ELSEVIER
Discrete Applied Mathematics 73 (1997) 251–260
DISCRETE APPLIED MATHEMATICS

An efficient algorithm for the Knight's tour problem

Ian Parberry *

Department of Computer Sciences, University of North Texas, P.O. Box 13886, Denton TX 76203–6886, USA

Received 17 October 1994; revised 31 October 1995

1. Introduction

A *knight's tour* is a series of moves made by a knight visiting every square of an $n \times n$ chessboard exactly once. The *knight's tour problem* is the problem of constructing such a tour, given n. 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. Define the *knight's graph* for an $n \times n$ chessboard to be the graph $G = (V, E)$, where $V = \{(i,j) \mid 1 \leqslant i,j \leqslant n\}$, and $E = \{((i,j),(k,\ell)) \mid \{|i-k|, |j-\ell|\} = \{1,2\}\}$. That is, there is a vertex for every square of the board and an edge between two vertices exactly when there is a knight's move from one to the other. Then, more formally, an open knight's tour is defined to be a Hamiltonian path, and a closed knight's tour is defined to be a Hamiltonian cycle on a knight's graph. A knight's graph has n^2 vertices and $4n^2 - 12n + 8$ edges.

The formal study of the knight's tour problem is said to have begun with Euler [10] in 1759, who considered the standard 8×8 chessboard. Rouse Ball and Coxeter [1] give an interesting bibliography of the history of the problem from this point. Dudeney [8, 9] contains a description of exactly which rectangular chessboards have knight's tours; in particular, an $n \times n$ chessboard has a closed knight's tour iff $n \geqslant 6$ is even,[1] and an open knight's tour iff $n \geqslant 5$. It is not clear who first proved this fact, but it appears to be part of the folklore of the subject (see, for example, [2]). There exist several independently conceived linear time (i.e. $O(n^2)$) algorithms for constructing knight's tours (see, for example, [5, 19]). Takefuji and Lee [22, 23] recently proposed a neural network solution to the knight's tour problem, although it appears to be of little use in practice (see [17, 18]). We will describe in this paper a new, simple, and fast algorithm for constructing knight's tours on square boards.

* Email: ian@cs.unt.edu

Research supported by the National Science Foundation under grant number CCR–9302917, and by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under grant number F49620–93–1–0100.

[1] It is easy to see that there is no closed knight's tour when n is odd since such a board has one more white square than black, or vice versa, and since the colours of the squares visited on a knight's tour must alternate.

PII S0166-218X(96)00010-8

Rouse Ball and Coxeter [1] describe a variant of the knight's tour problem in which the board is divided horizontally into two rectangular compartments. The tour must visit all of the squares in one compartment before proceeding to the second one. We will call this the *bisected knight's tour problem.* They give a solution to the 8×8 case due to Euler, and another due to Roget. Dudeney [8, 9] describes a further refinement of this problem in which the board is divided into four rectangular compartments. We will call this the *quadrisected knight's tour problem.* Dudeney sets the problem of constructing a quadrisected knight's tour on an 8×8 board as an exercise that "is not difficult, but will be found very entertaining and not uninstructive". We will describe a linear-time algorithm for generating quadrisected knight's tours on $n \times n$ boards for all even $n \geqslant 10$.

Domoryad [7] describes a quadrisected open tour on an 8×8 board, and also a closed tour on a 7×7 board that is missing the centre square. Hurd and Trautman [12] note that there exists an open knight's tour on a 4×4 board that is missing one of its corners. We will also present an algorithm for constructing closed knight's tours on $n \times n$ boards that are missing a corner square, for all odd $n \geqslant 5$.

Gardner [11, Chapter 14] gives a bisected tour on an 8×8 board due to Euler. He also notes that this tour is invariant under a rotation of 180° (that is, the transformation $(x, y) \rightarrow (n-x+1, n-y+1)$). We will see a linear time algorithm for generating such rotation-invariant knight's tours on an $n \times n$ board for all even $n \geqslant 10$. In contrast, Dejter [6] has shown that an $n \times n$ has a closed knight's tour that is invariant under a 90° rotation (that is, the transformation $(x, y) \rightarrow (y, n-x+1)$) iff $n \geqslant 6$ is divisible by 2 but not by 4. We will give a linear-time algorithm for all such $n \geqslant 10$.

Conrad et al. [3, 4] give a linear-time sequential algorithm for the more difficult problem of constructing open knight's tours (with arbitrary endpoints) that can also be adapted to give a parallel algorithm that runs in $O(1)$ time on $O(n^2)$ processors. We will give a new algorithm for closed knight's tours with the same resource bounds. The new algorithm has the following advantages: the sequential algorithm is easy to describe, and easy to implement, and the parallel version is easy to describe and analyze. In addition, both the sequential and parallel versions extend to the special cases described above: quadrisected tours, tours symmetric under 180° rotations, tours symmetric under 90° rotations, and tours on odd-sided boards that are missing a single square.

The remainder of this paper is divided into two sections. Section 2 describes a new divide-and-conquer algorithm for constructing regular knight's tours, quadrisected knight's tours, and tours that are invariant under rotations. Section 3 presents variants of the new divide-and-conquer algorithm for several popular parallel machine architectures. Throughout this paper, $\mathbb{N}$ denotes the set of natural numbers (including zero). Unless otherwise qualified, a "knight's tour" will mean a *closed* knight's tour.

2. A divide-and-conquer algorithm

This section is devoted to describing a new, particularly simple, linear-time divide-and-conquer algorithm for knight's tours of various types. A knight's

tour is said to be *structured* if it includes the knight's moves shown in Fig. 1.

Theorem 2.1. *For all even $n \geqslant 6$ there exists a structured knight's tour on an $n \times n$ and an $n \times (n+2)$ board. Such a tour can be constructed in time* $O(n^2)$.

Proof. The proof is by induction on n. The claim is easily seen to be true for $6 \leqslant n \leqslant 10$ by inspecting Fig. 2 (the knight's tours in this figure were obtained using the random walk algorithm described in Section 1).

Now suppose that $n \geqslant 12$ is even and that structured knight's tours exist on $m \times m$ and $m \times (m+2)$ boards for all $6 \leqslant m < n$. Divide the $n \times n$ board into four quadrants as evenly as possible. More precisely, each side of length $n = 4k$ for some $k \in \mathbb{N}$ is divided into two parts of length $2k$, and each side of length $4k+2$ for some $k \in \mathbb{N}$ is divided into a part of length $2k$ and a part of length $2(k+1)$. In the construction of an $n \times n$ board in which $n = 4k$ for some $k \in \mathbb{N}$, the four quadrants are each $2k \times 2k$. Alternatively, if $n = 4k+2$ for some $k \in \mathbb{N}$, then the four quadrants are either $2k \times 2k$, $2k \times 2(k+1)$, $2(k+1) \times 2k$, or $2(k+1) \times 2(k+1)$. In the construction

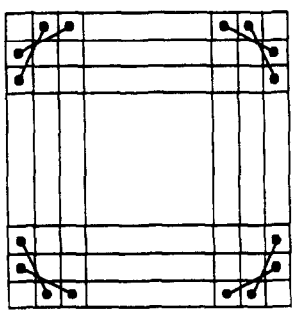

Fig. 1. Required moves for a structured knight's tour.

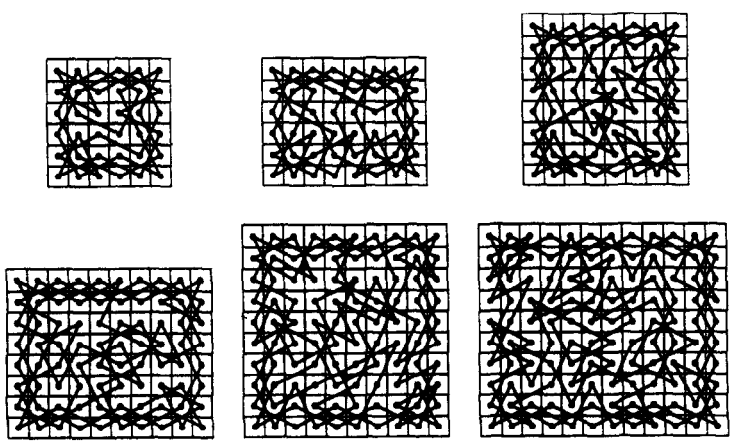

Fig. 2. Structured knight's tours for (in row-major order) 6×6, 6×8, 8×8, 8×10, 10×10, and 10×12 boards.

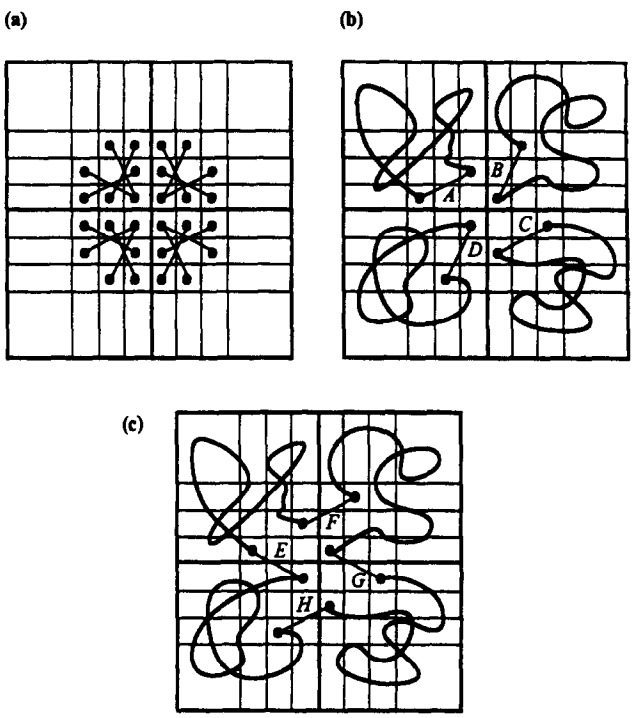

Fig. 3. How to combine four structured knight's tours into one: (a) the moves at the inside corners, (b) the edges A, B, C, D to be deleted, and (c) the replacement edges E, F, G, H.

of an $n \times (n+2)$ board in which $n = 4k$ for some $k \in \mathbb{N}$, the four quadrants are either $2k \times 2k$, $2k \times 2(k+1)$, or $2(k+1) \times 2k$. Alternatively, if $n = 4k+2$ for some $k \in \mathbb{N}$, then the four quadrants are either $2k \times 2k$, $2k \times 2(k+1)$, $2(k+1) \times 2k$, or $2(k+1) \times 2(k+1)$. Since $n \geqslant 12$ implies that $2k \geqslant 6$, knight's tours in each quadrant exist (by the induction hypothesis) in all of the above cases.

The moves at the inside corners of the quadrants are illustrated in Fig. 3(a). (Although the moves from the corner square were not specified in Fig. 1, note that there are no other choices for knight's moves out of a corner square.) The four tours are combined by deleting the edges A, B, C, D shown in Fig. 3(b) and replacing them with the four edges E, F, G, H shown in Fig. 3(c). Clearly, the result is a structured knight's tour. Fig. 4 illustrates the technique on a 16×16 board, constructed from four copies of the knight's tour on an 8×8 board in Fig. 2.

The technique described above can easily be implemented as a recursive algorithm with the tour represented in graph form using an adjacency matrix representation. The running time $T(n)$ required for the construction of a knight's tour on an $n \times n$ board is given by the following recurrence: $T(8) = O(1)$, and for $n \geqslant 16$ a power of 2, $T(n) = 4T(n/2) + O(1)$. This recurrence has solution $T(n) = O(n^2)$. Therefore (using the standard argument), the running time for all even $n \geqslant 6$ is $O(n^2)$. □

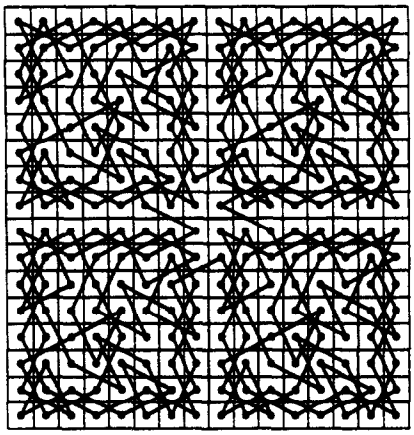

Fig. 4. A 16 × 16 knight's tour constructed from the 8 × 8 knight's tour in Fig. 2 using the technique of Theorem 2.1.

The algorithm of Theorem 2.1 is particularly easy to implement, and can be used to construct knight's tours of size up to 1000 × 1000 in under 11 s on a SUN SPARC 2.

The construction described in the proof of Theorem 2.1 can also be used to obtain some knight's tours with interesting properties. It should be clear that we have already solved the quadrisected knight's tour problem. The only even values of n for which we have not constructed quadrisected knight's tours are $n = 8, 10$. A quadrisected knight's tour is known for an 8 × 8 board (see Dudeney [7–9]) leaving $n = 10$ open.

Another minor modification to our algorithm also delivers symmetric knight's tours.

Theorem 2.2. *For all $n \times n$ boards where $n \geqslant 12$ is divisible by* 4, *there exists a quadrisected knight's tour that is symmetric under a 180° rotation. Such a tour can be constructed in time* $O(n^2)$.

Proof. If $n \geqslant 12$ is divisible by 4, the construction of Theorem 2.1 breaks the board into quadrants of equal size and uses the same initial tour within each quadrant. Instead, after constructing a tour in the first quadrant, rotate it three times through successive increments of 90°, once for each of the remaining quadrants in cyclic order. Complete the tour as before. The constructed tour will be symmetric under a 180° rotation. For example, Fig. 5 shows such a tour for a 16 × 16 and a 20 × 20 board. □

Dejter [6] has shown that there exists a knight's tour that is symmetric under a 90° rotation iff $n \geqslant 6$ is even but not divisible by 4. We will now give an algorithm for constructing such a tour for all such $n \geqslant 10$. First, we need some straightforward variations on Theorem 2.1.

Lemma 2.3. *For all $n \geqslant 6$ there exists a structured knight's tour on an $n \times (n + 1)$ board. Such a tour can be constructed in time* $O(n^2)$.

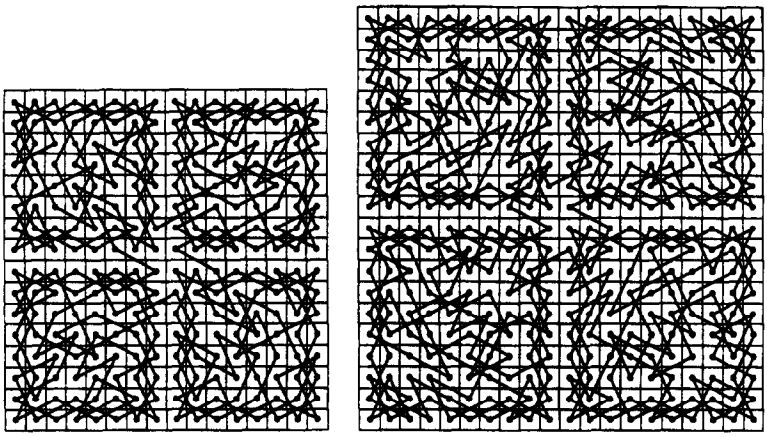

Fig. 5. 16×16, and 20×20 knight's tours that are symmetric under a 180° rotation, constructed using the technique of Theorem 2.2.

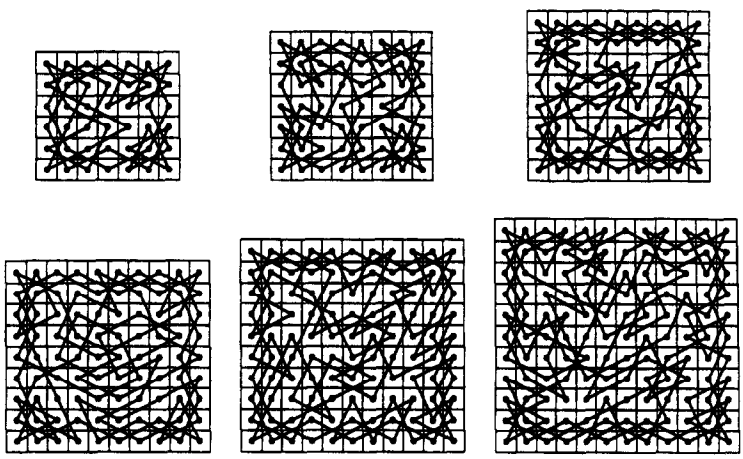

Fig. 6. Structured knight's tours for (in row-major order) 6×7, 7×8, 8×9, 9×10, 10×11, and 11×12 boards.

Proof. The construction is very similar to that of Theorem 2.1 and is left to the interested reader. The base of the recursive construction must be augmented with the structured knight's tours shown in Fig. 6. □

Lemma 2.4. *For all odd $n \geqslant 5$ there exists a structured knight's tour on an $n \times n$ board that is missing one of its corner squares. Such a tour can be constructed in time $O(n^2)$.*

Proof. The recursive construction is very similar to that of Theorem 2.1 and is left to the interested reader. The constructions of both Theorem 2.1 and Lemma 2.3 must be used as subroutines. The base of the recursive construction consists of the base cases

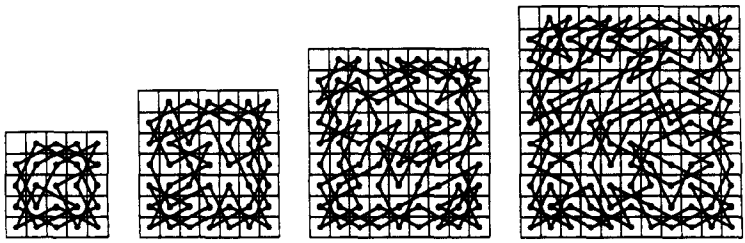

Fig. 7. Structured knight's tours for (from left to right) 5×5, 7×7, 9×9, and 11×11 boards that are missing the upper left corner square.

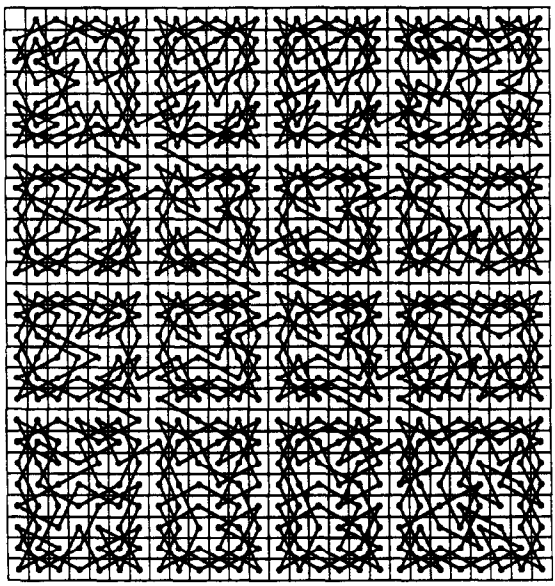

Fig. 8. A knight's tour on a 27×27 board that is missing one square.

from both of those results, plus the structured knight's tours shown in Fig. 7. Fig. 8 illustrates the construction when $n = 27$. □

Theorem 2.5. *For all $n \times n$ boards where $n \geqslant 10$ is divisible by 2 but not by 4, there exists a knight's tour that is symmetric under a 90° rotation. Such a tour can be constructed in time $O(n^2)$.*

Proof. The algorithm is similar to that of Theorem 2.5, but instead of breaking the board into four quadrants using the construction of Theorem 2.1 (which, since n is even but not divisible by 4, would not result in quadrants of equal size), it is to be broken evenly into four square quadrants of dimension $n/2 \times n/2$. Noting that $n/2$ is odd, we take a knight's tour on this board with a hole in one corner (constructed using Lemma 2.4), and place a copy in each quadrant, rotating by 90° each time so

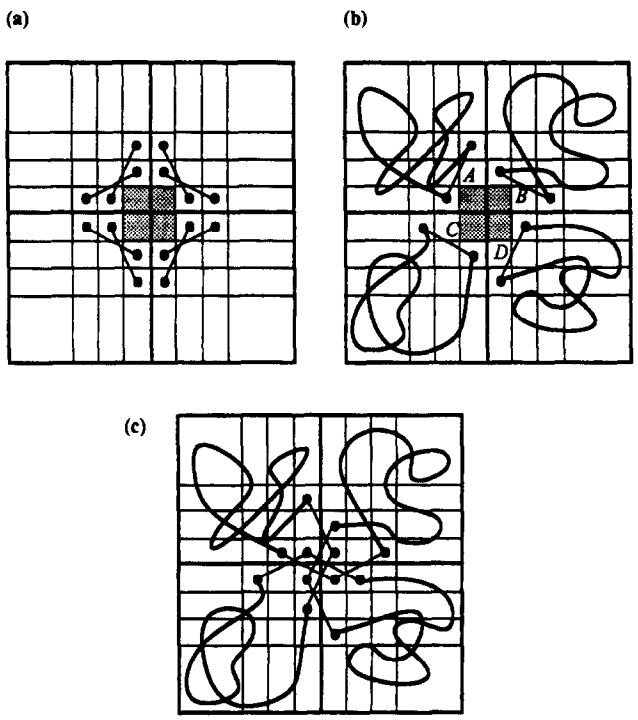

Fig. 9. How to combine four knight's tours into one: (a) the moves at the inside corners (the holes are shaded), (b) the edges A, B, C, D to be deleted, and (c) the replacement edges.

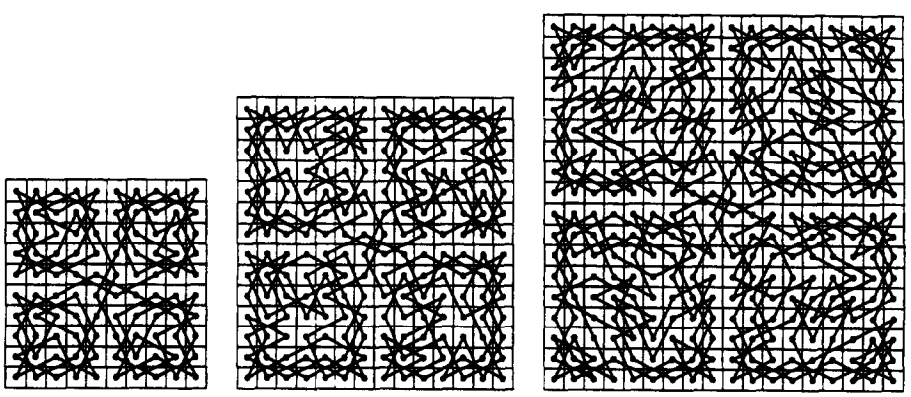

Fig. 10. 10×10, 14×14, and 18×18 knight's tours that are symmetric under a 90 degree rotation, constructed using the technique of Theorem 2.5.

that the holes are at the center of the board (see Fig. 9(a)). Finally, we remove the edges A, B, C, D shown in Fig. 9(b) and replace them with the eight edges shown in Fig. 9(c). Fig. 10 illustrates the technique for $n = 10, 14, 18$. □

Corollary 2.6. *For all $n \times n$ boards where $n \geqslant 8$ is even, there exists a knight's tour that is symmetric under a 180° rotation. Such a tour can be constructed in time* $O(n^2)$.

Proof. The result follows for $n \geqslant 10$ by Theorems 2.2 and 2.5. Gardner [11, Ch. 14] gives an 8×8 tour (attributed to Euler) invariant under a rotation of 180°.

3. Knight's tours in parallel

Our algorithms for constructing structured knight's tours can readily be implemented in parallel. We will consider four different architectures: bounded degree networks, CREW PRAMs, mesh-connected computers, and meshes with CREW row and column buses. We will content ourselves with simply stating the results here, since the proofs use mostly standard techniques. More details are available in [17].

A *bounded degree network* is a parallel machine in which the processors can communicate with at most a small constant number of neighbours (see, for example, [15] or [16]). Our algorithms can be implemented in time $O(n^2/p)$ on a p-processor bounded degree network, for all $p = O(n^2/\log n)$. A CREW PRAM is a parallel machine in which the processors can read and write into a shared memory. Concurrent reads of a single cell of shared memory are permitted but concurrent writes are not (see, for example, [13] or [16]). Our algorithms can be implemented in time $O(1)$ on an n^2-processor CREW PRAM.

A *mesh-connected computer*, or *mesh* for short, is a bounded degree network with a particularly simple interconnection pattern. An $n \times n$ mesh has n^2 processors, and for $0 \leqslant i, j < n$, processor $in + j$ is connected to processors $(i-1)n + j$, $(i+1)n + j$, $in + j - 1$, $in + j + 1$, provided these values are in the range 0 to $n^2 - 1$. Our algorithms can be implemented in time $O(n^2/p^2)$ on a p-processor mesh, for all $p \leqslant n^{2/3}$. A *mesh with* CREW *buses* is a mesh-connected computer with a bus for each row and each column. In addition to the standard mesh instructions, each processor may also write to or read from either its row bus or its column bus. Only one processor may write to each bus in any given step but concurrent reads from a bus are allowed (similar models were studied by Stout [20, 21]). Our algorithms can be implemented in time $O(1)$ on a n^2-processor mesh with CREW buses.

4. Conclusion

We have seen a new algorithm for constructing closed knight's tours on square boards. In addition to being simple to describe, implement, and analyze, the new algorithm can be used to find highly structured knight's tours (specifically, quadrisected tours and tours that are symmetric under rotations), and is amenable to implementation

as efficient parallel algorithms under various architectures. A similar construction can be used to prove exponential lower bounds on the number of closed knight's tours [14].

Acknowledgements

The author is indebted to Edwin Clark, Kevin Ford, and Rolf Wanka for helping with references, to Mike Sluyter for implementing the random walk algorithm and giving permission to use the idea shown in Fig. 9 that led the author to Theorem 2.5.

References

[1] W.W. Ball and H.S.M. Coxeter, Mathematical Recreations and Essays (University of Toronto Press, Toronto, 12th ed., 1974).
[2] C. Cole, The rec. puzzles Frequently Asked Questions List, Version 4, available by anonymous ftp from rtfm.mit.edu in /pub/usenet/news.answers/puzzles/archive (1993).
[3] A. Conrad, T. Hindrichs, H. Morsy and I. Wegener, Wie es dem springer gelang, schachbretter beliebiger groesse und zwischen beliebig vorgegebenen anfangs- und endfeldern vollstaendig abzuschreiten, Spektrum der Wissenschaft, (1992) p. 10–14.
[4] A. Conrad, T. Hindrichs, H. Morsy and I. Wegener, Solution of the knight's Hamiltonian path problem on chessboards, Discrete App. Math. 50 (1994) 125–134.
[5] P. Cull and J. DeCurtins, Knight's tour revisited, Fibonacci Quart., 16 (1978) 276–285.
[6] I.J. Dejter, Equivalent conditions for Euler's problem on Z_4-Hamilton cycles, Ars Combin. 16 (1983) 285–295.
[7] A.P. Domoryad, Mathematical Games and Pastimes (Pergamon Press, Oxford, 1964).
[8] H.E. Dudeney, Amusements in Mathematics (Thomas Nelson & Sons, 1917).
[9] H.E. Dudeney, Amusements in Mathematics (Dover, New York, 1970).
[10] L. Euler, Solution d'une question curieuse qui ne paroit soumise à aucune analyse, Mem. Acad. Sci. Berlin, (1959) 310–337.
[11] M. Gardner, Mathematical Magic Show (Alfred A. Knopf, 1977).
[12] S.P Hurd and D.A. Trautman, The knight's tour on the 15-puzzle, Math. Mag. 66 (1993) 159–166.
[13] R. Karp and V. Ramachandran, Parallel algorithms for shared-memory machines, in: J. van Leeuwen, ed., Algorithms and Complexity, vol. A, Handbook of Theoretical Computer Science (Elsevier and MIT Press, Amsterdam 1990).
[14] O. Kyek, I. Parberry and I. Wegener, Bounds on the number of knight's tours, unpublished manuscript (1994).
[15] F.T. Leighton, Introduction to Parallel Algorithms and Architectures: Arrays · Trees · Hypercubes (Morgan Kaufmann, Los Altos, CA, 1992).
[16] I. Parberry, Parallel Complexity Theory, Research Notes in Theoretical Computer Science (Pitman, London, 1987).
[17] I. Parberry, Algorithms for touring knights, Tech. Report CRPDC-94-7, Center for Research in Parallel and Distributed Computing, Dept. of Computer Sciences, University of North Texas, (1994).
[18] I. Parberry, Scalability of a neural network for the knight's tour problem, Neurocomputing, to appear.
[19] A.J. Schwenk, Which rectangular chessboards have a knight's tour? Math. Mag., 64 (1991) 325–332.
[20] Q.F. Stout, Mesh-connected computers with broadcasting, IEEE Tran. Comput. C–32 (1983) 826–830.
[21] Q.F. Stout, Meshes with multiple buses, in: 27th Annual Symp. on Foundations of Computer Science (IEEE Computer Society Press, New York, 1986) 264–273.
[22] Y. Takefuji, Neural Network Parallel Computing (Kluwer Academic Publishers, Dordrecht, 1992).
[23] Y. Takefuji and K.C. Lee, Neural network computing for knight's tour problems, Neurocomputing 4 (1992) 249–254.