Taylor & Francis
Taylor & Francis Group

Experimental Mathematics

ISSN: 1058-6458 (Print) 1944-950X (Online) Journal homepage: http://www.tandfonline.com/loi/uexm20

Heesch Numbers of Edge-Marked Polyforms

Casey Mann & B. Charles Thomas

To cite this article: Casey Mann & B. Charles Thomas (2016) Heesch Numbers of Edge-Marked Polyforms, Experimental Mathematics, 25:3, 281-294, DOI: 10.1080/10586458.2015.1096867

To link to this article: http://dx.doi.org/10.1080/10586458.2015.1096867

EXPERIMENTAL MATHEMATICS
2016, VOL. 25, NO. 3, 281–294
http://dx.doi.org/10.1080/10586458.2015.1096867

Taylor & Francis
Taylor & Francis Group

Heesch Numbers of Edge-Marked Polyforms

Casey Mann[a] and B. Charles Thomas[b]

[a]School of STEM, University of Washington Bothell, Bothell, WA, USA; [b]Department of Mathematics, The University of Texas at Tyler, Tyler, TX, USA

ABSTRACT
In this article, the results of a computerized search for edge-marked polyforms (tiles formed from either equilateral triangles, squares, or regular hexagons) with finite Heesch number are presented.

KEYWORDS
tiling; Heesch number; polyforms

2000 AMS SUBJECT CLASSIFICATION
52cxx; 52C20

1. Introduction

Tiles whose edges are marked with matching rules are the basis for some of tiling theory's most important examples, such as Wang tiles and their role in the decidability of the Tiling Problem [Berger 66, Robinson 71], aperiodic protosets [Berger 66, Goodman-Strauss 99, Grünbaum and Shephard 87, Penrose 80, Robinson 71, Socolar and Taylor 12], Heesch numbers [Mann 01, Mann 04], and anisohedral numbers [Myers 12, Rhoads 05]. Additionally, static and dynamic edge-marked tiles have found application in self-assembly models [Pelesko 07, Patitz 14]. Edge matching rules are often applied to a basic geometric shape that tiles the plane, such as a square or a regular hexagon, in order to enforce long-range structure, as in nonperiodicity and anisohedrality, or to restrict or specify certain kind of growth, as in Heesch numbers and self-assembly (Figure 1).

We begin with some definitions. A *tiling of the plane*, $\mathscr{T}$, is a countable family of closed topological disks $\mathscr{T} = \{T_1, T_2, \ldots\}$ that cover the Euclidean plane $\mathbb{E}^2$ without gaps or overlaps; that is, $\mathscr{T}$ satisfies:

1. $\bigcup_{i\in\mathbb{N}} T_i = \mathbb{E}^2$, and
2. $\text{int}(T_i)\cap\text{int}(T_j) = \emptyset$ when $i \neq j$.

The sets T_i of $\mathscr{T}$ are called *tiles*. A set of tiles $\mathcal{P} = \{\hat{T}_1, \ldots, \hat{T}_n\}$ is a *protoset* for $\mathscr{T}$ if for each $T_i \in \mathscr{T}$, there is a tile $\hat{T}_j \in \mathcal{P}$ such that T_i is congruent to $\hat{T}_j$. The tiles of $\mathcal{P}$ are called *prototiles*. $\mathcal{P}$ is said to *admit* the tiling $\mathscr{T}$, though $\mathcal{P}$ may admit other tilings as well. Given a single prototile $\hat{T}$ (thought of as a singleton protoset), $\hat{T}$ may or may not admit a tiling of the plane. If $\hat{T}$ does not admit a tiling of the plane, then there is a maximum number of "layers" or "coronas" formed from copies of $\hat{T}$ that may surround a centrally placed copy of $\hat{T}$. This maximum number of coronas is called the *Heesch number* of $\hat{T}$. To be precise, a *patch* of tiles is any finite collection of tiles whose union is a topological disk. A *first corona* $\mathcal{C}_1$ surrounding T is a collection of congruent copies of T not containing T that satisfies:

1. $T\cap U \neq \emptyset$ for all U in $\mathcal{C}_1$, and
2. $T \cup \bigcup_{U\in\mathcal{C}_1} U$ is a patch.

In Figure 2 is a tile surrounded by one corona. A *second corona* (or any subsequent coronas) can be defined inductively by replacing T in the definition above with the patch formed by the previous corona. For example, the edge-marked hexagons of Figures 3 and 4 show tiles surrounded by three coronas. For the tiles of Figures 2–4, those are the maximum number of coronas that can be formed, so their Heesch numbers are 1, 3, and 3, respectively. The tile of Figure 2 was described by Heesch in [Heesch 68], the paper in which the problem of finding tiles that do not tile the plane but can form one or more coronas around a centrally placed copy is described. If a tile can tessellate the entire plane, its Heesch number is ∞. *Heesch's Tiling Problem* is to find all positive integers n for which there is a prototile with Heesch number n. At this time, Heesch's Tiling Problem has been solved for $n = 1, 2, 3, 4$, and 5 [Mann 01, Mann 04, Fontaine 91]. In [Mann 01] and [Mann 04], several specific examples with Heesch numbers 1, 2, 3, 4, and 5 were reported, but at that time, no exhaustive search had been conducted (as will be presented here). Recently, hexagons with generalized matching rules have been found having Heesch numbers 1 through 9 and 11 [DeWeese 08].

CONTACT Casey Mann cemann@uw.edu School of STEM, University of Washington Bothell, 18115 Campus Way N.E., Box 358538, Bothell, WA 98011, USA.
Color versions of one or more figures in the article can be found online at www.tandfonline.com/uexm

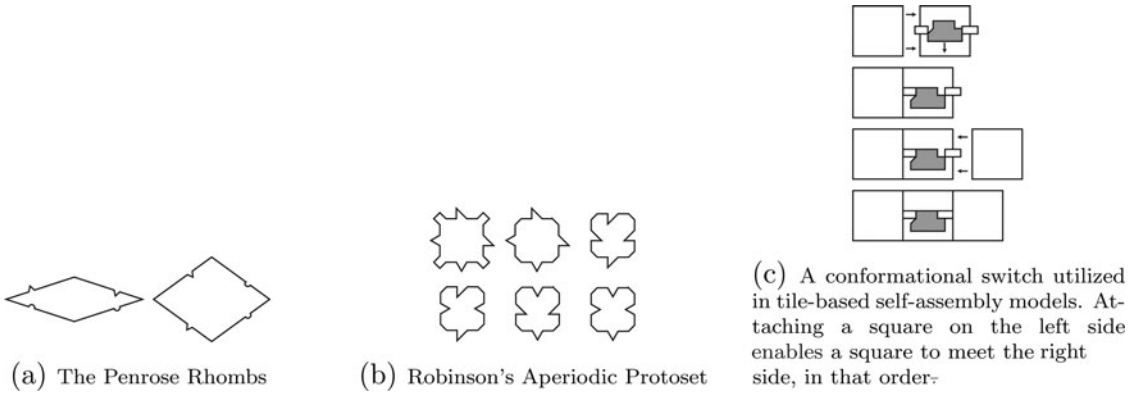

Figure 1. Well-known tiles with edge matching rules.

Polyforms are tiles formed as the union multiple regular polygons of a single kind meeting corner-to-corner. *Polyhexes* are formed from regular hexagons, *polyominoes* are formed from squares, and *polyiamonds* are formed from equilateral triangles. A polyform comprised of n base shapes is called an *n-hex, n-omino,* or an *n-iamond.* Polyforms, especially polyominoes, have been studied fairly extensively with regards to planar tilings, but polyforms whose edges carry matching rules are relatively unstudied. The main focus of this article is the presentation of examples of polyforms with finite Heesch number that were discovered using computers.

The *matching rules* of a tile can be defined in full generality as a boolean function on the Cartesian product of the set of sides of a tile that declares whether or not two sides of a tile may meet. A tile T's matching rules are *geometrically enforceable* if replacing each side of T with either a straight line segment, an *S-curve* (a nonstraight centrally symmetric arc), or a *J-curve* (a nonstraight curve with no nontrivial symmetry) yields exactly the same matching rules. In our study, we focused on a few specific types of matching rules – some geometrically enforceable and some not. In particular, we studied polyforms, some (or all) of whose edges are marked with bumps and nicks, colors, and directionality.

- *Bumps and nicks.* This is a geometrically enforceable edge marking device in which straight sides of a tile are replaced with centrally symmetric curves that have an outward protrusion (bump) or an inward indentation (nick). Two tiles so marked may meet along an edge when the curves forming those edges are congruent and when their parities are opposite (i.e., bumps must meet nicks).
- *Colors.* This edge matching condition is simply a labeling of edges with the rule that when two labeled sides meet the edges must be identically labeled.
- *Directionality.* A direction may be imposed on the edges of a tile so that two tiles may meet along an edge only if the two edges have the same direction.

While colored edges and oriented edges are not geometrically enforceable in isolation of other types of edge-markings, some combinations of these types of markings are geometrically enforceable. In particular, bumps/nicks + colors, bumps/nicks + directions, and bumps/nicks + colors + directions are geometrically enforceable. For example, combining bumps/nicks with colors is

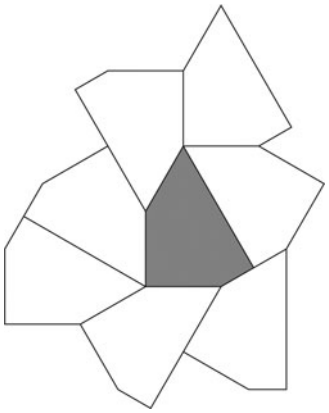

Figure 2. A polygon with Heesch number 1.

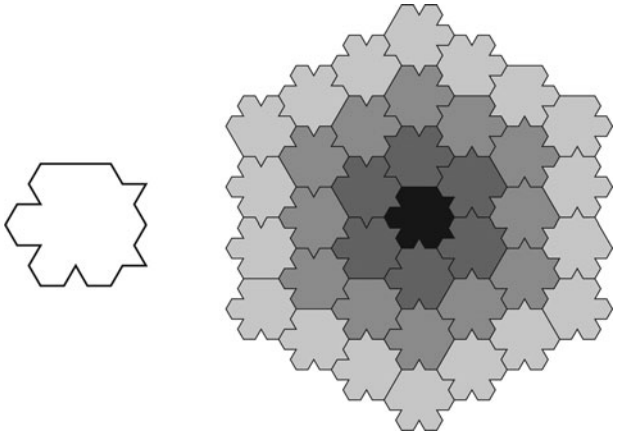

Figure 3. Ammann's Heesch number 3 hexagon.

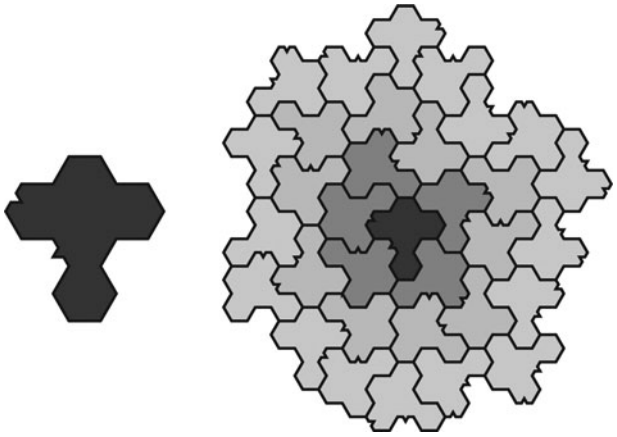

Figure 4. A 5-hex with Heesch number 3.

equivalent to using different shapes for the bumps/nicks corresponding to different colors, and combining bumps/nicks with directions is equivalent to asymmetric bumps/nicks (see Figure 5).

Using these types of edge matching rules, a given polyform can be marked in a multitude of different ways, some of which result in a tile that can tessellate the plane and some of which cannot. The purpose of this article is to describe some computer-generated results obtained by marking the edges of low-order polyforms in all possible ways and attempting to determine (via brute force computer search) the Heesch number of these marked polyforms.

2. Heesch numbers, aperiodic protosets, and the tiling problem

Heesch numbers, aperiodic protosets, and the Tiling Problem share some interesting connections. These connections and others are explored in more detail in [Goodman-Strauss 00], but we will briefly mention a few here. The common thread of these problems is that they investigate the complexity exhibited by individual tiles or sets of tiles. The *Tiling Problem* asks if there exists an algorithm that can decide in a finite number of steps whether an arbitrary input set of tiles can tessellate the plane. The

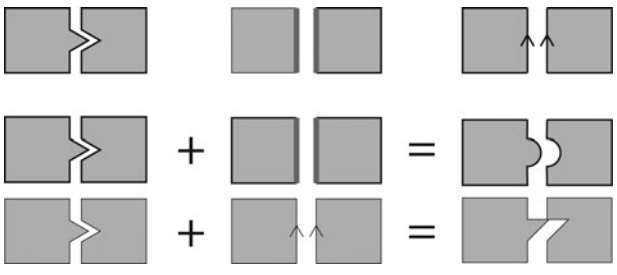

Figure 5. Common matching rules.

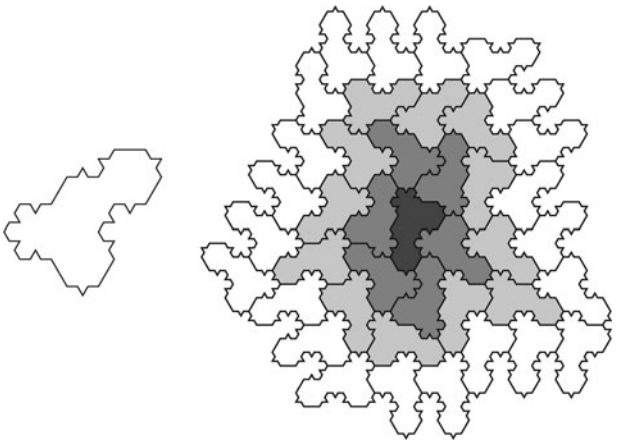

Figure 6. A combinatorially unbalanced 4-hex with Heesch number 3.

Tiling Problem has been shown to be undecidable [Berger 66], but the problem remains open when the input tile sets are restricted to singleton tile sets (i.e., *monotiles*). That is, it is unknown whether or not an algorithm exists that can take as input a monotile and determine whether it admits a tiling of the plane. To make this problem discrete, let us restrict it further by considering only curvilinear polygonal monotiles; that is, monotiles whose boundaries are comprised of a finite number of curvilinear arcs, and each vertex in a tiling admitted by such a monotile must be an endpoint of an arc from each tile containing the vertex. This restriction allows copies of the tile to fit together in only finitely many ways. We will call this problem the *Tiling Problem for Polygonal Monotiles (TPPM)*. Here are two interesting connections:

1. If TPPM is undecidable, then there exists an aperiodic monotile (i.e., a tile that admits tilings of the plane, but only nonperiod tilings).
2. If TPPM is undecidable, then there exist tiles with arbitrarily large Heesch numbers.

The first of these connections can be justified as follows. First, assume there are no aperiodic monotiles. A decision algorithm would go as follows:

i. For each positive integer k, there are only a finite number of ways to join k copies of a given monotile together, so find all such configurations of k copies of the monotile.
ii. For each configuration of k copies of the monotile, partition the boundary of the configuration into 4 arcs in every possible way, calling these arcs *top*, *bottom*, *left*, and *right* where the endpoints of these arcs are endpoints of the arcs comprising the boundaries of the constituent tiles.
iii. If top is a translate of bottom and left is a translate of right, then the monotile admits a periodic tiling.

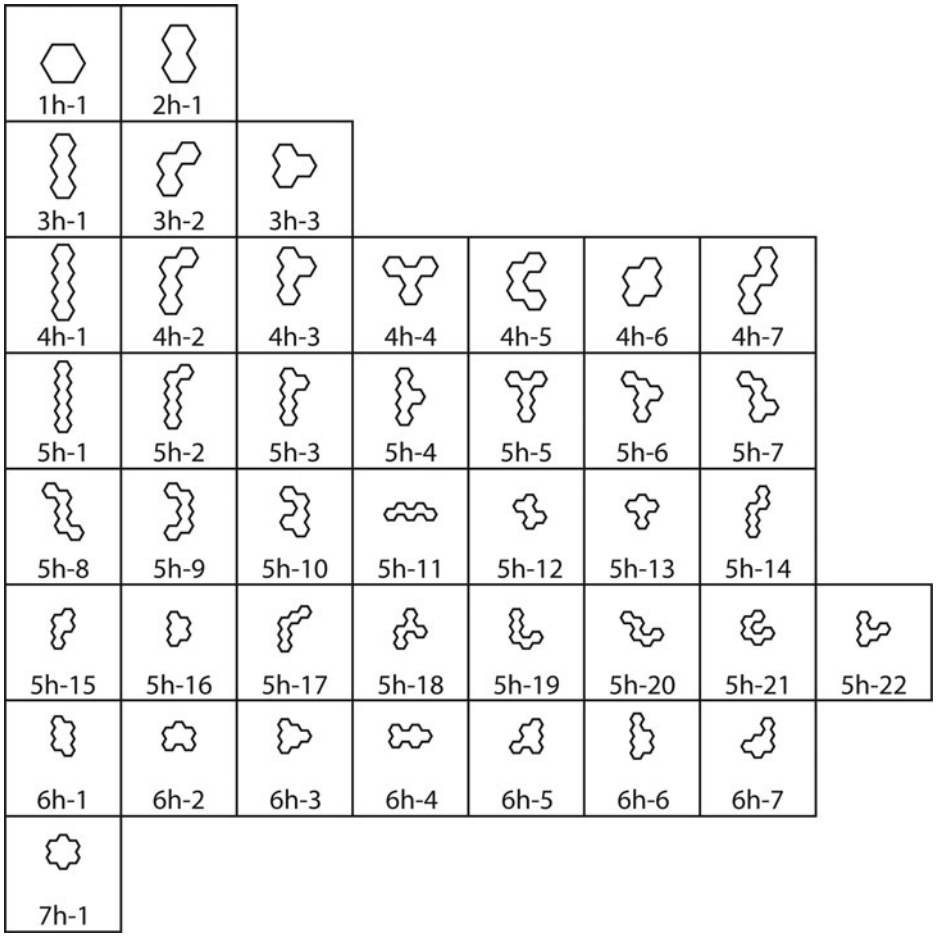

Figure 7. The examined polyhexes.

Eventually, this procedure will find a fundamental region for a periodic tiling, or else it will determine that the tile has finite Heesch number.

The second implication above involving Heesch numbers is astounding, for it is difficult to fathom a tile that can form a few trillion layers around a centrally placed copy, but somehow cannot form one more layer! To see why the second connection to TPPM is true, suppose that M is an upper bound on Heesch numbers for polygonal monotiles. Then a decision algorithm would involve exhaustively forming layers around a centrally placed copy of the tile until it is found that either $M + 1$ layers cannot be formed (in which case the monotile does not admit a tiling), or $M + 1$ layers can be formed (in which case, the monotile admits a tiling). The examples of tiles with "large" Heesch numbers shown in this article may be thought of as providing evidence that TPPM is undecidable.

3. Combinatorial imbalance

Of the marked tiles with Heesch number 2 or bigger, one sees a few recurrent themes as to the mechanism that allows a tile to have high Heesch number. One such mechanism is *combinatorial imbalance*, such as when a tile is marked with an unequal number of bumps and nicks. All of the polyforms studied here tile the plane when their edges are unmarked, so marking the edges has the effect of restraining the ability of the shapes to tile the plane. Endowing one of these polyforms with a combinatorial imbalance prevents the shape from tiling the plane, but when the combinatorial imbalance is slight, it will allow for high Heesch numbers. Figures 3 and 6 illustrate this phenomenon: Notice that, in both of these maximal coronal configurations, the prototiles are both marked with more nicks than bumps and boundary is populated almost entirely with nicks. Because the prototiles are combinatorially unbalanced, and configuration formed from these tiles must have unmatched nicks, and these unmatched nicks must necessarily appear on the boundary of the configuration.

Combinatorially unbalanced polyforms always have finite Heesch number.

Theorem 1. *Let T be a polyform with n edges each of which is marked with a bump, a nick, or is unmarked. If T is combinatorially unbalanced, then T has finite Heesch number.*

Table 1. Polyhexes marked with bumps and nicks.

Base shape (see Figure 7)	H1	H2	H3	H4	H5	Admits more than 5 coronas	Timed out
1h-1	66	0	12	0	0	65	3
2h-1	716	0	0	4	0	1227	18
3h-1	3514	32	0	4	0	17,786	49
3h-2	4630	28	2	0	0	11,689	42
3h-3	1542	51	0	0	0	4533	11
4h-1	22,002	58	0	0	4	183,000	241
4h-2	16,941	46	8	0	0	78,634	68
4h-3	5906	17	2	0	0	26,529	39
4h-4	10,560	0	0	0	0	114,703	66
4h-5	9567	2	0	0	0	33,946	260
4h-6	2256	12	0	0	0	16,566	56
4h-7	22,855	60	23	0	0	139,623	62
5h-1	181,299	104	4	0	4	2,318,733	2597
5h-2	81,915	36	0	0	0	578,632	380
5h-3	35,268	11	0	0	0	178,846	103
5h-4	42,358	2	0	0	0	196,165	255
5h-5	3756	0	0	0	0	72,185	2
5h-6	16,429	53	0	0	0	81,922	10
5h-7	31,735	103	0	0	0	125,147	74
5h-8	120,745	290	0	0	0	645,222	166
5h-9	66,677	0	0	0	0	204,393	119
5h-10	32,949	1	0	0	0	39,514	87
5h-11	109,215	20	0	0	0	628,908	173
5h-12	11,818	98	50	0	0	62,698	22
5h-13	12,120	116	26	0	0	41,457	33
5h-14	41,568	27	0	0	0	556,788	193
5h-15	11,030	88	0	0	0	68,775	34
5h-16	5085	34	0	0	0	23,271	37
5h-17	98,648	248	0	0	0	503,493	488
5h-18	60,009	481	33	0	0	198,269	33
5h-19	14,649	246	0	0	0	123,694	6
5h-20	23,864	393	3	0	0	88,175	22
5h-21	32,472	108	13	0	0	140,916	28
5h-22	28,940	6	0	0	0	144,137	243
6h-1	10,032	20	0	0	0	193,468	33
6h-2	8920	22	0	0	0	88,224	13
6h-3	19,083	0	0	0	0	105,722	56
6h-4	22,767	0	0	0	0	359,074	34
6h-5	16,214	0	0	0	0	214,745	12
6h-6	26,266	39	0	0	0	261,492	75
6h-7	19,546	9	0	0	0	299,711	34

Table 2. Polyominoes marked with bumps and nicks (order 1–5).

Base shape (see Figure 8)	H1	H2	H3	H4	Admits more than 4 coronas	Timed out
1s-1	8	0	0	0	14	0
2s-1	32	4	0	0	68	0
3s-1	154	4	0	0	312	2
3s-2	104	0	0	0	250	1
4s-1	498	0	4	0	1709	18
4s-2	110	0	0	0	755	12
4s-3	104	0	0	0	369	0
4s-4	82	0	0	0	1109	0
4s-5	218	3	0	0	1128	8
5s-1	1340	12	4	0	6890	26
5s-2	240	0	0	0	2345	0
5s-3	32	0	0	0	2455	0
5s-4	267	0	0	0	2138	1
5s-5	513	0	0	0	2418	6
5s-6	107	0	0	0	755	1
5s-7	288	0	0	0	2417	3
5s-8	284	0	0	0	2076	6
5s-9	140	0	0	0	3089	4
5s-10	151	0	0	0	1360	1
5s-11	134	0	0	0	482	0
5s-12	169	0	0	0	1946	8

Table 3. Polyominoes marked with bumps and nicks (order 6).

Base shape (see Figure 8)	H1	H2	H3	H4	H5	Admits more than 5 coronas	Timed out
6s-1	4760	36	4	0	0	31,456	148
6s-2	466	0	0	0	0	11,949	66
6s-3	208	16	0	0	0	1641	4
6s-4	294	0	0	0	0	7114	0
6s-5	176	0	0	0	0	3227	11
6s-6	200	0	0	0	0	3046	0
6s-7	693	0	0	0	0	4660	6
6s-8	471	0	0	0	0	4212	22
6s-9	1579	7	0	0	0	10,340	98
6s-10	68	0	0	0	0	3637	1
6s-11	536	0	0	0	0	9155	8
6s-12	65	0	0	0	0	3278	7
6s-13	296	0	0	0	0	16,071	14
6s-14	185	0	0	0	0	2793	4
6s-15	121	0	0	0	0	10,424	0
6s-16	192	0	0	0	0	16,377	3
6s-17	766	0	0	0	0	7614	45
6s-18	220	0	0	0	0	4591	1
6s-19	300	0	0	0	0	8015	20
6s-20	466	0	0	0	0	6895	10
6s-21	327	0	0	0	0	10,319	5
6s-22	660	1	0	0	0	9904	19
6s-23	421	0	0	0	0	9988	19
6s-24	359	0	0	0	0	10,326	18
6s-25	341	0	0	0	0	2087	22
6s-26	142	0	0	0	0	2416	6
6s-27	776	0	0	0	0	6423	25
6s-28	1327	0	0	0	0	4187	1
6s-29	624	0	0	0	0	4457	7
6s-30	192	0	0	0	0	1762	0
6s-31	83	0	0	0	0	2751	0
6s-32	71	0	0	0	0	3816	5
6s-33	545	0	0	0	0	5200	5
6s-34	98	0	0	0	0	1604	2
6s-35	718	0	0	0	0	5201	18

Proof. Suppose to the contrary that T is combinatorially unbalanced and admits a tiling $\mathscr{T}$ of the plane. Let $r > 0$ and P be any point in the plane. Let $\mathcal{A}(r, P)$ be the minimal simply connected patch that contains the closed disk of radius r centered at P. Set

$$t(r, P) = \text{the number of tiles in } \mathcal{A}(r, P)$$

and

$$E(r, P) = \text{the number of edges on the boundary of } \mathcal{A}(r, P).$$

If $\mathscr{T}$ is normal, the normality lemma [Grünbaum and Shephard 87] states that for every $x > 0$,

$$\lim_{r\to\infty} \frac{t(r + x, P) - t(r, P)}{t(r, P)} = 0.$$

The normality lemma is true even when the requirement of normality is relaxed; all that is required is that the tiles of the tiling are uniformly bounded topolgical disks, and certainly monohedral tilings by polyforms satisfy this requirement.

Let $x > 0$ and consider the annulus of tiles $\mathcal{A}(r + x, P) - \mathcal{A}(r, P)$. Observe that every boundary edge of $\mathcal{A}(r, P)$ is shared by some tile of $\mathcal{A}(r + x, P) - \mathcal{A}(r, P)$.

Table 4. Polyiamonds marked with bumps and nicks.

Base shape (see Figure 9)	H1	H2	H3	H4	H5	Admits more than 5 coronas	Timed out
1t-1	0	0	0	0	0	0	0
2t-1	0	0	0	0	0	0	0
3t-1	5	4	0	0	0	21	1
4t-1	10	0	0	0	0	67	1
4t-2	12	2	0	0	0	51	5
4t-3	21	0	0	0	0	57	1
5t-1	46	2	0	0	0	97	3
5t-2	4	0	0	0	0	29	0
5t-3	13	2	0	0	0	39	0
5t-4	26	3	0	0	0	36	1
6t-1	8	0	0	0	0	166	0
6t-2	28	0	0	0	0	215	7
6t-3	0	0	0	0	0	126	0
6t-4	21	0	0	0	0	192	0
6t-5	66	0	12	0	0	59	5
6t-6	42	2	0	0	0	283	3
6t-7	39	0	0	0	0	112	13
6t-8	28	0	0	0	0	159	10
6t-9	9	0	0	0	0	108	0
6t-10	17	0	0	0	0	116	0
6t-11	9	0	0	0	0	91	5
6t-12	12	0	0	0	0	96	1
7t-1	6	0	0	0	0	29	0
7t-2	25	2	0	0	0	42	6
7t-3	158	2	0	0	0	411	6
7t-4	21	0	0	0	0	83	6
7t-5	8	0	0	0	0	88	0
7t-6	56	0	0	0	0	143	11
7t-7	50	0	0	0	0	130	1
7t-8	23	0	0	0	0	126	1
7t-9	6	0	0	0	0	98	5
7t-10	9	0	1	0	0	79	0
7t-11	16	0	0	0	0	93	0
7t-12	8	0	0	0	0	89	1
7t-13	10	0	0	0	0	124	0
7t-14	29	0	0	0	0	53	0
7t-15	94	0	0	0	0	147	4
7t-16	8	0	0	0	0	81	0
7t-17	15	0	0	0	0	48	1
7t-18	11	0	0	0	0	99	1
7t-19	59	0	0	0	0	191	1
7t-20	32	0	0	0	0	104	2
7t-21	11	0	0	0	0	99	1
7t-22	54	0	0	0	0	127	5
7t-23	17	0	0	0	0	67	1

Therefore,

$$0 \le E(r, P) \le n \cdot [t(r + x, P) - t(r, P)].$$

Dividing this last inquality through by $t(r, P)$, letting $r \to \infty$, and applying the normality lemma implies that

$$\lim_{r\to\infty} \frac{E(r, P)}{t(r, P)} = 0. \tag{3–1}$$

T is combinatorially unbalanced (without loss of generality, suppose T has more nicks than bumps), so any patch $\mathcal{A}(r, P)$ must necessarily have more nicks than bumps, implying that there must be unmatched nicks on the boundary of $\mathcal{A}(r, P)$. Each copy of T contributes at least one unmatched nick. But, in light of equation (3–1), when r is sufficiently large there will be more unmatched nicks than boundary edges of $\mathcal{A}(r, P)$, a contradiction. □

Intuitively, it seems plausible that no unbalanced prototile can tile the plane due to the idea that the number of tiles in coronal patches would grow roughly quadriatically with respect to the number of coronas while the number of boundary edges would grow roughly linearly. However, it seems impossible to make this argument rigorous since the geometry of the patches is unpredictable. When the geometry of the coronal configurations is more predictable, it is possible to give an upper bound on the Heesch number of a combinatorially unbalanced tile [Mann 01].

4. Computer results

Because there are many ways to mark a particular tile with, say, bumps and nicks, a cluster of computers was needed for these searches (typically 100–150 CPUs were used in these searches). For example, if every side of the polyhex 5h-1 (Figure 7) is either unmarked or marked with a bump or a nick, then using Burnside's lemma and observing that interchanging bumps with nicks results in a combinatorially equivalently marked tile, it is seen that there are

$$\frac{1}{2} \cdot \frac{1}{4}(3^{22} + 2 \cdot 3^{11} + 3^{12}) = 3,922,743,168$$

distinct marking of this shape after factoring out the symmetries of that shape. The search algorithm, however, did not factor out symmetries which allowed some redundant cases to be checked, so the search space was actually much larger. If symmetries are not factored out in marking 5h-1 as previously discussed, there are $\lfloor \frac{1}{2} \cdot 3^{22} \rfloor = 15,690,529,804$ such tiles.

Tables 1–4 summarize our computer results for polyforms, each of whose edges are either unmarked straight edges or marked with a bump or a nick. The entries in the column labeled "shape" refer to the base polyform found in Figures 7–9 whose edges are marked in every possible way using bumps and nicks, directions, or colors. The columns "H1"–"H5" contain the numbers of edge-marked polyforms for each examined base shape with Heesch numbers 1 through 5. The column labeled "timed out" records the numbers of edge-marked polyforms for which the computer program took longer than 30 seconds to complete the exhaustive search for six coronas; again, among these timed out marked shaped there may be interesting examples. The column labeled "Admits more than 5 coronas" contains the numbers of edge-marked polyforms for each shape that can form more than 5 coronas; these edge-marked polyforms most likely admit tilings of the plane, but it is possible that there are examples among these marked shaped that have Heesch

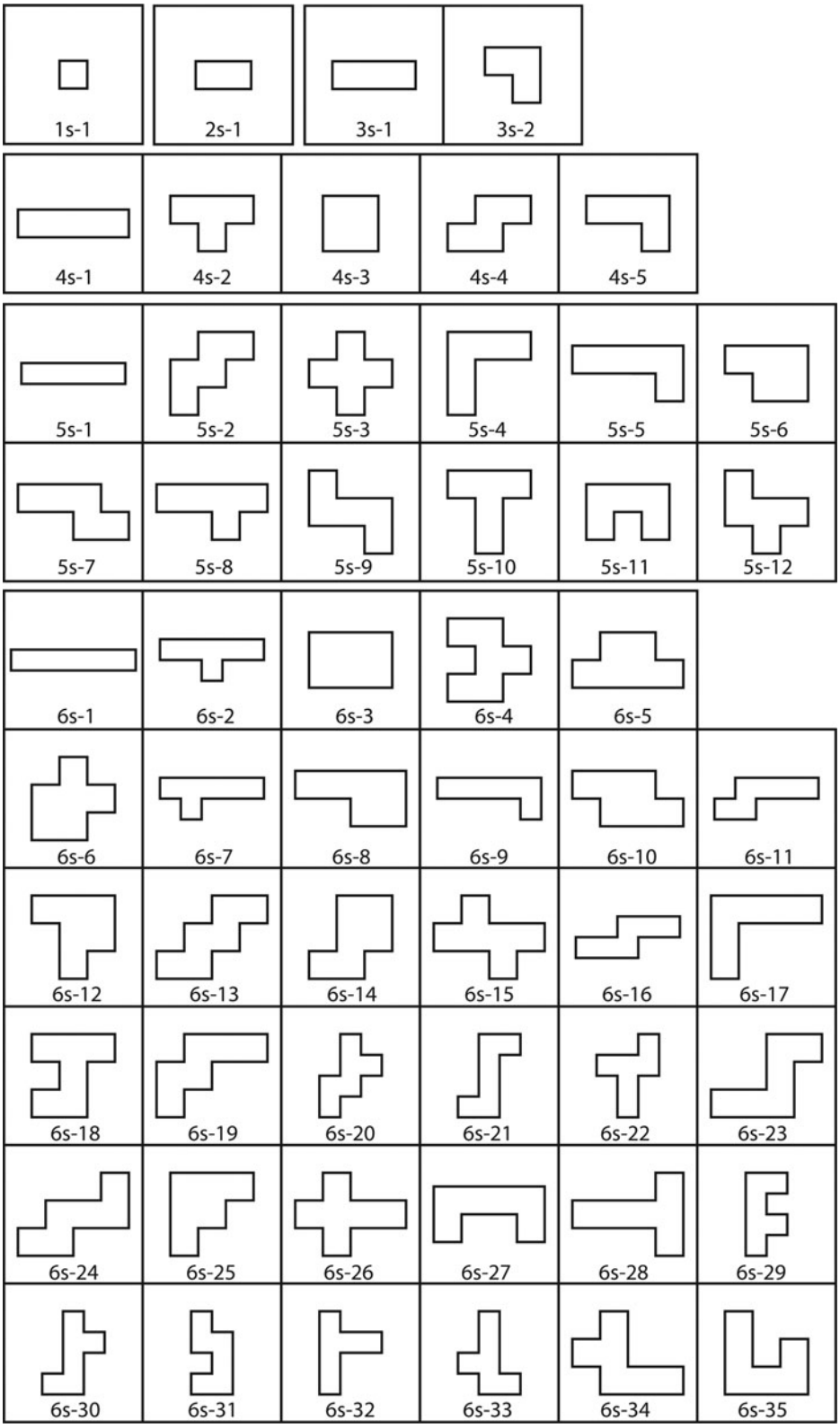

Figure 8. The examined polyominoes.

number greater than 5. These last two columns may contain a treasure trove of interesting examples, but extracting those examples is computationally intractable. A very large percentage of the tiles counted in those two columns tile the plane with low isohedral number. Identifying the isohedral number of a tile involves enumerating the partitions of all patches formed from 1, 2, 3, …, of these shapes, partitioning the boundaries of these patches into

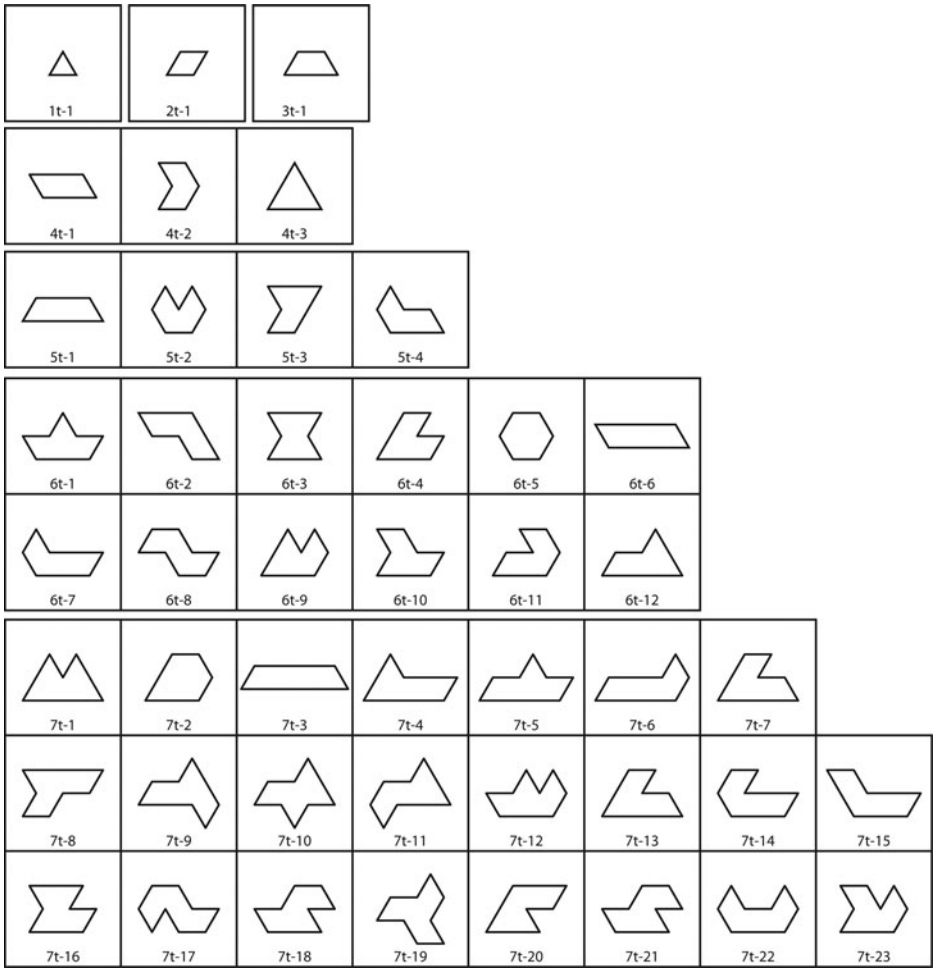

Figure 9. The examined polyiamonds.

3, 4, 5, or 6 arcs in every which way possible, and testing if the resulting partitions satisfy one of the 81 isohedrality conditions. For even small patches such computations become time-expensive. However, with more powerful computational hardware available in the future, the tiles populating these two columns could be revisited to separate the wheat from the chaff (Tables 1–7).

5. Interesting examples and observations

5.1. *Infinite families*

One goal of this exhaustive search through low-order edge-marked polyforms was to identify examples that can be generalized to examples with larger and larger Heesch numbers. While this search has not revealed families of tiles whose Heesch numbers grow without bound, a few infinite families of polyforms with large Heesch numbers have been found. The family of polyhexes in Figure 10(a) are called *hexapillars* because they remind the authors of caterpillars. The order-1 hexapillar has Heesch number 3 (ref Ammann), the hexapillars of orders 2 and 3 have Heesch number 4, and the hexapillars of order $n \geq 4$ all have Heesch number 5 [Mann 01, Mann 04]. The family of polyominoes in Figure 10(b) are called *polypillars*. The order-1 polypillar has Heesch number 1, the order-2 polypillar has Heesch number 2, and the polypillars with order $n \geq 3$ have Heesch number 3. Lastly, the polyiamonds of Figure 10(c) are called *grubs*. The grubs of orders 7 and 10 have Heesch number 3, and the grubs of order $n \geq 13$ have Heesch number 4. Figures 11–13 show maximal coronal configurations for representatives of each of these three families of tiles. It turns out that each of these three families exhibits a kind of behavior we call *forced grouping* that will be discussed shortly.

Table 5. Polyhexes marked with bumps and nicks and directions.

Base shape (see Figure 7)	H1	H2	H3	H4	H5	Admits more than 5 coronas	Timed out
1h-1	1412	48	72	0	0	6489	60
2h-1	38,102	88	0	24	0	731,275	112

Table 6. Polyhexes marked with colors.

Base shape (see Figure 7)	H1	H2	H3	H4	H5	Admits more than 5 coronas	Timed out
1h-1 (6 colors)	0	0	0	0	0	120	16
2h-1 (7 colors)	541	8	0	0	0	6453	50
3h-1 (4 colors)	5640	20	4	0	0	296,063	164
3h-2 (4 colors)	8404	16	0	0	0	188,212	122
3h-3 (8 colors)	1355	3	0	0	0	44,263	50
4h-1 (4 colors)	66,380	24	0	0	0	4,422,890	1227
4h-2 (4 colors)	45,207	36	8	0	0	1,558,916	243
4h-3 (4 colors)	18,542	25	1	0	0	383,396	101
4h-4 (4 colors)	61,750	0	0	0	0	1,021,438	88
4h-5 (4 colors)	30,074	6	0	0	0	1,042,778	455
4h-6 (5 colors)	3937	8	0	0	0	230,996	71
4h-7 (4 colors)	878,399	1568	0	0	0	35,665,927	12,543
5h-3 (3 colors)	66,690	12	0	0	0	896,001	325
5h-4 (3 colors)	90,178	60	0	0	0	1,018,315	330
5h-6 (3 colors)	29,785	54	0	0	0	460,057	45
5h-7 (3 colors)	34,733	110	0	0	0	654,043	165
5h-10 (3 colors)	67,860	4	0	0	0	409,704	193
5h-12 (4 colors)	40,870	124	48	0	0	1,222,723	61
5h-13 (4 colors)	57,621	115	28	0	0	1,316,624	105
5h-15 (4 colors)	51,808	110	0	0	0	1,823,716	132
5h-16 (4 colors)	12,919	30	0	0	0	474,881	85
5h-19 (3 colors)	28,125	243	0	0	0	544,544	39

Table 7. Polyhexes marked with directions.

Base shape (see Figure 7)	H1	H2	H3	H4	H5	Admits more than 5 coronas	Timed out
1h-1	42	24	0	0	0	152	14
2h-1	1333	36	0	0	0	1602	30
3h-1	7034	40	0	0	0	20,685	63
3h-2	9676	27	0	0	0	12,853	55
3h-3	2009	15	0	0	0	5502	23
4h-1	50,280	88	0	0	0	206,768	360
4h-2	31,594	56	8	0	0	87,648	59
4h-3	8012	29	2	0	0	30,465	40
4h-4	21,989	0	0	0	0	118,294	36
4h-5	13,430	4	0	0	0	33,674	267
4h-6	2772	12	0	0	0	18,855	54
4h-7	49,439	68	28	0	0	143,558	64

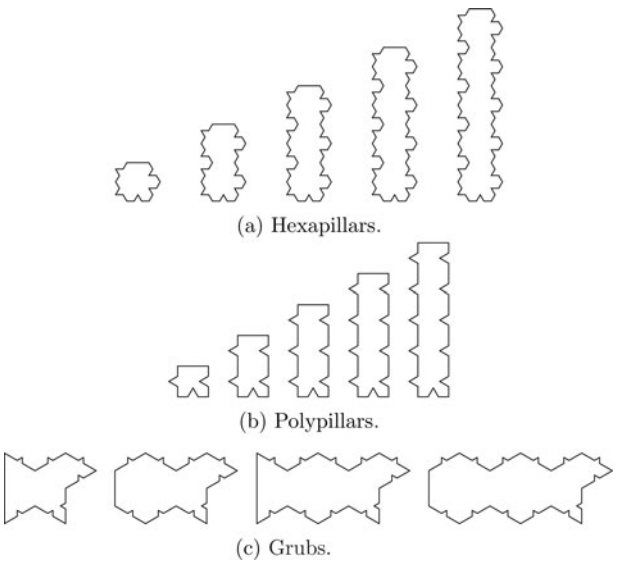

Figure 10. Infinite families with high Heesch numbers.

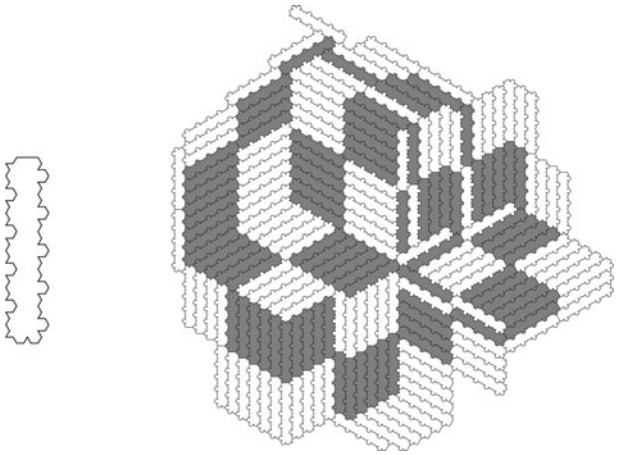

Figure 11. A Heesch number 5 polyhex.

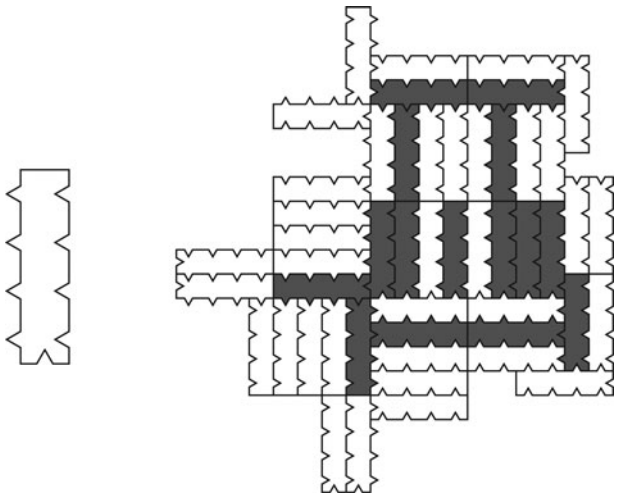

Figure 12. A 4-omino with Heesch number 3.

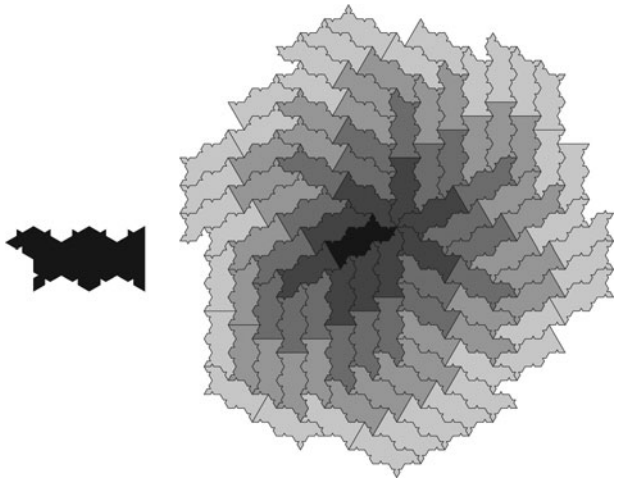

Figure 13. A 13-iamond with Heesch number 4.

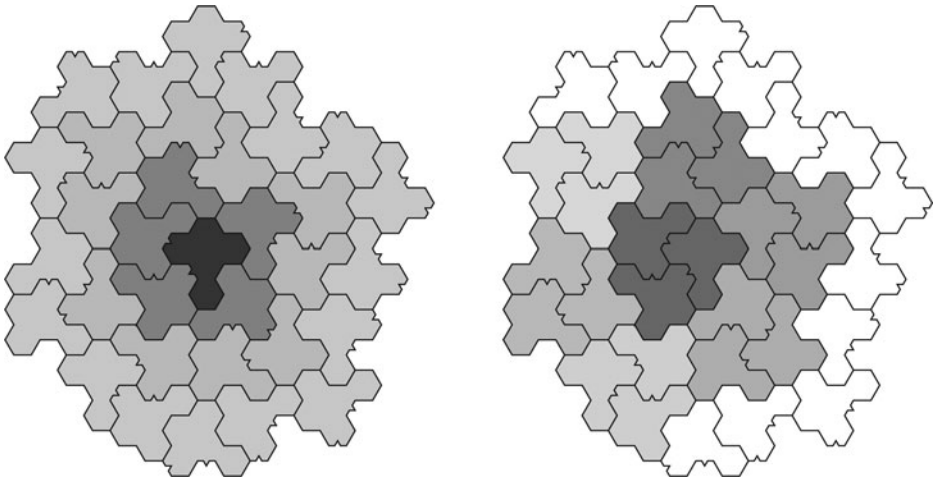

Figure 14. A prototile that exhibits forced grouping.

5.2. *Forced grouping*

Another mechanism that forces a tile to have large Heesch number is a behavior we will call *forced grouping*. Forced grouping occurs when the markings of the tile force copies to group together into certain kinds of larger configurations that do not tile the plane. This kind of behavior happens in a few distinct ways. For example, in Figure 14 is Heesch number 3 polyhex that, because of its edge markings, is forced to assemble into triangular shaped patches formed from three copies of the prototile. These triangular patches, thought of as a single prototile, have Heesch number 1, while the constituent polyhex itself has Heesch number 3. *Notice that the prototile of Figure 14 is not combinatorially unbalanced, so we cannot ascribe the fact that this prototile does not tile the plane to that mechanism.* Similar groupings can be forced using colors or directions, as in Figure 15.

More complicated clustering behavior has been observed. The Heesch number 5 family of prototiles represented by the polyhexes in Figure 11 is forced to assemble into a very large hexagonal shaped patch. By "forced," we mean that when assembling a patch of tiles from this polyhex, if a copy of the tile is placed in a way that does not match how it should fit into the large hexagonal structure, an "unfillable" spot is introduced on the boundary of the patch that cannot be continued. For example, the tile of Figure 16 has Heesch number 4, and one can see that almost 4 coronas of copies of that tile contained in the large forced hexagonal structure. Similar large forced patches formed from polyominoes (Figure 17) and from polyiamonds (the polyiomond of Figure 13 forms a large hexagonal structure) has been observed.

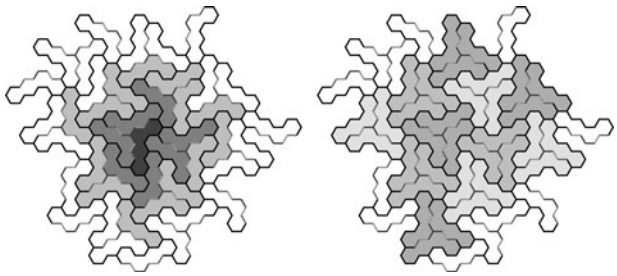

Figure 15. Grouping enforced by colored edges.

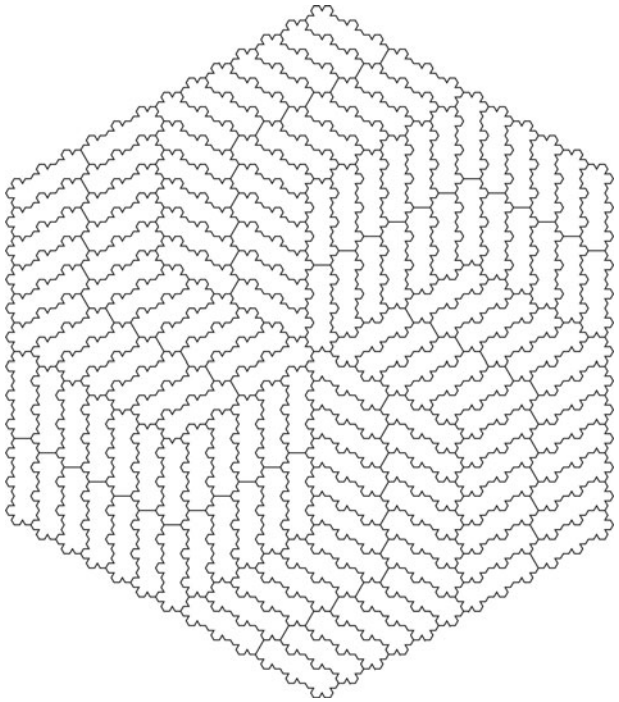

Figure 16. One of two maximal hexagonal patches admitted by the Heesch number 4 3-hex.

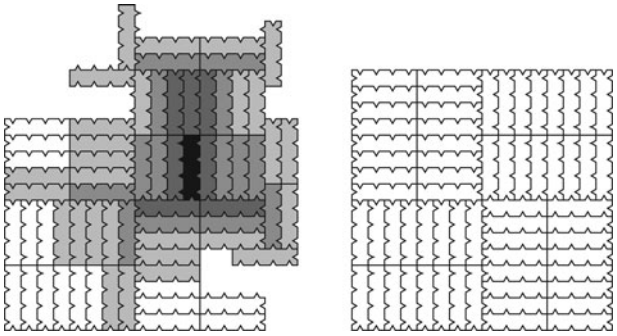

Figure 17. A Heesch number 3 polyomino that exhibits forced grouping into a large square-shaped patch.

5.3. *Complex behavior*

The program identified many examples with finite Heesch number for which there is no identifiable mechanism at play. These prototiles seem to be artifacts of the emergent complexity in the space of partial tessellations of the plane. For example, the prototile in Figure 18 has Heesch number 3 for no identifiable reason – *it just is*. It is examples like these that make us suspect that, even if no infinite families of prototiles with ever increasing Heesch numbers are identified (as we hoped for the families in Figure 10), examples with larger and larger Heesch numbers may exist, even though it may be essentially computationally impossible (at the moment) to find them.

6. Details of the algorithms

Polyhexes will be used to illustrate the data structures and algorithms, but these structures and algorithms are similar for polyiamons and polyominoes. The primary data structures utilized in the computer code correspond to the *input prototile* P (including its edge matching rules), an *orientation list* $\mathcal{L}(P)$ of all possible orientations of the prototile, and a *tiling board* B onto which the different orientations of the tile are placed.

- A prototile P is a list of base shapes (hexagons, squares, triangles). Each of these base shapes is an ordered list of edges with the top edge being first and counting clockwise around the hexagon. These edges contain the following information: bump/nick setting (–1 for nick, 0 for flat, or 1 for bump), color setting (a nonnegative integer), direction setting (–1, 0, or 1), and a fuse setting (a nonnegative integer). The fuse setting of n for an edge indicates that that particular edge of the base shape is fused to the base shape of index n in the list of base shapes comprising P, and $n = -1$ if that edge is a boundary edge of the prototile. See Figure 19. The rules embedded on each edge are enforced in the obvious way; two edges may meet if their bump/nick settings sum to 0, if their color settings are equal, and if their direction settings sum to 0.
- An orientation list for a prototile P is a list $\mathcal{L}(P)$ containing every feasible orientation of the prototile. This corresponds to cyclic permutations of the list of edges of each of the constituent base shapes comprising the prototile (along with reverse orderings for reflections). See Figure 20.
- Conceptually, the tiling board B is an arrangement of (initially) empty slots in a triangular, square, or hexagonal grid where the copies of the prototile will be placed. In computer memory, these slots are stored in an indexed array. Each element of this array will contain three types of information, the first being the coordinates of the corresponding place on

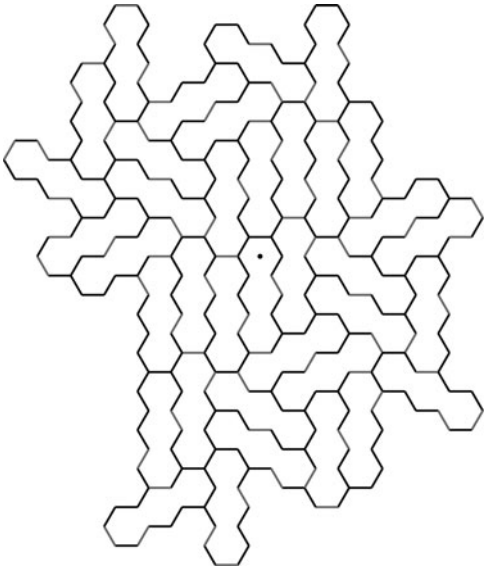

Figure 18. A Heesch number 3 polyhex.

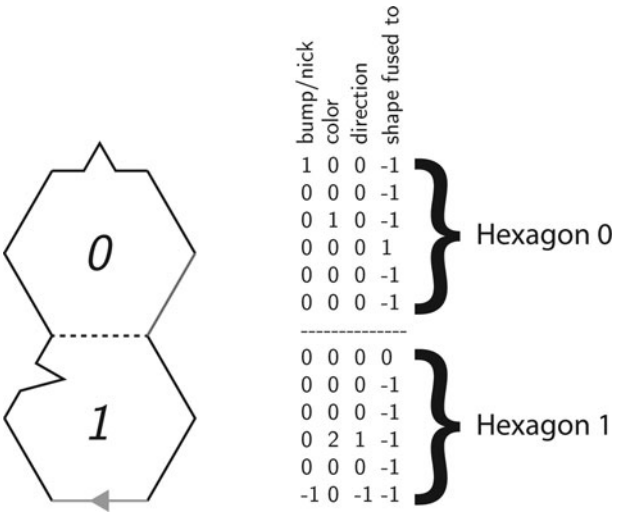

Figure 19. The prototile data structure.

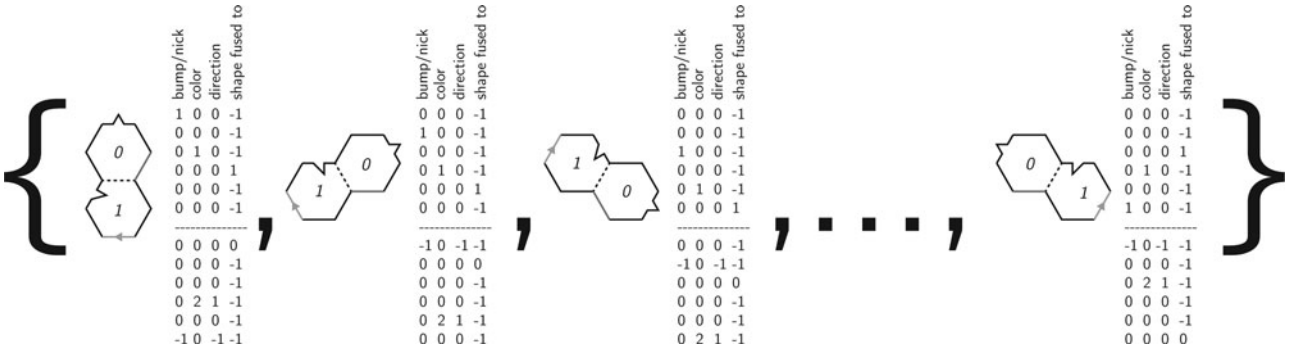

Figure 20. The orientation list data structure.

the board, the second being an array of two integers that stores the orientation index and index of the constituent base shape of that orientation that occupies this spot on the board, and third is a boolean variable that indicates whether or not this board location is occupied or empty. See Figure 21.

In Figure 21, it is seen that the indices of the board slots are assigned in a spiral fashion. The coordinates stored at each board slot are the coordinates of the center point of the board shape expressed in terms of a convenient basis. An algebraic cross-referencing function can then quickly determine the index in the board array given the coordinates (as opposed to simply searching through the array of board slots to find a given set of coordinates, which is more time consuming). Next, a brief overview of the tiling algorithm is given, and this overview will be followed by psuedocode.

The first thing that the algorithm does is to place the first orientation of the prototile so that this orientation's first base shape fills slot 0 of the board. The other base shapes comprising this orientation of the prototile will fill other board slots whose indices are determined by first using vector addition to identify the coordinates of these slots and then cross referencing the indices by the coordinates.

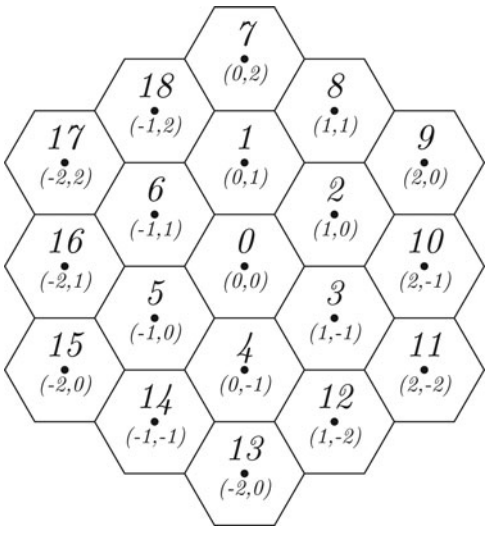

Figure 21. The indexing and coordinates of the tiling board for polyhexes.

After placing the first tile, the algorithm generates an array of board slot indices corresponding to those board slots that surround the first tile placed on the board. If the algorithm can successfully fill those slots with copies of the prototile, it will have found a first corona of tiles surrounding the centrally placed copy. After finding a first corona, the algorithm will generate an array of board slot indices corresponding to the board slots surrounding the first corona, and it will then go on to try and cover those board slots with copies of the prototile, and so on.

Each time the algorithm identifies the next board slot to cover with a copy of the prototile, it looks at the orientation array that is stored in that board slot. This array keeps an accounting of the orientation of the prototile that it has previously attempted to place there. This array has two components, one for the orientation index, the other for the base shape of the prototile to be placed in this board slot. For example, suppose that the prototile is a 4-hex, and at some point in the middle of the process of building coronas, the algorithm decides that the next board slot to be filled has index 27. So it looks up the orientation array at index 27 of the board and finds (2, 1), which indicates that orientation (2, 1) was already placed at this board slot previously but that copy of the prototile has subsequently been removed from the board. The array (2, 1) indicates "orientation index = 2, base shape index = 1." The algorithm will first increment that array (in the natural way) and check if the next orientation fits, and if not it increments and checks again, and so on, until it either successfully places the tile or it exhausts the possible orientations of the tile.

$$(2,1) \xrightarrow{\text{increment}} (2,2) \xrightarrow{\text{if it did not fit}} (2,3) \xrightarrow{\text{if it still did not fit}} (3,0) \xrightarrow{\cdots}$$

If the algorithm exhausts the possible orientations of the prototile to be placed at a particular board slot, that means that the prototile simply does not fit at that spot, and so the previously placed tile must be removed from the board. At this point, the process of attempting to place a tile resumes at the new first empty board slot (which was just made available by the removal of the copy of the tile from the board).

Any time the algorithm fills all of the slots that surround the previously generated corona, it increments a counter to keep track of hvvow many coronas it has generated. Also, when tiles are removed from the board, it is checked to see if the tile removed was part of the previous corona, in which case the program decrements the number of coronas found so far and it focuses on filling the slots of the board surrounding the previous corona.

Each time the algorithm is checking to see if a particular orientation of the tile can be placed at certain board slot, it checks things in the obvious way: First, it checks to see if this particular orientation of the tile being placed at this particular spot fits on the board (i.e., it makes sure that none of the constituent base shapes of this copy of the prototile will be forced to occupy slots on the board that are already occupied). This is done using vector addition and cross referencing the indices of the board slots that would have to be occupied by that tile were it placed on the board. Next, if this particular orientation of the tile does fit on the board, the edge matching rules are checked to see if they are compatible with what is already placed on the board. For each edge of the prototile being placed, vector addition determines if that edge is being placed against an edge that is already placed on the board. If so, it checks that the corresponding edge matching rules are compatible.

Next, psuedocode for the algorithm that can determine the Heesch number of a polyhex with finite Heesch number is presented (along with descriptions of a few functions needed in the psuedocode).

- **PlaceTile(int slotIndex)**. This function attempts to place a copy of the prototile at slot "slotIndex" on the board. It returns TRUE if it successfully places a tile, FALSE otherwise.
- **GenerateBoundarySlots(board).** This function generates an array of indices of the annulus of empty slots surrounding the current board configuration.
- **FindFirstEmptySlot().** This function finds the first index of an unfilled boundary board slot.
- **RemoveTile().** This function removes the last tile placed on the board.

Algorithm 1 Calculate the Heesch Number of a Tile

```
Require: An integer M = the maximum number of coronas to form, a prototile P.
Ensure: The Heesch number of P (if less than M)
  numCoronasFormed = 0
  maxNumCoronasFormed = 0
  place L(P)[0] at tiling board index 0
  BoundarySlots(0) = GenerateBoundarySlots(board)
  while −1 < numCoronasFormed < M do
    FirstEmpty = FindFirstEmptySlot()
    DidItFit = PlaceTile(FirstEmpty)
    if DidItFit = TRUE then
      if BoundarySlots[i] is occupied for every i then
        numCoronasFormed++
        if numCoronasFormed > maxNumCoronasFormed then
          maxNumCoronasFormed++
          BoundarySlots(numCoronasFormed) = GenerateBoundarySlots(board)
        end if
      end if
    else
      if every orientation of the tile has been tested at board slot FirstEmpty then
        removeTile()
        if the attempted orientations at first board slot in BoundarySlots(numCoronasFormed) are exhausted then
          delete BoundarySlots(numCoronasFormed)
          numCoronasFormed = numCoronasFormed - 1
        end if
      end if
    end if
  end while
  print maxNumCoronasFormed
```

References

[Berger 66] R. Berger. "The Undecidability of the Domino Problem." *Mem. Amer. Math. Soc. No.* 66 (1966), 72.

[DeWeese 08] M. DeWeese. "Personal Communication." (2008).

[Fontaine 91] A. Fontaine. "An Infinite Number of Plane Figures with Heesch Number Two." *J. Combin. Theory Ser. A* 57(1) (1991), 151–156.

[Goodman-Strauss 99] C. Goodman-Strauss. "A Small Aperiodic Set of Planar Tiles." *Eur. J. Combin.* 20(5) (1999), 375–384.

[Goodman-Strauss 00] C. Goodman-Strauss. "Open Problems in Tiling." Available online http://comp.uark.edu/~strauss/papers/survey.pdf, 2000.

[Grünbaum and Shephard 87] B. Grünbaum and G. C. Shephard. *Tilings and Patterns*. New York, NY: W. H. Freeman and Company, 1987.

[Grünbaum and Shephard 98] G. Grünbaum and G. C. Shephard. "Some Problems on Plane Tilings." In *Mathematical Recreations*, edited by D. A. Klarner, pp. 167–196. Dover, 1998.

[Heesch 68] H. Heesch. *Reguläres Parkettierungsproblem*. Arbeitsgemeinschaft für Forschung des Landes Nordrhein-Westfalen, Heft 172. Cologne: Westdeutscher Verlag, 1968.

[Mann 01] C. Mann. "Heesch's Problem and Other Tiling Problems." PhD thesis, University of Arkansas, 2001.

[Mann 04] C. Mann. "Heesch's Tiling Problem." *Amer. Math. Monthly* 111(6) (2004), 509–517.

[Myers 12] J. Myers. "Polyomino, Polyhex and Polyiamond Tiling." Available online http://www.polyomino.org.uk/mathematics/polyform-tiling/, 2012.

[Patitz 14] M. Patitz. "Self-Assembly." Available online http://self-assembly.net, 2014.

[Pelesko 07] J. A. Pelesko. *Self Assembly: The Science of Things That Put Themselves Together*. Chapmann and Hall/CRC, 2007.

[Penrose 80] R. Penrose. "Pentaplexity: A Class of Nonperiodic Tilings of the Plane." *Math. Intelligencer* 2(1) (1979/1980), 32–37.

[Rhoads 05] G. C. Rhoads. "Planar Tilings by Polyominoes, Polyhexes, and Polyiamonds." *J. Comput. Appl. Math.* 174(2) (2005), 329–353.

[Robinson 71] R. M. Robinson. "Undecidability and Nonperiodicity of Tilings of the Plane." *Inventiones Math* 12 (1971), 177–909.

[Socolar and Taylor 12] J. E. S. Socolar and J. M. Taylor. "Forcing Nonperiodicity with a Single Tile." *Math. Intelligencer* 34(1) (2012), 18–28.