Taming the Knight's Tour: Minimizing Turns and Crossings

Juan Jose Besa
University of California, Irvine, CA, USA
jbesavi@uci.edu

Timothy Johnson
University of California, Irvine, CA, USA
tujohnso@uci.edu

Nil Mamano
University of California, Irvine, CA, USA
nmamano@uci.edu

Martha C. Osegueda
University of California, Irvine, CA, USA
mosegued@uci.edu

Abstract

We introduce two new metrics of "simplicity" for knight's tours: the number of turns and the number of crossings. We give a novel algorithm that produces tours with $9.5n + O(1)$ turns and $13n + O(1)$ crossings on a $n \times n$ board, and we show lower bounds of $(6 - \varepsilon)n$ and $4n - O(1)$ on the respective problems of minimizing these metrics. Hence, our algorithm achieves approximation ratios of $19/12 + o(1)$ and $13/4 + o(1)$. We generalize our techniques to rectangular boards, high-dimensional boards, symmetric tours, odd boards with a missing corner, and tours for $(1, 4)$-leapers. In doing so, we show that these extensions also admit a constant approximation ratio on the minimum number of turns, and on the number of crossings in most cases.

2012 ACM Subject Classification Human-centered computing → Graph drawings; Theory of computation → Computational geometry; Mathematics of computing → Approximation algorithms

Keywords and phrases Graph Drawing, Chess, Hamiltonian Cycle, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.FUN.2021.4

Related Version `https://arxiv.org/abs/1904.02824`

Supplementary Material `https://nmamano.github.io/MinCrossingsKnightsTour/index.html`

Funding The authors were supported by NSF Grant CCF-1616248 and NSF Grant 1815073.

1 Introduction

The game of chess is a fruitful source of mathematical puzzles. The puzzles often blend an appealing aesthetic with interesting and deep combinatorial properties [32]. An old and well-known problem is the knight's tour problem. A *knight's tour* in a generalized $n \times m$ board is a path through all nm cells such that any two consecutive cells are connected by a "knight move" (Fig. 1). For a historic treatment of the problem, see [2]. A knight's tour is *closed* if the last cell in the path is one knight move away from the first one. Otherwise, it is *open*. This paper focuses solely on closed tours, so henceforth we omit the distinction. The knight's tour problem is a special case of the Hamiltonian cycle problem, in which we find a simple cycle that visits all the nodes for a specific class of graphs. These graphs are formed by representing each cell on the board as a node and connecting cells a knight move apart. Existing work focuses on the questions of existence, counting, and construction algorithms.

© Juan Jose Besa, Timothy Johnson, Nil Mamano, and Martha C. Osegueda;
licensed under Creative Commons License CC-BY

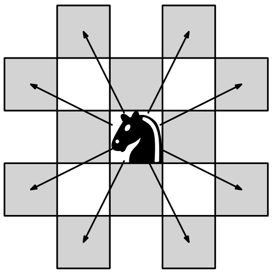


Figure 1 A knight moves two units on one axis and one on the other.

In general, the goal of existing algorithms is to find *any* knight's tour. We propose two new metrics that capture simplicity and structure in a knight's tour. We associate each cell in the board with a point (i, j) in the plane, representing the row and column respectively.

▶ **Definition 1** (Turn). *Given a knight's tour, a* turn *is* a triplet of consecutive cells with non-colinear coordinates.

▶ **Problem 1** (Minimum turn knight's tour). *Finding the knight's tour with the smallest number of turns for a given rectangular $n \times m$ board.*[1]

▶ **Definition 2** (Crossing). *Given a knight's tour, a* crossing *occurs* when the two line segments corresponding to moves in the tour intersect. *E.g. if $\{c_1, c_2\}$ and $\{c_3, c_4\}$ are two distinct pairs of consecutive cells visited along the tour, a crossing happens if the open line segments (c_1, c_2) and (c_3, c_4) intersect.*

▶ **Problem 2** (Minimum crossing knight's tour). *Finding the knight's tour with the smallest number of crossings for a given rectangular $n \times m$ board*[1]

Knight's tours are typically visualized by connecting consecutive cells by a line segment. Turns and crossings make the sequence harder to follow. Minimizing crossings is a central problem in *graph drawing*, the sub-field of graph theory concerned with the intelligible visualization of graphs (e.g., see the survey in [13]). Problem 2 is the natural adaptation for knight's tours. Problem 1 asks for the (self-intersecting) polygon with the smallest number of vertices that represents a valid knight's tour.

1.1 Our contributions

We propose a novel algorithm for finding knight's tours with the following features.

- $9.5n + O(1)$ turns and $13n + O(1)$ crossings on a $n \times n$ board.
- A $19/12 + o(1)$ approximation factoron the minimum number of turns (Problem 1).
- A $13/4 + o(1)$ approximation factor on the minimum number of crossings (Problem 2).
- $O(nm)$ run-time on a $n \times m$ board, i.e., linear to the number of cells, which is optimal.
- The algorithm is fully parallelizable, it can be executed in $O(1)$ time with $O(nm)$ processors in the CREW PRAM model. Since the cell at any given index in the tour sequence (or, conversely, the index of a given cell) can be determined in constant time.

[1] Assuming a knight's tour exists for this board

- It can be generalized to most typical variations of the problem: higher-dimensional cubical boards, rotationally symmetric tours, tours in odd-sized boards that skip a corner cell, and tours for *giraffes*, which move a cell in one dimension and four in another.
- The algorithm can be simulated by hand with ease. This is of particular interest in the context of recreational mathematics and mathematics outreach.

The paper is organized as follows. In Section 1.2, we give an overview of the literature on the knight's tour problem and its variants. We describe the algorithm in Section 2. We prove the approximation ratios in Section 3. We describe how our algorithm extends to related problems in Section 4. We conclude in Section 6.

The tours produced by the algorithm can be generated interactively for different board dimensions at `https://nmamano.github.io/MinCrossingsKnightsTour/index.html`.

1.2 Related Work

Despite being over a thousand years old [32], the knight's tour problem is still an active area of research. We review the key questions considered in the literature.

Existence. In rectangular boards, a tour exists as long as one dimension is even and the board size is large enough; no knight's tour exists for dimensions $1 \times n$, $2 \times n$ or $4 \times n$, for any $n \geq 1$ and, additionally, none exist for dimensions 3×6 or 3×8 [29]. In three dimensions or higher, the situation is similar: a tour exists only if at least one dimension is even and large enough [9, 10, 11]. In the case of open knight's tours, a tour exists in two dimensions if both dimensions are at least 5 [7, 6].

Counting. The number of closed knight's tours in an even-sized $n \times n$ board is at least $\Omega(1.35^{n^2})$ and at most 4^{n^2} [20]. The exact number of knight's tours in the standard 8×8 board is $26,534,728,821,064$ [23]. Furthermore algorithms for enumerating multiple [30] and enumerating all [1] knight's tours have also been studied.

Algorithms. Greedy algorithms have been popular in tour construction. Usually doing so by selecting one step at a time according to a heuristic rule. Historically, greedy algorithms have been popular. The idea is to construct the tour in order, one step at a time, according to some heuristic. Warnsdorff's rule and its refinements [27, 1, 31] work well in practice for small boards, but do not scale to larger boards [25]. The basic idea is to choose the next node with fewest continuations, which is also useful for the Hamiltonian cycle problem [27].

To our knowledge, all efficient algorithms for arbitrary board sizes before this paper are based on a divide-and-conquer approach. The tour is solved for a finite set of small, constant-size boards. Then, the board is covered by these smaller tours like a mosaic. The small tours are connected into a single one by swapping a few knight moves. This can be done in a bottom-up [29, 6, 9, 10, 16] or a top-down recursive [26, 21] fashion. This process is simple and can be done in time linear on the number of cells and, like ours, are also highly parallelizable [6, 26] since they are made of repeating patterns.

Divide-and-conquer is not suitable for finding tours with a small number of turns or crossings. Since each base solution has constant size, a $n \times n$ board is covered by $\Theta(n^2)$ of them, and each one contains turns and crossings. Thus, the divide-and-conquer approach necessarily results in $\Theta(n^2)$ turns and crossings. In contrast, our algorithm has $O(n)$.

Extensions. The above questions have been considered in related settings. Extensions can be classified into three categories, which may overlap:

- **Tours with special properties.** Our work can be seen as searching for tours with special properties. *Magic knight's tours* are also in this category: tours such that the indices of each cell in the tour form a magic square (see [3] for a survey).
  The study of symmetry in knight's tours dates back at least to 1917 [4]. Symmetric tours under 90 degree rotations exist in $n \times n$ tours where $n \geq 6$ and n is of the form $4k+2$ for some k [8]. Parberry extended the divide-and-conquer approach to produce tours symmetric under 90 degree rotations [26]. Jelliss provided results on which rectangular board sizes can have which kinds of symmetry [15].
  Both of our proposed problems are new, but minimizing crossings is related to the *uncrossed knight's tour problem*, which asks to find the longest sequence of knight moves without *any* crossings [34]. This strict constraint results in incomplete tours. This problem has been further studied in two [14, 12] and three [19] dimensions.
- **Board variations.** Besides higher dimensions, knight's tours have been considered in other boards, such as torus boards, where the top and bottom edges are connected, and the left and right edges are also connected. Any rectangular torus board has a closed tour [33]. Another option is to consider boards with odd width and height. Since boards with an odd number of cells do not have tours, it is common to search for tours that skip a specific cell, such as a corner cell [26].
- **Move variations.** An (i, j)-*leaper* is a generalized knight that moves i cells in one dimension and j in the other (the knight is a $(1, 2)$-leaper) [24]. Knuth studied the existence of tours for general (i, j)-leapers in rectangular boards [18]. Tours for *giraffes* ($(4, 1)$-leapers) were provided in [8] using a divide-and-conquer approach. Chia and Ong [5] study which board sizes admit generalized (a, b)-leaper tours. Kamčev [16] showed that any board with sufficiently large and even size admits a $(2, 3)$-, $(2, 5)$-, and a $(a, 1)$-leaper tour for any even a, and generalized this to any higher dimensions. Note that a and b are required to be coprime and not both odd, or no tour can exist [16].

2 The Algorithm

Given that one of the dimensions must be even for a tour to exist, we assume, without loss of generality, that the width w of the board is even, while the height h can be odd. We also assume that $w \geq 16$ and $h \geq 12$. The construction still works for some smaller sizes, but may require tweaks to its most general form described here.

Quartet moves. What makes the knight's tour problem challenging is that knight jumps leave "gaps". Our first crucial observation is that a quartet of four knights arranged in a square 2×2 formation can move "like a king": they can move horizontally, vertically, or diagonally without leaving any gaps (Figure 2).

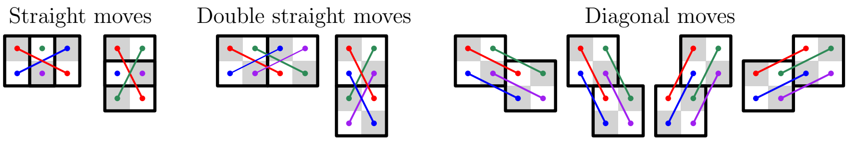


Figure 2 Quartet of knights moving in unison leaving no unvisited squares. In a straight move, the starting and ending position of the quartet overlap because two knights remain in place.

Algorithm 1 Knight's tour algorithm for even width $w \geq 16$ and height $h \geq 12$.

1. Fill the corners of the board as follows:
Bottom-left: first junction in Figure 4.
Top-right: junction of height $5 + ((w/2 + h - 1) \bmod 4)$ in Figure 4 except the first one.
Bottom-right: Sequence $(w/2 + 2) \bmod 4$ in Figure 5.
Top-left: Sequence $(3 - h) \bmod 4$ in Figure 5 rotated 180 degrees.
2. Connect the four corners using formation moves, by moving along diagonals from the bottom-left corner to the top-right corner as in Figure 3. To transition between diagonals:
Vertical edges: use a double straight up move (Figure 2).
Horizontal edges: use Sequence 1 in Figure 5.

By using the "formation moves" depicted in Figure 2, four knights can easily cover the board moving vertically and horizontally while remaining in formation. Of course, the goal is to traverse the entire board in a single cycle, not four paths. We address this issue with special structures placed in the bottom-left and top-right corners of the board, which we call *junctions*, and which tie the paths together to create a single cycle. Note that using only straight formation moves leads to tours with a large number of turns and crossings. Fortunately, two consecutive diagonal moves in the same direction introduces no turns or crossings, so our main idea is to use as many diagonal moves as possible. This led us to the general pattern shown in Figure 3.

The full algorithm is given in Algorithm 1. The formation starts at a junction at the bottom-left corner and ends at a junction at the top-right corner. To get from one to the other, it zigzags along an odd number of parallel diagonals, alternating between downward-right and upward-left directions. The junctions in Figure 4 have a *height*, which influences the number of diagonals traversed by the formation.

At the bottom-left corner, we use a junction with height 5. At the top-right corner, we use a junction with height between 5 and 8. Choosing the height as in Algorithm 1 guarantees that, for any board dimensions, an odd number of diagonals fit between the two junctions. Sequence 1 in Figure 5, which we call the *heel*, is used to transition between diagonals along the horizontal edges of the board. The two non-junction corners require special sequences of quartet moves, as depicted in Figure 5. Sequences $1, 2, 3$, and 0 are used when the last heel ends $0, 2, 4$, and 6 columns away from the vertical edge, respectively. These variations and the top-right junction are *predictable* because they cycle as the board dimensions grow, so in Algorithm 1 we give expressions for them in terms of w and h.

2.1 Correctness

It is clear that the construction visits every cell, and that every node in the underlying graph of knight moves has degree two. However, it remains to be argued that the construction is actually a single closed cycle. For this, we need to consider the choice of junctions.

A junction is a pair of disjoint knight paths whose four endpoints are adjacent as in the quartet formation. Thus, the bottom-left junction connects the knights into two pairs. Denote the four knight positions in the formation by tl, tr, bl, br, where the first letter indicates top/bottom and the second left/right. We consider the three possible *positional matchings* with respect to these positions: horizontal matching $H = (tl, tr), (bl, br)$, vertical matching $V = (tl, bl), (tr, br)$, and cross matching $X = (tl, br), (tr, bl)$. Let $\mathcal{M} = \{H, V, X\}$ denote the set of positional matchings. We are interested in the effect of formation moves on the positional matching. A formation move does not change which knights are matched with which, but a non-diagonal move changes their positions, and thus their matchings. For instance, a horizontal matching becomes a cross matching after a straight move to the right.

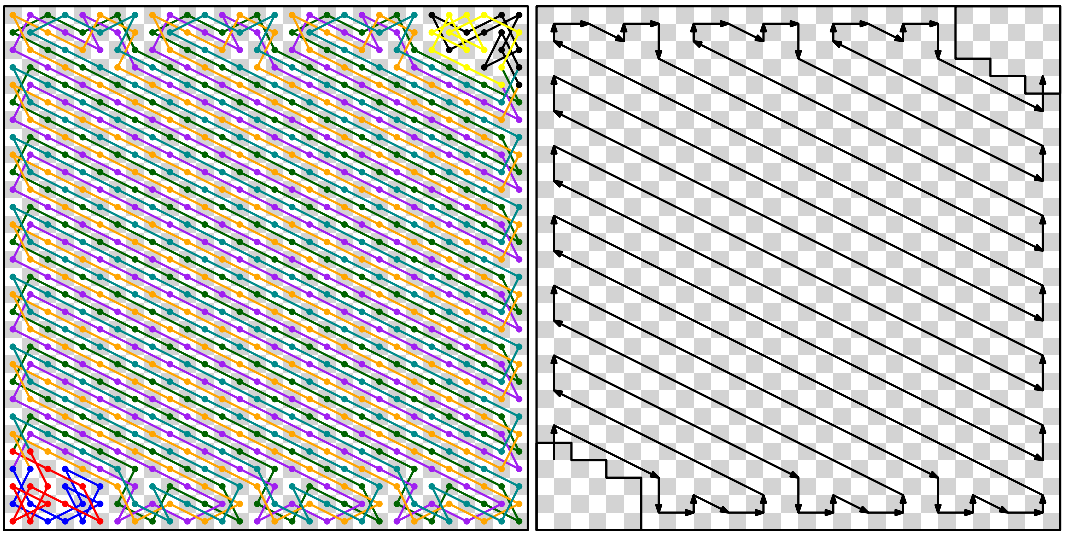


Figure 3 Side by side comparison between the knight's tour **(left)** and the underlying quartet moves **(right)** in a 30×30 board. Arrows illustrate sequences of consecutive formation moves. Starting from the bottom-left square of the board, the single knight's tour follows the colored sections in the order: red, green, yellow, purple, blue, orange, black, cyan, and back to red.

It is easy to see that a straight move upwards or downwards has the same effect on the positional matching. Similarly for left and right straight moves. Thus, we classify the formation moves in Figure 2 (excluding double straight moves, which are a composition of two straight moves) into vertical straight moves ↕, horizontal straight moves ↔, and diagonal moves ⤢. Let $\mathcal{S} = \{$↕, ↔, ⤢$\}$ denote the three types of quartet moves. We see each move type $s \in \mathcal{S}$ as a function $s : \mathcal{M} \to \mathcal{M}$ (see Table 1). Note that the diagonal move ⤢ is just the identity. Given a sequence of moves $S = (s_1, \ldots, s_k)$, where each $s_i \in \mathcal{S}$, let $S(M) = s_1 \circ \cdots \circ s_k(M)$. The move types ↕, ↔, ⤢ seen as functions are, in fact, permutations (Table 1). It follows that *any* sequence of formation moves permutes the positional matchings, according to the composed permutation of each move in the sequence. There are six possible permutations of the three positional matchings, three of which correspond to the "atomic" formation moves ⤢, ↕, and ↔. The other three permutations can be obtained by composing atomic moves, for instance, with the compositions ↕↔, ↔↕, and ↕↔↕ (Table 1). Thus, any

Table 1 Result of applying each type of formation move, as well as three compositions of sequences of moves, to each formation matching.

	⤢	↕	↔	↕↔	↔↕	↕↔↕
V	V	X	V	H	X	H
H	H	H	X	X	V	V
X	X	V	H	V	H	X

Table 2 Cayley table for the group of positional matching permutations.

	⤢	↕	↔	↕↔	↔↕	↕↔↕
⤢	⤢	↕	↔	↕↔	↔↕	↕↔↕
↕	↕	⤢	↕↔	↔	↕↔↕	↔↕
↔	↔	↔↕	⤢	↕↔↕	↕	↕↔
↕↔	↕↔	↕↔↕	↕	↔↕	⤢	↔
↔↕	↔↕	↔	↕↔↕	⤢	↕↔	↕
↕↔↕	↕↔↕	↕↔	↔↕	↕	↔	⤢

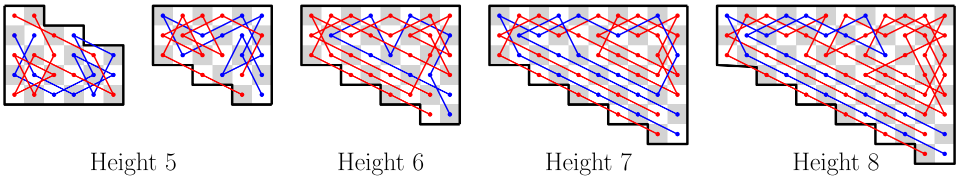


Figure 4 Junctions used in our construction.

sequence of moves permutes the positional matchings in the same way as one of the sequences in the set {⤢, ↕, ↔, ↕↔, ↔↕, ↕↔↕}. This is equivalent to saying that this set, under the composition operation, is isomorphic to the symmetric group of degree three. Table 2 shows the Cayley table of this group.

Let $T_{w,h}$ be the sequence of formation moves that goes from the bottom-left junction to the top-right one in Algorithm 1 in a $w \times h$ board.

▶ **Lemma 3.** *For any even $w \geq 16$ and any $h \geq 12$, $T_{w,h}(H) = H$.*

Proof. We show that the entire sequence of moves $T_{w,h}$ is either neutral or equivalent to single vertical move, depending on the board dimensions. We refer to a sequence of moves as *neutral* when the knights have the same formation at the beginning and end. sAccording to Table 1, this suffices to prove the lemma.

The sequence $T_{w,h}$ consists mostly of diagonal moves, which are neutral. The transition between diagonals along the vertical edges consist of two vertical moves, which are also neutral (↕↕= ⤢). The heel is also neutral, as it consists of the sequence ↕↕↔↔↕↔↔↕ (omitting diagonal moves) which is again equivalent to ⤢. This is easy to see by noting that any two consecutive vertical or horizontal moves cancel out. It is depicted in detail in Figure 6. Thus, $T_{w,h}$ reduces to composing the sequences in the bottom-right and top-left corners. As mentioned, Sequence 1 (the heel) is neutral. It is easy to see that the other sequences (counting each part of Sequence 2 separately) is equivalent to ↕. Thus, we get that $T_{w,h}$ is simply the composition of zero to four vertical moves, depending on the width and height of the board. This further simplifies to zero or one vertical moves. ◀

From these it is easy to see that the algorithm produces a correct tour, however we proceed with a more detail proof.

▶ **Theorem 4** (Correctness). *Algorithm 1 outputs a valid knight's tour in any board with even width $w \geq 16$ and with height $h \geq 12$.*

Proof. Clearly, the formation moves in our construction yield a set of disjoint cycles in the underlying knight-move graph. We prove that after attaching them to the junctions they actually form one cycle. Given a set of disjoint cycles in a graph, *contracting* a node in one of the cycles is the process of removing it and connecting its two neighbors in the cycle. Contracting a node in a cycle of length ≥ 3 does not change the number of cycles. Thus, consider the remaining graph if we contract all the nodes except the four endpoints of the top-right junction.

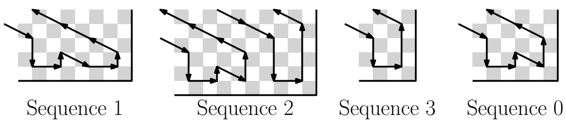


Figure 5 The four possible cases for the bottom-right corner.

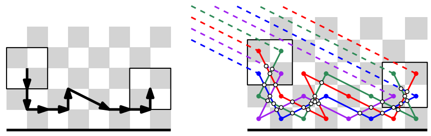


Figure 6 Visualization of the heel: permutation of knights' positions (in color) and its 32 crossings (white disks).

Note that we use a horizontal matching in the bottom-left junction and a vertical matching in the top-right junction. Contracting the non-endpoint nodes inside the top-right junction leaves the two edges corresponding to the vertical matching. By Lemma 3, contracting the nodes outside the top-right junction leaves the edges corresponding to a horizontal matching. Thus, the resulting graph is a single cycle of four nodes. ◀

3 Lower Bounds and Approximation Ratios

In this section, we analyze the approximation ratio that our algorithm achieves for Problem 1 and Problem 2. For simplicity, we restrict the analysis to square boards. Furthermore, it is worth noting that the way in which the input is encoded affects the complexity analysis of the problem, we have included a thorough discussion of these in Appendix A

3.1 Number of Turns

All the turns in our construction happen near the edges. The four corners account for a constant number of turns. The left and right edges have eight turns for each four rows. As it can be seen in Figure 6, the heel has 22 turns, so the top and bottom edges have 22 turns each for each eight columns. Therefore, the number of turns in our construction is bounded by $2\frac{8}{4}n + 2\frac{22}{8}n + O(1) = 9.5n + O(1)$.

Lower bound. First, note that every cell next to an edge *must* contain a turn. This accounts for $4n - 4$ turns. A simple argument, sketched in Appendix B, improves this to a $4.25n - O(1)$ lower bound. Here we focus on the main result, a lower bound of $(6 - \varepsilon)n$ for any $\varepsilon > 0$. We start with some intermediate results.

We associate each cell in the board with a point (i, j) in the plane, where i is the row of the cell and j is the column. An edge cell only has four moves available. We call the directions of these moves D_1, D_2, D_3, and D_4, in clockwise order. For an edge cell c, let $r_i(c)$, with $1 \leq i \leq 4$, denote the ray starting at c and in direction D_i. That is, the ray that passes through the cells reachable from c by moving along D_i.

Let a and b be two cells along the left edge of the board, with a above b. The discussion is symmetric for the other three edges. Given two intersecting rays r and r', one starting from a and one from b, let $S(r, r')$ denote the set of cells in the region of the board *bounded* by r and r': the set of cells below or on r and above or on r'. We define the *crown* of a and b as the following set of cells (see Figure 7):

$$\text{crown}(a, b) = S(r_2(a), r_1(b)) \cup S(r_3(a), r_2(b)) \cup S(r_4(a), r_3(b)).$$

We can associate, with each edge cell c, the two maximal sequences of moves without turns in the tour that have c as an endpoint. We call them the *legs* of c. We say that legs begin at c and end at their other endpoint. We say two legs of different cells *collide* if they end at the same cell. Let $C_{a,b}$ denote the set of edge cells along the right edge between a and b (a and b included). The following is easy to see.

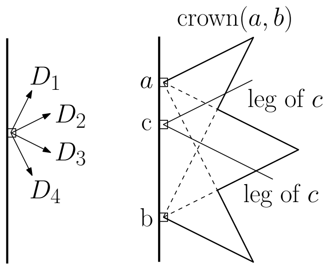


Figure 7 Terminology for the lower bound. Note that c is a clean cell (with respect to the crown of a and b) because both of its legs escape it.

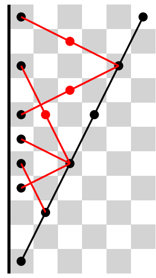


Figure 8 The black leg collides would collide with all the red legs.

▶ Remark 5. Any collision between the legs of edge cells in $C_{a,b}$ happens inside crown(a, b).

We say that a leg of a cell in $C_{a,b}$ *escapes* the crown of a and b if it ends outside the crown. We say an edge cell in $C_{a,b}$ is *clean*, with respect to $C_{a,b}$, if both of its legs escape. We use the following observation, to later obtain the lower bound and show that there is only a constant number of clean cells inside a crown.

▶ Remark 6. Let $m = |C_{a,b}|$ and k be the number of clean cells in $C_{a,b}$. The number of turns inside crown(a, b) is *at least* $m + (m - k)/2$.

Proof. Each edge cell is one turn. Further, each of the $m - k$ non-clean cells have a leg that ends in a turn inside the crown. This turn may be because it collided with the leg of another edge cell in the crown. Thus, there is at least one turn for each two non-clean edge cells. ◀

▶ **Lemma 7.** *Let a, b be two cells along the left edge of the board, with a above b. There are at most 122 clean cells inside crown(a, b).*

Proof. First we show that there are at most 60 clean cells such that one of their legs goes in direction D_1. For the sake of contradiction, assume that there are at least 61. Then, there are two, c and d, such that c is $60r$ rows above d, for some $r \in \mathbb{N}, r \geq 1$. The contradiction follows from the fact that the other leg of c, which goes along D_2, D_3, or D_4, would collide with the leg of b along D_1. This is because, for any $l \geq 1$, the leg of b along D_1 collides with (see Figure 8):

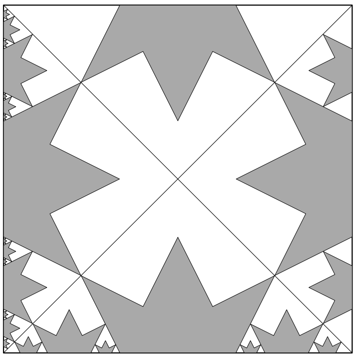
- any leg along D_2 starting from a cell $3l$ rows above b,
- any leg along D_3 starting from a cell $5l$ rows above b, and
- any leg along D_4 starting from a cell $4l$ rows above b.

Since $60r$ is a multiple of $3, 4$, and 5, no matter what direction the other leg of c goes, it collides with the leg of d. As observed, this collision happens inside the crown. Thus, c and d are not clean. By a symmetric argument, there are at most 60 clean cells such that one of their legs goes in direction D_4.

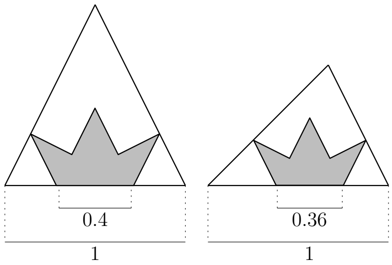
Finally, note that there can only be two clean cells with legs in D_2 *and* D_3. This is because, by a similar argument, there cannot be two such cells an even distance of each other; the leg along D_3 of the top one would collide against the leg along D_2 of the bottom one. ◀

▶ **Corollary 8.** *Suppose that the crown of a and b has $m \geq 122$ edge cells. Then, there are at least $(m - 122)/2$ turns inside the crown at non-edge cells.*

Now, consider the iterative process depicted in Figure 9, defined over the unit square. The square is divided in four sectors along its main diagonals. Whereas earlier we used the term "crown" to denote a set of cells, here we use it to denote the polygon with the *shape* of



■ **Figure 9** Each sector of the square shows the process after a different number of iterations: $1, 2, 3$, and 4 iterations on the top, right, bottom, and left sectors, respectively.



■ **Figure 10** Lower bounds on two ratios.
Left: the ratio between the gap between consecutive crowns and the base of the maximum-size crown that fits in the gap is > 0.4.
Right: the ratio between the gap between a crown and a main diagonal and the base of the maximum-size crown that fits in the gap is > 0.36.

a crown. On the first step, a maximum-size crown is placed on each sector. At step $i > 1$, we place 2^{i-1} more crowns in each sector. They are maximum-size crowns, subject to being disjoint from previous crowns, in each gap between previous crowns and between the crowns closest to the corners and the main diagonals.

▶ **Lemma 9.** *For any $1 > \varepsilon > 0$, there exists an $i \in \mathbb{N}$ such that at least $(1 - \varepsilon)$ of the boundary of the unit square is inside a crown after i iterations of the process.*

Proof. At each iteration, a constant fraction larger than 0.36 of the length on each side that is not in a crown is added to a new crown (Figure 10). This gives rise to a series A_i for the fraction of the side inside crowns after i iterations: $A_1 = 1/3$, $A_{i+1} > A_i + 0.36(1 - A_i)$ for $i > 1$; this series converges to 1. ◀

▶ **Theorem 10** (Lower bound)**.** *For any constant $\varepsilon > 0$, there is a sufficiently large n such that any knight's tour on a $n \times n$ board requires $(6 - \varepsilon)n$ turns.*

Proof. We show a seemingly weaker form of the claim: that there is a sufficiently large n such that any knight's tour on a $n \times n$ board requires $(6 - 2\varepsilon)n - C_\varepsilon$ turns, where C_ε is a constant that depends on ε but not on n. This weaker form is in fact equivalent because, for sufficiently large n, $C_\varepsilon < \varepsilon n$, and hence $(6 - 2\varepsilon)n - C_\varepsilon > (6 - 3\varepsilon)n$. Thus, the claim is equivalent up to a multiplicative factor in ε, but note that it is a claim about arbitrarily small ε, so it is not affected by a multiplicative factor loss.

Let i be the smallest number of iterations of the iterative process in Figure 9 such that at least $(1 - \varepsilon)$ of the boundary of the unit square is inside crown shapes. The number i exists by Lemma 9. Fix S to be the corresponding set of crown shapes, and $r = |S|$. Note that $r = 4(2^i - 1)$ is a constant that depends only on ε. Now, consider a square $n \times n$ board with the crown shapes in S overlaid in top of them. Let the board size n be such that the smallest crown in S contains more than 122 edge cells. Then, by Corollary 8, adding up the turns at non-edge cells over all the crowns in S, we get at least $4n(1 - \varepsilon)/2 - 61r$ turns. Adding the $4n - 4$ turns at edge cells, we get that the total number of turns is at least $(6 - 2\varepsilon)n - 61r - 4$. To complete the proof, consider $C_\varepsilon = 61r + 4$. ◀

▶ **Corollary 11.** *Algorithm 1 achieves a $19/12 + o(1)$ approximation on the minimum number of turns.*

Proof. In a $n \times n$ board, let $ALG(n)$ denote the number of turns in the tour produced by Algorithm 1 and $OPT(n)$ denote the minimum number of turns. Let $\varepsilon > 0$ be an arbitrarily small constant. We show an n_0 exists such that $\forall n \geq n_0$, $ALG(n)/OPT(n) < 19/12 + \varepsilon$. As mentioned, for any even $n \geq 16$, $ALG(n) < 9.5n + c$ for some small constant c. In addition, by Theorem 10, for sufficiently large n, $OPT(n) > (6 - \varepsilon)n$. Thus, $ALG(n)/OPT(n) < (9.5n + c)/(n(6 - \varepsilon))$ Furthermore, for large enough n, $c/((6 - \varepsilon)n) < \varepsilon/2$, so

$$\frac{ALG(n)}{OPT(n)} < \frac{9.5}{6-\varepsilon} + \frac{\varepsilon}{2} = \frac{19}{12 - 2\varepsilon} + \frac{\varepsilon}{2} < \frac{19 + 6\varepsilon}{12} + \frac{\varepsilon}{2} = \frac{19}{12} + \varepsilon.$$ ◀

3.2 Number of Crossings

Similarly to the case of turns, all the crossings in our construction happen near the edges. The four corners account for a constant number of crossings. The left and right edges have 10 crossings for each four rows. The top and bottom edges have 32 crossings for each eight columns (Figure 6). Therefore, the number of turns in our construction is bounded by $2\frac{10}{4}n + 2\frac{32}{8}n + O(1) = 13n + O(1)$.

▶ **Lemma 12.** *Any knight's tour on an $n \times n$ board has at least $4n - O(1)$ crossings.*

Proof. Let T be an arbitrary knight's tour on an $n \times n$ board. We show that T has $n - O(1)$ crossings involving knight moves incident to the cells along the left edge of the board. An analogous argument holds for the three other edges of the board, which combined yield the desired bound.

We partition the edge cells along the left-most column into sets of three consecutive cells, which we call *triplets* (if n is not multiple of three, we ignore any remaining cells, as they only account for a constant number of crossings). Two triplets are *adjacent* if they contain adjacent cells. Each triplet has six associated knight moves in the tour T, two for each of its cells. We call the choice of moves the *configuration* of the triplet. Since there are $\binom{4}{2} = 6$ choices of moves for each cell, there are $6^3 = 216$ possible configurations of each triplet.

Consider a weighted directed graph G with a node for each of the 216 possible triplet configuration and an edge from every node to every node, including a loop from each node to itself. The graph has weights on both vertices and edges. Given a node v, let $C(v)$ denote its associated configuration. The weight of v is the number of crossings between moves in $C(v)$. The weight of each edge $v \to u$ is the number of crossings between moves in $C(v)$ and moves in $C(u)$ when $C(v)$ is adjacent and above $C(u)$.

Each path in G represents a choice of move configurations for a sequence of consecutive triplets. Note that if two knight moves in T with endpoints in edge cells cross, the edges cells containing the endpoints are either in the same triplet or in adjacent triplets. Thus, the sum of the weights of the vertices and edges in the path equals the total number of crossings

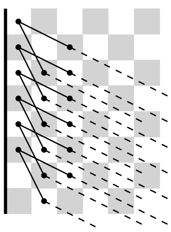


Figure 11 The configuration pattern along the board's edge with the minimum number of crossings. Triplet configurations (solid) and their continuation (dashed) without extra crossings.

among all of these moves. Since G is finite, any sufficiently long path eventually repeats a vertex. Given a cycle c, let $w(c)$ be the sum of the weights of nodes and edges in c, divided by the length of c. Let c^* be the cycle in G minimizing w. Then, $w(c^*)$ is a lower bound on the number of crossings per triplet along to edge.

By examining G, we can see that $w(c^*) = 3$. Figure 11 shows an optimum cycle, which in fact uses only one triplet configuration. The cycle minimizing w can be found using Karp's algorithm for finding the minimum mean weight cycle in directed graphs [17], which runs in $O(|V| \cdot |E|)$ time in a graph with vertex set V and edge set E. However, this requires modifying the graph G, as Karp's algorithm is not suitable for graphs that also have node weights. We transform G into a directed graph G' with weights on only the edges and which preserves the optimal solution, as follows. We double each node v in G into two nodes v_{in}, v_{out} in G'. We also add an edge $v_{in} \to v_{out}$ in G' with weight equal to the weight of v in G. For each edge $v \to u$ in G, we add an edge $v_{out} \to u_{in}$ in G'. ◀

Since we only counted crossings between moves incident to the first column, a question arises of whether the lower bound can be improved by considering configurations spanning more columns (e.g., the two or three leftmost columns). The answer is negative for any constant number of columns. Figure 11 shows that the edges can be extended to paths that cover any fixed number of rows away from the edge without increasing the number of crossings.

▶ **Corollary 13.** *Algorithm 1 achieves a $13/4 + o(1)$ approximation on the minimum number of crossings.*

4 Extensions

The idea of using formation moves to cover the board and junctions to close the tour is quite robust to variations of the problem. We show how this can be done in some of the most popular generalizations of the problem.

A variant that we do not consider is torus boards (where opposite edges are connected). The problem of finding tours with a small number of turns seems easier on a torus board, because one is not forced to make a turn when reaching an edge. Nonetheless, in a square $n \times n$ torus board, $\Omega(n)$ turns are still required, because making n consecutive moves in the same direction brings the knight back to the starting position, so at least one turn is required for each n visited cells. The tours for torus boards in [33] match this lower bound up to constant factors, at least for some board dimensions (see the last section in [33]). Crossings are not straightforward to define in torus boards.

4.1 High-dimensional boards

We extend our technique to three and higher dimensions. In d dimensions, a knight moves by 1 and 2 cells along any two dimensions, and leaves the remaining coordinates unchanged. A typical technique to extend a knight's tour algorithm to three dimensions is the "layer-by-layer" approach [32, 9, 10]: a 2D tour is reused on each level of the 3D board, adding the minimal required modifications to connect them into a single tour. We also follow this approach. (Watkins and Yulan [28] consider a generalizations of knight moves where the third coordinate also has a positive offset, but this is not as common.) For illustration purposes, we start with the 3D case, and later extend it to the general case. We require one dimenson to be even and ≥ 16 and another dimension to be ≥ 12, which we assume w.l.o.g. to be the first two. The rest can be any size. Note that at least one dimension must be even, or no tour exists [11].

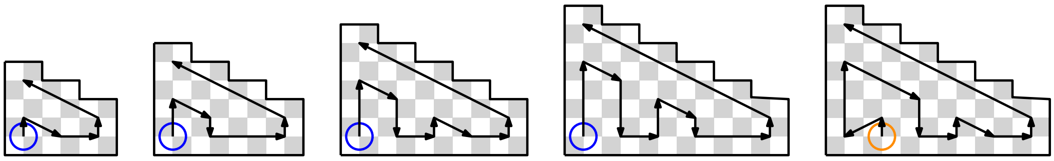


Figure 12 Corners where the knights stay in formation and end at specific positions.

The construction works as follows. The 2D construction is reused at each level. However, there are only two actual junctions, one on the first layer, of height 5, and one on the last layer, which may have any of the four heights in Figure 4. Every other junction is replaced by a sequence of formation moves. At every layer except the last, the formation ends adjacent to the corner using one of the sequences of moves in Figure 12 (note that we show sequences for 4 different heights, thus guaranteeing that one shape fits for any board dimensions). The layers are connected with a formation move one layer up and two cells to the side, as in Figure 14. At every layer except the first, the formation starts with the rightmost sequence of moves in Figure 12. A full example is illustrated in Figure 13.

Note that, since the sequence of moves between junctions is more involved than in two dimensions, Lemma 3 may not hold. There is, however, an easy fix: if the entire sequence is not a single cycle, replace the first junction with one that has a vertical matching (second junction in Figure 4, rotate 180 degrees). This then makes a cycle.

If the number of dimensions is higher than three, simply observe that the same move used between levels can also be used to jump to the next dimension; instead of changing by 1 the third coordinate, change the fourth. After the first such move, the formation will be at the "top" of the second 3D board, which can be traversed downwards. This can be repeated any number of times, and generalizes to any number of dimensions.

Note that in a n^d board, $\Omega(n^{d-1})$ turns are needed, because there are n^d cells and a turn must be made after at most $n/2$ moves. Note that our construction has $O(n^{d-1})$ turns, as it consists of n^{d-2} iterations of the 2D tour. Thus, it achieves a constant approximation ratio on the minimum number of turns. We do not know of any lower bound on the number of crossings in higher dimensions.

4.2 Odd boards

We show how to construct a tour for a 2D board with odd dimensions which visits every cell except a corner cell. This is used in the next section to construct a tour that is symmetric under 90° rotations.

Let the board dimensions be $w \times h$, where $w > 16$ and $h > 12$ are both odd. First, we use Algorithm 1 to construct a $(w-1) \times h$ tour which is missing the leftmost column. Then, we extend our tour to cover this column, except the bottom cell, with the variations of our construction depicted in 15. In particular, for the top-left corner, recall that we use sequence $(3-h) \bmod 4$ in Figure 5. Here, the height h is odd, so we only need adaptations for Sequences 2 and 0.

4.3 90 Degree Symmetry

In this section, we show how to construct a symmetric tour under 90 degree rotations. We say a tour is symmetric under a given geometric operation if the tour looks the same when the operation is applied to the board.

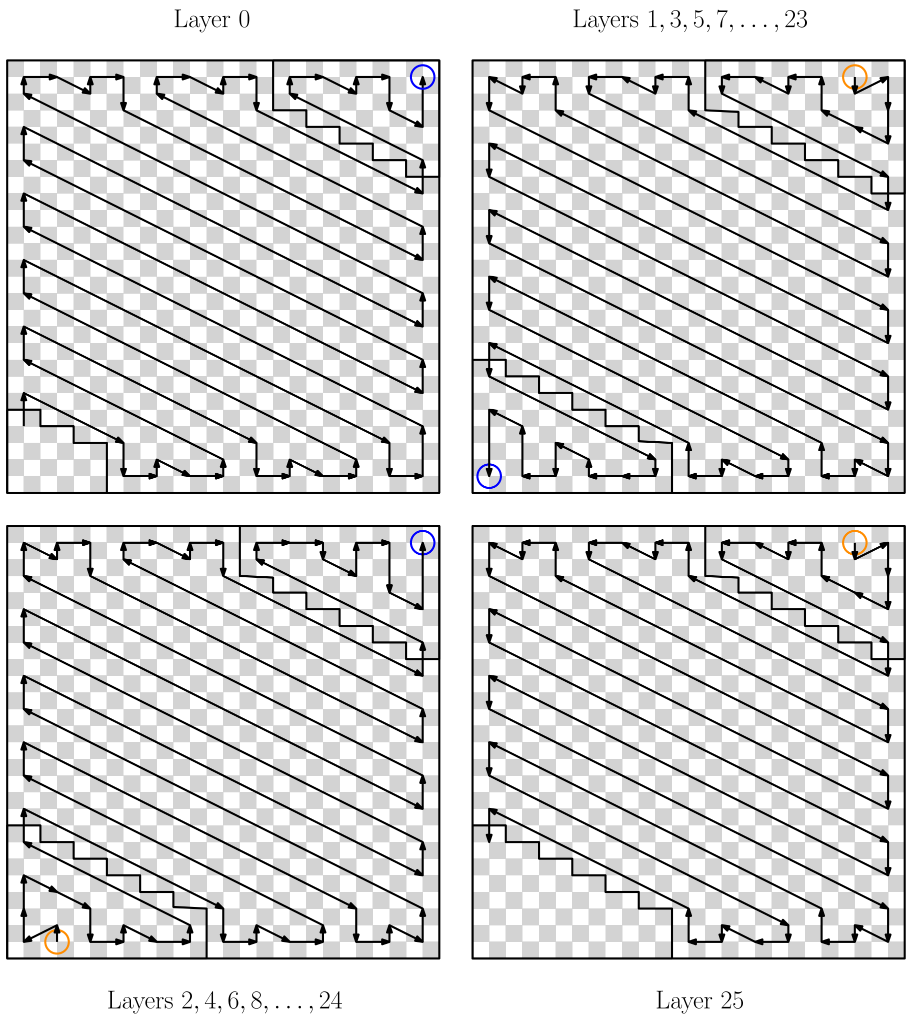


Figure 13 Quartet moves for a $3D$ tour in a $26 \times 26 \times 26$ board. The quartet can move from the blue circle at each layer to the orange circle in the next layer by a quartet move.

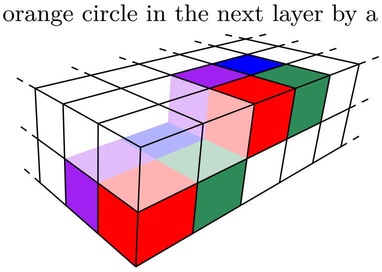


Figure 14 Formation move across layers. Each color shows the starting and ending position of one of the knights.

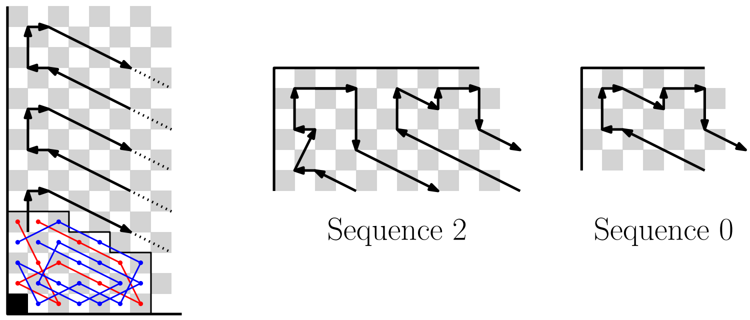


Figure 15 Adaptations required to add a row to the left of the normal construction, with a missing cell in the junction.

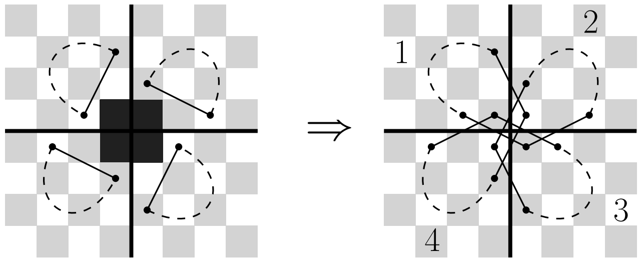


Figure 16 This transformation appears in [26]. **Left:** four tours missing a corner square and containing a certain edge. The dashed lines represent the rest of the tour in each quadrant, which cover every square except the dark square. **Right:** single tour that is symmetric under 90° rotations. The numbers on the right side indicate the order in which each part of the tour is visited, showing that the tour is indeed a single cycle.

As a side note, our construction is already nearly symmetric under 180° rotations. For board dimensions such as 30×30 where opposite corners have the same shape, the only asymmetry is in the internal wiring of the junctions. However, the construction cannot easily be made fully symmetric. It follows from the argument in the proof of Lemma 3 that if the two non-junction corners are equal, the entire sequence of formation moves from one junction to the other is neutral. Thus, using the same junction in both corners, as required to have symmetry, would result in two disjoint cycles.

Symmetric tours under 90° rotations exist only for square boards where the size $n = 4k+2$ is a multiple of two but not a multiple of four [8]. In [26], Parberry gives a construction for knight's tours missing a corner cell and then shows how to combine four such tours into a single tour symmetric under 90° rotations. We follow the same approach to obtain a symmetric tour with a number of turns and crossings linear on n, and thus constant approximation ratios.

In our construction from Section 4.2, cell $(0,0)$ is missing, and edge $e = \{(0,1),(2,0)\}$ is present. This suffices to construct a symmetric tour. Divide the $2n \times 2n$ board into four equal quadrants, each of which is now a square board with odd dimensions. Use the construction for odd bords to fill each quadrant, rotated so that the missing cell is in the center. Finally, connect all four tours as in Figure 16.

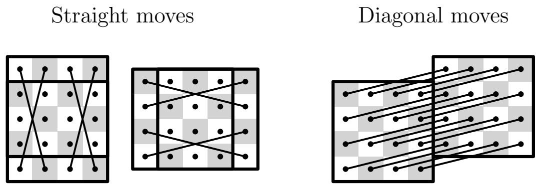


Figure 17 Formation of 16 giraffes moving together without leaving any unvisited squares.

5 Giraffe's tour

A giraffe is a leaper characterized by the move $(1, 4)$ instead of $(1, 2)$. Giraffe's tours are known to exist on square boards of even size $n \geq 104$ [16] and on square boards of size $2n$ when n is odd and bigger than 4 [8]. Our result extends this to some rectangular sizes.

We adapt our techniques for finding giraffe's tours with $O(w + h)$ turns and crossings, where w and h are the width and height of the board. We use a formation of 4×4 giraffes. Figure 17 shows the formation moves, Figure 18 shows the analogous of a heel to be used to transition between diagonals, and Figure 19 shows the two junctions. Figure 20 shows how these elements are combined to cover the board.

We restrict our construction to rectangular boards where $w = 32k + 20$, for some $k \geq 1$, and $h = 8l + 14$, for some $l \geq 1$ (extending the results to more boards would require additional heel variations).

We start at the bottom-left junction as in the knight's tour. We transition between diagonals along the bottom edge with a giraffe heel, and along the top edge with a flipped giraffe heel. We transition between diagonals along the left and right edges with four consecutive upward moves. The junction has width 20 and each heel has width 32, so there are k heels along the bottom and top edges. The junction has height 11 and the tip of the heel has height 3, so there are l sequences of four upward moves along each side (see Figure 20).

It is easy to see that the construction visits every cell. As in the case of knight's tours, for the result to be a valid giraffe's tour it should be a cycle instead of a set of disjoint cycles. Note that the matchings in the two junctions form a cycle. Thus, if the formation reaches the top-right junction in the same matching as they left the bottom-left junction, the entire construction is a single cycle (by an argument analogous to Theorem 4).

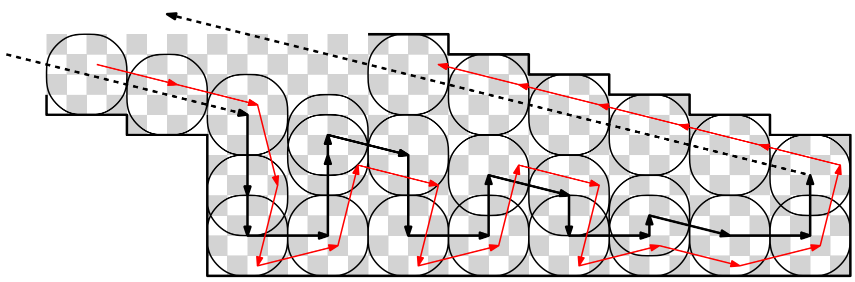


Figure 18 A giraffe heel. The formation moves are shown with black arrows (grouping up to four sequential straight moves together) The intermediate positions of the formation are marked by rounded squares, showing that every cell is covered. Note that the tip of the heel fits tightly under the next heel. The red line shows the path of one specific giraffe.

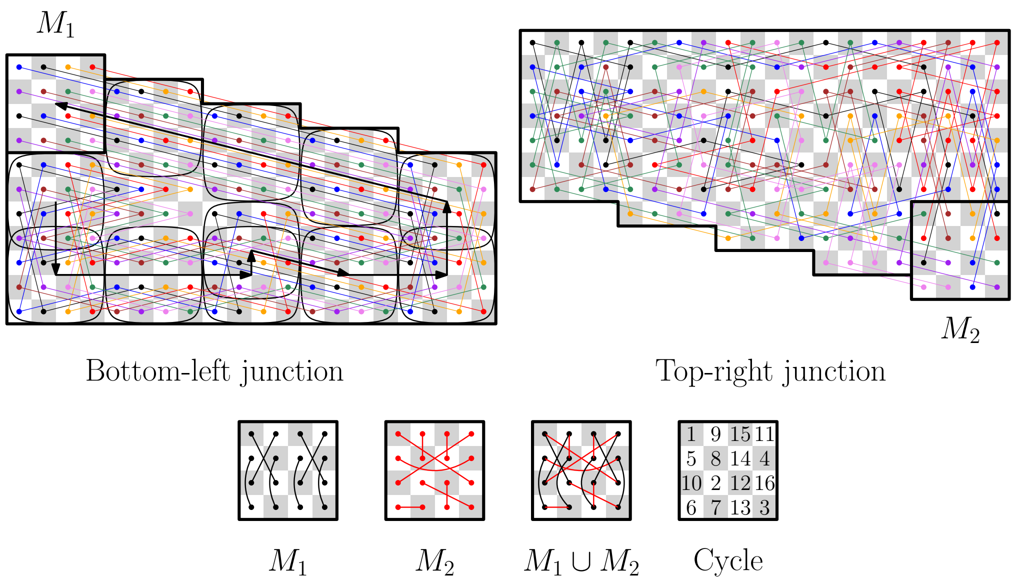


Figure 19 Two giraffe junctions, their corresponding matchings, and the union of their matchings. The bottom-left junction consists mostly of formation moves, whereas the top-right one was computed via brute-force search. The cycle through the edges of the union is shown with the index of each node.

Let H, F, and U denote the sequences of formation moves in the heel, in the flipped heel, and the sequence of four upward moves, respectively. Let $T_{w,h}$ denote the entire sequence of moves from one junction to the other, where $w = 32k + 20$, for some $k \geq 1$, and $h = 8l + 14$, for some $l \geq 1$. Note that $T_{w,h}$ is a concatenation, in some order, of H k times, F k times, and U $2l$ times (we can safely ignore diagonal moves, which do not change the coordinates of the giraffes within the formation). Let M be the matching of the bottom-left junction. We want to argue that, after all the moves in $T_{w,h}$, the giraffes are still in matching M, that is, $T_{w,h}(M) = M$ using the notation from Section 2.

We show that not only the giraffes arrive to the other junction in the same matching but, in fact, they arrive in the same coordinates in the formation as they started. First, note that U has the effect of flipping column 1 with 2 and column 3 with 4 in the formation. Perhaps surprisingly, H and F have the same effect. This is tedious but can be checked for each

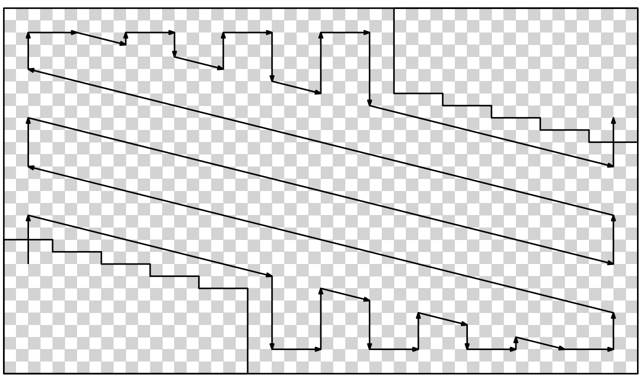


Figure 20 The formation moves of a giraffe's tour on a 52×30 board.

giraffe (Figure 18 shows one in red). Therefore, $T_{w,h}$ is equivalent to U $2(l+k)$ times in a row. Note that after eight consecutive upward moves, or U twice, each giraffe ends where it started. Thus, this is true of the entire tour.

6 Conclusions

We have introduced two new metrics of "simplicity" for knight's tours: the number of turns and the number of crossings. We provided an algorithm which is within a constant factor of the minimum for both metrics. In doing so, we found that, in a $n \times n$ board, the minimum number of turns and crossings is $O(n)$. Prior techniques such as divide-and-conquer necessarily result in $\Theta(n^2)$ turns and crossings, so at the outset of this work it was unclear whether $o(n^2)$ could be achieved at all.

The ideas of the algorithm, while simple, seem to be new in the literature, which is interesting considering the history of the problem. Perhaps it was our *a priori* optimization goal that led us in a new direction. The algorithm exhibits a number of positive traits. It is simple, efficient to compute, parallelizable, and amenable to generalizations (see Section 4). We conclude with some open questions:

- Our tours have $9.5n + O(1)$ turns and $13n + O(1)$ crossings, and we showed respective lower bounds of $(6 - \varepsilon)n$ and $4n - O(1)$. The main open question is closing or reducing these gaps, as there may still be room for improvement in both directions. We conjecture that the minimum number of turns is at least $8n$.
- Are there other properties of knight's tours, besides turns and crossings, that might be interesting to optimize?
- Our method relies heavily on the topology of the knight-move graph. Thus, it is not applicable to general Hamiltonian cycle problems. Are there other graph classes with a similar enough structure that the ideas of formations and junctions can be useful?

References

1 Karla Alwan and Kelly Waters. Finding re-entrant knight's tours on n-by-m boards. In *Proceedings of the 30th Annual Southeast Regional Conference*, ACM-SE 30, pages 377–382, New York, NY, USA, 1992. ACM. doi:10.1145/503720.503806.

2 W.W. Rouse Ball and H.S.M. Coxeter. *Mathmatical Recreations & Essays: 12th Edition*. University of Toronto Press, 1974. URL: http://www.jstor.org/stable/10.3138/j.ctt15jjcrn.

3 John D. Beasley. Magic knight's tours. *The College Mathematics Journal*, 43(1):72–75, 2012. URL: http://www.jstor.org/stable/10.4169/college.math.j.43.1.072.

4 Ernest Bergholt. Three memoirs on knight's tours. *The Games and Puzzles Journal*, 2(18):327–341, 2001.

5 G.L. Chia and Siew-Hui Ong. Generalized knight's tours on rectangular chessboards. *Discrete Applied Mathematics*, 150(1):80–98, 2005. doi:10.1016/j.dam.2004.11.008.

6 Axel Conrad, Tanja Hindrichs, Hussein Morsy, and Ingo Wegener. Solution of the knight's hamiltonian path problem on chessboards. *Discrete Applied Mathematics*, 50(2):125–134, 1994. doi:10.1016/0166-218X(92)00170-Q.

7 Paul Cull and Jeffery De Curtins. Knight's tour revisited. *Fibonacci Quarterly*, 16:276–285, June 1978.

8 Italo J. Dejter. Equivalent conditions for euler's problem on z_4-hamilton cycles. *Ars Combinatoria*, 16–B:285–295, 1983.

9 Joe DeMaio. Which chessboards have a closed knight's tour within the cube? *the electronic journal of combinatorics*, 14(1):32, 2007.

10 Joe DeMaio and Mathew Bindia. Which chessboards have a closed knight's tour within the rectangular prism? *the electronic journal of combinatorics*, 18(1):14, 2011.

11 Joshua Erde, Bruno Golénia, and Sylvain Golénia. The closed knight tour problem in higher dimensions. *the electronic journal of combinatorics*, 19(4):9, 2012.
12 Alexander Fischer. New records in nonintersecting knight paths. *The Games and Puzzles Journal*, 2006.
13 Ivan Herman, Guy Melançon, and M. Scott Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on visualization and computer graphics*, 6(1):24–43, 2000.
14 George P. Jelliss. Non-intersecting paths by leapers. *The Games and Puzzles Journal*, 2(17):305–310, 1999.
15 George P. Jelliss. Symmetry in knight's tours. *The Games and Puzzles Journal*, 2(16):282–287, 1999.
16 Nina Kamčev. Generalised knight's tours. *the electronic journal of combinatorics*, 21(1):32, 2011.
17 Richard M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete mathematics*, 23(3):309–311, 1978.
18 Donald E. Knuth. Leaper graphs. *The Mathematical Gazette*, 78(483):274–297, 1994. URL: http://www.jstor.org/stable/3620202.
19 Awani Kumar. Non-crossing Knight's Tour in 3-Dimension. *ArXiv e-prints*, March 2008. arXiv:0803.4259.
20 Olaf Kyek, Ian Parberry, and Ingo Wegener. Bounds on the number of knight's tours. *Discrete Applied Mathematics*, 74(2):171–181, 1997. doi:10.1016/S0166-218X(96)00031-5.
21 Shun-Shii Lin and Chung-Liang Wei. Optimal algorithms for constructing knight's tours on arbitrary $n \times m$ chessboards. *Discrete Applied Mathematics*, 146(3):219–232, 2005. doi:10.1016/j.dam.2004.11.002.
22 Stephen R. Mahaney. Sparse complete sets for np: Solution of a conjecture of berman and hartmanis. *Journal of Computer and System Sciences*, 25(2):130–143, 1982. doi:10.1016/0022-0000(82)90002-2.
23 Brendan D. McKay. Knight's tours of an 8× 8 chessboard. Technical report, Australian National University, Department of Computer Science, February 1997.
24 Crispin Nash-Williams. Abelian groups, graphs and generalized knights. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 55(3), pages 232–238. Cambridge University Press, 1959.
25 Ian Parberry. Scalability of a neural network for the knight's tour problem. *Neurocomputing*, 12(1):19–33, 1996. doi:10.1016/0925-2312(95)00027-5.
26 Ian Parberry. An efficient algorithm for the knight's tour problem. *Discrete Applied Mathematics*, 73(3):251–260, 1997.
27 Ira Pohl. A method for finding hamilton paths and knight's tours. *Commun. ACM*, 10(7):446–449, July 1967. doi:10.1145/363427.363463.
28 Yulan Qing and John J. Watkins. Knight's tours for cubes and boxes. *Congressus Numerantium*, January 2006.
29 Allen J. Schwenk. Which rectangular chessboards have a knight's tour? *Mathematics Magazine*, 64(5):325–332, 1991.
30 Jefferey A. Shufelt and Hans J. Berliner. Generating knight's tours without backtracking from errors. Technical report, Carnegie-Mellon University, School of Computer Science, 1993.
31 Douglas Squirrel and Paul Cull. A warnsdorff-rule algorithm for knight's tours on square chessboards, 1996.
32 John J. Watkins. *Across the Board: The Mathematics of Chessboard Problems*. Princeton Puzzlers. Princeton University Press; Reissue edition, 2012.
33 John J. Watkins and Rebecca L. Hoenigman. Knight's tours on a torus. *Mathematics Magazine*, 70(3):175–184, 1997. doi:10.1080/0025570X.1997.11996528.
34 L. D. Yarbrough. Uncrossed knight's tours. *Journal of Recreational Mathematics*, 1(3):140–142, 1969.

A Computational Complexity

Consider the following decision versions of the problems: is there a knight's tour on an $n \times n$ board with at most k turns (resp. crossings)? We do not know if these problems are in P. Furthermore, it may depend on how the input is encoded. Technically, the input consists of two numbers, n and k, which can be encoded in $O(\log n + \log k)$ bits. However, it is more natural to do the analysis as a function of the board size (or, equivalently, of the underlying graph on which we are solving the Hamiltonian Cycle problem), that is, $\Theta(n^2)$. It is plausible that the optimal number of turns (resp. crossings) is a simple, arithmetic function of n. This would be the case if the optimal tour follows a predictable pattern like our construction (counting the number of turns or crossings achieved by our algorithm does not require finding the tour). In this case, the problems are in P, regardless of the input's encoding.

If the input is represented using $\Theta(n^2)$ space, the problems are clearly in NP, as a tour with k turns/crossings acts as a certificate of polynomial length. However, unless P = NP, the problems are not NP-hard.

Consider the language $\{1^n 0 1^k \mid$ a tour exists with at most k turns in an $n \times n$ board$\}$, and analogously for crossings. These languages are sparse, meaning that, for any given word length, there is a polynomial number of words of that length in the language. Mahaney's theorem states that if a sparse language is NP-complete, then P = NP [22]. This suggests that the problems are in P, though technically they could also be NP-intermediate.

If the input is represented using $O(\log n + \log k)$ bits, then the problems are in NEXP because the "unary" versions above are in NP. Note that, in this setting, simply listing a tour would require time exponential on the input size.

B Easy Lower Bound for Turns

A sketch for a simple proof providing a loose lower bound on the number of turns.

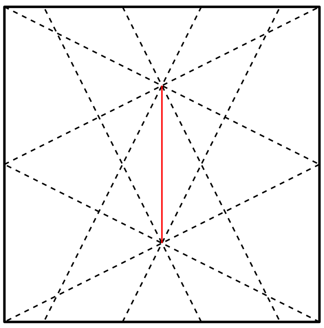


Figure 21 A lower bound of $4.25n - O(1)$ on the number of turns required by any knight's tour on a $n \times n$ board can be seen as follows. Consider the cells in one of the two central columns (does not matter which one), and in a row in the range $(n/4, 3n/4)$. They are shown in red. These cells have the property that every maximal sequence of knight moves without turns through them reaches opposite facing edges. The maximal sequences of knight moves through the first and last red cells are shown in dashed lines. Because n must be even, one of the two endpoints of each maximal sequence through a red cell is not an edge cell. It follows that each red cell is part of a sequence of knight's moves that ends in a turn at a non-edge cell. Thus, there is at least one turn at a non-edge cell for each pair of red cells. Since there are $0.5n$ red cells, we get the mentioned lower bound.