgements

We thank the reviewers for their valuable comments and suggestions. This research was supported in part by a Grant NSC89-2213-E-003-007 from National Science Council, ROC. Our gratitude also goes to the Academic Paper Editing Clinic, National Taiwan Normal University.

References

[1] W.W.R. Ball, Mathematical Recreations and Essays, eleventh ed., MacMillan, New York, 1939.
[2] W.W.R. Ball, H.S.M. Coxeter, Mathematical Recreations and Essays, twelfth ed., University of Toronto Press, Toronto, 1974.
[3] R. Cannon, S. Dolan, The knight's tour, Mathematical Gazette 70 (1986) 91–100.
[4] A. Conrad, T. Hindrichs, H. Morsy, I. Wegener, Wie es dem Springer gelingt, Schachbretter beliebiger Größe und zwischen beliebig vorgegebenen Anfangs-und Endfeldern vollständig abzuschreiten, Spektrum der Wissenschaft (1992) 10–14.
[5] A. Conrad, T. Hindrichs, H. Morsy, I. Wegener, Solution of the knight's Hamiltonian path problem on chessboards, Discrete Appl. Math. 50 (1994) 125–134.
[6] A.P. Domoryad, Mathematical Games and Pastimes, Pergamon Press, Oxford, 1964.
[7] H.E. Dudeney, Amusements in Mathematics, Thomas, Springfield, 1917.
[8] H.E. Dudeney, Amusements in Mathematics, Dover, New York, 1970.
[9] L. Euler, Sulution d'une question curieuse qui ne paroit soumise a aucune analyse, Mem. Acad. Sci. Berlin (1759) 310–337.
[10] T. Filler, The Solution [online]. In Knight's Tour at: http://home.earthlink.net/t˜filler/bts.htm, Accessed January 25, 2002.

[11] S.P. Hurd, D.A. Trautman, The knight's tour on the 15-puzzle, Math. Mag. 66 (3) (1993) 159–166.
[12] I. Parberry, An efficient algorithm for the knight's tour problem, Discrete Appl. Math. 73 (1997) 251–260.
[13] A. Ralston, Knight's tours on odd boards, J. Recreational Math. 28 (3) (1996) 194–200.
[14] G.H.J. van Rees, Knight's tours and circuits on the $3 \times n$ chessboard (classroom notes), Bull. ICA 16 (1996) 81–86.
[15] A. Schwenk, Which rectangular chessboards have a knight's tour?, Math. Mag. 64 (5) (1991) 325–332.
[16] M. Valtorta, M.I. Zahid, Warnsdorff's tours of a knight, J. Recreational Math. 25 (4) (1993) 263–275.