Structured Knight's Tours

Robert Bosch and Zejian Huang

Oberlin College, Oberlin, Ohio, USA; rbosch@oberlin.edu

Abstract

We discuss discrete optimization methods for constructing what we refer to as *structured knight's tours*: knight's tours that are nearly symmetric, contain repeated geometric motifs, and/or express hidden (or in some cases, not so hidden) mathematical structures.

Introduction

The left-hand panel of Figure 1 shows an 8×8 chessboard. The center panel displays the 8×8 knight's-move chessboard graph, which has 64 vertices (dots)—one for each square of the board—and 168 edges (line segments)—one for each possible undirected knight's move. The right-hand panel contains an open knight's tour of the board. We can imagine a chess knight starting in the bottom left square and then making one knight's move after another (traversing one edge of its tour after another) until at the end of its journey it has visited every square of the board once and only once and has arrived at its final square, the square that's diagonally adjacent to the bottom right square.

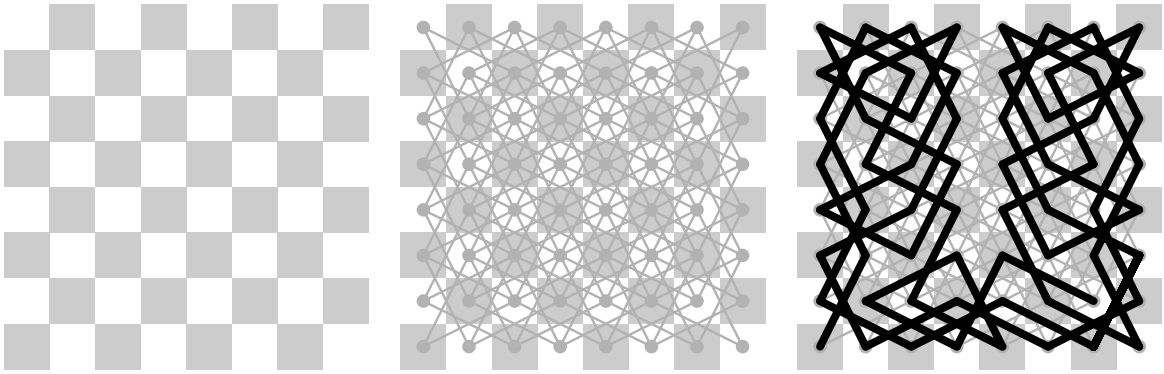

Figure 1: *(left) An* 8×8 *chessboard, (center) the* 8×8 *knight's-move chessboard graph, and (right) a nearly symmetric open knight's tour* $O_{8\times8}$ *of an* 8×8 *chessboard.*

Figure 2 displays four members of a family of what we call *structured knight's tours*: knight's tours that are nearly symmetric, contain repeated geometric motifs, and/or express hidden (or in some cases, not so hidden) mathematical structures. In each one, the knight's tour starts in the bottom left corner and ends in the square that is diagonally adjacent to the square in the bottom right corner. The 8×8 tour shown in the top left panel of Figure 2 is the same as $O_{8\times8}$, the tour that appears in the right-hand panel of Figure 1. This tour appears as a motif in the other three tours, not only in its original form $O_{8\times8}$ but also in two other forms: form $D^{+}_{8\times8}$, which is $O_{8\times8}$'s reflection through the 8×8 board's upward-sloping diagonal, and form $D^{-}_{8\times8}$, which is $O_{8\times8}$'s reflection through the 8×8 board's downward-sloping diagonal. In fact, one recipe for the construction of $O_{16\times16}$, the 16×16 tour shown in the top right panel of Figure 2, is to place a copy of

$O_{8\times8}$ in the 16×16 board's top left quadrant, another copy of $O_{8\times8}$ in its top right quadrant, a copy of $D^{+}_{8\times8}$ in its bottom left quadrant, and a copy of $D^{-}_{8\times8}$ in its bottom right quadrant. The final step of the recipe is to introduce three additional knight's-move edges to stitch together these four 8×8 "quadrant" tours: one edge to join together the tours in the bottom left and top left quadrants, a second "connector" to link the tours in the top left and top right quadrants, and a third to glue together the tours that lie in the top right and bottom right quadrants.

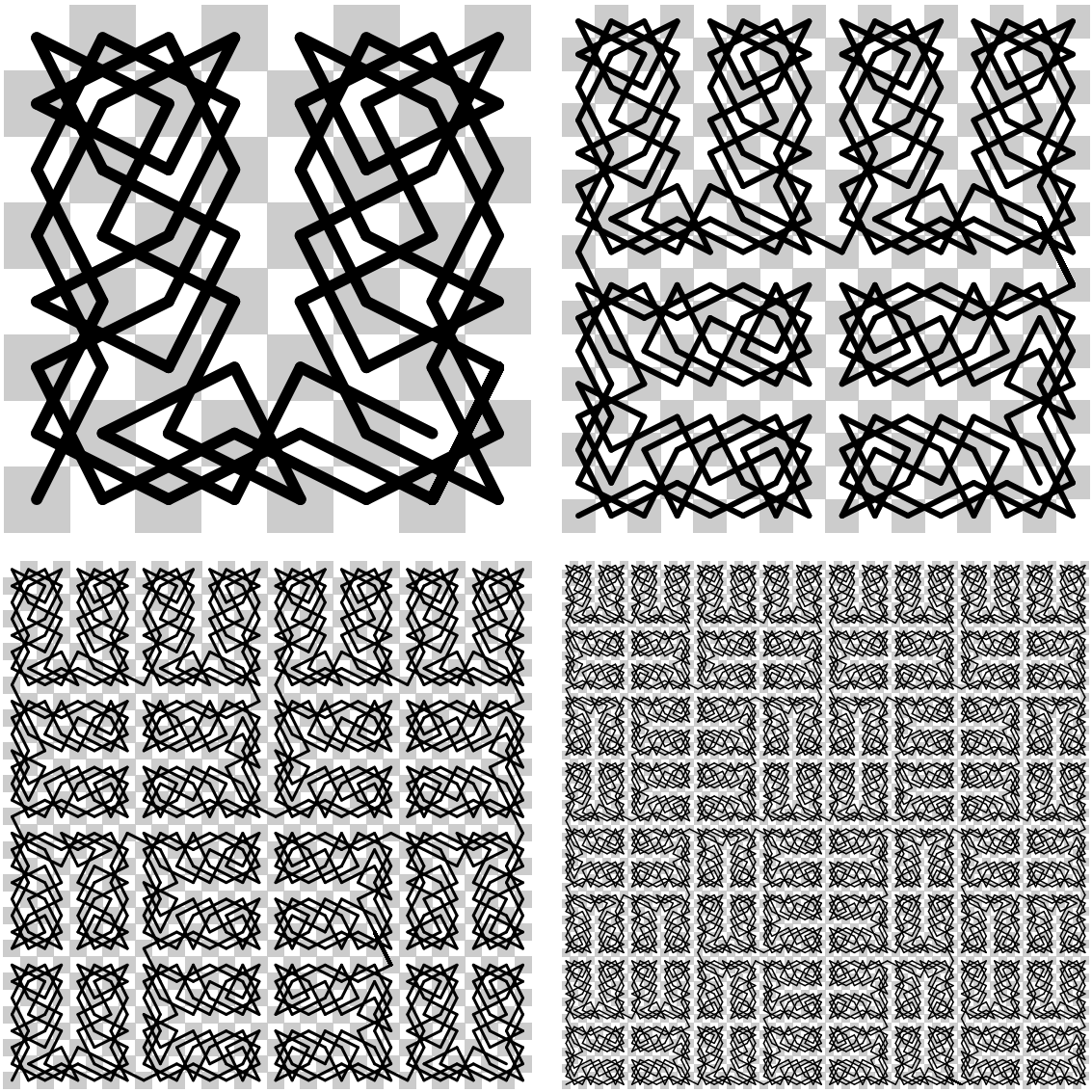

Figure 2: *Four members of a family of open knight's tours of* $8 \cdot 2^k \times 8 \cdot 2^k$ *chessboards (for* $k = 0, 1, 2, 3$*): (top left)* $O_{8\times8}$*, (top right)* $O_{16\times16}$*, (bottom left)* $O_{32\times32}$*, and (bottom right)* $O_{64\times64}$*.*

This recipe can be modified to construct additional members of the family. To form $O_{32\times32}$, shown in Figure 2 (bottom left), we can place copies of $O_{16\times16}$ in the top left and top right quadrants of a 32×32 board, a copy of $D^{+}_{16\times16}$ in the bottom left quadrant, and a copy of $D^{-}_{16\times16}$ in the bottom right quadrant. Again, we include three connector edges to stitch together the four 16×16 quadrant tours. To form $O_{64\times64}$, shown in Figure 2 (bottom right), we can follow the same steps but with two copies of $O_{32\times32}$, a copy of $D^{+}_{32\times32}$, and a

copy of $D^{-}_{32\times32}$. In fact, by repeatedly using the recipe—a substitution rule—we can produce open knight's tours of all $8 \cdot 2^k \times 8 \cdot 2^k$ chessboards with $k \geq 0$. So the recipe provides a new way of establishing that there exist open knight's tours of arbitrarily large chessboards.

If the recipe seems familiar, it is because it is essentially the same algorithm as is used to construct the Hilbert curve. (For a definitive treatment of the Hilbert curve and other space filling curves, see Doug McKenna's beautifully written and programmed e-book [5].) Figure 3 illustrates one iteration of the Hilbert curve construction algorithm. The initial stage, which lives in a unit square, is shown in the left-hand panel of Figure 3. To construct the subsequent stage, which is shown in the right-hand panel of Figure 3, we take a new unit square and divide it into four quadrants. We then place half-sized copies of the initial stage in the top left and top right quadrants of the new square. We then reflect a half-sized version of the initial stage in its square's upward sloping diagonal and place the result in the bottom left quadrant of the new square. We then reflect a half-sized version of the initial stage in its square's downward sloping diagonal and place the result in the bottom right quadrant of the new square. Finally, we introduce connectors to join together the quadrant paths into a single path. (The connectors are drawn in gray.)

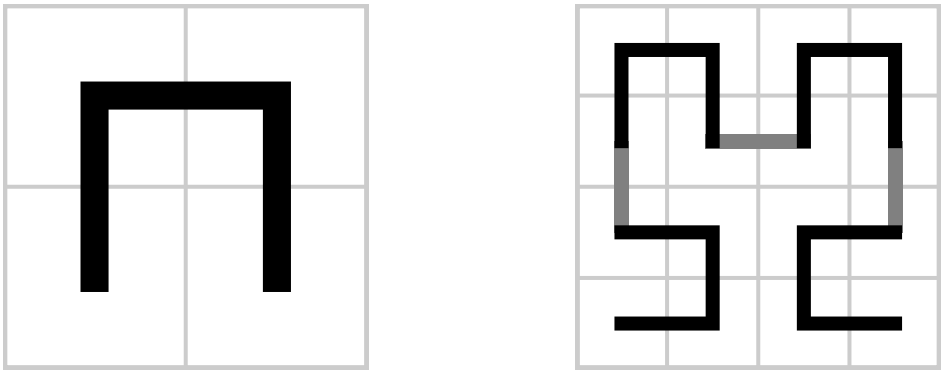

Figure 3: *The Hilbert curve construction process: (left) the initial stage, (right) the subsequent stage.*

Consequently, for each of the open knight's tours displayed in Figure 2 (as well as the rest of the members of the family), it can be said that the knight visits the individual 8×8 blocks of the board in a Hilbert-curve-like fashion. This means that each of these knight's tour has, in a well-defined sense, a stage of the Hilbert curve hidden within the path it takes from its start square to its end square.

Figure 4 displays four more "hidden-Hilbert" open knight's tours of a 64×64 chessboard. Like the tours shown in Figure 2, each one was constructed from a single motif, a nearly symmetric open knight's tour of an 8×8 chessboard. Although each motif is the same length and requires the same amount of ink to draw it—each is an open tour of an 8×8 chessboard and therefore consists of 63 knight's moves—each of the resulting 64×64 tours looks quite different from the others. This is due to the different ways in which the 63 knight's moves are distributed in the 8×8 motifs. In some cases, the white space (or spaces) in the motif and the way in which the recipe arranges the copies of the motif (in its original form and its reflected forms) results in emergent patterns that we find quite pleasing to the eye.

In this paper we describe how to construct motifs for structured knight's tours on chessboards, cylindrical chessboards, and chessboards that live on Möbius strips. For thorough treatments of knight's tours, we direct the reader to John Watkin's wonderful book [8] and George Jellis's marvelous webpages [4].

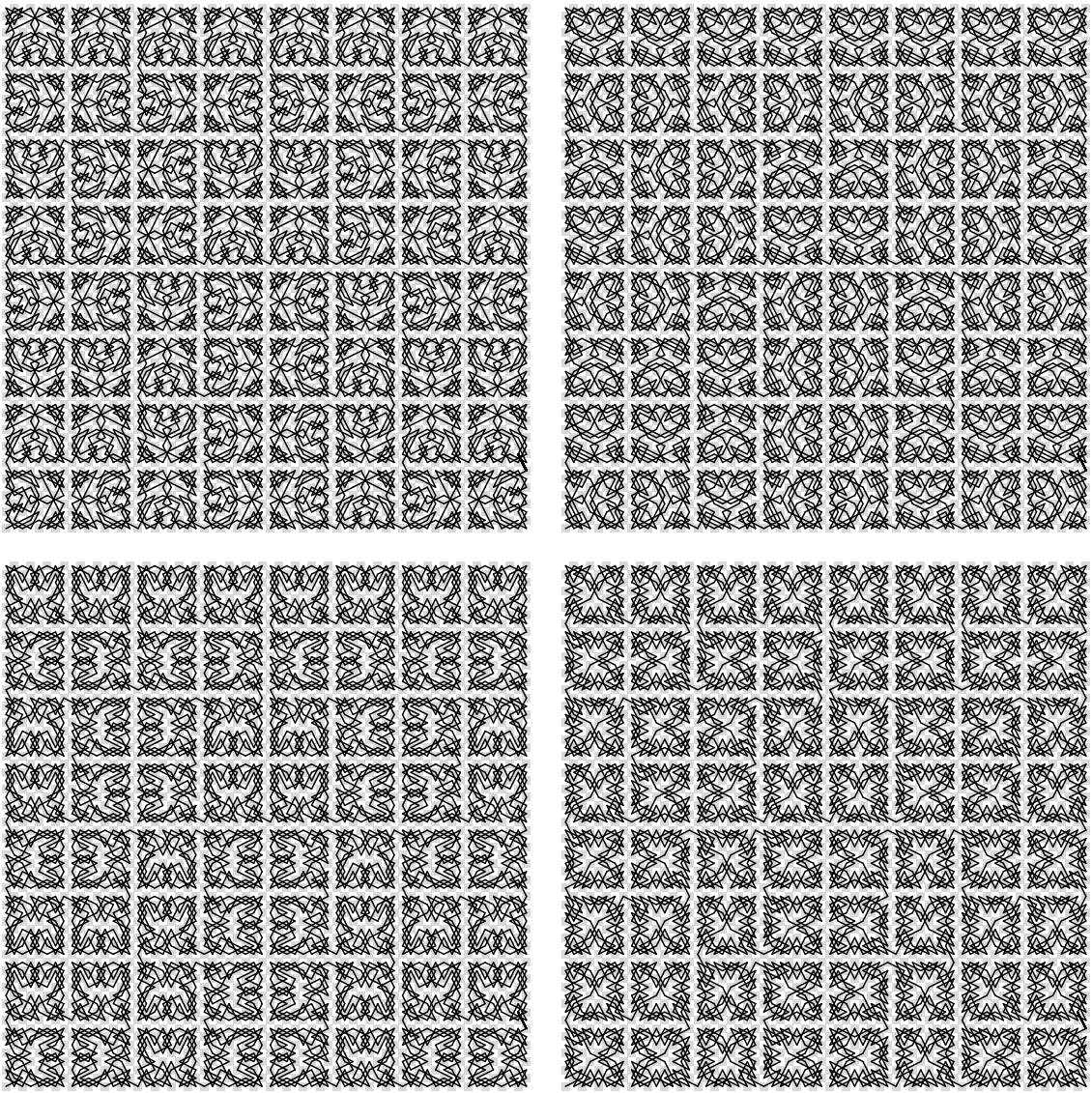

Figure 4: *Four additional hidden-Hilbert open knight's tours of a* 64×64 *chessboard.*

Motif Design via Discrete Optimization

To find a motif like $O_{8\times 8}$, which is one of the open knight's tours of an 8×8 chessboard that is as close as is possible to having vertical mirror symmetry, we follow the example of [1] and introduce additional Boolean variables into the Dantzig-Fulkerson-Johnson integer programming (IP) formulation of the Traveling Salesperson Problem (TSP) [2]. In addition to having a Boolean variable x_e that equals 1 if we select the undirected knight's-move edge e to be in the tour and 0 if we don't, we include a Boolean symmetry-detection variable v_e that equals 1 if e is a part of an instance of vertical mirror symmetry and 0 if it isn't.

We let V denote the set of all vertices (the set of all squares of an 8×8 chessboard), and we let $s \in V$ and $t \in V$ denote the start and end squares respectively. We let E denote the set of all possible edges (the set of all possible undirected knight's-moves on the board). For each $e \in E$, we let e' stand for the edge that is the reflection of e through the board's vertical bisector. To ensure that both e and its symmetric counterpart

e' appear in the tour when v_e equals 1, we impose two linear inequalities: $v_e \le x_e$ and $v_e \le x_{e'}$. Because the variables are Boolean, we can think of these linear inequalities as logical implications. The first one states that if v_e equals 1, then x_e equals 1, and the second one states that if v_e equals 1, then $x_{e'}$ equals 1.

With the v_e symmetry-detection variables and the linear inequalities that link them to the x_e edge-usage variables, we can try to maximize the number of instances of vertical mirror symmetry in the tour by solving the following IP:

$$\begin{aligned}
\text{maximize} \quad & \sum (v_e \,:\, e \in E) \\
\text{subject to} \quad & v_e \le x_e \quad \textit{and} \quad v_e \le x_{e'} \qquad \textit{for all } e \in E, \\
& \sum (x_e \,:\, e \textit{ is incident to } s) = 1, && (1) \\
& \sum (x_e \,:\, e \textit{ is incident to } t) = 1, && (2) \\
& \sum (x_e \,:\, e \textit{ is incident to } v) = 2 \qquad \textit{for all } v \in V - \{s, t\}. && (3)
\end{aligned}$$

The numbered equations are called *degree constraints*. Equation (1) ensures that the knight's tour includes precisely one edge incident to the start square, and equation (2) ensures that it includes precisely one edge incident to the end square. After all, the knight has to be able to start its tour by leaving the start square and finish its tour by entering the end square. The equations in (3) make sure that every other square of the board is incident to precisely two edges of the tour. While on its tour, the knight has to enter and exit each square that's not the start or end square.

Usually, additional constraints are needed. When we solve the IP (with something like the Gurobi Optimizer [3]), we may very well discover that the solution is not a tour but instead contains two or more *subtours*. An example of a four-vertex subtour is displayed in the left-hand side of Figure 5.

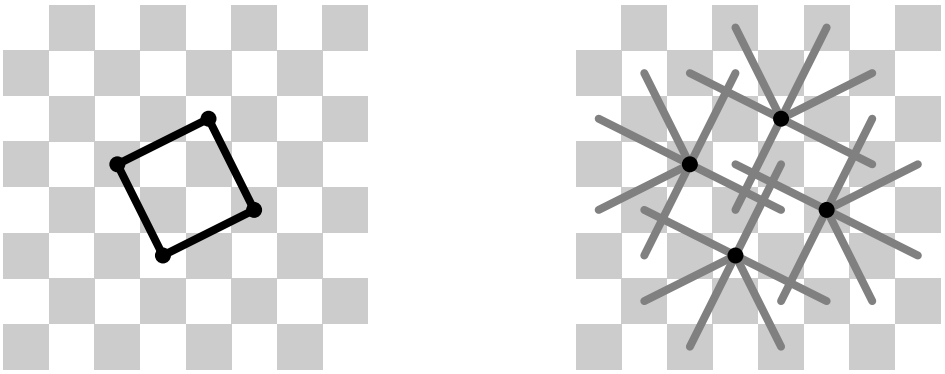

Figure 5: *(left) A subtour and its vertex set S, (right) the set of all edges $\delta(S)$ that connect S to $V - S$.*

To eliminate this subtour, we start by identifying the vertices it visits. We call this set of vertices S. We then find all of the edges that are incident to precisely one vertex in S. We call this set of edges $\delta(S)$. An example of a $\delta(S)$ set is shown in the right-hand side of Figure 5 (the gray edges). Note that none of the subtour's edges belong to $\delta(S)$, as each of them is incident to two vertices in S. To break the subtour (or equivalently, to prevent it from appearing again when we try to find a new solution), we can impose just one additional linear inequality:

$$\sum (x_e \,:\, e \in \delta(S)) \ge 2.$$

This *subtour elimination constraint* breaks the subtour by ensuring that the solution includes at least two edges that connect the vertices in S to the other vertices (the vertices in $V - S$).

If the solution has multiple subtours, we impose multiple subtour elimination constraints. Each time we impose one or more subtour elimination constraints, we re-solve the IP. We keep doing this—handling the subtours in a "whack-a-mole" fashion, eliminating them as they pop up—until we finally obtain a solution that has no subtours. And if we want to find additional tours, we can impose linear constraints of the form

$$\sum(x_e \,:\, e \in E_k) \leq |E_k| - 1,$$

where E_k is the set of all edges that belong to the kth tour, to rule out previously obtained tours.

Tessellation-Like Tours

It is relatively straightforward to modify the IP formulation described above so that it can be used to find larger motifs and/or open (or closed) knight's tours that have nearly fourfold symmetry. In addition, we can impose additional constraints that force certain knight's-tour edges to appear in the tour as well as other constraints that force much the interior of the tour to have translational symmetry and thereby resemble a tessellation. Figure 6 presents six examples of nearly fourfold "tessellation-like" structured knight's tours.

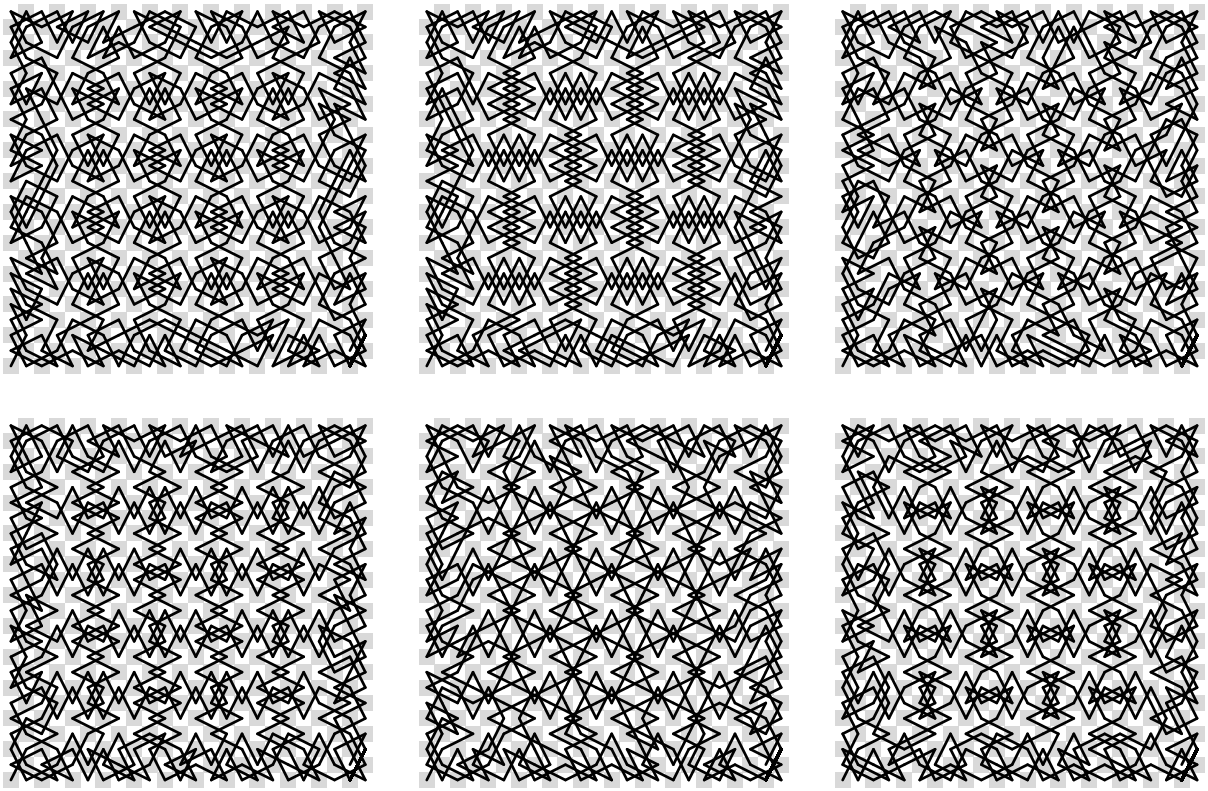

Figure 6: *Six nearly fourfold tessellation-like open knight's tours of a* 24×24 *chessboard.*

Although we can (and do) join together tessellation-like tours to form larger hidden-Hilbert tours, in Figure 7 we display something distinctly different: two 8×8 tours, one 16×16 tour, one 24×24 tour, one 40×40 tour, and one 64×64 tour all stitched together to form an open knight's tour of a 64×104 chessboard. The tour can be considered to be a visual proof of the Fibonacci identity $1^2 + 1^2 + 2^2 + 3^2 + 5^2 + 8^2 = 8 \cdot 13$.

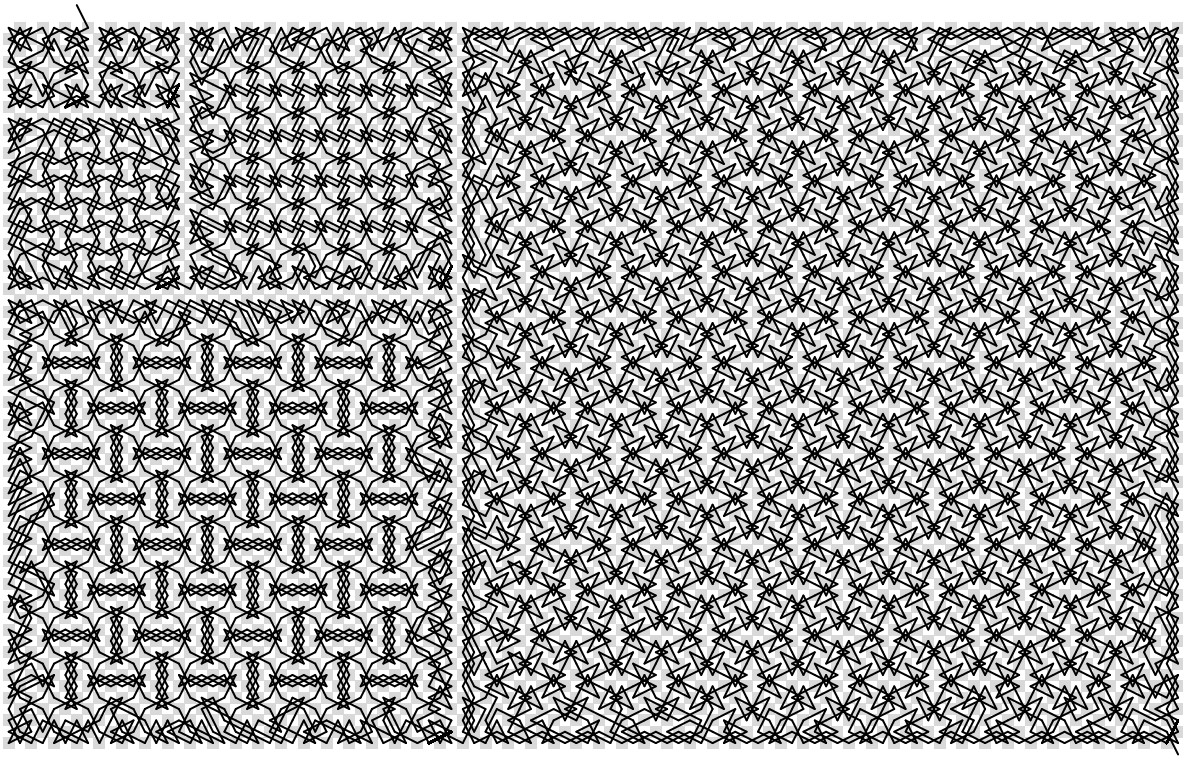

Figure 7: *An open knight's tour of a* 64×104 *chessboard, with two extra edges to make it easier to find the start and end of the tour. The tour can be considered to be a visual proof of the Fibonacci identity* $1^2 + 1^2 + 2^2 + 3^2 + 5^2 + 8^2 = 8 \cdot 13$.

Knight's Tours on Cylindrical Boards and Möbius Strips

With some minor modifications, the optimization modeling techniques discussed above can be applied to cylindrical chessboards and also boards that live on Möbius strips. On a cylindrical board, the leftmost and rightmost columns of squares are considered to be next to one other, which means that on a cylindrical board, a knight can (for example) move directly from a square in the board's leftmost column to a square in the board's rightmost column provided that the square it lands on is either two rows up or two rows down from the row in which it started, or directly from a square in the leftmost column to a square in the second-to-rightmost column provided that the square it lands on is either one row up or one row down from the row in which it started. Of course, the terms "leftmost" and "rightmost" and "second-to-rightmost" only make sense when the board is drawn on a rectangular strip. It is only when we have finished gluing the left edge of the rectangular strip to its right edge that the board takes on a cylindrical shape. Note that if we make make a half twist in the rectangular strip before we glue together its left and right edges, we will end up with a Möbius strip instead of a cylinder.

Figure 8 displays photos of 3D printed versions of four closed knight's tours of cylindrical and Möbius-strip chessboards. We did the 3D modeling work in OpenSCAD [6], and we ordered the 3D prints from Shapeways [7]. The cylindrical examples came from an 8×64 chessboard. The left-hand Möbius-strip example came from an 8×40 chessboard, and its right-hand counterpart from an 8×72 chessboard.

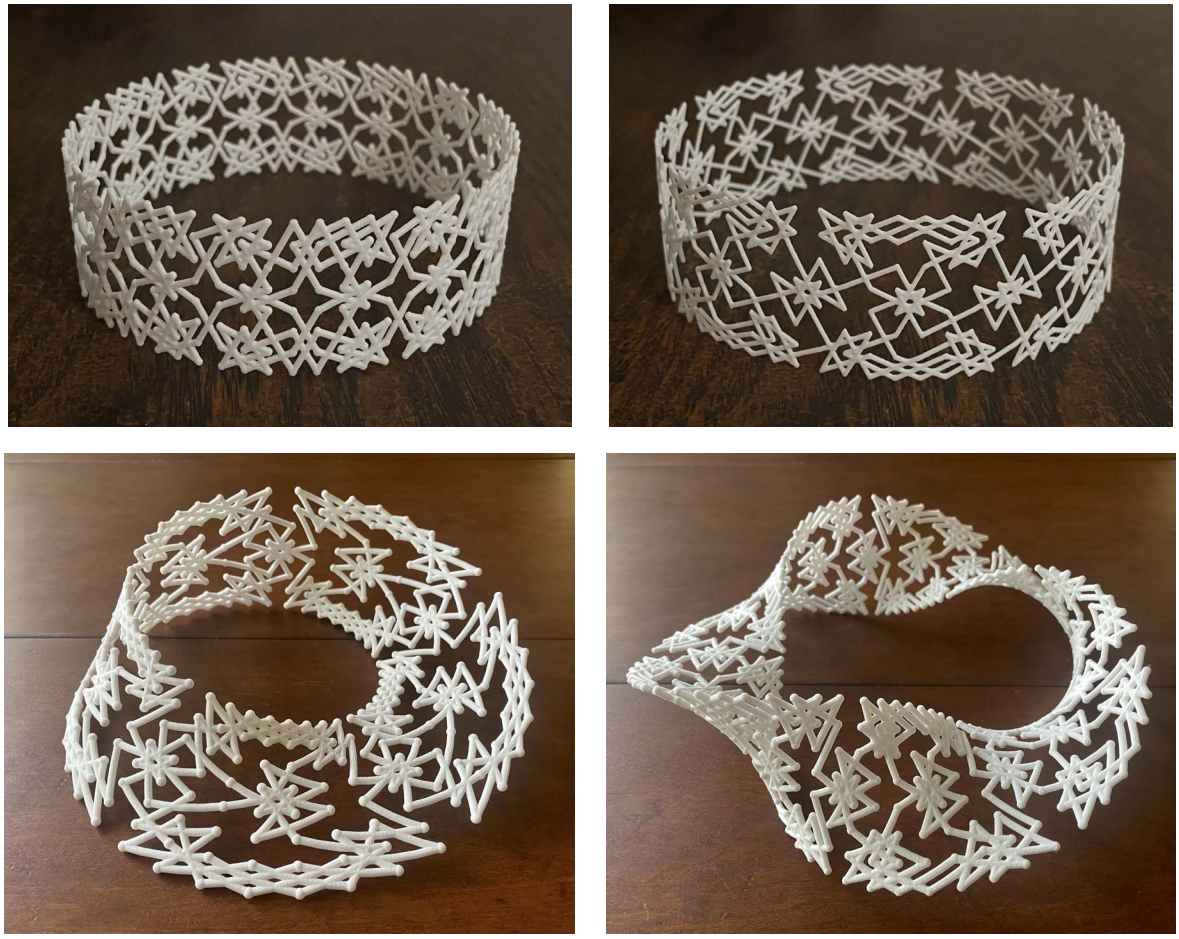

Figure 8: *3D printed closed knight's tours of cylindrical and Möbius-strip chessboards.*

References

[1] Robert Bosch. *Opt Art: From Mathematical Optimization to Visual Design*. Princeton University Press, 2019.

[2] William J. Cook. *In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation*. Princeton University Press, 2012.

[3] Gurobi optimization. http://www.gurobi.com/products/gurobi-optimizer, as of Feb. 1, 2021.

[4] George Jellis. "Knight's Tour Notes." Available online at https://www.mayhematics.com/t/t.htm, as of Feb. 1, 2021.

[5] Douglas M. McKenna, *Hilbert Curves: Outside In and Inside Gone*, Mathemaesthetics, Inc., 2019, ISBN: 978-1-7332188-0-1 (eBook/app for iOS, https://apps.apple.com/us/app/hilbert-curves/id1453611170, as of Feb. 1, 2021).

[6] OpenSCAD: The Programmers' Solid 3D CAD Modeler. https://www.openscad.org/, as of Feb. 1, 2021.

[7] Shapeways. https://www.shapeways.com/, as of Feb. 1, 2021.

[8] John J. Watkins. *Across the Board: the Mathematics of Chessboard Problems*. Princeton University Press, 2004.