Tilings and Amalgamations

Thesis

UNRESTRICTED

Tilings and Amalgamations

William Jonathan Griffith Wingate, MA
Faculty of Mathematics
The Open University

A thesis submitted for the degree of Master of Philosophy

December 1988

Date of submission: 19 December 1988
Date of award: 20 March 1989

Abstract

This thesis investigates tilings of the Euclidean Plane and their amalgamations. For any tiling an *amalgamation* is a tiling produced by joining together tiles of the original tiling. This definition can be extended to cover any suitable adjacency structure, such as a graph.

The first chapter of the thesis reviews some of the basic concepts and results in the theory of tilings. Chapter 2 introduces amalgamations, both of tilings and of infinite graphs. Chapter 3 discusses *tesseral arithmetic*, and shows how the theory of amalgamations can be used to produce addressing systems of the plane.

The second part of the thesis concentrates on classifying and enumerating amalgamators. In chapter 4, we list the possible types of amalgamation of each of the eleven Laves nets. In chapter 5 an algorithm to enumerate a particular class of amalgamations is developed, and the results of running this on a computer are presented. Chapter 6 contains some theoretical results about *tiling hierarchies*, sequences of tilings produced by successive amalgamations.

Acknowledgements

I would like to thank the following for their help in the preparation of this thesis:

- my supervisor, Dr. Fred Holroyd, for his constant help and patience;
- Dr. Marjorie Senechal, for giving up a week to discuss the work, as well as the authorities of Smith College, Northampton, MA, and the Five College Applied Mathematics Committee for their hospitality and generosity.

I would also like to acknowledge the generous financial support provided by the Open University.

Contents

List of Figures

List of Tables

Part I

Tilings, Amalgamations and Tesserals

Chapter 1

Tilings

We review some of the basic definitions and properties of tilings. For a very complete survey of the theory of tilings and patterns see [6]. We have not been able to discover a previous reference for the result given as theorem 1.19 and its corollary.

1.1 Definitions

Throughout this thesis we shall be concerned solely with tilings of the Euclidean Plane, $\mathbf{E}^2$, although some of the ideas presented are capable of generalisation to higher dimensional Euclidean space (and possibly other spaces too).

Definition 1.1 *A* tiling *of the Euclidean plane is a set*

$$T = \{t_i | i = 1, 2, 3, \ldots\}$$

of closed topological discs called tiles, *whose interiors are pairwise disjoint, and whose union covers the plane.*

Definition 1.2 *If T is a tiling, and X is any subset of T, then we denote by $[X]$ the region of $\mathbf{E}^2$ which is the union of tiles belonging to X. If $[X]$ is itself a topological disc, then it is said to be a* molecule *of T. The set of all molecules of T is denoted $\langle T \rangle$. A molecule consisting of n tiles of T will be called an n-*molecule.

We now consider various conditions which can be placed on a tiling.

Condition 1 *A tiling is* locally finite *if any bounded subset of* E^2 *meets only finitely many tiles of* T.

Condition 2 *A tiling is* simply adjacent *if the intersection of any two tiles is a simplex (which in the case of a two-dimensional tiling may be a point, a line segment, or empty).*

All the tilings we will consider will satisfy these two criteria unless otherwise stated. In particular, this implies that every molecule is the union of finitely many tiles of T.

Definition 1.3 *We define a* vertex *of* T *to be a point common to three or more tiles, and an* edge *of* T *to be the closure of one of the arcs into which the vertices partition the boundaries of the tiles.*

Hence if T is simply adjacent, then an edge of T is the intersection of two tiles; otherwise the intersection of two tiles will in general be a collection of edges and vertices.

Definition 1.4 *We define the edges and vertices of a tile* t *of* T *to be the edges and vertices of* T *contained in* t. *A tile with* r *edges is an* r*-tile.*

Condition 3 *A tiling* T *is* normal *if there exist parameters* u, U *such that every tile contains a disc of radius* u, *and is contained in a disc of radius* U.

Condition 4 *A tiling* T *is* polygonal *if all the tiles are polygons, so that all edges are polygonal arcs.*

To avoid confusion we will speak where necessary of the *sides* and *corners* of a polygon. This is necessary as a tiling may well have different numbers of edges and sides: consider the tilings in figure 1.1 which are respectively tilings by dodecagons which are 4-tiles, and by quadrilaterals which are 6-tiles.

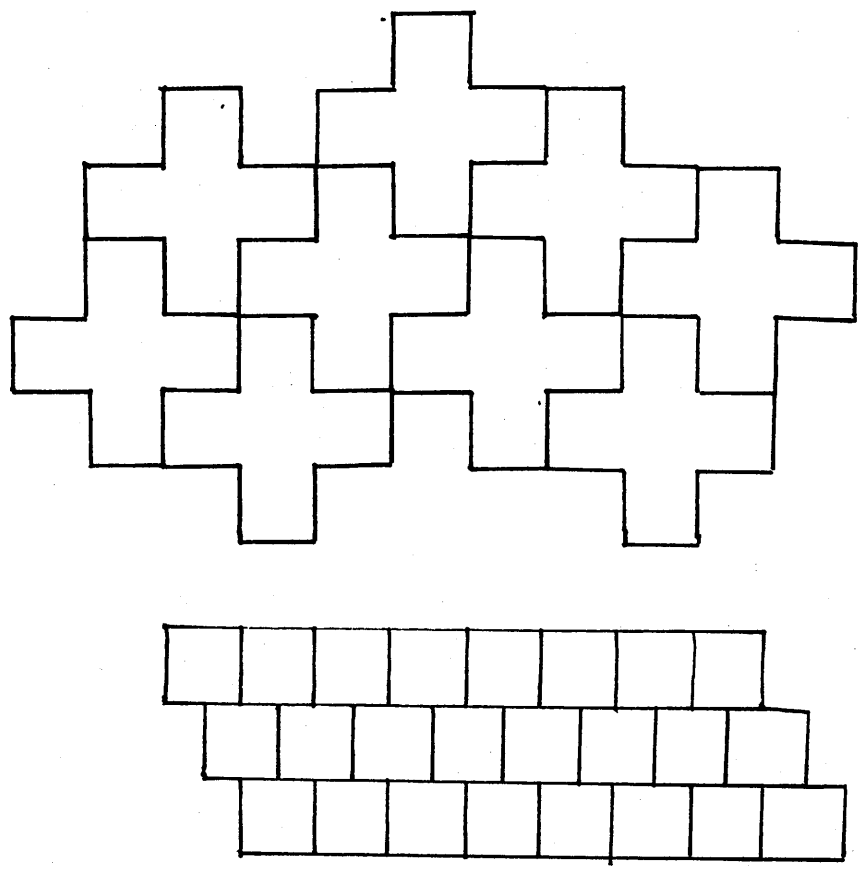


Figure 1.1 "Latin cross" and "brick wall" tilings

Condition 5 *A tiling T is* straight *or* proper *if all edges are straight line segments.*

Condition 6 *A tiling T is* convex *if all tiles are convex.*

Clearly convex $\Rightarrow$ straight $\Rightarrow$ polygonal.

Condition 7 *A straight tiling is* edge-to-edge *if the intersection of any two tiles is either empty, or a corner, or a side of each.*

In an edge-to-edge tiling an r-tile is always an r-gon, and *vice-versa.*

Definition 1.5 *The* 1-skeleton *or* net *of T, $\Gamma(T)$, is the graph formed by the edges and vertices of T: the degree of a vertex is then its degree in this graph. The net of a tiling will always be 3-connected.*

Definition 1.6 *A tile t is said to be of* type $v_1.v_2.\ldots.v_r$ *if it is an r-tile and if its vertices taken in cyclic order have degrees $v_1, v_2, \ldots, v_r$.*

Condition 8 *A normal tiling T is* homogeneous *of type $[v_1.v_2.\ldots.v_r]$ if each tile is of type $v_1.v_2.\ldots.v_r$.*

Condition 9 *A tiling T is* monohedral *if all tiles of T are congruent.*

Definition 1.7 *A vertex v of T is said to be of* type $p_1.p_2.\ldots.p_q$ *if it is surrounded by a p_1-tile, a p_2-tile, ..., a p_q-tile.*

Condition 10 *A tiling T is* semi-regular *of type $(p_1.p_2.\ldots.p_q)$ if each vertex is of type $p_1.p_2.\ldots.p_q$.*

Remark 1.8 *It is interesting to note that conditions 1,3–7,9 are metric conditions, whereas conditions 2,8,10 are combinatorial conditions.*

1.2 Transformations of tilings

1.2.1 Combinatorial mappings

We now define the concept of a (combinatorial) isomorphism between tilings. We choose to define this in terms of molecules, as this simplifies matters later.

Definition 1.9 *Let T, S be tilings. Then a* combinatorial isomorphism *between T and S is a bijection $\phi : \langle T \rangle \to \langle S \rangle$ which satisfies the following conditions:*

1. *$\phi|_T$ (the restriction of ϕ to T) is an adjacency preserving bijection between T and S;*
2. *For any molecule $[X]$ of T,*

$$\phi([X]) = [\phi(x) | x \in X]$$

Other definitions are possible: for example [6] gives the following:

Definition 1.10 *A combinatorial isomorphism between two tilings T and S is an inclusion-preserving map between $\mathcal{E}(T)$ and $\mathcal{E}(S)$, where $\mathcal{E}(T)$ denotes the set of* elements *of T—the tiles, edges and vertices.*

Proposition 1.11 *These definitions are equivalent, in the sense that they define mappings $\langle T \rangle \to \langle S \rangle$ and $\mathcal{E}(T) \to \mathcal{E}(S)$ which, when restricted to $T \to S$ define the same class of mappings.*

Proof (sketch): Any vertex in a tiling can be specified uniquely and unambiguously by the molecule consisting precisely of those tiles which meet at that vertex. An edge can similarly be specified by the two tiles in whose intersection it lies: if the tiling is not simply adjacent it is necessary to give the endpoints as well. It is thus possible to extend an isomorphism in the sense of 1.9 to act on the vertices and edges of a tiling, and conditions 1 and 2 ensure that it is inclusion preserving. The converse follows by considering the edges and vertices lying in a particular molecule. □

An *automorphism* of a tiling T is an isomorphism $\phi : \langle T \rangle \to \langle T \rangle$.

Definition 1.12 *A* symmetry *of a tiling is an isometry of the plane which preserves T, in the sense that it maps tiles to T to tiles of T.*

Clearly a symmetry of T induces an automorphism of T: we call such an automorphism a *symmetric automorphism* or *combinatorial symmetry.* (In general, an isomorphism between two tilings which is induced by an isometry of the plane will be called a *symmetric isomorphism.*) If all the automorphisms of a tiling are symmetric, then the tiling is said to be *maximally symmetric.*

The combinatorial symmetry and automorphism groups of T will be denoted $\Sigma(T)$ and $A(T)$ respectively.

Remark 1.13 *The symmetry group of a tiling must clearly be isomorphic to either a finite cyclic or dihedral group, or one of the seven frieze groups, or one of the seventeen plane crystallographic groups (see, e.g. [16]).*

Remark 1.14 *We shall frequently speak of an isomorphism $\phi : T \to S$ when in fact we mean the more general isomorphism defined above; similarly we may also confuse the symmetry group of T (which is a group of affine transformations of the plane) and the group of combinatorial symmetries of T (which is an infinite permutation group).*

Definition 1.15 *If T is a tiling on which the action of $\Sigma(T)$ is transitive, then T is said to be* isohedral.

Remark 1.16 *From now on we shall be concerned mainly with isohedral tilings.*

Definition 1.17 *Two molecules $[X]$ and $[Y]$ of T are said to be* equivalent *under a subgroup G of $A(T)$ there is an automorphism of T in G which maps $[X]$ to $[Y]$.*

Definition 1.18 *If T, S are tilings, then a bijection $\phi : T \to S$ between T and S which also satisfies*

$$\phi\Sigma(T)\phi^{-1} = \Sigma(S).$$

is said to be compatible *with the symmetry groups of S and T. A compatible isomorphism is called an* isomerism.

1.2.2 Topological mappings

If T, S are tilings, let f be a homeomorphism of the plane which takes T to S. Then it is clear, from the continuity of f, that f induces an isomorphism between T and S, which we will denote $\tilde{f}$.

If $S = T$, then f will be called a *topological symmetry* of T, and in this case $\tilde{f}$ is an automorphism of T. The group of topological symmetries of T will be denoted $H(T)$, so that $\tilde{H}(T) \leq A(T)$.

Let $E(T)$ be the subgroup of $H(T)$ consisting of the isometries in $H(T)$. Then $\tilde{E}(T) = \Sigma(T)$, and in this case $\sim$ is clearly a bijection between $E(T)$ and $\Sigma(T)$. If f is a homeomorphism from the plane of T to the plane of S which is such that

$$fE(T)f^{-1} = E(S)$$

then f is said to be a *homeomerism.*

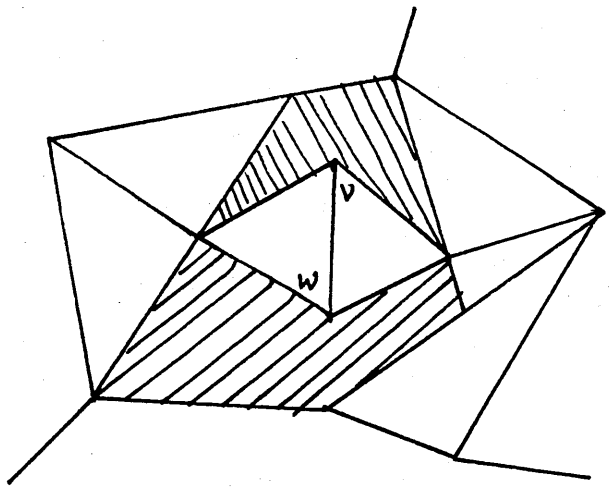
There is a strong connection between topological and combinatorial mappings of tilings: see statements 4.1.1 of [6], which states that for normal tilings, the concepts of topological and combinatorial equivalence coincide.

1.2.3 Transformations of the net of a tiling

The final type of transformation of a tiling which is of interest is the automorphism group of the net of the tiling, $G(T)$.

Clearly, any automorphism of T induces an automorphism of the graph $\Gamma(T)$. The converse, however, is not necessarily true. Consider, for example, a tiling which includes the configuration shown in figure 1.2 Transposing v and w gives an automorphism of $\Gamma(T)$ which does not act on the tiles of T (consider what happens to the shaded tiles). Notices that in this case the tiling is not simply adjacent, since the two shaded tiles intersect at a pair of vertices, x and z. In fact we have the following result.

Theorem 1.19 *If T is a simply adjacent tiling, then $G(T)$ acts on T, in the sense that if $v_1, v_2, \ldots, v_n$ are the vertices of a tile in T, then for any $\gamma \in G(T), \gamma(v_1), \gamma(v_2), \ldots, \gamma(v_n)$ are also the vertices of a tile in T.*



v and w both have degree 3

Figure 1.2

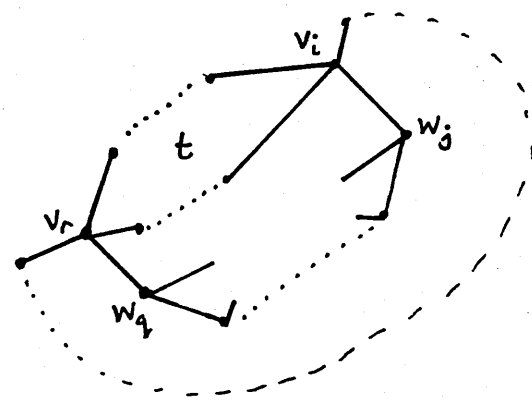


Figure 1.3

Proof: Let T be as above. Suppose that there exists a $\gamma \in G(T)$ and vertices $v_1, \ldots, v_n$ of T such that $v_1, \ldots, v_n$ are the vertices of a tile $t \in T$, but $\gamma(v_1), \ldots, \gamma(v_n)$ are not.

Now, $v_1, \ldots, v_n$ form a cycle in $\Gamma(T)$, and hence so do $\gamma(v_1), \ldots, \gamma(v_n)$: thus they also define a molecule of T. A finite number of vertices lie inside this cycle (in $\Gamma(T)$), as we are assuming that Condition 1 holds, and they will be the images under γ of vertices $w_1, \ldots, w_k$, say.

Now, $\Gamma(T)$ is a plane graph, and so the w_j's are adjacent only to v_i's and w_j's. Hence there is some cycle C in $\Gamma(T)$ consisting of v_i's and w_j's such that all v_i's and w_j's lie inside this cycle. The only way this could not be the case is if the subgraph induced by the v_i's and w_j's contained a bridge: this cannot be the case, however, as they are the vertices of the tiles making up a general tile, and hence this subgraph must be 2-edge-connected.

Some part of this cycle (see figure 1.3) must be of the form

$$\ldots, v_i, \quad \overbrace{w_j, \ldots, w_q}^{\text{one or more } w_j\text{'s}} \quad , v_r, \ldots$$

Since $w_j, \ldots, w_z$ are not adjacent to any vertices outside C, this part of the

cycle must form part of the boundary of a tile, which is adjacent to t at both v_i and v_q, but not between them. So T is not simply adjacent. □

Corollary 1.20 *The net of a simply adjacent tiling has only one normal plane embedding, up to equivalence. Equivalently, if a 3-connected infinite planar graph admits a normal plane embedding with no infinite face, then this embedding is unique up to isomorphism of the tiling it induces.*

Proof. Two embeddings of the graph will be related by a graph automorphism. But, from above, this is a tiling isomorphism. □

1.3 Classification of Tilings

The coarsest possible classification of isohedral tilings is to classify them up to isomorphism. It is easy to see that if two homogeneous tilings are of the same type, then they will be isomorphic, and it is well known that there are eleven possible types, corresponding to the eleven Laves nets. The eleven types are: $[3^6]$, $[3^4.6]$, $[3^3.4^2]$, $[3^2.4.3.4]$, [3.4.6.4], [3.6.3.6], [3.12.12], $[4^4]$, [4.6.12], $[4.8^2]$, $[6^3]$. For a proof of this result see result 4.3.1 of [6]. Two tilings which are isomorphic to the same Laves net are said to be of the same *topological type.*

In [5] Grünbaum and Shepherd classified all of the isohedral tilings of the plane up to isomerism. Their result was as follows:

Theorem 1.21 (Grünbaum and Shepherd) *Up to isomerism there are 93 types of marked isohedral tiling of the plane (that is tilings in which the tilings may be marked in some finitary way), 81 of which can be realised as unmarked types.*

In fact, they defined two isohedral tilings to have the same isohedral type if and only if their *incidence symbols* differed trivially. In chapter 7 of [6] they show that this is equivalent to classification up to homeomerism (or isomerism).

The 93 isohedral types, presented as marked tilings, rather than shaped tilings, are listed in appendix A.

Remark 1.22 *We shall make heavy use of this classification in the sequel: when we speak of the isohedral type of an isohedral tiling, we will mean the type as in the paper cited above.*

One other method of classification of isohedral tilings is of interest; namely classification by *henomeric* type: the concept of henomerism was introduced by Grünbaum and Shepherd for the purpose of classifying patterns, and is defined as follows:

Definition 1.23 *Two tilings (patterns) are said to have the same* pattern type, *or to be* henomeric, *if they have the same symmetry group, the same induced subgroup (tile group), and the same set of motif-transitive subgroups. The "motifs" here are interpreted as tile interiors.*

In the table below we classify the 93 isohedral tilings by pattern type (using information given in [6]). Marked tilings are indicated by italics.

It is interesting to note how tilings with different topological types may still be of the same pattern type.

Table 1.1 Classification of isohedral tilings by pattern type

Pattern type	Isohedral type(s)	Symmetry group	Tile group	Topological type(s)
01	1,41	p1	c1	$[3^6],[4^4]$
02	2,3,43,44	pg	c1	$[3^6],[4^4]$
03	42	pm	c1	$[4^4]$
04	64		d1	$[4^4]$
05	22,45,83	cm	c1	$[3^3.4^2],[4^4],[6^3]$
06	12,14,68		d1	$[3^6],[4^4]$
07	4,23,46,47,84	p2	c1	$[3^6],[3^3.4^2],[4^4],[6^3]$
08	8,57		c2	$[3^6],[4^4]$
09	5,6,25,27,51,52,53,86	pgg	c1	$[3^6],[3^3.4^2],[3^2.4.3.4],[4^4],[6^3]$
10	9,59		c1	$[3^6],[4^4]$
11	24,49,50,85	pmg	c1	$[3^3.4^2],[4^4],[6^3]$
12	58		c2	$[4^4]$
13	13,15,66,69		d1	$[3^6],[4^4]$
14	*48*	pmm	c1	$[4^4]$
15	*65*		d1	$[4^4]$
16	72		d2	$[4^4]$
17	54,78	cmm	c1	$[4^4],[4.8^2]$
18	*60*		c2	$[4^4]$
19	26,67,91		d1	$[3^3.4^2],[4^4],[6^3]$
20	17,74		d2	$[3^6],[4^4]$
21	7,33	p3	c1	$[3^6],[3.6.3.6]$
22	10		c3	$[3^6]$
23	30,38	p31m	c1	$[3.4.6.4],[3.12^2]$
24	*89*		c3	$[6^3]$
25	16,36		d1	$[3^6],[3.6.3.6]$
26	18		d3	$[3^6]$
27	*87*	p3m1	c1	$[6^3]$
28	*35*		d1	$[3.6.3.6]$
29	*19*		d3	$[3^6]$

Table 1.2 Classification of isohedral tilings by pattern type cont.

Pattern type	Isohedral type(s)	Symmetry group	Tile group	Topological type(s)
30	28,55,79	p4	c1	$[3^2.4.3.4]$,$[4^4]$,$[4.8^2]$
31	61		c2	$[4^4]$
32	62		c4	$[4^4]$
33	56,81	p4g	c1	$[4^4]$,$[4.8^2]$
34	*63*		c4	$[4^4]$
35	29,71		d1	$[3^2.4.3.4]$,$[4^4]$
36	73		d2	$[4^4]$
37	*80*	p4m	c1	$[4.8^2]$
38	82		d1	$[4.8^2]$
39	*70*		d1	$[4^4]$
40	*75*		d2	$[4^4]$
41	76		d4	$[4^4]$
42	21,31,39,88	p6	c1	$[3^4.6]$,$[3.4.6.4]$,$[3.12^2]$,$[6^3]$
43	34		c2	$[3.6.3.6]$
44	90		c3	$[6^3]$
45	11		c6	$[3^6]$
46	77	p6m	c1	$[4.6.12]$
47	*92*		d1	$[6^3]$
48	32,40		d1	$[3.4.6.4]$,$[3.12^2]$
49	37		d2	$[3.6.3.6]$
50	93		d3	$[6^3]$
51	20		d6	$[3^6]$

Chapter 2

Amalgamations and amalgamators

We review the basic definitions of amalgamations and amalgamators, and present new work extending the theory to "infinite trees".

2.1 General amalgamations

Definition 2.1 *The most general definition of an* amalgamation A *of a set* S*, is simply a partition of* S*: in other words a subset of the power set* $\mathcal{P}S$ *which satisfies*

1. $a, a' \in A,\ a \neq a' \Rightarrow a \cap a' = \emptyset$;
2. $\bigcup A = S$.

An amalgamator *is then a bijection* $\alpha : S \to A$*, where* A *is an amalgamation of* S.

If $|a| > 1$ *for each* $a \in A$*, then* A, α *are said to be* strict.

If for some integer k*,* $|a| = k, \forall a \in A$*, then* A, α *are said to be* k*-uniform.* k *is called the* aperture. *If* $k = 1$ *then* A, α *are said to be* trivial.

Remark 2.2 *Note that a non-trivial amalgamator can exist only if* S *is infinite. For a given amalgamation,* A*, an amalgamator* $\alpha : S \to A$ *(which, if it exists, may not be unique) should be distinguished from the unique* inclusion function, $\iota : S \to A$ *which maps each* $s \in S$ *to the unique* $a \in A$ *which contains* s*: this map is surjective but not injective, unless* A *is trivial.*

In itself, this definition is not very interesting. An amalgamation is so far simply a partition, and an amalgamator is is some sense the inverse of a surjection.

The theory becomes much more interesting if we impose some sort of structure on the set S which can be inherited by an amalgamation of S, and impose further conditions on amalgamators. We can then try to classify amalgamations of S, and investigate the existence of amalgamators. If we require α to be an isomorphism between S and its amalgamation then we can investigate *hierarchies*, $\{S, \alpha(S), \alpha^2(S), \ldots\}$ of amalgamations.

As an example, let $\langle S, R\rangle$ be a relational structure with a single binary relation R which is symmetric and non-reflexive. Denote by R^t the transitive extension of R. We say that a subset X of S is *connected* if

$$\forall x, x' \in X, x(R|X)^t x'$$

We will assume that S is infinite, but that for each $s \in S$,

$$N_R(s) \stackrel{\text{def}}{=} \{x \in S | sRx\}$$

is finite.

Now we define an *amalgamation* A of S to be a partition of S consisting of finite, connected sets.

We define a relation R^* on A as follows:

$$\forall a, a' \in A,\ aR^*a' \Leftrightarrow a'R^*a \Leftrightarrow \exists s \in a, s' \in a' \text{ s.t. } sRs'$$

This gives us a new relational structure $\langle A, R^*\rangle$.

We also require an amalgamator α to satisfy the additional condition

$$\forall s, s' \in S, \alpha(s)R^*\alpha(s') \Leftrightarrow sRs'$$

Consider now the possible amalgamators on S. We have the following proposition:

Proposition 2.3 *If C is a connected component (maximal connected set) of S, and $s, s' \in C$, then $\alpha(s), \alpha(s')$ will be members of the same component of A which, moreover, will be of the same cardinality as C.*

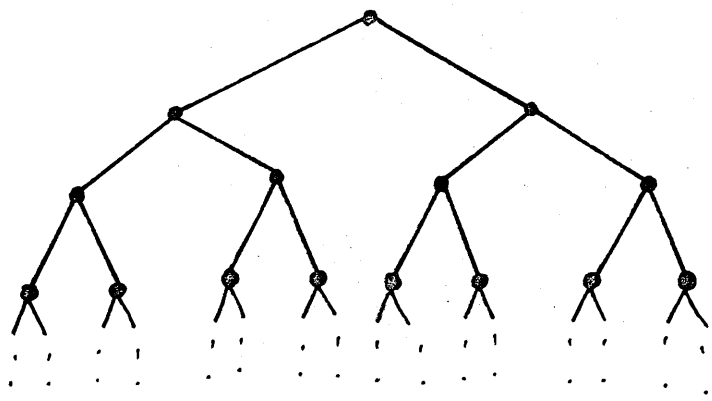


Figure 2.1 Infinite binary tree

Corollary 2.4 *The restriction of α to any finite component has aperture 1.*

Clearly the above could be rephrased more simply in the language of graph theory: our reason for talking in terms of relational structures is that the concept of amalgamations could possibly be applied to structures other than those with just a single binary symmetric non-reflexive relation. We now give a graph-theoretical example of a situation where there are no non-trivial amalgamators.

Consider the infinite binary tree in figure 2.1. Two possible amalgamations of this tree are shown in figure 2.2. In the first, at each level we amalgamate any unamalgamated vertex with its "left hand" filial vertex. This gives us a ternary tree. In the second, we amalgamate with both filial vertices, giving a quaternary tree.

It easy to see, however, that in spite of its regularity, there are no non-trivial amalgamators on this tree, because identifying any two or more vertices gives a vertex of degree greater than three.

An interesting problem is to characterise those trees which *do* have non-trivial amalgamators—figure 2.3 shows one such tree.

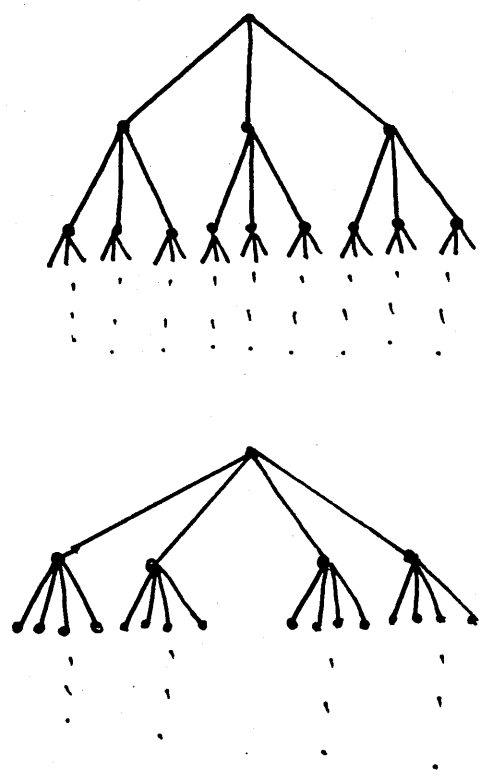


Figure 2.2 Two amalgamations of an infinite binary tree

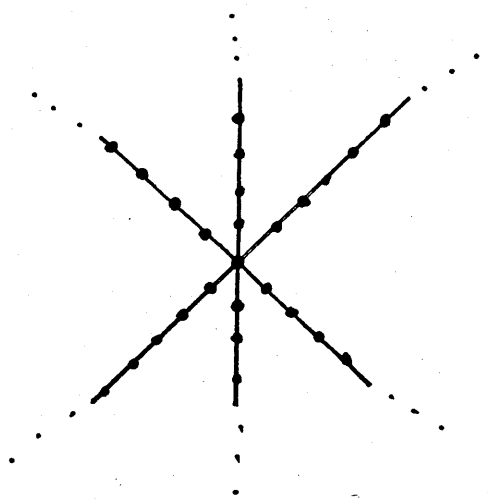


Figure 2.3 Tree with non-trivial amalgamators

We have the following theorem:

Theorem 2.5 *Let G be an infinite, acyclic graph of minimum degree $\delta > 2$. The G has no non-trivial uniform amalgamators.*

Proof: Suppose that G has a non-trivial k-uniform amalgamator α, and let ι be the corresponding inclusion function (see remark 2.2). Let $v \in V(G)$ be a vertex of degree δ: we have $|\iota^{-1}\alpha(v)| = k > 1$. Let H be the induced subgraph on $\iota^{-1}\alpha(v)$. Because G is acyclic, we must have

$$\begin{aligned} \partial_G(v) &= \partial_{\alpha(G)}(\alpha(v)) \\ &= \#\{\text{edges joining vertices of } H \text{ to vertices of } G \setminus H\} \quad (2.1) \end{aligned}$$

Now, since H is connected, finite and acyclic, H has $m = k - 1$ edges. Also, by the handshaking lemma,

$$m = \frac{1}{2} \sum_{w \in V(H)} \partial_H(w)$$

But from equation 2.1 above,

$$\sum_{w \in V(H)} \partial_H(w) = \sum_{w \in V(H)} \partial_G(w) - \partial_G(v)$$

Hence we have

$$k - 1 = \frac{1}{2}\left(\sum_{w \in V(H)} \partial_G(w) - \partial_G(v)\right)$$

i.e.

$$\sum_{w \in V(H)} \partial_G(w) = 2k - 2 + \partial_G(v)$$

But, by assumption, $\sum_{w \in V(H)} \partial_G(w) \geq \delta k$. Hence

$$2k - 2 + \partial_G(v) \geq \delta k$$

from which we get

$$\begin{aligned} \partial_G(v) &\geq k(\delta - 2) + 2 \\ &\geq 2\delta - 2 \\ &> \delta \end{aligned}$$

since $k > 1$ and $\delta > 2$. But this contradicts our original assumption that v is of minimum degree. □

If T is an infinite acyclic graph with no end-vertices, then it is possible to construct a uniform amalgamation of T of any aperture k using a greedy algorithm: start at any vertex and amalgamate sets of k vertices. If T has end-vertices this may fail: for example a vertex adjacent to r end-vertices precludes the existence of amalgamations of aperture r or less.

The following proposition demonstrates how to construct infinite acyclic graphs (not connected) which possess uniform amalgamators.

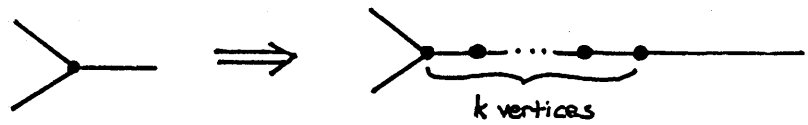
Proposition 2.6 *Let T be an infinite acyclic graph with no end-vertices. Then for any $k > 1$ T is a subgraph of a graph G possessing a uniform k-amalgamator.*

Proof. Construct a family $\{T_i | i \in \mathbf{Z}\}$ of graphs as follows:

(a) $T_0 \sim T$;

(b) for $i < 0$, T_i is a uniform k-amalgamation of T_{i+1} constructed as above;

(c) for $i > 0$, T_i is constructed from T_{i-1} by replacing each vertex of T_i with k vertices as follows:



We now construct G by taking the disjoint union $\bigcup_{i \in \mathbb{Z}} T_i$. It is clear that we can construct a k-amalgamator on G which maps T_i to T_{i+1}. □

Definition 2.7 *If G is an infinite graph then an* ω-tail *of G is a one-way infinite path consisting entirely of vertices of degree 2. We say that G is a* porcupine *if every vertex is adjacent to an ω-tail.*

Theorem 2.8 *A porcupine has uniform k-amalgamators for all $k > 1$.*

Proof. Let P be a porcupine. Now P can have at most one end-vertex; if it does have an end-vertex then it is a one-way infinite path, and the theorem is true. Suppose, then, that P has no end-vertices, and let X be the set of vertices which do not lie on an ω-tail: we have $\partial(x) > 2$ for all $x \in X$. We now define a k-amalgamator α as follows: for each $x \in X$, set $\alpha(x)$ to be the vertex in the amalgamated graph obtained by amalgamating x with the first $k-1$ vertices of one of the ω-tails to which x is adjacent. The vertices not so far considered now form a set of one-way infinite paths, and it is clear how to extend α to the whole of P in such a way that $\alpha(P)$ is isomorphic to P. □

It is not known whether any connected graph admitting k-amalgamators for all k must be a porcupine.

An example of an infinite graph which does have non-trivial uniform amalgamators is the square lattice: see figure 2.4. In fact, the results of chapter 4 will show that the infinite graphs which are the duals of all the Laves nets except for $[3.12^2]$ have non-trivial uniform amalgamators.

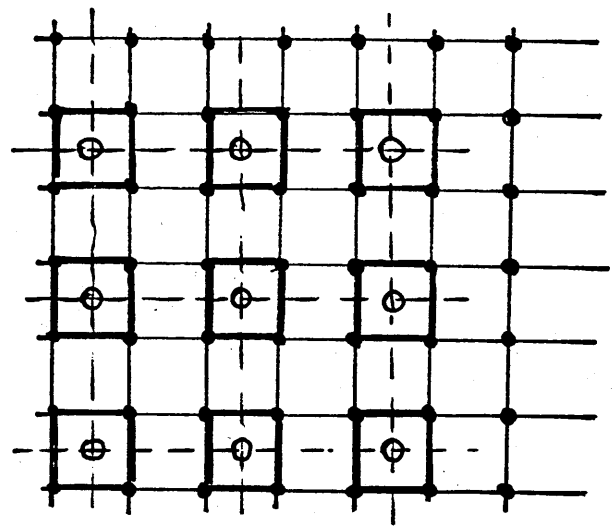


Figure 2.4 The infinite square lattice, and a 4-uniform amalgamation

2.2 Amalgamations of Tilings

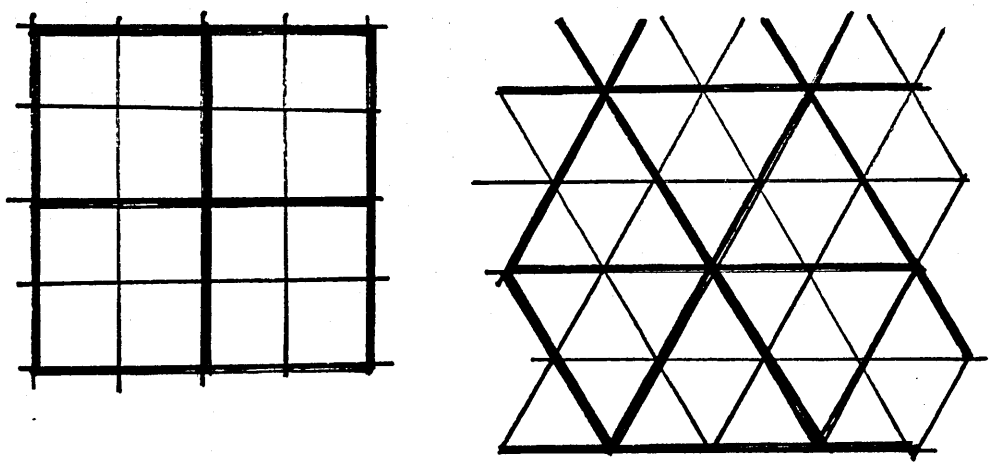
We now apply the definitions of the previous section to tilings of the plane. The definition of an amalgamation now becomes:

Definition 2.9 *Let T be a tiling of the plane. An* amalgamation *of T is a tiling S, each of whose tiles is a molecule of T. Hence $S \subset \langle T \rangle$.*

Definition 2.10 *If S is an amalgamation of T, and S is isohedral, then we say that S is an* isohedral amalgamation *of T.*

Definition 2.11 *If every tile of S is an n-tile of T, then we say that the amalgamation is n-*uniform*, or that it is an* n-amalgamation*. Unless otherwise stated, all amalgamations we consider will be uniform amalgamations.*

Definition 2.12 *If S is an amalgamation of T, and S is isomorphic to T, then we say that S is a* iso-amalgamation *of T.*



The thicker lines show the tiles of the amalgamation

Figure 2.5 Examples of iso-amalgamations

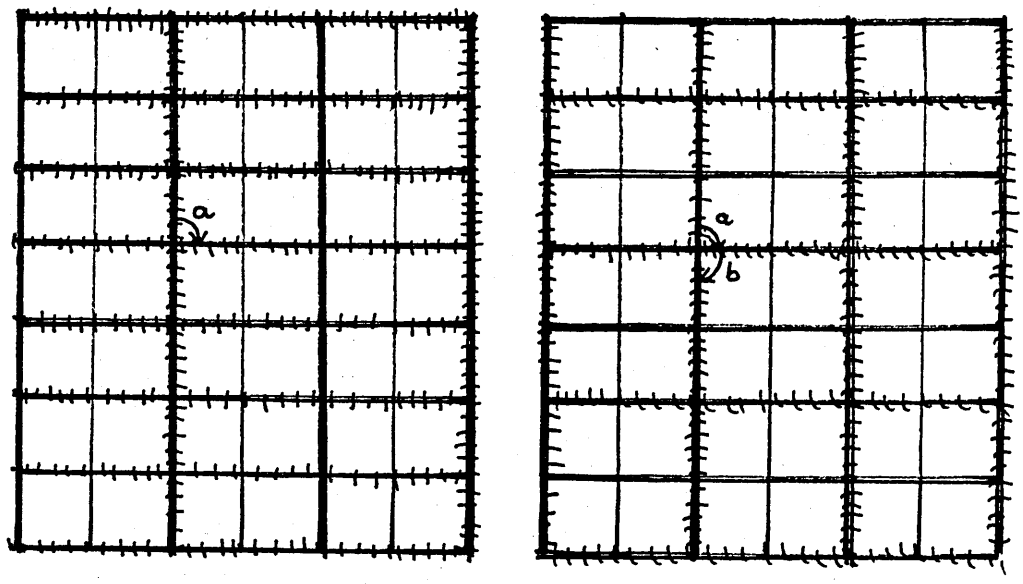
Examples: Figure 2.5 shows examples of these. If we have an iso-amalgamation S of a tiling T, then clearly there is an isomorphism $\alpha : T \to S$.

Definition 2.13 *An* amalgamator *of a tiling T is an isomorphism between T and an amalgamation of T.*

Remark 2.14 *It will usually clear from the context when the amalgamations under discussion are iso-amalgamations.*

A problem of notation arise with both amalgamations and amalgamators. With amalgamations we are usually content with showing a small area of the amalgamation and assuming an "obvious" extension to the rest of the tiling: more precise ways of specifying the amalgamation may be possible when, for example, we have a co-ordinate system defined on the tiling.

In the case of amalgamators, more information is needed, to show how the tiles of the tiling T are mapped onto those of $\alpha(T)$ by the amalgamator α. In this case we mark a directed angle a on the tiling, together with its image b (unless the two coincide). The fact that α is an isomorphism means that this is sufficient to specify its action. It is important to note that two



The bold lines indicate the first level tilings, and the hatched lines indicate the second level tilings

Figure 2.6 Two amalgamators having the same first level tiling

amalgamators α and β may be such that $\alpha(T)$ and $\beta(T)$ are congruent, whereas $\alpha^2(T)$ and $\beta^2(T)$ are not: see, for example, figure 2.6, which also illustrates the method of specifying amalgamators just described.

It can happen that an amalgamation of a tiling can have symmetries which are not generic, in the sense that they do not arise from the symmetries of the tiling itself: they are not restrictions of the symmetries of the tiling. These symmetries are known as *spurious symmetries.* Figure 2.7 shows two examples of this. In the first case the amalgamation contains two inequivalent types of general tile. Here the spurious symmetries arise because the sides of the rectangular tiles are in the ratio 2:1. In the second case the tiles of the amalgamation are all equivalent, and we have the situation that an automorphism of the basic tiling has become a symmetry of the amalgamation.

We make the following definitions:

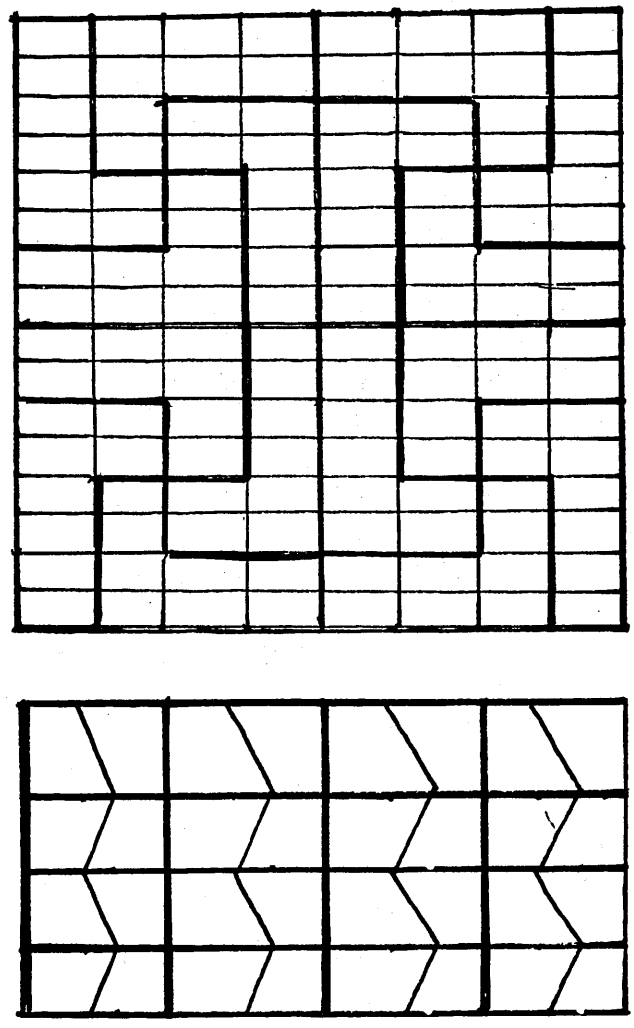


Figure 2.7 Examples of spurious symmetries

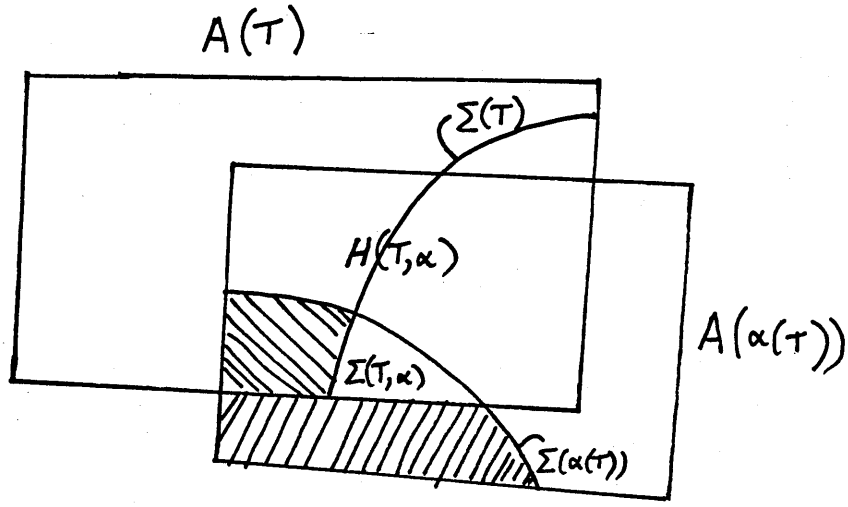


Figure 2.8 Groups acting on a tiling and an amalgamation

Definition 2.15 *We denote by $H(T,S)$ the group of automorphisms of T which act as automorphisms of S. Similarly, we denote by $\Sigma(T,S)$ the group of symmetries of T which act as symmetries of S.*

We abbreviate $H(T,\alpha(T))$ and $\Sigma(T,\alpha(T))$ to $H(T,\alpha)$ and $\Sigma(T,\alpha)$ respectively.

By an abuse of notation, we can write:

$$H(T,\alpha) = A(T) \cap A(\alpha(T))$$

$$\Sigma(T,\alpha) = \Sigma(T) \cap \Sigma(a(T))$$

The diagram in figure 2.8 makes clearer the relationship between the various groups, although this is not strictly a Venn diagram of sets, for the reasons noted above (Remark 1.14 etc.). The shaded areas show the two different types of spurious symmetries noted above: note that there are also areas which correspond to what might be called "spurious automorphisms"—these are rather trickier to imagine as it is less easy to "see" whether an automorphism of $\alpha(T)$ arises from an automorphism of T.

The behaviour of these groups is fundamental to the study of amalgamations.

Definition 2.16 *If S is an isohedral amalgamation of a tiling T then S is said to be* strongly isohedral *if $\Sigma(T, S)$ acts transitively on S.*

2.3 Tiling Hierarchies

Suppose that we have a tiling, T, and an amalgamator α on T. It follows from the way that we have defined isomorphisms that we can apply α successively to T, to obtain a series of tilings $T, \alpha(T), \alpha^2(T), \ldots$. We then have the following definition:

Definition 2.17 *If T is a tiling, and α is an amalgamator on T, then the sequence $T, \alpha(T), \alpha^2(T), \ldots$ of tilings is called the* tiling hierarchy *generated by T and α, and is denoted $[T, \alpha]$ or $\mathcal{T}(T, \alpha)$.*

Definition 2.18 *Two tiling hierarchies $\mathcal{T}(T, \alpha)$ and $\mathcal{T}(S, \beta)$ are said to be isomorphic (isomeric) if there is an isomorphism (isomerism) ϕ such that $\beta = \phi\alpha\phi^{-1}$.*

The paper by Holroyd [11] gives a summary of definitions and results concerning tiling hierarchies. We will return to this subject when we come to discuss the classification of hierarchies.

Chapter 3

Tesseral Addressing Systems

In this chapter we briefly survey the work which has been done on *tesseral addressing*—it was this work which originally motivated the study of amalgamations of tilings.

3.1 Quadtrees

Remark 3.1 *The word quadtree has become a rather vague term used to describe a class of hierarchical data structures which are based on the principal of recursively decomposing space. A full survey of this field can be found in [22]. In this section we will be describing how a* region quadtree *can be used to analyse an image.*

The problem of amalgamations of tilings originally arose in connection with a problem in the field of digital image processing.

A digitized image is divided up into cells, called *picture elements* or *pixels*, of sufficiently small size, and to each pixel we can then associate various attributes such as colour, intensity, etc. In the simplest case we would simply assign 1 or 0, where 1 indicates that a pixel lies in a "region of interest", where this could be a road, river, populated area etc. We now want to analyse this image in some way: for example to trace the path of a road to its intersection with another road.

This immediately raises the first problem, which is how we should divide up the picture area; in other words what shape should the pixels be? The

most commonly used shape is, of course, the square (i.e. the Laves net of type $[4^4]$). However the hexagonal tiling offers some advantages over this: for example each tile is adjacent at an edge to six rather than four tiles, and there is only one type of adjacency (at an edge) rather that the two (at an edge and at a corner) exhibited by the square tiling. On the other hand, it is not possible to subdivide a hexagon into smaller hexagons, which means that hierarchies are "one way". Another possibility is to use a tiling of type $[3^6]$, but with square tiles: this is used in some colour televisions. Note that this does not overcome the problem of subdivision; if T is a type $[3^6]$ tiling by squares then subdividing the squares will not produce another $[3^6]$ tiling. In [2] a list of such criteria is presented, and various possibilities are considered.

Having chosen a tiling, we now consider how the data for an image is stored: there are three possibilities.

Raster Representation. The information corresponding to the pixels is stored as a list, corresponding to some particular ordering of the pixels.

Vector Representation. We store a list of ordered pairs consisting of the address and the corresponding attribute.

Hierarchical Representation. This is the method described below, in which the image is divided into regions, which are in turn subdivided, continuing until we have extracted as much detail as possible.

The first task in the processing of an image by computer, then, known as *segmentation*, is to identify subsets of interest from the scanned image. Klinger and Dyer [3] describe how this may be carried out by successive *regular decomposition* of the picture, and we describe this process now.

Consider the "image" shown in figure 3.1. Notice that there is a large area of open land, which is basically uniform: we would not, therefore, be interested in storing details of every pixel of this area. On the other hand, there are areas with more detail which need to be analysed with more precision. To analyse this we represent the picture as spatial subsets of

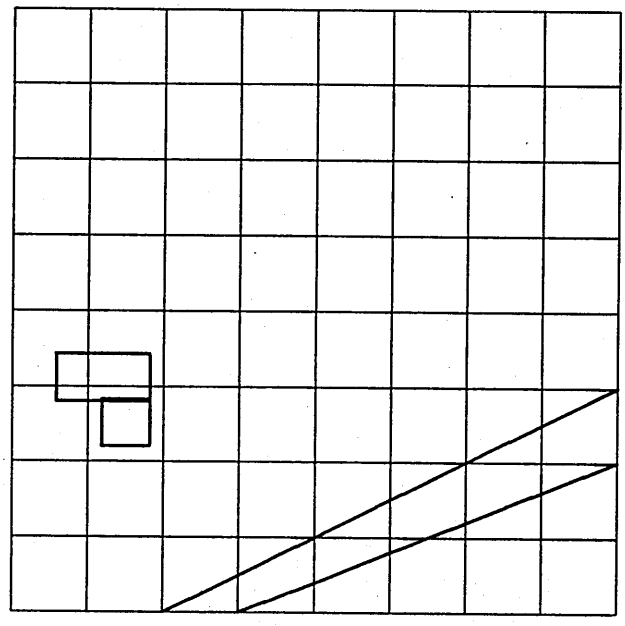


Figure 3.1 A sample "image"

different sizes, each of which we mark as "informative" or "uninformative", so that we can discard pixels belonging to "uninformative" subsets.

Specifically, we start by considering the whole picture as a quadrant. There are basically two possibilities: either the area contains nothing of interest (for example it might be a large area of open land), or it contains detail which needs to be analysed further, in which case we subdivide the quadrant and repeat the analysis. We continue until either no quadrants require further subdivision, or the quadrant size becomes the same as the pixel size.

To make this clearer, we will analyse the "map" in figure 3.1. We will assume, rather unrealistically, that the map is 8 pixels by 8 pixels.

Clearly the map contains detail of interest, so we subdivide the map, as in figure 3.2 . Notice that the top two quadrants contain no detail of interest. We do not, therefore, consider them further. The bottom two quadrants do, so we subdivide them further: figure 3.3 We now subdivide once more in

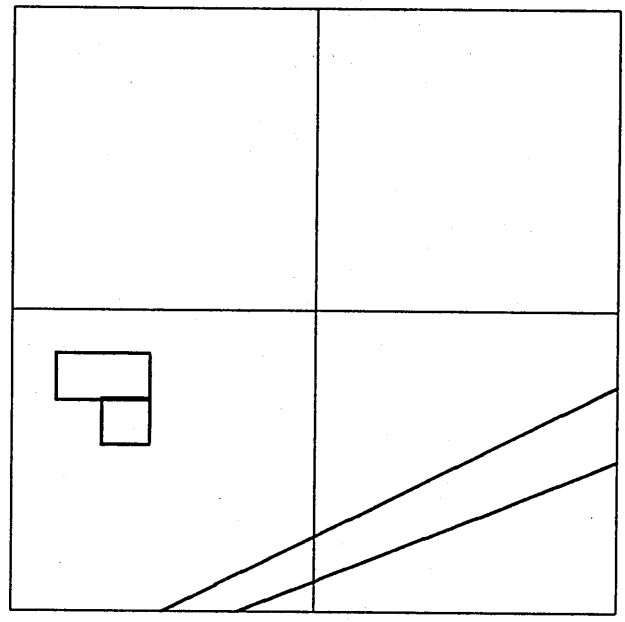


Figure 3.2 The first subdivision

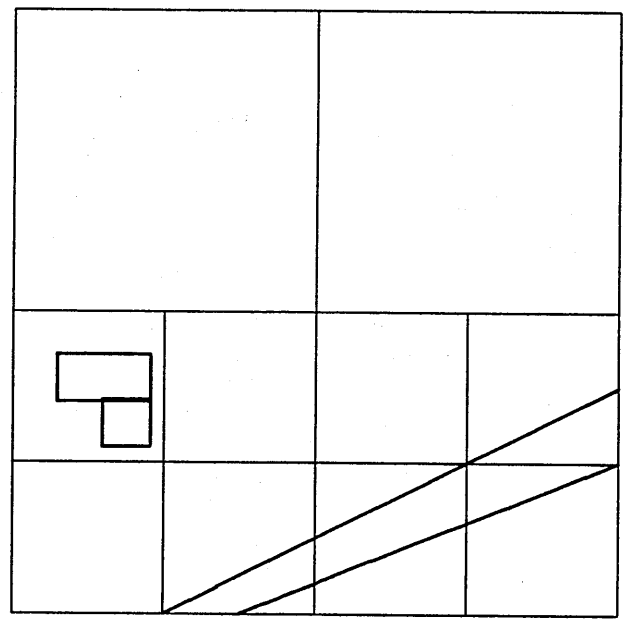


Figure 3.3 The second subdivision

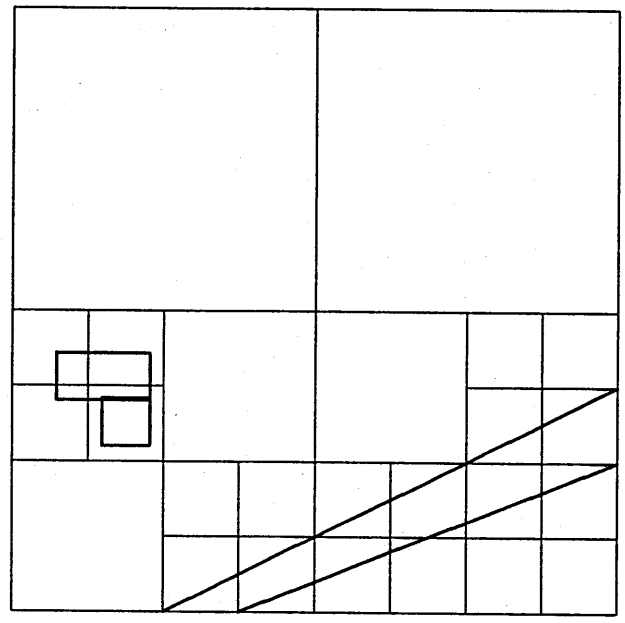


Figure 3.4 The third subdivision

those quadrants which contain detail of interest.

Figure 3.5 shows a binary array corresponding to the image, where pixels marked 1 lie in regions of interest.

This can also be represented by a binary tree (hence the term quadtree), as shown in figure 3.6.

This tree can now be stored on a computer: for example for each node we could store the co-ordinates of a suitable reference point (e.g. the top left hand corner), together with either a colour (black or white) or a set of pointers to the filial nodes. A node which is not leaf is known as a *grey node*.

This representation has two advantages over raster and vector representations. Firstly, large homogeneous areas are usually recognised (although the shape and orientation of the area will determine how much it is split up: this is another example of how different shapes of pixel may be more appropriate in a given situation), and secondly quadtrees provide an areal notion of locality, rather than a linear one: in the usual raster representa-

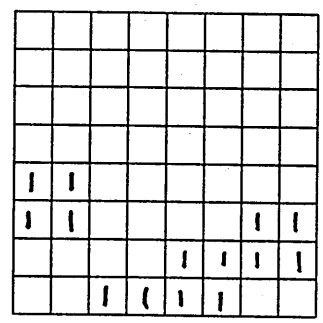


Figure 3.5 Binary image corresponding to figure 3.1

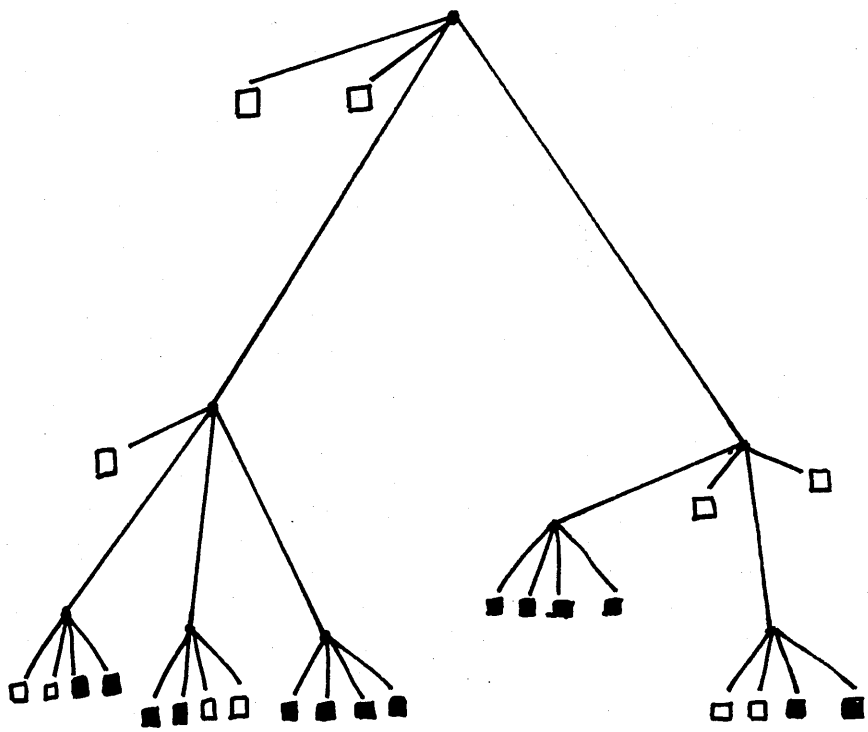


Figure 3.6 Quadtree corresponding to figure 3.1

tion, horizontally adjacent pixels will be adjacently stored while vertically adjacent pixels will be stored at addresses some distance apart. The hierarchical nature of the quadtree representation makes the adjacency structure of the image much clearer.

3.2 Tesseral Addressing

In [4] Gargantini proposed a method of representing a quadtree that did not involve the use of pointers to filial nodes, as had been used so far, but represented each node with a quaternary integer chosen to reflect the hierarchical structure of the quadtree. This was later termed *tesseral addressing* by other workers in the field.

To construct the tesseral address of a point in the image we proceed as follows. (The addressing system given here is a slightly modified version of that given by Gargantini, in that it covers NE rather that the SE quadrant of the plane. The significance of this becomes clear when negative tesseral addresses are considered.) For each subdivision of the image we encode the SW quadrant with the digit 0, the SE quadrant with the digit 1, the NW quadrant with the digit 2, and the NE quadrant with the digit 3. Each pixel is then encoded with a weighted quaternary code $\ldots a_{n-1}a_{n-3}\ldots a_2a_1a_0$, where a_i identifies the quadrant to which the pixel belongs at the $(n-i)$th subdivision, where the image is of size $2^n \times 2^n$ pixels. Figure 3.7 shows the addresses of the pixels in our 8×8 image space.

We now have an address for each pixel in our region. To represent a larger quadrant consisting of 2^k pixels, we introduce a "joker", X, which means "0,1,2 and 3". Thus, to represent the square containing the pixels labelled 120,121,122,123, we would use the address 12X. We might also use the notation 1X2 to represent the pixels 102,112,122 and 132.

We can now represent a digitised image simply as a list of black pixels. For example the image considered would be represented by the list 02X, 010, 011, 10X, 112, 113, 130, 131.

222	223	232	233	322	323	332	333
220	221	230	231	320	321	330	331
202	203	212	213	302	303	313	313
200	201	210	211	300	301	310	311
022	023	032	033	122	123	132	133
020	021	030	031	120	121	130	131
002	003	012	013	102	103	112	113
000	001	010	011	100	101	110	111

Figure 3.7 Tesseral addresses

3.3 Tesserals and Amalgamators

It is clear that tesseral addressing systems and amalgamators are intimately related, and we now show how the theory of amalgamations can be used to put the above discussion on a firmer footing.

Firstly, we should reiterate that the square tiling with the corresponding "2×2" amalgamator is by no means the only possibility for constructing a tesseral addressing system. In [2] several further possibilities are illustrated. The primary requirement, however, is that the tessellation should possess a non-trivial uniform amalgamator, and this gives a practical reason for classifying all possible amalgamations of plane tessellations.

Secondly, it is intuitively clear that the addressing system described above could be expanded to cover the whole of the NE quadrant of the plane, but the "recipe" given does not give a rigorous way to do this: to achieve this it is necessary to use amalgamations.

Consider first the addressing system shown in figure 3.7, but ignoring all

2	3	2	3	2	3	2	3
0	1	0	1	0	1	0	1
2	3	2	3	2	3	2	3
0	1	0	1	0	1	0	1
2	3	2	3	2	3	2	3
0	1	0	1	0	1	0	1
2	3	2	3	2	3	2	3
0	1	0	1	0	1	0	1

Figure 3.8 Atomic Addressing Pattern

but the right-most digit, a_0—see figure 3.8.

It is clear how to extend this pattern to the whole NE quadrant (and in fact to the whole plane). Now by repeated applications of the amalgamator α to this marked tiling, we can define the digits $a_1, a_2, \ldots$. This gives an infinite address for each tile: however since every tile in the NE quadrant is contained in $\alpha_n(t_0)$ for some n, where t_0 is the tile whose SW corner is the origin, we have $a_k = 0$ for $k > n$. We can therefore assign a finite address to every point in the NE quadrant. This corresponds to the fact that any non-negative integer can be uniquely represented as $\sum_{i=0}^{\infty} a_i 10^i$ where $0 \leq a_i \leq 9$, so any particular non-negative integer is expressed as a finite string of digits.

In general, let T be a tiling of the plane which admits a non-trivial uniform strongly isohedral amalgamator α with aperture s. An *atomic addressing pattern* is a function $\phi_0 : T \to \{0, \ldots, s-1\}$ which satisfies the following conditions.

1. For any tile $t \in T$, no two tiles in $\alpha(t)$ have the same label.

2. $\Sigma(T, \alpha)$ has a subgroup H which acts regularly[1] on $\alpha(T)$ and is such that, for any $\gamma \in H$, we have

$$\phi_0(\gamma(t)) = \phi_0(t) \ \ \forall t \in T.$$

Following the example above, we use the amalgamator α to "lift" the atomic addressing system as follows: for each k any tile t will lie in the image under α^k of some tile t', say. We let the kth digit (from the right) of the address of t be $\phi_0(t')$. More precisely, for each k we define a function ϕ_k by

$$\phi_k = \phi_0 \circ \alpha^{-k}.$$

We now define the address of a particular tile to be:

$$\ldots a_n a_{n-1} \ldots a_1 a_0$$

where $a_i = \phi_i(t)$. Notice that, as it stands, this is an infinite string: in practice this is not a problem, as we shall see.

There are two desirable properties we should like our addressing system to possess:

1. it should be *unambiguous*: i.e. no two tiles have the same address;

2. it should be finitary: that is to say, there is a constant c (which we can arrange to be 0) such that for any tile t there is an integer K for which $\phi_k(t) = c$ for all $k > K$.

There are various ways in which an addressing system can fail to have the properties: for example the system based on the Gargantini system is not finitary.

An example of an addressing system that is both unambiguous and finitary is the addressing system for the $[3^6]$ tiling known as *generalised balanced ternary*, or GBT: this is based on an amalgamation of aperture 7, and is illustrated in figure 3.9 This system is due to Lucas and is described in [17].

[1]The action of a group on a set is regular if it is transitive and the stabiliser of any point is trivial.

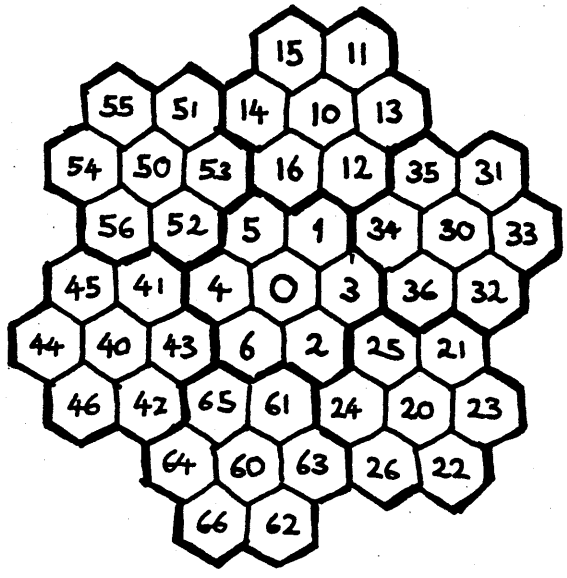


Figure 3.9 Generalised Balanced Ternary system

Notice that in this case the underlying amalgamator has the property that, if t_0 is the tile labelled 0, we have

$$t_0 \subseteq \alpha(t_0) \subseteq \alpha^2(t_0) \subseteq \ldots$$

and

$$\bigcup_{k=1}^{\infty} \alpha^k(t_0) = \mathbf{R}^2.$$

In contrast, the Gargantini addressing system is unambiguous, but not finitary: the plane is divided into four sectors, with $\phi_k(t) = 0$ for k large enough in the NE quadrant, $\phi_k(t) = 1$ for k large enough in the NW quadrant, with the SE and SW quadrants being similarly characterised by 2 and 3 respectively.

If we introduce a rotation between levels, it is possible to obtain a system which is unambiguous, but where the 'tail' is cyclic.

It is also possible to obtain systems with finite or infinite ambiguity. Non-finitary systems always divide the plane up into a finite number of sectors: we conjecture that the number of sectors is always 1,2,3,4 or 6.

The behaviour of an addressing system resulting from an amalagamator can be described in terms of the number of *origins* that the amalgamator has.

Definition 3.2 *An* origin *of an amalagamator α on a tiling T is a tile t_0 such that $\alpha(t_0)$ contains t_0.*

We now have the following:

Proposition 3.3 *For any origin t_0 of an amalgamator α we have:*

$$t_0 \subseteq \alpha(t_0) \subseteq \alpha^2(t_0) \subseteq \ldots$$

We can now make the following definition:

Definition 3.4 *The* sector *of an amalgamator α corresponding to an origin t_0 is the subset of the plane defined by:*

$$Sec(t_0) = \bigcup_{k=1}^{\infty} \alpha^k(t_0)$$

For any normal tiling, the sectors of an amalgamator will be unbounded, so long as there is no tile for which $\alpha(t) = t$.

We can now classify amalgamators into three basic types.

1. An amalgamator may have no origins. In this case it is said to be *0-degenerate.*

2. An amalgamator may have finitely many origins.

3. An amalgamator may have infinitely many origins. In this case it is said to be *ω-degenerate.*

Degenerate amalgamators are of less interest for producing addressing systems than non-degenerate ones. A 0-degenerate amalgamator will produce non-finitary addressing systems: typically there will be a "cyclic" infinite tail to each address: an example is shown in figure 3.10. An ω-

232	233	212	213	332	333	312	313
230	231	210	211	330	331	310	311
222	223	202	203	322	323	302	303
220	221	200	201 b	320 a	321	300	301
032	033	012	013	132	133	112	113
030	031	010	011	130	131	110	111
022	023	002	003	122	123	102	103
020	021	000	001	120	121	100	101

Figure 3.10 Addressing system defined by 0-degenerate amalgamator

100	101	110	111	000	001	010	011
100	101	110	111	000	001	010	011
100	101	110	111	000 a	001	010	011
100	101	110	111	000	001	010	011

Figure 3.11 Addressing system defined by ω-degenerate amalgamator

degenerate amalgamator, on the other hand, will produce an addressing system with infinite ambiguity: an example is shown in figure 3.11.

We conjecture that a non-degenerate uniform amalgamator on an isohedral tiling has either 1,2,3,4,6 or 12 origins.

In the examples above, the Gargantini system is based on a amalgamator with 4 origins, whereas the GBT system is based on an amalgamator with 1 origin. It is clear that an amalagamator with n origins will produce addressing systems with at most n-fold ambiguity. However, since in practice it is only ever necessary to address a finite portion of the plane this may not always be a problem, particularly if the addressing system produced has other features which make it attractive.

Connected with the subject of origins of amalgamators is the subject of fixed points. In the same way as there are topological transformations corresponding to automorphisms of tilings, so there is corresponding to any isomorphism, including an amalgamation α, a topological transformation f whose action on the elements of the tiling T is the same as that of the amalgamator (clearly there will be infinitely many of these): that is to say $\tilde{f} = \alpha$. If the inverse of this transformation is a contraction mapping on $\mathbf{R}^2$ then it will have a fixed point x_0, and x_0 will also be a fixed point of f and will lie either inside a tile of T, or on an edge of T, or on a vertex of T. It is clear that any tile such that $x_0 \in t$ (recall that t is a closed set) will be an origin of α.

3.4 Tesseral Arithmetic

We now describe how it is possible to impose an arithmetic onto certain tesseral addressing systems, which is similar in some ways to 'normal' arithmetic. We also describe some of the work which has arisen out of this.

3.4.1 One-dimensional arithmetic

Before moving on to the two dimensional case, we note that that when we express the integers 'to base b' for some b, we are in fact using a tesseral

system, based on the obvious amalgamator of aperture b. This could be seen to be the one dimensional analogue of the Gargantini system, and shares the problem of not uniquely addressing the whole number line (it has two origins)—we need to introduce the concept of negative numbers.

Notational systems for the integers which eliminate the need for negative numbers have been devised. For example, a form of ternary notation, with three symbols corresponding to 0, -1 and 1, and based on the 3-amalgamation under which $0 \mapsto \{-1, 0, 1\}$ is the one- dimensional analogue of the Generalised Balanced Ternary system described above. Each integer has a unique representation in this system, with no use of minus signs needed.

Another system which has been devised is the system of *Colson Numbers*, named after John Colson FRS (1680–1760), who first proposed the idea in 1726. These use an amalgamator of aperture 10, using $\bar{4}$, $\bar{3}$, $\bar{2}$, $\bar{1}$, 0, 1, 2, 3, 4, 5 as digits, corresponding to $-4, -3, -2, -1, 0, 1, 2, 3, 4, 5$ respectively[2]. These ten numbers constitute the image of 0 under the amalgamation which generates the Colson numbers. Once again the need for minus signs is eliminated, and we have, for example, $-1726 \equiv \bar{2}3\bar{3}4$, and $1726 \equiv 2\bar{3}3\bar{4}$. The arithmetic of Colson numbers, which is a tesseral arithmetic, is developed by Morgan in [19].

Given the above, it is perhaps less surprising that we can develop an arithmetic on a discrete two-dimensional structure which shares with decimal arithmetic such features as a place-value system with carry digits.

3.4.2 Addition

From now on we will assume that the tilings under consideration are *translational*: that is to say that the symmetry group has a transitive subgroup of type p1. If we fix the origin to lie on one of the tiles, t_0, say of T, then the centroids of the tiles of such a tiling T must form a two dimensional lattice which is closed under vector addition in $\mathbf{R}^2$ (to see this let the p1 subgroup

[2]Colson also had an additional digit, $\bar{5}$, representing -5: this introduces ambiguity as we have, for example, $2\bar{5} \equiv 15$.

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	12	3	34	5	16	0
2	2	3	24	25	6	0	61
3	3	34	25	36	0	1	2
4	4	5	6	0	41	52	43
5	5	16	0	1	52	53	4
6	6	0	61	2	43	4	65

Table 3.1 Addition table for GBT

act on the centroid of a particular tile), and we can define an addition operation on the tiles of T which corresponds to vector addition on $\mathbf{R}^2$. In turn, we can carry this over to define an addition operation on the address strings which are the addresses of the tiles.

The simplest case to consider is when the amalgamator α has one origin t_0, which lies at the origin of the vector space. In this case, as we have seen, every tile is contained in $\alpha^k(t_0)$ for some k, and so we can assign a finite address to each tile. This is true, for example in the case of the GBT system described above.

If α has aperture s, we can take the s tiles lying in $\alpha(t_0)$ and construct an addition table: we show the table for the GBT system in table 3.1.

Any two finite address strings can be added by using this table at every position, and treating the carry digits as one does in everyday addition. For example, to add 344 and 614:

$$\begin{array}{ccc} 3^5 & 4^4 & 4 \\ 6 & 1 & 4 \\ \hline & 2 & 1 \end{array}$$

This is possible because of the following three facts:

1. addition is commutative and associative;

2. the atom whose address is $a_k a_{k-1} \ldots a_1 a_0$ is the same as

$$a_k \underbrace{0\ldots0}_{k} + a_{k-1} \underbrace{0\ldots0}_{k-1} + \cdots + a_1 0 + a_0$$

$+$	0	1	2	3	$\dot{1}$	$\dot{2}$	$\dot{3}$
0	0	1	2	3	$\dot{1}$	$\dot{2}$	$\dot{3}$
1	1	10	3	12	0	$\dot{2}3$	$\dot{2}$
2	3	20	21	$\dot{1}3$	0	$\dot{1}$	
3	3	12	21	30	2	1	0
$\dot{1}$	$\dot{1}$	0	$\dot{1}3$	2	$\dot{1}0$	$\dot{3}$	$\dot{3}2$
$\dot{2}$	$\dot{2}$	$\dot{2}3$	0	1	$\dot{3}$	$\dot{2}0$	$\dot{3}1$
$\dot{3}$	$\dot{3}$	$\dot{2}$	$\dot{1}$	0	$\dot{3}2$	$\dot{3}1$	$\dot{3}0$

Table 3.2 Addition table for Gargantini addressing system

3. addition of digits obeys the same rules at each level: in other words if $a + b = c$ then $a0 + b0 = c0$ etc.

The first fact follows directly from the fact that addition is essentially vector addition, but the second and third rely on the particular amalgamator chosen: namely it should possess at least one origin which receives the label 0, and lies at the origin for the addition operation.

If the amalgamator α on which the addressing system is based has more than one origin, then the situation becomes slightly more complicated.

We need to introduce the concept of *dotted digits*, analagous to recurring decimal digits. A tile not in the NE quadrant of the Gargantini addressing space will have an address such as $\ldots 1111123$. We will denote this by $\dot{1}23$. We can now extend the addition table to include dotted digits, as shown in table 3.2, and the place value addition continues to apply.

Introducing dotted digits also introduces a potential source of ambiguity, in that if an arbitrary mixture of dotted and undotted digits is allowed in an address, then each atomic tile has infinitely many addresses: e.g.

$$\dot{1} = \dot{1}1 = \dot{1}11 = \dot{1}111 = \ldots$$

$$\dot{1}\dot{1} = \dot{1}0 + \dot{1}1 = \dot{1}01$$

If we insist that only the leftmost digit is dotted, and that the shortest such expression is used, then we get uniqueness.

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Table 3.3 Multiplication table for GBT

3.5 Multiplication

As defined above, tesseral addition has a natural geometric interpretation: adding one tile to another corresponds to translating the first by the vector corresponding to the displacement of the second from the origin (i.e. it corresponds to vector addition). This corresponds to real and complex addition in one and two dimensions respectively.

It is natural to ask whether we can define a multiplication on an addressing system, which would have the corresponding geometrical interpretation of 'rotation plus scaling' (as in complex arithmetic).

In fact, for both the systems defined above this is possible, and the corresponding multiplication tables are given in tables 3.3 and 3.4. Notice that for the GBT system, no carry digits are generated.

The reason that it is possible to define these multiplication operations is that in each case the tile centroids can be arranged to form a sub- ring of the set of complex numbers: $\mathbf{Z}[1, \sqrt{3}i/2]$ and $\mathbf{Z}[1, i]$ respectively.

3.5.1 Extension to higher dimensions

The question of whether tesseral arithmetic can be extended to higher dimensions is considered in [26]. It is shown that, whereas a Gargantini type addressing system, with addition corresponding to vector addition, can be defined in any number of dimensions, the same is not true of multiplication:

$\times$	0	1	2	3	$\dot{1}$	$\dot{2}$	$\dot{3}$
0	0	0	0	0	0	0	0
1	0	1	2	3	$\dot{1}$	$\dot{2}$	$\dot{3}$
2	0	2	$\dot{1}$	$\dot{1}3$	$\dot{2}$	1	$\dot{2}3$
3	0	3	$\dot{1}3$	20	$\dot{3}$	$\dot{2}3$	$\dot{2}0$
$\dot{1}$	0	$\dot{1}$	$\dot{2}$	$\dot{3}$	1	2	3
$\dot{2}$	0	$\dot{2}$	1	$\dot{2}3$	2	$\dot{1}$	$\dot{1}3$
$\dot{3}$	0	$\dot{3}$	$\dot{2}3$	$\dot{2}0$	3	$\dot{1}3$	20

Table 3.4 Multiplication table for Garagantini system

once we go higher than two dimensions the only possible non-distorting multiplication is quaternion multiplication in 4 dimensions, and this is non-commutative. Multiplications *can* be defined in higher dimensions, but they will be distorting (angles are not preserved) or singular (all multiples of a single point lie in the same hyperplane).

3.5.2 Radix systems

Suppose now that we allow multiplication to be a distorting operation, and possibly non-commutative, but continue to exist that it is associative, distributive over addition, and non-singular (in other words if a and b are non-zero, then so is $a \times b$). We are now searching for divisor-free rings over the set of atomic tiles. However, we still require a place-value system to hold.

For an atomic tile a, denote by $0_\alpha a$ the atom of $\alpha(a)$ whose zero-level address is 0. Then for a place value system to work we need the following condition to hold:

$$0_\alpha(a \times b) = 0_\alpha a \times b = a \times 0_\alpha b \tag{3.1}$$

If a multiplicative identity e exists then this is equivalent to

$$0_\alpha(e \times a) = 0_\alpha e \times a = e \times 0_\alpha a \tag{3.2}$$

where a, b are atomic tiles.

Definition 3.5 *A* radix system *on a tiling T is a combination of a non-degenerate addressing system, together with a divisor-free ring structure, sharing the same origin, labelled 0, and obeying condition 3.1 if no multiplicative identity exists, or condition 3.2 if one does. In the latter case we assume that the addressing system is such that the multiplicative identity is labelled 1. The* radix *of the system is the element 10.*

3.6 The future of tesseral addressing

We have seen that addressing images using quadtrees leads to a representation of the image which can preserve areal locality, and which can recognize large homogeneous areas, thus possibly reducing the storage required. Tesseral arithmetic was developed as a means of manipulating these stored images; it was hoped that the development of special hardware—"tesseral machines"—would speed up the processing of stored images. It is certainly the case that such a machine could implement transformations of the plane more efficiently than a conventional computer using cartesian addressing with two co-ordinates.

Holroyd, in [14], suggests a way of constructing a general integer linear transformation machine, which would be even more efficient. He concludes that tesseral arithmetic may not be the way forward, and may possibly remain only of theoretical interest.

Part II

Generating Amalgamations

Introduction

This part of the thesis is devoted to classifying and enumerating amalgamations and amalgamators. The ultimate goal would be to obtain, for each IH type of tiling, a list of the possible amalgamations and amalgamators, and to classify the possible tiling hierarchies up to isomerism.

The results presented here are only partial, however, and fall into two main categories: firstly generating strongly isohedral amalgamations, and secondly describing the groups acting at different levels of a tiling hierarchy, which is an approach towards the classification of tiling hierarchies.

Chapter 4

Topological Classification of Amalgamations

In this chapter we consider the types of homogeneous uniform amalgamations of homogeneous tilings possible when tilings are classified simply by topological type.

There is a fundamental lemma which excludes particular amalgamations:

Lemma 4.1 *The maximum vertex-degree (i.e. number of tiles meeting at a vertex) of any amalgamation of a tiling is less than or equal to the maximum vertex-degree of the tiling.*

The proof of this lemma is obvious on considering what happens to a vertex on amalgamation.

We now go on to prove the following theorem:

Theorem 4.2 *If T is a homogeneous tiling, then T has homogeneous uniform amalgamations which are regular of every type allowed by the maximum-degree lemma, except when T has type* $[3.12^2]$, *when amalgamations of types* $[4.8^2]$, $[3.12^2]$ *and* $[4.6.12]$ *are not possible.*

Proof: We prove this theorem by considering each Laves net in turn.

The hexagonal tiling $[3^6]$ has maximum vertex-degree 3, and is the only type with this property. It follows that it only has amalgamations of type $[3^6]$.

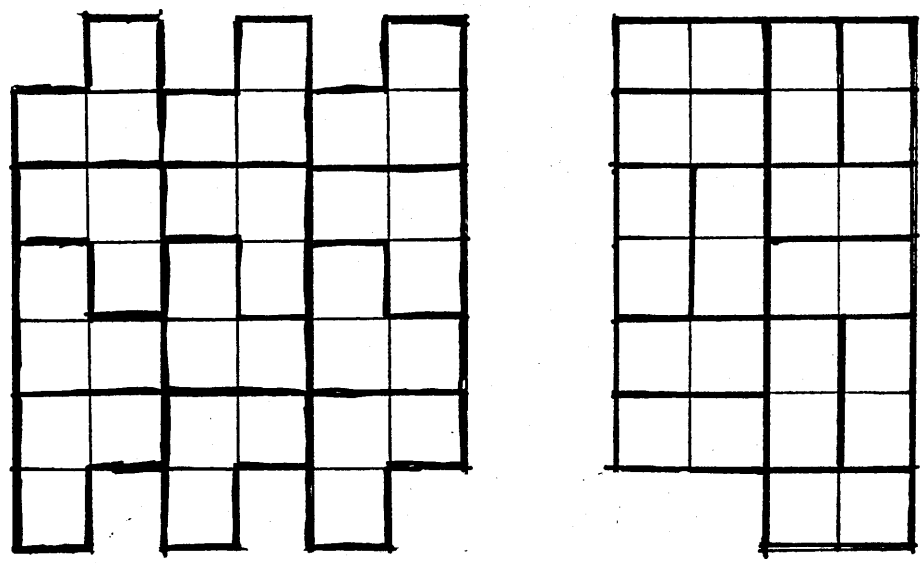


Figure 4.1

The square tiling $[4^4]$ has maximum vertex-degree 4, and thus the lemma excludes all types except for $[3^6]$, $[4^4]$, $[3^3.4^2]$ and $[3^2.4.3.4]$. The first two can be realised by the "brick wall" and "square" tilings repectively. For the second two, see figure 4.1.

Tilings $[3^3.4^2]$ and $[3^2.4.3.4]$ have amalgamations of the same types, for they both amalgamate to $[4^4]$ (see figure 4.2) and hence to the other types; once again the lemma excludes all other types.

We now consider tiling type $[6^3]$. It is obvious how to amalgamate 6 tiles to get $[3^6]$; by amalgamating pairs of tiles one gets type $[4^4]$, from which by previous results one can get $[3^3.4^2]$ and $[3^2, 4, 3, 4]$. Figure 4.3 shows amalgamations of types $[3^4.6]$, $[3.6.3.6]$, $[6^3]$, and $[3.4.6.4]$. All other types are excluded by the lemma.

Figure 4.4 shows amalgamations of types $[3^4.6]$, $[3.4.6.4]$ and $[3.6.3.6]$ to type $[6^3]$: it follows that these tilings have amalgamations of all types not excluded by the maximum degree lemma.

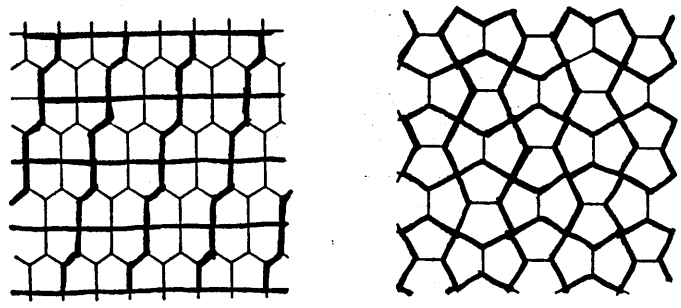


Figure 4.2

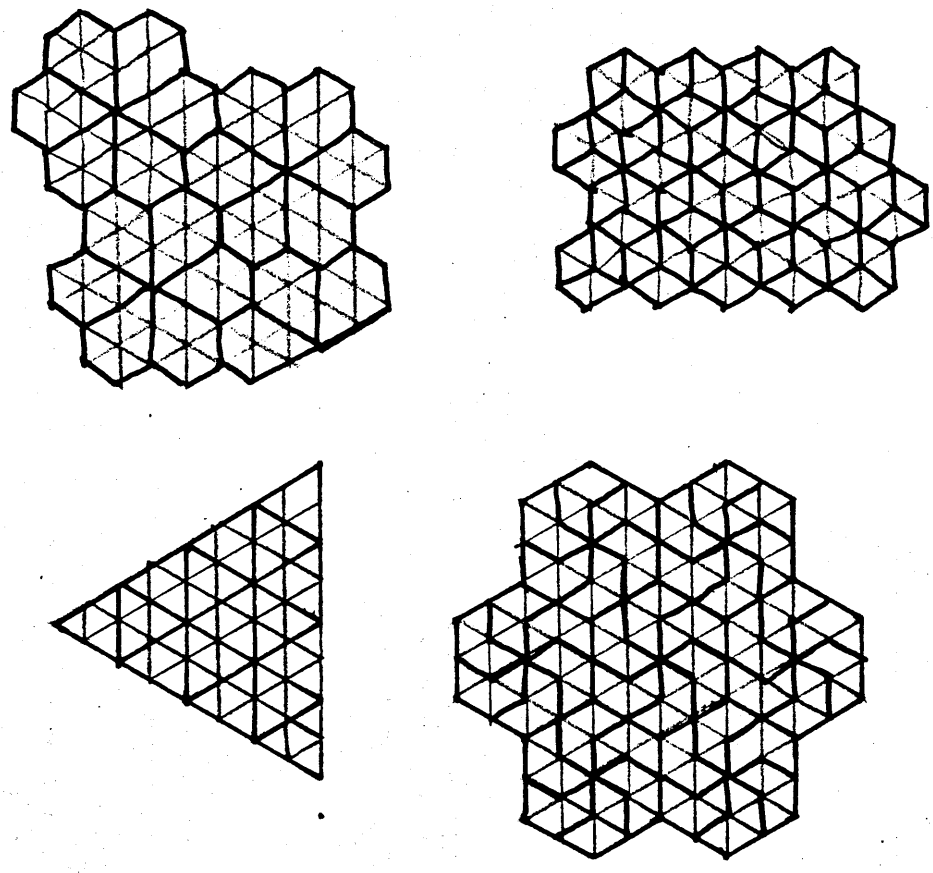
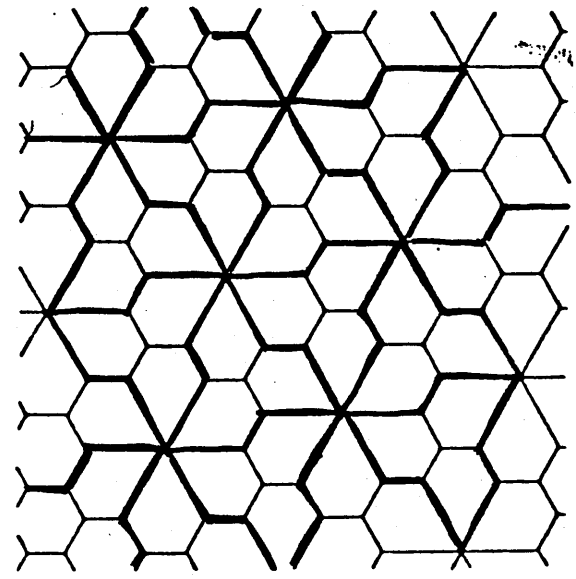
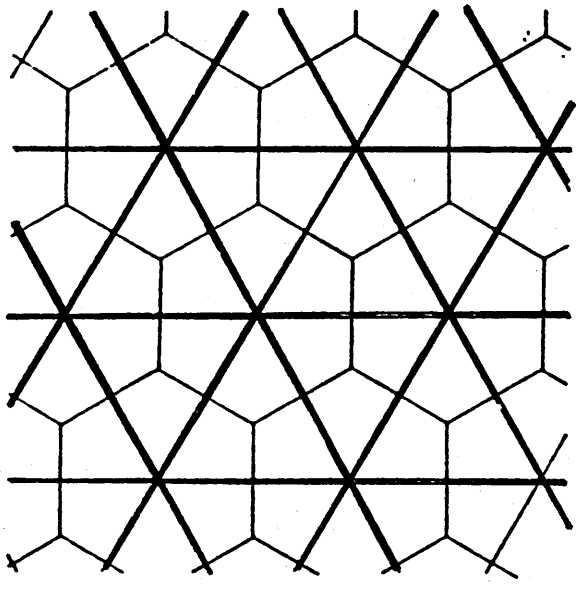


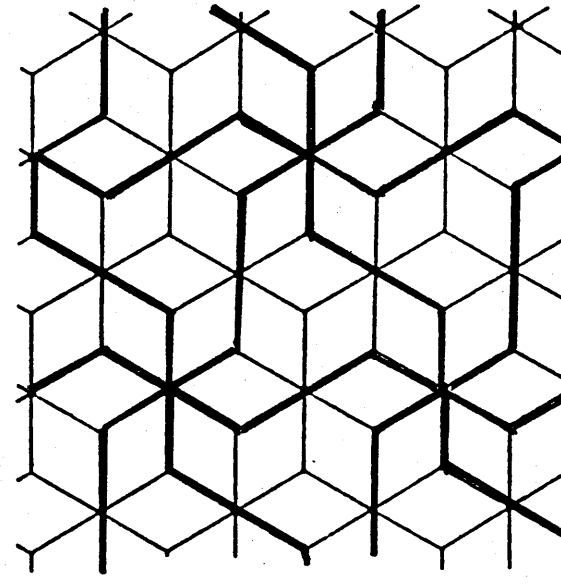
Figure 4.3



$[3^4.6]$



[3.4.6.4]



[3.6.3.6]

Figure 4.4

Type $[4.8^2]$ has obvious 4-amalgamations to types $[4^4]$, $[3^6]$ and $[4.8^2]$, and once again by previous results it has amalgamations to all types not excluded by the lemma.

Now consider type $[3.12^2]$. There is an obvious 3-amalgamation to $[6^3]$, and hence to all types except $[4.8^2]$, $[3.12^2]$ and $[4.6.12]$. Amalgamations of types $[3.12^2]$ and $[4.6.12]$ are excluded as follows. Consider a vertex of type 12 in a uniform, non-trivial amalgamation of a $[3.12^2]$ tiling. It must have arisen from a vertex of degree 12 in the original tiling. But there are only six tiles adjacent to the tiles surrounding a vertex of degree 12 and therefore available for amalgamation with them. But even a 2-amalgamation would require the existence of 12 such tiles. Hence an amalgamation of a $[3.12^2]$ tiling cannot contain a vertex of degree 12.

If one now considers a vertex of degree 8, it is clearly possible to join four pairs of tiles to create a vertex of degree 8, and so it appears that a 2-amalgamation with vertices of degree 8 might be possible (although it is easy to see that the tiles will not all have the same shape). Consider, however, the hexagon surrounding a vertex of order 12 which becomes a vertex of order 8. Only 16 of the 18 tiles in this hexagon will be used in such an amalgamation and so this too is impossible (although the two remaining tiles may possibly amalgamate with corresponding tiles in other hexagons in a 2-amalgamation, the new 2-molecules will lack vertices of degree 8, and so the amalgamation will not be homogeneous).

Finally, consider type $[4.6.12]$. An obvious 2-amalgamation to type $[6^3]$ gives amalgamations to all types except $[4.8^2]$, $[3.12^2]$, and $[4.6.12]$. Amalgamations of these types are shown in figure 4.5

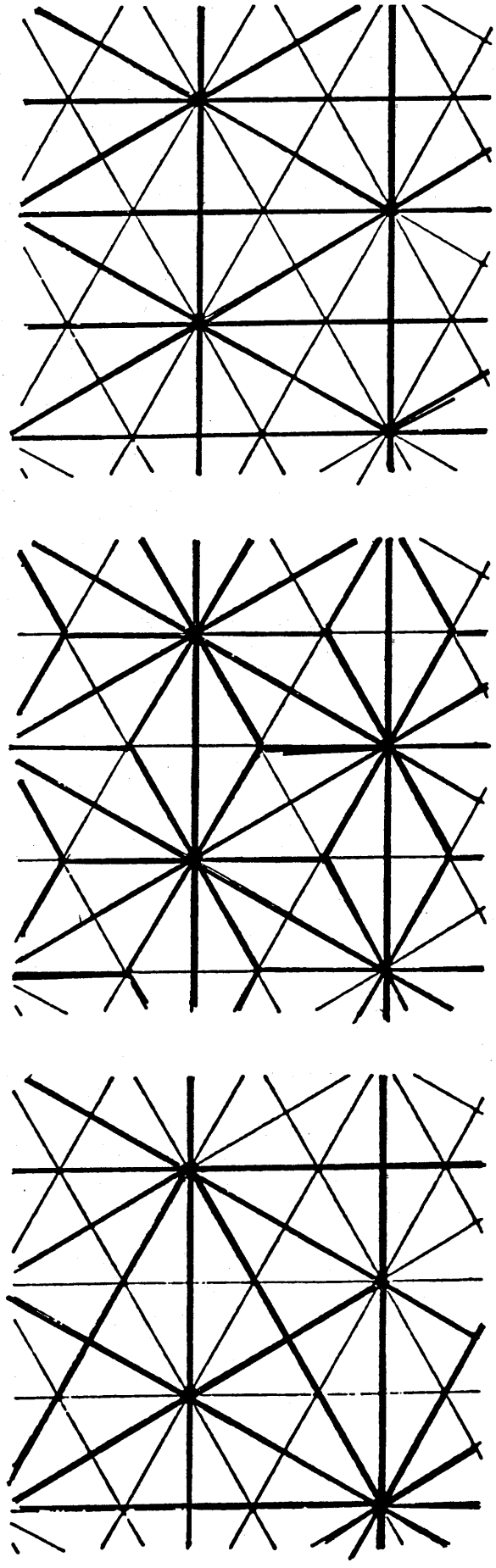


Figure 4.5

Chapter 5

Generation of strongly isohedral amalgamations

In this chapter we will consider an algorithm for the generation of strongly isohedral amalgamations of an isohedral tiling, and show how this can be implemented on a computer to produce strongly isohedral tilings of a *translational* tiling. The algorithms are due to Holroyd [13,12].

5.1 The basic algorithm

Recall that if S is an isohedral amalgamation of a tiling T then S is said to be *strongly isohedral* if $\Sigma(T, S)$, the group of symmetries of T which act as symmetries of S, acts transitively on S.

Definition 5.1 *A subgroup F of $A(T)$ is said to act* effectively *(or semi-regularly) on T if the stabiliser of each tile under the action of F is the identity. If F is both effective and transitive on T, then F is said to act* regularly *on T.*

Theorem 5.2 *Let T be an isohedral tiling, and let F be a subgroup of $\Sigma(T)$ (or the corresponding group of permutations: see remark 1.14 above) which is effective on T, and which partitions T into n orbits. Then:*

1. *any n-tile of T consisting of exactly one tile from each F-orbit generates, by the action of F, a strongly isohedral amalgamation of T on which F acts regularly;*

2. *every strongly isohedral amalgamation of T on which F acts regularly arises in this way.*

Proof:

1. Let t' be an n-tile constructed as described. The every tile of T is the image of one of the tiles making up t' under some element of F; but since F acts effectively, no two F-images of t' can coincide or overlap. Thus the F-images of t' form a strongly isohedral amalgamation of T on which F acts regularly.

2. Let T' be a strongly isohedral amalgamation of T on which F acts regularly. Then any tile $t' \in T'$ must consist of at least one tile from each F-orbit. Since F acts transitively on each F-orbit of T, and regularly on T', t' cannot contain more than one tile from each F-orbit, and is therefore as in the construction of part 1. □

Theorem 5.3 *For every isohedral tiling T, there is at least one subgroup F of $\Sigma(T)$ that acts regularly on T.*

Proof: The method of proof is to examine the list of isohedral types and give a corresponding type which describes a regularly acting subgroup. Types 1–7, 21–25, 27–28, 30–31, 33, 38–39, 41–56, 77–81, 83–88 have symmetry groups which are themselves regular. Types 8, 10–12, 14, 17–20, 57, 62, 64, 68, 72, 74, 76 have transitive subgroups of type p1. Corresponding groups for the remaining isohedral types are given in table 5.1.

Definition 5.4 *A set S of tiles of T is said to be* edge-connected *if, for any two elements $s, t \in S$, there is a string $s_1, s_2, \ldots, s_k$ of elements of S, each s_i being adjacent to s_{i-1} along an edge, such that $s = s_1$ and $t = s_k$.*

Theorem 5.5 *With the notation of theorem 5.2, let S be an edge-connected set of m tiles from different F-orbits, where $m < n$. Then S can be extended to an n-tile consisting of a tile from each F-orbit.*

Isohedral type	Regular subgroup
9	2
13	3
15	2
16	7
26	22
29	28
32	31
34	33
35	33
36	33
37	33
40	39
58	43
59	44
60	45
61	55
63	52
65	48
66	47
67	46
69	46
70	48
71	51
73	44
75	45
82	79
89	87
90	88
91	84
92	87
93	88

Table 5.1 Proof of theorem 5.3

Proof: The union of all F-images of $\mathcal{S}$ cannot cover the whole of $\mathbf{E}^2$, so there must be some F-image of $\mathcal{S}$ which is adjacent along a tile edge to a tile in an F-orbit not represented in $\mathcal{S}$. By transitivity within each orbit, this must be true of $\mathcal{S}$ itself. Thus we may extend $\mathcal{S}$ by steps to an edge-connected set $\mathcal{S}_1$ of n tiles, one from each F-orbit.

It remains to show that $\mathcal{S}_1$ is a topological disc. Suppose it is not. Let $\mathcal{S}_2$ be the smallest topological disc containing $\mathcal{S}_1$ (it will be a union of tiles). The tiles of $\mathcal{S}_2 \setminus \mathcal{S}_1$ belong to F-images of $\mathcal{S}_1$ that are disjoint from $\mathcal{S}_1$; but each of these images is edge-connected, and must also contain tiles not in $\mathcal{S}_2 \setminus \mathcal{S}_1$. This is a contradiction. □

Theorems 5.2, 5.3, 5.5 allow us, for each $n \geq 2$, to generate every strongly isohedral n-amalgamation of T, as follows.

1. Select a fixed tile $a_0 \in T$.

2. For every effective subgroup F of $\Sigma(T)$ which partitions the tiles into n orbits (there are only finitely many of these), and for every n-tile t' containing a_0 which can be constructed as in theorem 5.2 from that partition into orbits, apply F to t' to obtain a tiling.

This algorithm will generate each strongly isohedral amalgamation of T exactly once for each subgroup of $\Sigma(T)$ which acts regularly on that amalgamation.

5.2 Translational Tilings

We now restrict our attention to tilings of the plane which are *translational*, in other words those whose symmetry groups have a transitive subgroup of type p1. Such tilings will always have topological type $[4^4]$ or $[3^6]$. As we have seen, it is only on these tilings that we can define addressing systems which admit a tesseral arithmetic.

In this section we will consider the problem of generating translational amalgamations of tilings which have symmetry group of type p1. To obtain the corresponding list for a tiling with a larger symmetry group it is necessary to eliminate molecules equivalent under that group.

We now consider in detail how to apply the algorithm to a translational tiling T to determine all possible molecular tiles of size s.

Firstly we fix an origin at the centroid of some tile t_0. The centroids of the tiles lie on a lattice generated by two vectors, which we use as a basis, so that the centroids of the tiles are precisely the points $\{(n,m)|n,m \in \mathbf{Z}\}$. From now on we will identify a tile by the coordinates of its centroid, so that t_0 is the tile (0,0).

The next step is to list all of the subgroups of $\Sigma(T)$ of type p1. Any element g of such a subgroup H can be represented with respect to the lattices defined above by a column vector $\mathbf{v}(g)$ with integer coordinates: g is the map $\mathbf{p} \mapsto \mathbf{p} + \mathbf{v}(g)$. If $\mathbf{F} = \{\mathbf{f_1}, \mathbf{f_2}\}$ is an integral basis for H, then let $\mathbf{A}(\mathbf{F})$ be the 2×2 matrix whose columns are $\mathbf{f_1}, \mathbf{f_2}$. $\mathbf{A}$ then specifies H, but $\mathbf{A}$ is not unique: if $\mathbf{A_1}, \mathbf{A_2}$ are two such matrices then

$$\mathbf{A_1} = \mathbf{A_2}\mathbf{L}$$

where $\mathbf{L} \in \mathrm{GL}(2, \mathbf{Z})$ (see [25]). This gives an equivalence relation on the set of 2×2 integer matrices: we need to produce one element from each equivalence class. From theorem A1 of [12] we obtain the following: each equivalence class contains precisely one matrix of the form

$$\begin{pmatrix} p & t \\ 0 & q \end{pmatrix}.$$

where $0 \leq t < q$. For the purposes of this section we will say that a matrix in the above form is in *normal form.*

Since $|\Sigma(T) : H| = |A|$, we only need to list all normal matrices with $pq = s$ in order to get the required enumeration of subgroups, and this is straightforward.

The next step is to attach orbit labels to the tiles of T. We need to label all points within such a distance of t_0 that we can form all s-tiles containing t_0 and consisting of one point from each orbit. The set of tiles contained in the same orbit as t_0, which we call the 0-orbit is the set

$$\{\mathbf{A}(H)\mathbf{X}|\mathbf{X} \text{ is a column vector with integer entries}\}$$

It is sufficient to let the entries of X run from $-s$ to s. Once we have determined the 0-orbit as far out as necessary we determine the other orbits by picking an unlabelled tile (r, s), say, and forming the set $\mathrm{Orb}(t_0) + (r, s)$.

In order to describe the adjacency structure of the lattice of centroids, we define the set of *adjacency vectors* of the lattice as follows. Let v be the degree of the adjacency graph of the lattice, i.e. the number of points adjacent to any point. Then there will be v vectors, $\mathbf{d_1}, \mathbf{d_2}, \ldots, \mathbf{d_v}$ such that for any point P, the points adjacent to P will be precisely the translates of P by the vectors $\mathbf{d_i}$.

Now we can construct all connected molecules m containing t_0 and comprising one tile from each orbit as follows. We start the process with the molecule containing the single tile t_0. Suppose that we have constructed a k-molecule ($k < s$). We add each of the adjacency vectors in turn to each tile in the molecule. If the resulting tile lies in an orbit not already represented, then we can add it to the k-molecule to get a molecule of size $k + 1$. Proceeding in this way we can build up all possible s-molecules.

It only remains to eliminate translational duplicates from the list, and to determine the adjacency structure of the amalgamation generated by the molecule: this process is explained below. Note that since the amalgamation will itself be a translational tiling, it will have topological type $[4^4]$ or $[3^6]$, and so it suffices to determine the number of adjacent tiles in order to determine the adjacency structure.

5.3 Implementation

The algorithm was programmed in PASCAL and run on the DECSYSTEM-20 at the Open University, and on the IBM 3084 at the University of Cambridge. The code is reproduced in appendix A.

We now describe the program section by section.

5.3.1 Declarations

The constant s is the size of molecule we are generating; `offset` and `arraysize` are declared for convenience later on. The constant `valency` will be equal to 4 for the $[4^4]$ tiling, and 6 for the $[3^6]$ tiling.

In order the reduce the space used, we will declare all numbers to lie in the range `0..63`, so that they need a maximum of 6 bits of memory. The program uses as its workspace an square array of size $4 * s + 1$: this is the type `windowsize`. The variable `offset` is used to ensure that all indices are positive. The program works with linked lists of molecules: thus the type `molecule` is in fact a record, consisting of a variable of type `tileset` (representing a set of tiles) together with a pointer, of type `link`, which points to another variable of type `molecule`.

The variable `points` is the program's workspace: this represents an array of tiles centred at the origin, and the entries will correspond to the orbit to which the tile belongs. The variable `adjvecs` will contain the adjacency vectors of the tiling under consideration: notice that since no assumption is made about the orthogonality of the coordinate system used, it is possible to represent the hexagonal tiling in the same way as the square tiling as long as we change the adjacency vectors; this means that the program is virtually identical in each case.

We now describe the procedures of the program in the order they are called by the main section of the program (at the end of the code). This starts by setting up the adjacency vectors, and the initial values of `p`, `q` and `t`, which are the parameters of the subgroup as described in the previous section.

5.3.2 `zeroarray`

Resets the array `points`.

5.3.3 zero_orbit

This procedure determines the zero orbit of the subgroup under consideration. This is done by calculating

$$\begin{pmatrix} p & t \\ 0 & q \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

for x, y lying in the interval $[-2s, 2s]$ and setting the corresponding entries in points to 1 (the kth orbit is stored as $k+1$).

5.3.4 otherorbits

Determines the remaining orbits, as described in comment.

5.3.5 printorbits

Prints out the array points.

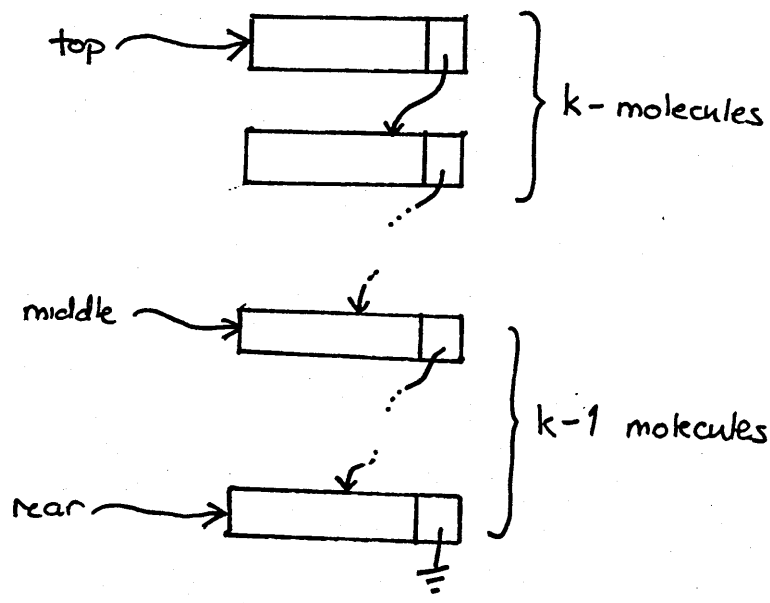
5.3.6 initlist

The molecules generated are placed in a linked list: this is to save space. The molecules are generated by successively producing molecules of size 1,2,3,...,s, and the use of a list structure means that space no longer required can be recovered. The structure of the list is as shown in figure 5.1. This procedure initialises the pointers, and sets the molecules to which they point to "zero".

5.3.7 expand

The procedure expands a molecule by one tile. The adjacency vectors are used to determine the tiles adjacent to the molecule. In each case, if the tile found is in an orbit not so far represented in the tile, then the tile so formed is added to the list. In this way the molecules are built up one stage at a time.

Figure 5.1 Structure of list used in program



5.3.8 addlist

This procedure adds the molecules formed to the list. The molecules are ordered so that molecules which have already been formed are not added to the list.

5.3.9 duplicates and checkdups

The procedures eliminate translational copies of the molecules.

5.3.10 printlist and findstruct

The procedure printlist prints out the molecules which have been found, counting them at the same time. The procedure findstruct is used to determine the adjacency structure of the amalgamation resulting from the molecule. This is only necessary when considering type $[4^4]$, as all amalgamations of a tiling of type $[3^6]$ will also be of this type. It works by counting the number of molecules adjacent to the molecule in the list.

5.3.11 nextnorm

This produces the "next" normal matrix with the specified value of s.

5.4 Results

5.4.1 Enumerative Results

The following tables present the enumerative results from the program. They show the number of molecules arising for each possible set of values of p, q and t for the values of s shown.

5.4.2 Square Lattice

$s = 3$

(p,q)	(1,3)	(3,1)
t		
0	1	1
1		0
2		0

$s = 4$

(p,q)	(1,4)	(2,2)	(4,1)
t			
0	1	1	1
1		2	0
2			2
3			0

$s = 5$

(p,q)	(1,5)	(5,1)
t		
0	1	1
1		0
2		1
3		1
4		0

$s = 6$

(p,q)	(1,6)	(2,3)	(3,2)	(6,1)
t				
0	1	5	5	1
1		4	3	0
2			3	3
3				4
4				3
5				0

$s = 7$

(p,q)	(1,7)	(7,1)
t		
0	1	1
1		0
2		4
3		4
4		4
5		4
6		0

$s = 8$

(p,q)	(1,8)	(2,4)	(4,2)	(8,1)
t				
0	1	13	13	1
1		14	8	0
2			6	8
3			8	6
4				14
5				6
6				8
7				0

5.4.3 Hexagonal Lattice

$s = 3$

(p,q)	(1,3)	(3,1)
t		
0	4	4
1		5
2		4

$s = 4$

(p,q)	(1,4)	(2,2)	(4,1)
t			
0	8	11	8
1		10	10
2			10
3			8

$s = 5$

(p,q)	(1,5)	(5,1)
t		
0	16	16
1		21
2		21
3		21
4		16

$s = 6$

(p,q)	(1,6)	(2,3)	(3,2)	(6,1)
t				
0	32	47	47	32
1		47	47	42
2			42	47
3				47
4				42
5				32

$s = 7$

(p,q)	(1,7)	(7,1)
t		
0	64	64
1		85
2		96
3		85
4		96
5		85
6		64

$s = 8$

(p,q)	(1,8)	(2,4)	(4,2)	(8,1)
t				
0	128	198	198	128
1		198	196	170
2			198	196
3			170	198
4				198
5				196
6				170
7				128

5.4.4 Square Lattice with no checking for adjacency structure

For $3 \leq s \leq 8$ the program was run on a square lattice, but without checking that the adjacency stucture of the amalgamation generated was of type $[4^4]$. This produces *all* possible strongly isohedral translational amalgamations of the $[4^4]$ tiling.

$s = 3$

(p,q)	(1,3)	(3,1)
t		
0	1	1
1		4
2		4

$s = 4$

(p,q)	(1,4)	(2,2)	(4,1)
t			
0	1	5	1
1		6	8
2			6
3			8

$s = 5$

(p,q)	(1,5)	(5,1)
t		
0	1	1
1		16
2		13
3		13
4		16

$s = 6$

(p,q)	(1,6)	(2,3)	(3,2)	(6,1)
t				
0	1	17	17	1
1		18	23	32
2			23	23
3				18
4				23
5				32

$s = 7$

(p,q)	(1,7)	(7,1)
t		
0	1	1
1		64
2		44
3		44
4		44
5		44
6		64

$s = 8$

(p,q)	(1,8)	(2,4)	(4,2)	(8,1)
t				
0	1	53	53	1
1		54	80	128
2			84	80
3			80	84
4				54
5				84
6				80
7				128

5.4.5 Summary table

The table below gives the total number of molecules generated for each value of s in each of the three cases listed above.

$s =$	3	4	5	6	7	8
Square lattice	2	7	4	32	18	106
Hexagonal lattice	17	65	111	504	639	2670
Square, no check	10	35	60	228	306	1044

5.5 Molecule Enumeration

5.5.1 Theoretical Results

In this section we present some theoretical enumerative results on molecule enumeration.

Firstly, note that both in these results, and in the numerical results, we are counting molecules which generate a strongly isohedral amalgamation of a tiling which has p1 as symmetry group, and on which p1 is effective; the tiles of the tiling have trivial symmetry group and stabiliser. The molecules generated by a subgroup of the symmetry group of the tiling are the edge-connected orbit transversals of that subgroup.

The first result gives a crude upper bound on the number of possible molecules of size s of a tiling by tiles with v edges.

Theorem 5.6 *If $\mathcal{M}_{v,s}$ is the number of possible s-molecules, counted up to equivalence, of a tiling by tiles with v edges, then*

$$\mathcal{M}_{v,s} \leq \frac{1}{2^{s-1}} \prod_{k=1}^{s-1} [k(v-2)+2].$$

Proof. Imagine building up an s-molecule in stages; at each stage we add a tile to the existing partial molecule. At each stage we increase the number of edges of the partial molecule, and hence the number of candidate tiles to be added, by at most $v-2$. Hence

$$\mathcal{M}_{v,s} \leq v(v+(v-2))(v+2(v-2))\ldots(v+(s-2)(v-2))$$

$$
\begin{aligned}
&= v(2v-2)(3v-4)(4v-6)\ldots((s-1)v-2(s-2)) \\
&= \prod_{k=1}^{s-1}[kv-2(k-1)] \\
&= \prod_{k=1}^{s-1}[k(v-2)+2]
\end{aligned}
$$

However, even after eliminating translational duplicates, each possible molecule will be produced more than once in this way. Given an s-molecule, consider the number of ways of deleting tiles successively, so that at each stage we still have a molecule. Each distinct sequence of deletions corresponds to a distinct way of building up the s-molecule. Clearly at each stage except the final one there are two possible tiles which can be deleted; hence each molecule is produced at least 2^{s-1} times, and the result follows. In particular we have

$$\mathcal{M}_{4,s} = \frac{1}{2^{s-1}} \prod_{k=1}^{s-1} 2(k+1) = s!$$

$$\mathcal{M}_{6,s} = \frac{1}{2^{s-1}} \prod_{k=1}^{s-1} 2(2k+1) < (2s-1)!$$

□

The upper bound presented above is crude because it takes no account of the action of the symmetry groups. In one or two special cases we can get exact formulae for the number of molecules.

In each of the three cases below, we consider the molecules generated by the action of a subgroup H of the symmetry group of a translational tiling T whose tiles have trivial symmetry group.

Theorem 5.7 *If H has matrix of the form* $\begin{pmatrix} s & 0 \\ 0 & 1 \end{pmatrix}$ *or* $\begin{pmatrix} 1 & 0 \\ 0 & s \end{pmatrix}$ *and T is of type $[4^4]$ then there is precisely one molecule possible.*

Proof. The orbits of H in these cases are infinite horizontal or vertical strips; it is clear that the only possible edge-connected orbit transversal is a strip of s tiles. All such strips are translationally equivalent. □

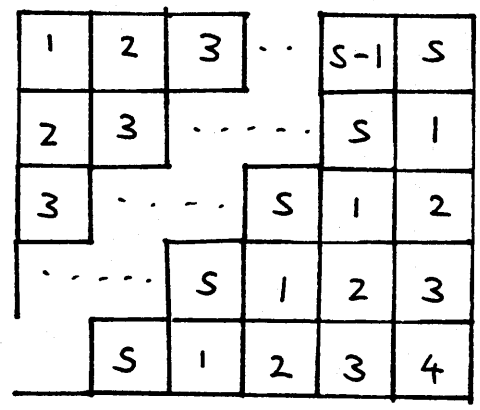


Figure 5.2

Theorem 5.8 *If H has matrix of the form* $\begin{pmatrix} s & 1 \\ 0 & 1 \end{pmatrix}$ *and T is of type* $[4^4]$ *then there are $2^{(s-1)}$ possible molecules, all of which, however, give rise to amalgamations of type* $[3^6]$.

Proof. The orbits of H in this case form diagonal strips (see figure 5.2). This means that every molecule has a unique "top left" tile; in other words the leftmost tile in the topmost horizontal strip of the tile is also the topmost tile of the leftmost vertical strip of the tile. Without loss of generality we can choose this tile to be in the orbit labelled 1. Imagine building up the molecule tile by tile, always maintaining connectedness. It is clear that there will always be two possible choices of tile to add, those to the right and below the tile last added. Hence there are $2^{(s-1)}$ possible molecules. Since, however, at each vertex two tiles belong to the same orbit, no amalgamation by one of these molecules could contain a vertex of degree 4, and hence there are no amalgamations of type $[4^4]$. □

Theorem 5.9 *If H has matrix of the form* $\begin{pmatrix} s & 0 \\ 0 & 1 \end{pmatrix}$ *or* $\begin{pmatrix} 1 & 0 \\ 0 & s \end{pmatrix}$ *and T*

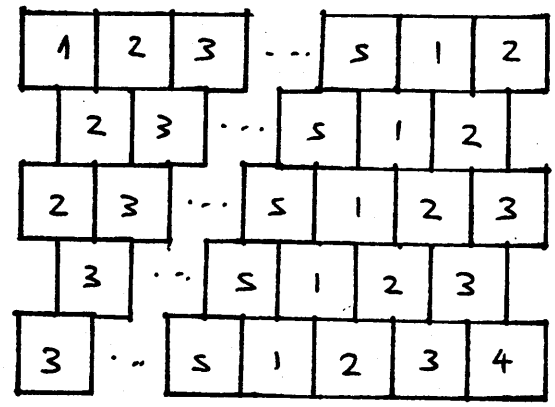


Figure 5.3

is of type $[3^6]$ *then there are* $2^{(s-1)}$ *possible molecules.*

Proof. The orbits again form diagonal strips (see figure 5.3), and the same argument as in the previous proof applies. □

5.5.2 Larger symmetry groups

We reiterate that the results above are limited in the sense that they asssume that the underlying tiling T has symmetry group p1. As such they can be used to derive all molecules of any translational tiling which give rise to translational amalgamations, by eliminating those molecules which are equivalent under the symmetry group of the tiling (the amount of work required would be reduced by only considering one representative of each conjugacy class of effective subgroups).

In order to enumerate molecules of a non-translational tiling, and/or those which give rise to non-translational amalgamations, it would be necessary to generate *all* conjugacy classes of effective subgroups, and not just those of type p1.

5.5.3 Related work

One area clearly related to the work described here is the enumeration of polyominoes, although most authors seem to allow polyominoes to contain "holes". See, for example, [21,18,15,20]. The algorithm described by Redelmeier in [21] bears some resemblance to the one described here, in that it builds up polyominoes tile by tile, storing the partial polyominoes in a linked list. Another paper of interest is [1], in which it is shown that every hexomino tiles the plane, and that of the 35 hexominoes, 24 tile the plane translationally.

Another related area is the enumeration of carbon compounds, since these can be seen as polyominoes with a particular structure. Work in this field is described in [10,7,8,9].

In neither of the above cases is the action of the effective subgroup important. One question of interest in which it is important involves the concept of a *perfect colouring* (see [23] and elsewhere).

A *k-colouring* of a tiling is a map $\chi : T \to \{1, 2, \ldots, k\}$. A *coloured tiling*, $\hat{T}$ is a pair (T, χ), where χ is a colouring of T. A *colour symmetry* of $\hat{T}$ is a pair $\hat{\gamma} = (\gamma, \theta)$, where $\gamma \in \Sigma(T)$, $\theta \in S(k)$, the permutation group on $\{1, 2, \ldots, k\}$, and γ and θ are *compatible* in the sense that for any tile $t \in T$,

$$\theta(\chi(t)) = \chi(\theta(t)).$$

We say that γ and θ are *associated* with $\hat{\gamma}$ (and with each other): if every element $\gamma \in \Sigma(T)$ is associated with a colour symmetry $\hat{\gamma}$ we say that the colouring is *perfect*.

Clearly every labelling by subgroup orbits produced above is a colouring of the underlying tiling: the question then arises as to when that colouring is perfect. The answer for the regular translational tilings follows from results 8.7.1 and 8.7.2 of [6], and is as follows:

Theorem 5.10 *If T is the Laves net of type $[4^4]$ then the translational subgroups of $\Sigma(T)$ with matrices* $\begin{pmatrix} n & 0 \\ 0 & n \end{pmatrix}$ *or* $\begin{pmatrix} 2n & 0 \\ 0 & n \end{pmatrix}$ *induce perfect*

colourings of T. If T is the Laves net of type $[3^6]$ then the corresponding matrices are $\begin{pmatrix} n & 0 \\ 0 & n \end{pmatrix}$ *or* $\begin{pmatrix} 3n & 0 \\ 0 & n \end{pmatrix}$.

See [24] for further results about perfect colourings of isohedral tilings.

5.6 Enumerating amalgamations

The amalgamations resulting from the molecules given above all lie in distinct symmetric isomorphism classes of tilings. However, since they all have symmetry group p1 and tile group c1, they are of isohedral type 1 or 41, according as they are of topological type $[3^6]$ or $[4^4]$.

By considering different underlying tilings T we can generate all possible translational, strongly isohedral, amalgamations. Once again, by considering one representative of each conjugacy class of effective subgroups, and eliminating molecules equivalent under $\Sigma(T)$, we will obtain precisely one representative of each symmetric isomorphism class of amalgamations. It is not immediately clear how to proceed to a coarser classification of these amalgamations (e.g. up to isomerism): one possibility might be by the use of the *incidence symbols* devised by Grünbaum and Shepherd (see [6]).

5.7 Enumerating amalgamators

We now consider how the above algorithm enables us to enumerate amalgamators of a tiling T. More particularly, we can derive from the lists of molecules generated above all *affine* amalgamators of T, where an affine amalgamator α of a translational tiling T is an amalgamator such that $\alpha(T)$ is also translational.

We will say that two amalgamators α, β of T are *equivalent* if there is a symmetry $\gamma \in \Sigma(T)$ with $\beta = \gamma^{-1}\alpha\gamma$.

We have already noted that the tiles of T lie on a lattice with origin t_0, generated by two vectors, which we have taken as our basis. Suppose now

that m_0 is a molecule, generated by the effective p1 subgroup H of $\Sigma(T)$, which generates an amalgamation of T which is isomorphic to T.

Now, m_0 contains t_0, by construction. The images of t_0 under H, in other words the orbit of H containing t_0, also lie on a lattice with origin t_0, generated by the vectors $\mathbf{f}_1, \mathbf{f}_2$, the columns of the normal matrix $\mathrm{A}(H)$ of H. For the moment, identify a molecule of the amalgamation by the co-ordinate of the (unique) image of t_0 which it contains. Then we can define a map from T to its amalgamation by

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto x\mathbf{f}_1 + y\mathbf{f}_2.$$

This map is an isomorphism, and hence is an amalgamator α.

Now let $\mathrm{M}(m_0)$ be the matrix whose columns are the position vectors of the constituent atomic tiles of m_0. Then α is precisely the map

$$P \mapsto \mathrm{M}(m_0) + \mathrm{A}(H)P$$

for any integer column vector P, representing a tile of t.

Notice also that if X is any integer column vector, and $[\mathrm{X}]$ is the $2 \times s$ matrix $(X \ldots X)$, then

$$P \mapsto \mathrm{M}(m_0) + [\mathrm{X}] + \mathrm{A}(H)P$$

is also an amalgamator. Furthermore, any affine amalgamator of T arises this way. Hence the algorithm above gives us a way of generating, for each aperture s, a finite set $\mathcal{M}$ of matrix pairs (M, A) such that

$$\mathcal{S} = \{(\mathrm{M} + [\mathrm{X}], \mathrm{A}) : (\mathrm{M}, \mathrm{A}) \in \mathcal{M}, \mathrm{X} \in \mathbf{Z}^2\}$$

represents the set of all affine amalgamators of T of aperture s.

It remains now to act on $\mathcal{S}$ with $\Sigma(T)$, to produce one member of each equivalence class of amalgamators.

Any symmetry $\gamma \in \Sigma(T)$ can be represented by a pair $(\mathbf{S}, \mathbf{Z})$, where $\mathbf{S}$ is an integer matrix representing a rotation or reflection and $\mathbf{Z}$ is an integer column vector representing a translation (since we are assuming that T is one of the Laves nets $[4^4]$ or $[3^6]$, $\Sigma(T)$ contains no glide reflections).

Now, suppose that $\mathbf{P}$ is an integer column vector representing the atom p, and that $(\mathbf{M}, \mathbf{A}) \in \mathcal{S}$. Then $\gamma^{-1}(p)$ is represented by the column vector $\mathbf{S}^{-1}\mathbf{P} - \mathbf{S}^{-1}\mathbf{Z}$. The molecule $\alpha\gamma^{-1}(p)$ is represented by the matrix

$$\mathbf{A}[\mathbf{S}^{-1}\mathbf{P} - \mathbf{S}^{-1}\mathbf{Z}] + \mathbf{M}$$

and thus $\gamma\alpha\gamma^{-1}(p)$ by the matrix

$$\begin{aligned} & SA[S^{-1}P - S^{-1}Z] + SM + [Z] \\ = \quad & SAS^{-1}[P] + (SM + (SAS^{-1} - I)[Z]). \end{aligned}$$

It follows that the matrix pair representing $\gamma\alpha\gamma^{-1}$ is

$$(\mathbf{SM} + (\mathbf{SAS}^{-1} - \mathbf{I})[\mathbf{Z}], \mathbf{SAS}^{-1})$$

So the set of equivalence classes of amalgamators is given by:

$$\mathcal{S}/\Sigma(T) = \left\{ \left\{ (\mathbf{SM} + (\mathbf{SAS}^{-1} - \mathbf{I})[\mathbf{Z}], \mathbf{SAS}^{-1}) \right\}_{(\mathbf{S},\mathbf{Z}) \in \Sigma(T)} \right\}_{(\mathbf{M},\mathbf{A}) \in \mathcal{S}}$$

Chapter 6

Classification of amalgamators by sequence

In this chapter we define various numerical sequences which could be used to classify amalgamators and the resulting tiling hierarchies, summarise what is know so far about their behaviour, and indicate possible directions for further investigation.

6.1 Tiling Hierarchies

In chapter 2 we introduced the concept of an *amalgamation* of a tiling T, and defined the concept of a *tiling hierarchy* arising from an amalgamator α. We also defined the groups $H(T,\alpha)$, and $\Sigma(T,\alpha)$, consisting of those automorphisms and symmetries respectively of T which act as automorphisms and symmetries of $\alpha(T)$.

Suppose now that T is a tiling, and α is an amalgamator on T. Then $\mathcal{T}(T,\alpha)$ is the tiling hierarchy generated by T and α; i.e. the sequence of tilings $\{T, \alpha(T), \alpha^2(T), \ldots\}$. We refer to T as the *base level tiling*, and to $\alpha^k(T)$ as the kth level tiling.

Definition 6.1 *If $\mathcal{T}(T,\alpha)$ is a tiling hierarchy, let j and k be integers with $0 \leq j < k$ and let ϕ be an automorphism of $\alpha^j(T)$. The tiles of $\alpha^k(T)$ are molecules of $\alpha^j(T)$, and so we can consider the restriction of ϕ to this tiling, $\phi|_{\alpha^k(T)}$, which we will denote $\phi|_k$. This is an isomorphism from $\alpha^k(T)$ to*

some tiling S which is an amalgamation of $\alpha^j(T)$ (and hence of T itself).

If it is actually the case that $S = \alpha^k(T)$, so that $\phi|_k$ is an automorphism of $\alpha^k(T)$, then ϕ is said to act at level k.

Conversely, if ψ is an automorphism of $\alpha^k(T)$, and also $\psi = \phi|_k$ for some automorphism ϕ of $\alpha^j(T)$, then ψ is said to extend to level j.

If L is a subgroup of $A(\alpha^j(T))$ each of whose elements acts at level k, then we denote the corresponding subgroup of $A(\alpha^k(T))$ by $L|_k$.

Remark 6.2 *If $\mathcal{T}(T,\alpha)$, j,k,ϕ are as above then, since $\alpha^k \circ \alpha^{-k}$ is the identity automorphism on $\alpha^k(T)$, $\phi \circ \alpha^k \circ \alpha^{-k}$ is well-defined and equal to $\phi|_k$.*

Definition 6.3 *An automorphism of $\alpha^k(T)$ which extends to each of the lower levels is said to be an* hierarchical automorphism.

Similarly, a symmetry of $\alpha^k(T)$ which extends to each of the lower levels is said to be an hierarchical symmetry.

We denote the hierarchical automorphism and symmetry groups of $\alpha^k(T)$ by $H_k(T,\alpha)$ and $\Sigma_k(T,\alpha)$ respectively (we may shorten these to H_k, Σ_k where no confusion can be caused).

Definition 6.4 *The kth level tiling of a tiling hierarchy $\mathcal{T}(T,\alpha)$ is said to be* hierarchically transitive *(respectively* hierarchically isohedral*) if H_k (respectively Σ_k) acts transitively on the kth level tiling.*

If a group acts transitively on the tiles of a tiling, then the tiling must be isohedral, and the action of the group must be equivalent to the action of one of the 93 IH types. If we adopt the convention that the IH type of a group which is not transitive is 0, then we can make the following definitions.

Definition 6.5 *The numeral corresponding to the IH type equivalent to the action of H_k (respectively Σ_k) is called the* hierarchical algebraic type *(respectively* hierarchical isohedral type*) of $\alpha^k(T)$, denoted by h_k (respectively σ_k).*

The simple isohedral type *of $\alpha^k(T)$ is its isohedral type considered without reference to the lower level tilings; it is denoted s_k.*

Definition 6.6 *For a given tiling hierarchy $\mathcal{T}(T,\alpha)$ we define the following three sequences: the* hierarchical algebraic sequence

$$\tilde{h}(T,\alpha) = \{h_0, h_1, h_2, \ldots\};$$

the hierarchical isohedral sequence

$$\tilde{\sigma}(T,\alpha) = \{\sigma_0, \sigma_1, \sigma_2, \ldots\};$$

and the simple isohedral sequence

$$\tilde{s}(T,\alpha) = \{s_0, s_1, s_2, \ldots\}.$$

Definition 6.7 *Given two non-zero IH types n, m we say that n* dominates *m, and write $n \succeq m$ if a group whose action is of type n contains a subgroup whose action is of type m. We also adopt the convention that $0 \preceq n \ \forall n, 0 \leq n \leq 93$.*

Remark 6.8 *The arrangement of the 93 IH types has the consequence that*

$$n \succeq m \Longrightarrow n \geq m.$$

6.2 Properties of the sequences

We firstly summarise the results presented in [11].

Theorem 6.9 *1. $\tilde{h}(T,\alpha)$ is determined by the isomorphism class of $\mathcal{T}(T,\alpha)$.*

2. $\tilde{\sigma}(T,\alpha)$ is determined by the isomerism class of $\mathcal{T}(T,\alpha)$.

Theorem 6.10 *1. $\tilde{\sigma}$ is dominated by both $\tilde{h}$ and $\tilde{s}$.*

2. If T is maximally symmetric then $\tilde{\sigma} = \tilde{h}$.

Theorem 6.11

$$h_0 \succeq h_1 \succeq h_2 \succeq h_3$$

$$h_k = h_3 \ \forall k \geq 3$$

This last theorem implies that the hierarchical algebraic sequence settles down very quickly. The same is not true of the two isohedral sequences: indeed the existence of spurious symmetries implies that the simple isohedral sequence is not necessarily determined by the isomerism class of the hierarchy, but may depend on particular geometric properties of the base tiles, such as the relative lengths of their sides.

It is, however, possible to determine the ultimate behaviour of the hierarchical algebraic sequence: we have the following theorem.

Theorem 6.12 *For any tiling hierarchy $\mathcal{T}(T,\alpha)$ the hierarchical isohedral sequence $\tilde{\sigma}(T,\alpha)$ is eventually either constant or periodic.*

Proof. Firstly we define the groups R_1, L_1 as follows:

$$R_1 = \{\phi \in A(T) : \alpha\phi = \psi\alpha \text{ for some } \psi \in A(T)\}$$

$$L_1 = \{\psi \in A(T) : \alpha\phi = \psi\alpha \text{ for some } \phi \in A(T)\}.$$

We can define an isomorphism $\lambda : R_1 \to L_1$ as follows:

$$\phi \mapsto \psi \Leftrightarrow \alpha\phi = \psi\alpha.$$

Lemmas 1 and 2 of [11] state that L_k is the group of automorphisms of T which act at all levels up to and including k, and that

$$L_k|_k = H_k = \alpha^k R_k \alpha^{-k}$$

from which is follows that the action of R_k on T is of the same IH type as the action of H_k on $\alpha^k(T)$.

Now, for any positive integer k, we denote the domain of λ^k by R_k, and set $L_k = \lambda^k(R_k)$. Finally, set $R_0 = L_0 = A(T)$.

Assume that $\mathcal{T}(T,\alpha)$ is actually hierarchically isohedral, so that

$$\sigma_k > 0 \; \forall k > 0.$$

Then it is certainly hierarchically transitive. It follows that

$$R_k = R_3 \;\; \forall k \geq 3.$$

Let $k \geq 3$. Now, from Lemma 1 of [11], we know that L_k is the group of automorphisms of T that act at all levels up to and including level k, so that $L_k|_k$ is simply H_k, the hierarchical automorphism group of $\alpha^k(T)$. Furthermore, $L_k \cap \Sigma(T)$ is the group of symmetries of T that act at all levels up to and including level k, so that

$$(L_k \cap \Sigma(T))|_k = \Sigma_k,$$

the hierarchical symmetry group of $\alpha^k(T)$.

Thus, Σ_k is a subgroup of L_k. Since, for the moment, we are assuming that Σ_k is transitive on $\alpha^k(T)$, the index of Σ_k in L_k must divide the order of the stabiliser in L_k of a tile, which means it must be 1,2,3,4,6,8 or 12. It follows that there only are finitely many possibilities for Σ_k, for each k. Now, $\hat{\Sigma_k} = \lambda^{-k}(\Sigma_k)$ is a subgroup of R_k, which (from above) is equal to R_3. Hence there are only a finite number of possibilities for $\hat{\Sigma}_k$ for *all* $k \geq 3$.

Now it follows that for some $k \geq 3$ and $l \geq 1$ we have $\hat{\Sigma}_k = \hat{\Sigma}_{k+l}$. By considering the hierarchies $\mathcal{T}(T, \alpha)$ and $\mathcal{T}(\alpha^l(T), \alpha)$ it is easy to see that $\hat{\Sigma}_{k+1} = \hat{\Sigma}_{k+l+1}$, and so we get periodicity.

Now consider the case when $\sigma_p = 0$ for some value of p. Either $\sigma_k = 0 \ \forall k > p$, in which case $\tilde{\sigma}$ is eventually constant, or there is an integer q such that $\sigma_q = 0$, but $\sigma_{q+1} > 0$. Without loss of generality, we can assume that $q > 2$, so that, as above, there are only a finite number of possibilities for $\hat{\Sigma}_{q+1}$. If $\tilde{\sigma}$ is not eventually zero, then we must have k, l such that $\hat{\Sigma}_k = \hat{\Sigma}_{k+l}$, and periodicity follows as above. □.

6.3 Enumeration of sequences

Since the hierarchical algebraic and isohedral sequences of a tiling hierarchy are determined by, respectively, the isomorphism and isomerism classes of that hierarchy, theoretical results about the behaviour of these sequences would be of value in obtaining enumerative results about tiling hierarchies. There is still a long way to go in this field, in particular in trying to get some hold on the behaviour of the *simple* isohedral sequence. We finish this

chapter with two results from [11] which give specific results about possible values of $\tilde{h}$ (which, since it is completely determined by its first four values, is probably the most tractable of the three sequences to deal with).

Theorem 6.13 *If T is a tiling such that $A(T)$ is of type 21, 26, 76, 77, 82, or 93, then there is a strict amalgamator α on T such that every term of $\tilde{h}(T,\alpha)$ is equal to $h_0(T,\alpha)$. This is not the case if $A(T)$ is of type 20, 29, 32, 37 or 40. Indeed, if $A(T)$ is of type 40, then T posesses no strict amalgamators. (These IH types are in fact the IH types of the Laves nets, i.e. the maximally symmetric types of the eleven topological types.)*

This theorem is proved using argument similar to those in chapter 4.

Theorem 6.14 *There exist tiling hierarchies with the following hierarchical algebraic sequences:*

1. $\{20, 11, 11, 11, \ldots\}$
2. $\{20, 17, 17, 17, \ldots\}$
3. $\{20, 18, 18, 18, \ldots\}$
4. $\{29, 27, 27, 27, \ldots\}$
5. $\{29, 28, 28, 28, \ldots\}$
6. $\{32, 30, 30, 30, \ldots\}$
7. $\{32, 31, 31, 31, \ldots\}$
8. $\{37, 34, 34, 34, \ldots\}$

   There do not exist tiling hierarchies with the following hierarchical algebraic sequences:

9. $\{20, 19, 19, 19, \ldots\}$
10. $\{37, 35, 35, 35, \ldots\}$
11. $\{37, 36, 36, 36, \ldots\}$

The results establishes "the best we can do" in the way of preserving symmetry if $A(T)$ is of type 20, 29, 32 or 37.

Bibliography

[1] F.W. Barnes. Every hexomino tiles the plane. *Journal of Combinatorics, Information and System Sciences*, 8(2):113–115, 1983.

[2] Sarah B.M. Bell, Bernard M. Diaz, Fred Holroyd, and Michael J. Jackson. Spatially referenced methods of processing raster and vector data. *Image and Vision Computing*, 1:211–220, 1983.

[3] C.R. Dyer and A. Klinger. Experiments on picture representation using regular decomposition. *Computer Graphics and Image Processing*, 5:68–105, 1976.

[4] I. Gargantini. An effective way to represent quadtrees. *Communications of the ACM*, 25:905–910, 1982.

[5] Branko Grünbaum and G.C. Shephard. The eighty one types of isohedral tilings in the plane. *Mathematical Proceedings of the Cambridge Philosophical Society.*, 82:177–196, 1977.

[6] Branko Grünbaum and G.C. Shephard. *Tilings and Patterns.* W.H. Freeman, San Francisco, 1987.

[7] Henry R. Henze and Charles M. Blair. The number of isomeric carbons of the methane series. *J. Amer. Chem. Soc.*, 53:3077–3085, August 1931.

[8] Henry R. Henze and Charles M. Blair. The number of stereoisomeric and non-stereoisomeric mono- substitution products of the paraffins. *J. Amer. Chem. Soc.*, 54:1098–1106, March 1932.

[9] Henry R. Henze and Charles M. Blair. The number of stereoisomeric and non-stereoisomeric paraffin hydrocarbons. *J. Amer. Chem. Soc.*, 54:1538–1545, April 1932.

[10] Henry R. Henze and Charles M. Blair. The number of structurally isomeric alcohols of the methanol series. *J. Amer. Chem. Soc.*, 53:3042–3046, August 1931.

[11] F. Holroyd. The geometry of tiling hierarchies. *Ars Combinatoria*, 16B:211–244, December 1983.

[12] F.C. Holroyd. *Addressing Systems for Digital Pictures.* Discussion Paper 2, Centre for Configurational Studies, The Open University, Milton Keynes, July 1985.

[13] F.C. Holroyd. Amalgamations of isohedral tilings. The Open University.

[14] F.C. Holroyd. A design for fast hardware implementation of integer matrix operations. April 1987. The Open University.

[15] D.A. Klarner and R.L. Rivest. A procedure for improving the upper bound for the number of n-ominoes. *Canad. J. Math.*, 25(3):585–602, 1973.

[16] E.H. Lockwood and R.H. Macmillan. *Geometric Symmetry.* Cambridge University Press, 1978.

[17] D. Lucas. A multiplication in n-space. *Proc. Amer. Math. Soc.*, 74:1–8, 1979.

[18] W.F. Lunnon. Counting polyominoes. In A.O.L. Atkin and B.J. Birch, editors, *Computers in Number Theory*, pages 347–372, Academic Press, London, 1971.

[19] Dylan Morgan. Arithmetic made easy. *New Scientist*, 94(1302):221–223, April 1982.

[20] B.M.I. Rands and D.J.A. Welsh. Animals, trees and renewal sequences. *IMA Journal of Applied Mathematics*, 27:1–17, 1981.

[21] D. Hugh Redelmeier. Counting polyominoes: yet another approach. *Discrete Mathematics*, 36(2):191–203, 1981.

[22] Hanan Samet. The quadtree and related hierarchical data structures. *ACM Computing Surveys*, 16(2), June 1984.

[23] R.L.E. Schwarzenberger. Colour symmetry. *Bull. Lond. Math. Soc.*, 16:209–240, 1984.

[24] M. Senechal. Color groups. *Discrete Appl. Math.*, 1:51–73, 1979.

[25] M. Senechal. A simple characterization of the subgroups of space groups. *Acta Cryst.*, A36:845–650, 1980.

[26] W.J.G. Wingate. Tesseral arithmetic in more than two dimensions. In B.M. Diaz and S.B.M. Bell, editors, *Spatial Data Processing using Tesseral Methods*, NERC Unit for Thematic Information Systems, National Environmental Research Council, Swindon, 1986.

Appendix A

A table of the 93 marked isohedral tilings

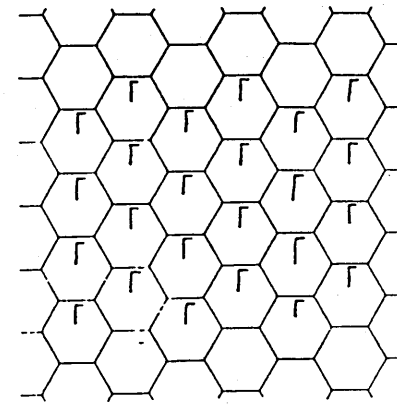
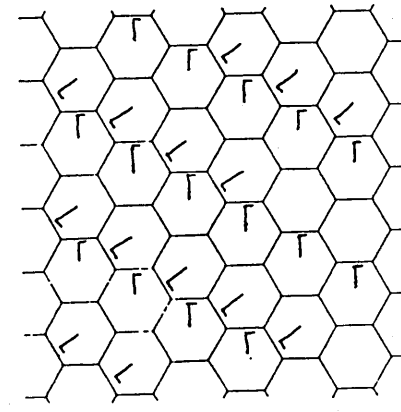
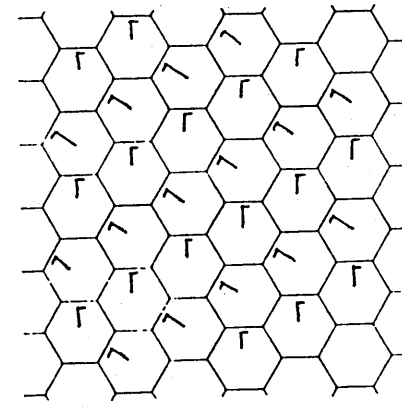
The classification is that given in [5].

For each isohedral type, the incidence symbol, symmetry group and tile stabiliser are given, together with a marked tiling of that type. The mark chosen is based on an asymmetric L, such that the short edge of the L always lies along the edge which corresponds to the letter a in the tile symbol.

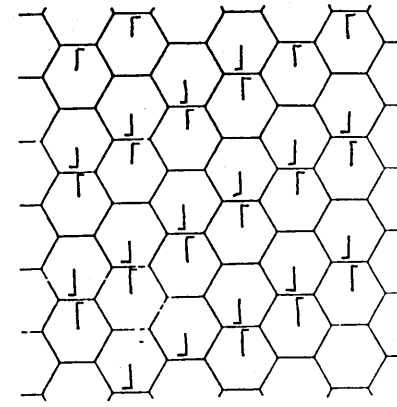
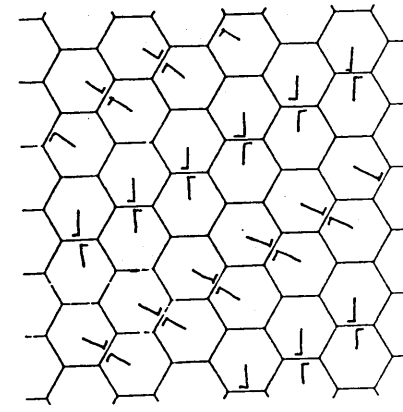
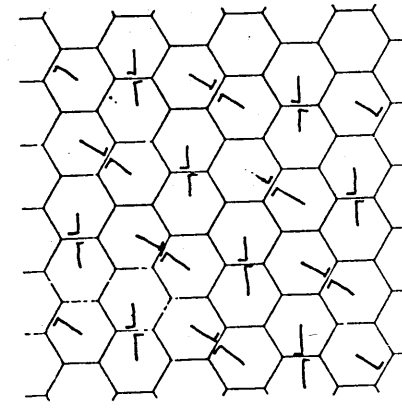
For types 20, 21, 26, 29, 32, 37, 40, 76, 77, 82 and 93 which include the apropriate Laves tiling, no markings have been given.

The following isohedral types have no unmarked representations: 19, 35, 48, 60, 63, 65, 70, 75, 80, 87, 89, 92.

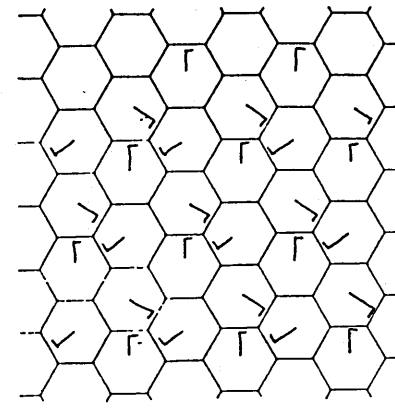
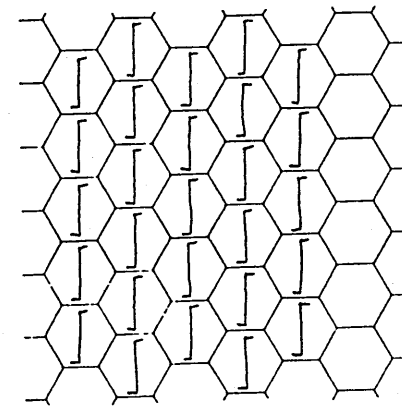
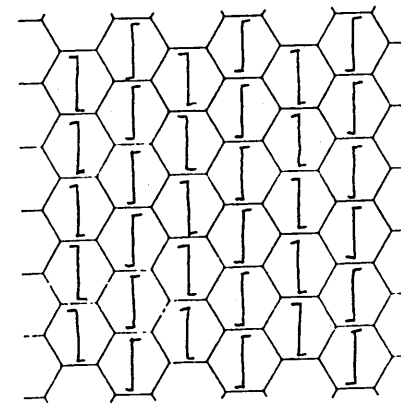
topological type: $[3^6]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
1	$[a^+b^+c^+d^+e^+f^+,\ d^+e^+f^+a^+b^+c^+]$	$p1$ e	
2	$[a^+b^+c^+d^+e^+f^+,\ b^-a^-f^+e^-d^-c^+]$	pg e	
3	$[a^+b^+c^+d^+e^+f^+,\ c^-e^+a^-f^-b^+d^-]$	pg e 90	

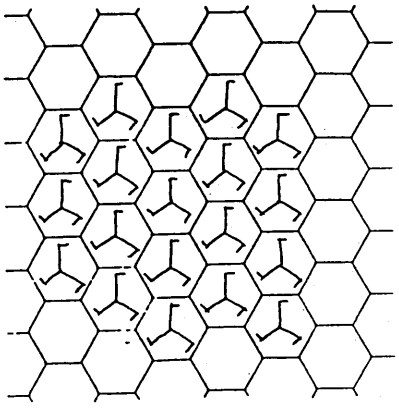
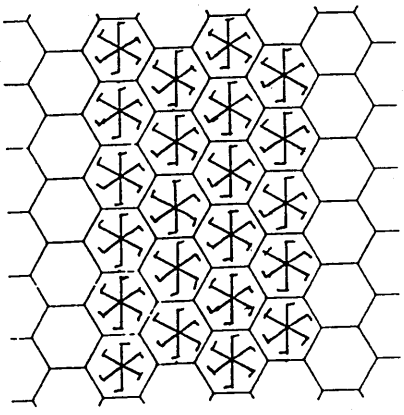
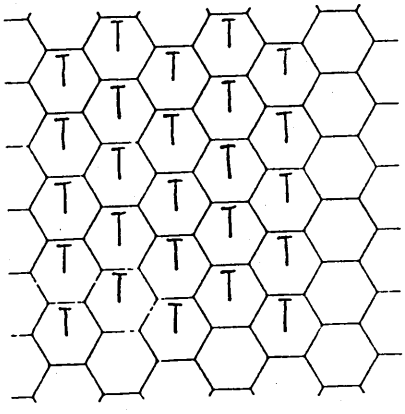
topological type: $[3^6]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
4	$[a^+b^+c^+d^+e^+f^+;\ a^+e^+c^+d^+b^+f^+]$	p2 e	
5	$[a^+b^+c^+d^+e^+f^+;\ a^+e^+d^-c^-b^+f^+]$	pgg e	
6	$[a^+b^+c^+d^+e^+f^+;\ a^+e^-c^+f^-b^-d^-]$	pgg e	

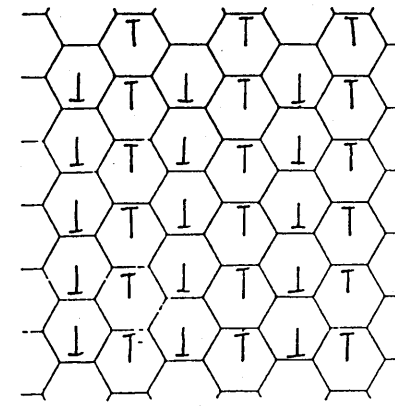
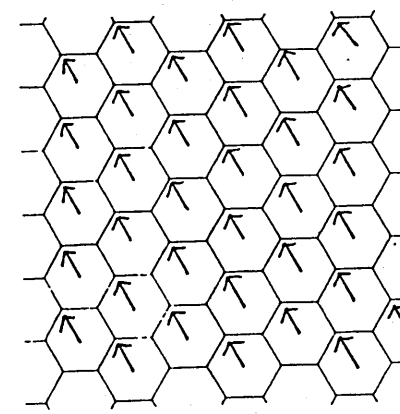
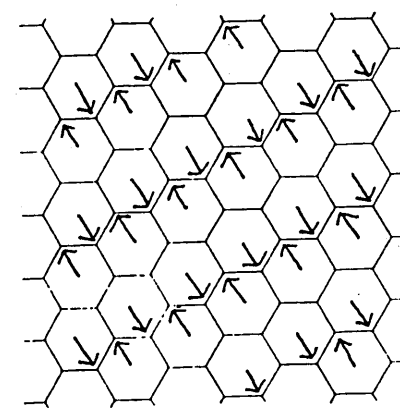
topological type: $[3^6]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
7	$[a^+b^+c^+d^+e^+f^+;\ b^+a^+d^+c^+f^+e^+]$	p3 e	
8	$[a^+b^+c^+a^+b^+c^+,\ a^+b^+c^+]$	p2 c2	
9	$[a^+b^+c^+a^+b^+c^+;\ a^+c^-b^-]$	pgg c2	

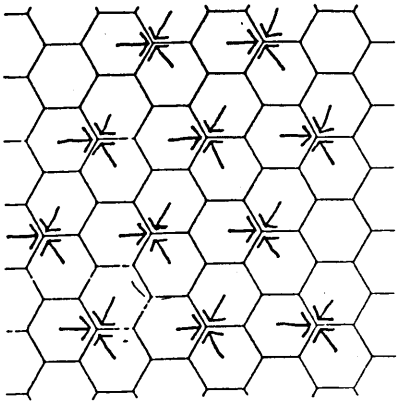
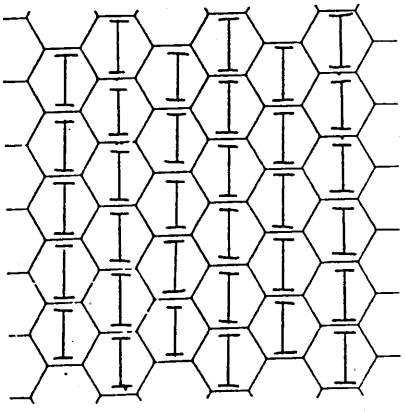
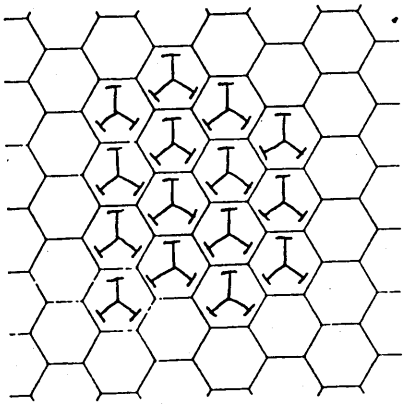
topological type: $[3^6]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
10	$[a^+b^+a^+b^+a^+b^+;\ b^+a^+]$	$p3$ $c3$	
11	$[a^+a^+a^+a^+a^+a^+;\ a^+]$	$p6$ $c6$	
12	$[ab^+c^+dc^-b^-;\ dc^-b^-a]$	$d1$ cm	

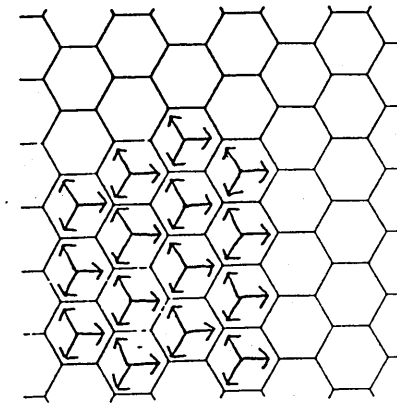
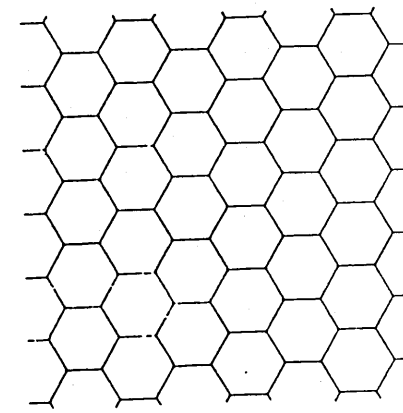
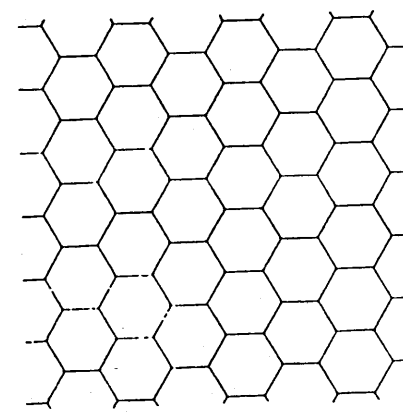
topological type: $[3^6]$.

IH type	incidence symbol	symmetry group stabiliser	marked representation
13	$[ab^+c^+dc^-b^-;\ db^+c^+a]$	pmg d1	
14	$[a^+b^+c^+c^-b^-a^-]$.	cm d1	
15	$[a^+b^+c^+c^-b^-a^-;\ a^+b^-c^+]$	pmg d1	

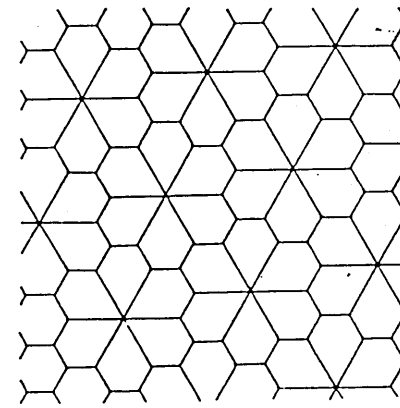
topological type: $[3^6]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
16	$[a^+b^+c^+c^-b^-a^- ; a^-c^+b^+]$	p31m d1	
17	$[ab^+b^-ab^+b^- ; ab^+]$	cmm d2	
18	[ababab ; ba]	p31m d3	

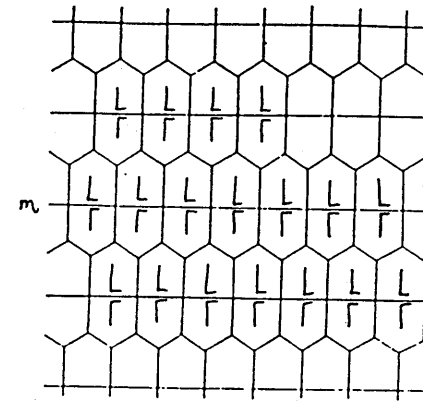
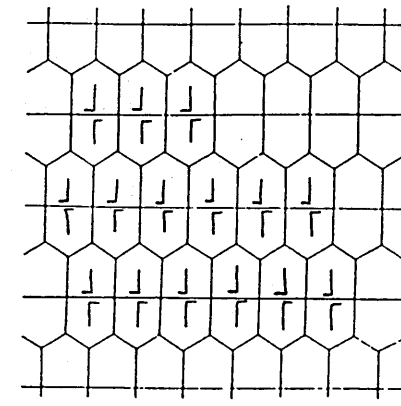
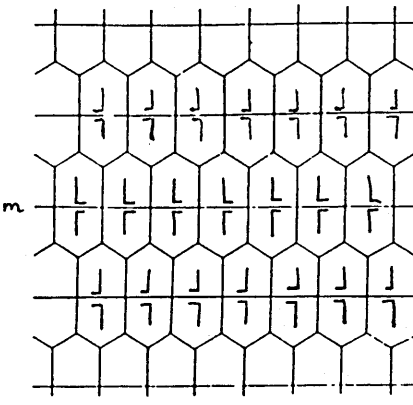
topological type: $[3^6]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
19	$[a^+a^-a^+a^+a^+a^-;\ a^-]$	p3m1 d3	
20	$[aaaaaa;\ a]$	p6m d6	
			

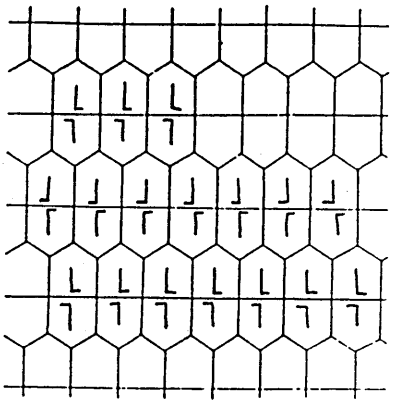
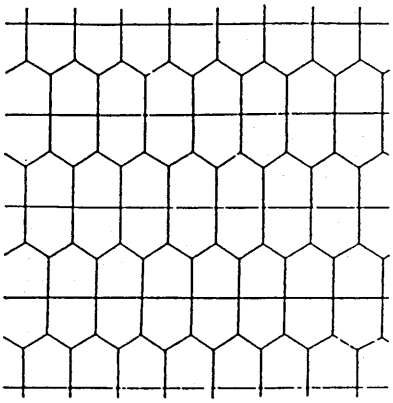
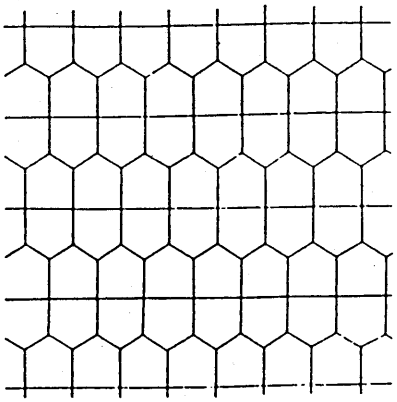
topological type : $[3^4.6]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
21	$[a^+b^+c^+d^+e^+; e^+c^+b^+d^+a^+]$	p6 e	

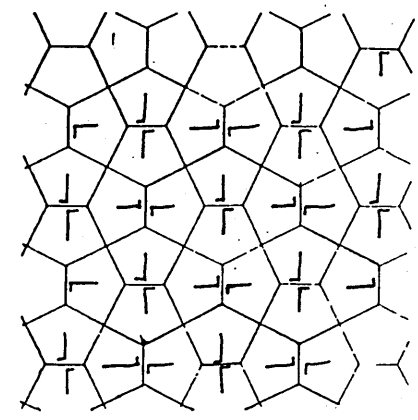
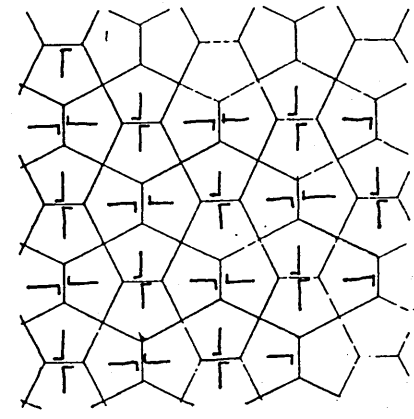
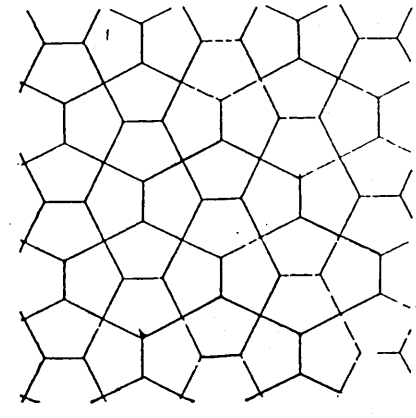
topological type: $[3^3.4^2]$.

IH type	incidence symbol	symmetry group stabiliser	marked representation
22	$[a^+b^+c^+d^+e^+;\ a^-e^+d^-c^-b^+]$	cm e	
23	$[a^+b^+c^+d^+e^+;\ a^+e^+c^+d^+b^+]$	p2 e	
24	$[a^+b^+c^+d^+e^+;\ a^-e^+c^+d^+b^+]$	pmg e	

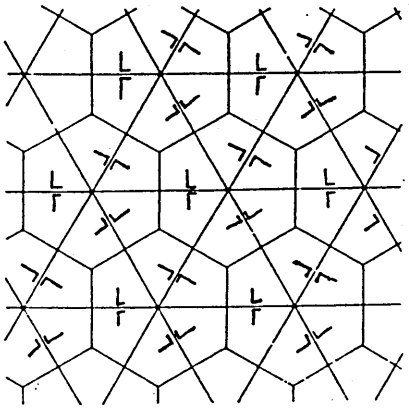
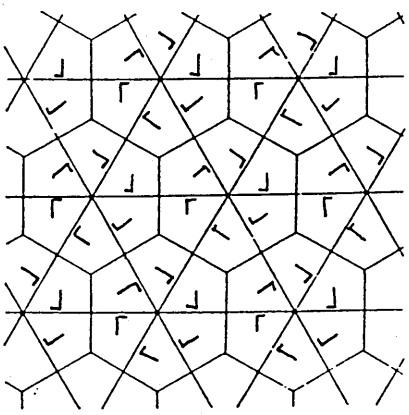
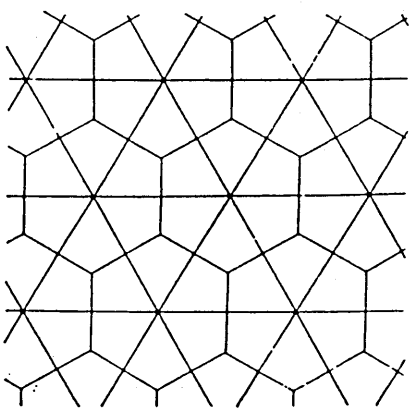
topological type : $[3^3 . 4^2]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
25	$[a^+b^+c^+d^+e^+ ; a^+e^+d^-c^-b^+]$	pgg e	
26	$[a^+b^+c^+c^-b^- ; ab^-c^+]$	cmm d1	
			

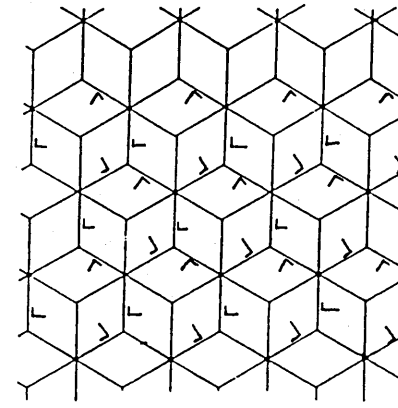
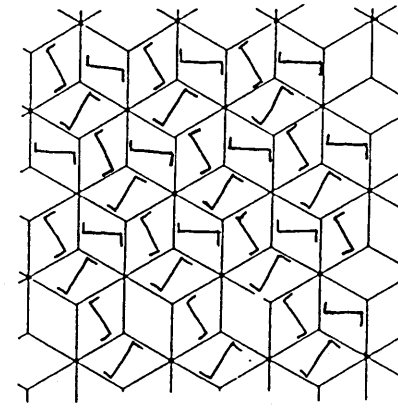
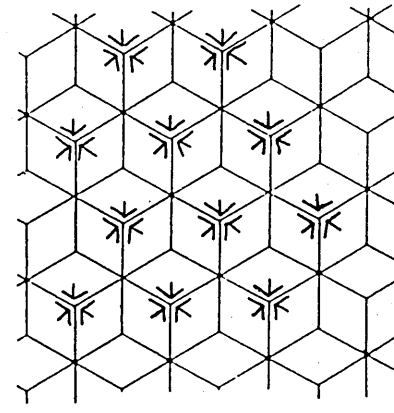
topological type : $[3^2.4.3.4]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
27	$[a^+b^+c^+d^+e^+;\ a^+d^-e^-b^-c^-]$	pgg e	
28	$[a^+b^+c^+d^+e^+;\ a^+c^+b^+e^+d^+]$	p4 e	
29	$[ab^+c^+c^-b^-;\ ac^+b^+]$	p4g d1	

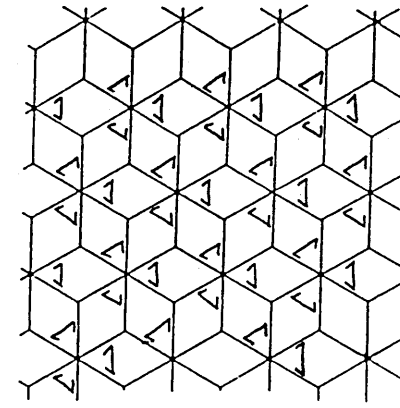
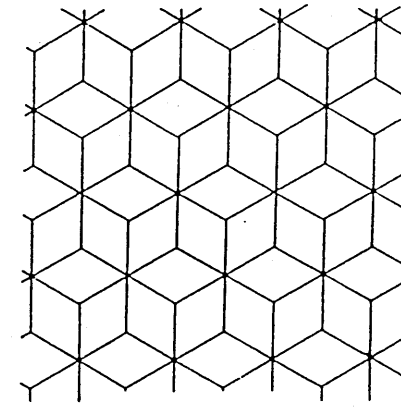
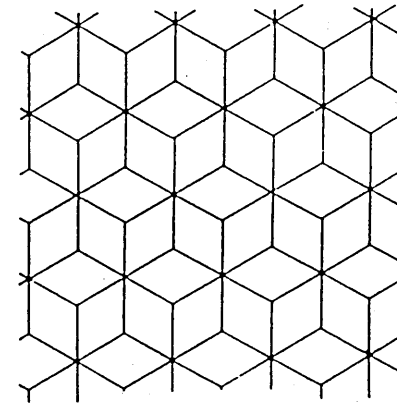
topological type: [3.4.6.4]

IH type	incidence symbol	symmetry group stabiliser	marked representation
30	$[a^+b^+c^+d^+ ; a^-b^-d^+c^+]$	p31m e	
31	$[a^+b^+c^+d^+ ; b^+a^+d^+c^+]$	p6 e	
32	$[a^+a^-b^+b^- ; a^-b^-]$	p6m d1	

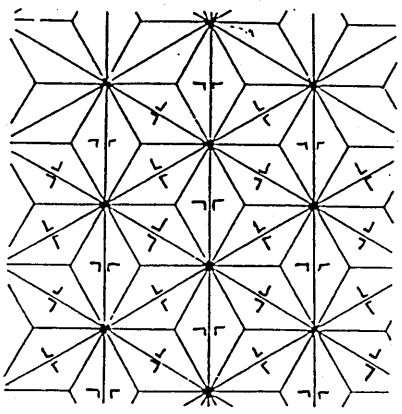
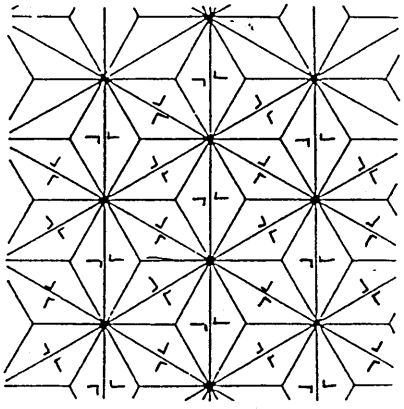
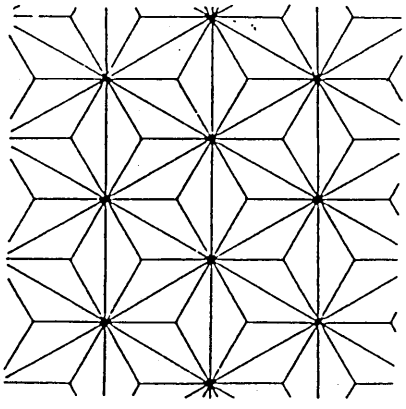
topological type: [3.6.3.6]

IH type	incidence symbol	symmetry group stabiliser	marked representation
33	$[a^+b^+c^+d^+;\ d^+b^+c^+a^+]$	p3 e	
34	$[a^+b^+a^+b^+;\ b^+a^+]$	p6 c2	
35	$[a^+b^+b^-a^-;\ a^-b^-]$	p3m1 d1 102	

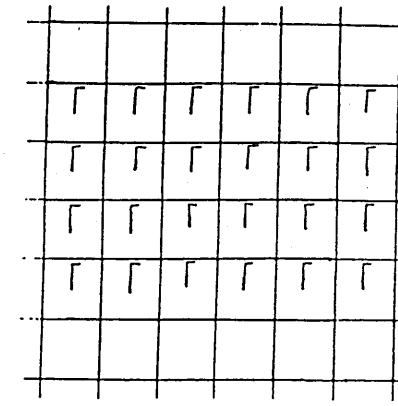
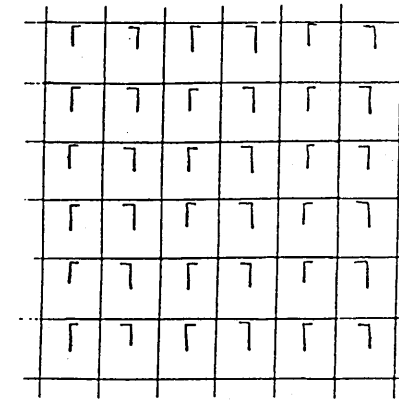
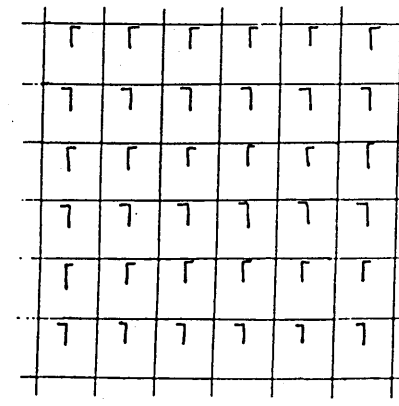
topological type: [3.6.3.6]

IH type	incidence symbol	symmetry group stabiliser	marked representation
36	$[a^+a^-b^+b^-;\ b^-a^-]$	$p3lm$ $d1$	
37	$[a^+a^-a^+a^-;\ a^-]$	$p6m$ d_2	
			

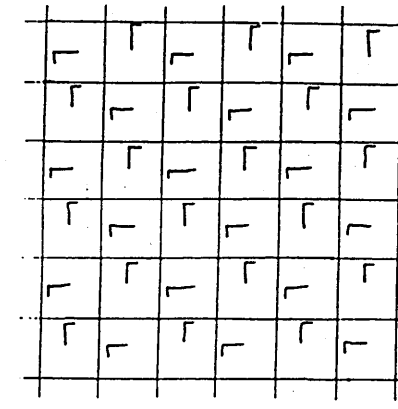
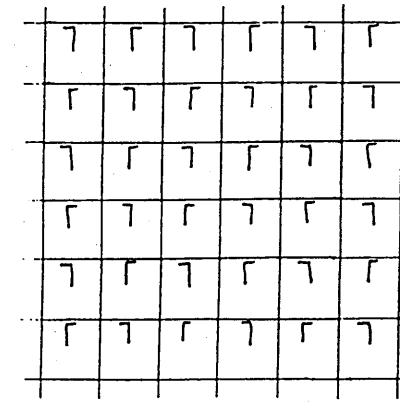
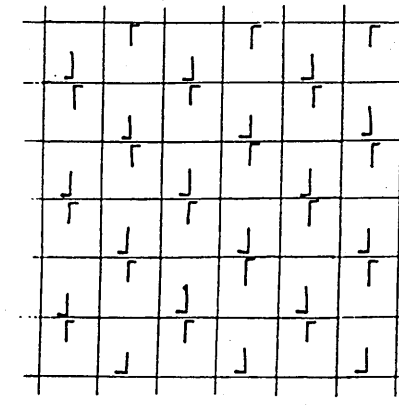
topological type : $[3.12^2]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
38	$[a^+b^+c^+;\ a^-c^+b^+]$	p31m e	
39	$[a^+b^+c^+;\ a^+c^+b^+]$	p6 e	
40	$[ab^+b^-;\ ab^-]$	p6m d1	

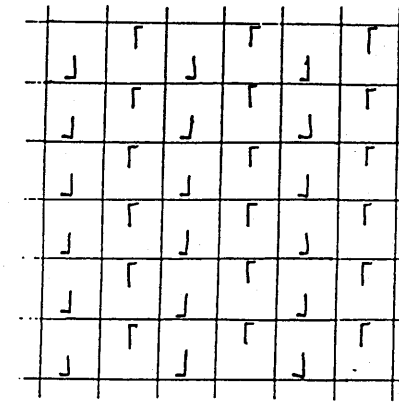
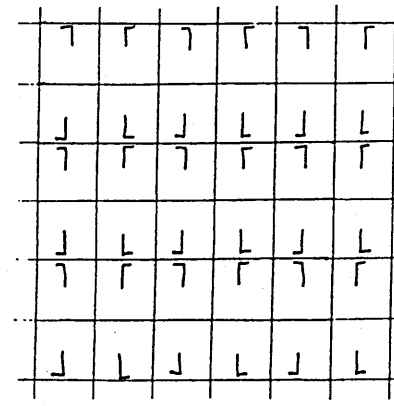
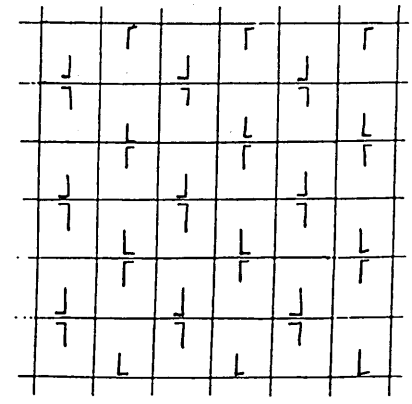
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
41	$[a^+b^+c^-d^-;\ c^+d^+a^+b^+]$	p^1 e	
42	$[a^+b^+c^-d^+;\ c^+b^-a^+d^-]$	pm e	
43	$[a^+b^+c^+d^+;\ c^-d^+a^-b^+]$	pg e	

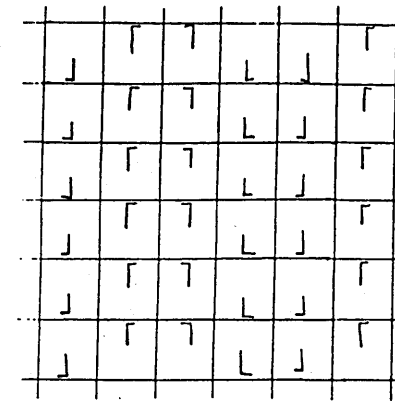
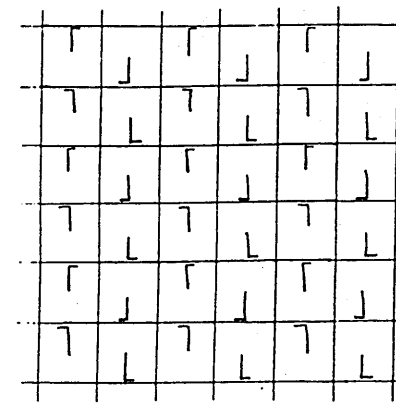
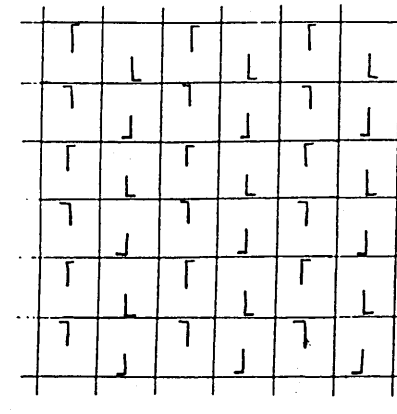
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
44	$[a^+b^+c^+d^+;\ b^-a^-d^-c^-]$	pg e	
45	$[a^+b^+c^+d^+;\ c^-b^-a^-d^-]$	cm e	
46	$[a^+b^+c^+d^+;\ a^+b^+c^+d^+]$	p2 e	

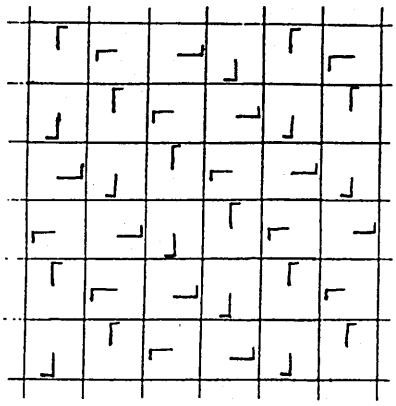
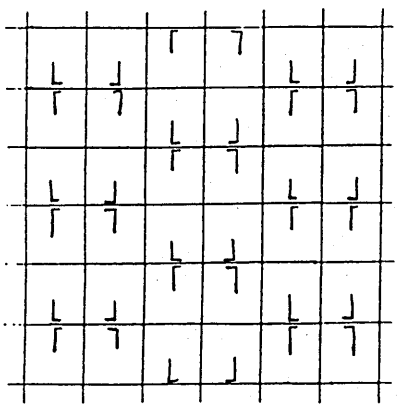
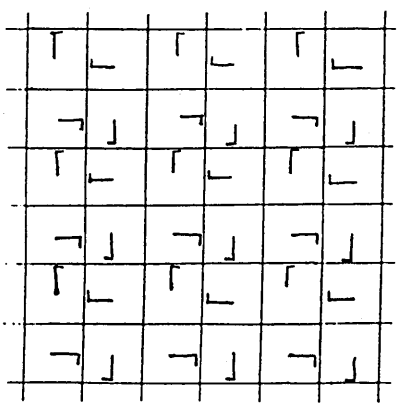
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
47	$[a^+b^+c^+d^+ ; c^+b^+a^+d^+]$	p^2	
		e	
48	$[a^+b^+c^+d^+ ; a^-b^- \ -d^-]$	pmm	
		e	
49	$[a^+b^+c^+d^+ ; a^-b^+c^-d^+]$	pmg	
		e	

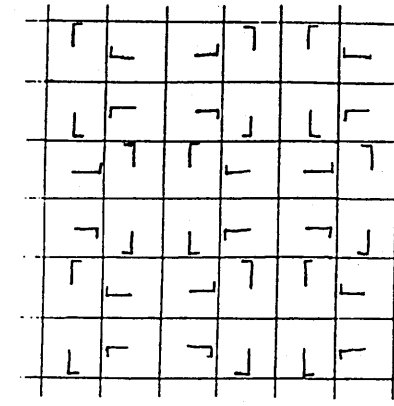
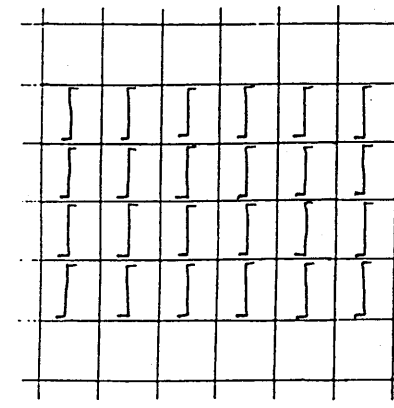
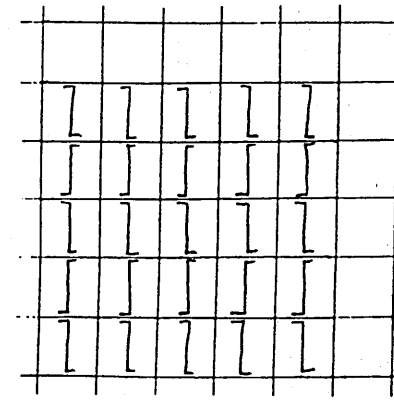
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
50	$[a^+b^+c^+d^+;\ c^+b^-a^+d^+]$	pmg e	
51	$[a^+b^+c^+d^+;\ c^-b^+a^-d^+]$	pgg e	
52	$[a^+b^+c^+d^+;\ c^-d^-a^-b^-]$	pgg e	

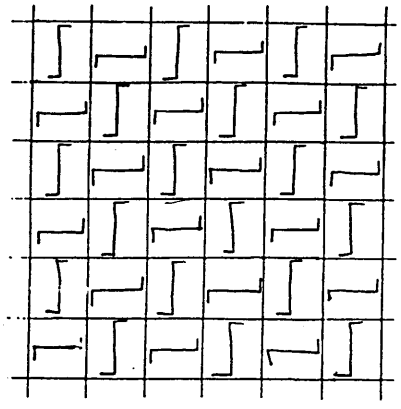
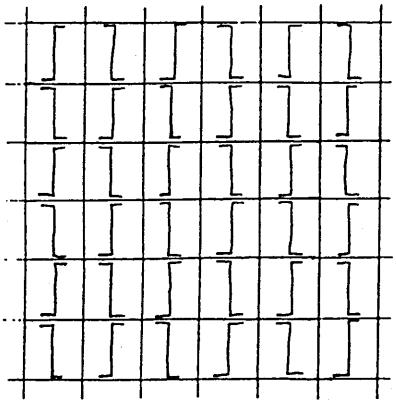
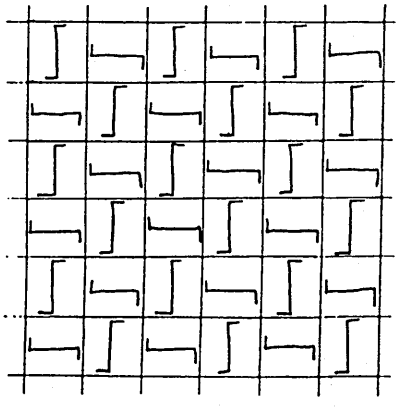
topological type : $[4^4]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
53	$[a^+b^+c^+d^+;\ b^-a^-c^+d^+]$	pgg e	
54	$[a^+b^+c^+d^+;\ a^-b^-c^-d^+]$	cmm e	
55	$[a^+b^+c^+d^+;\ b^+a^+d^+c^+]$	p4 e	

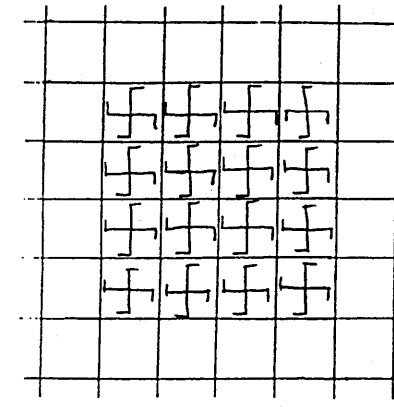
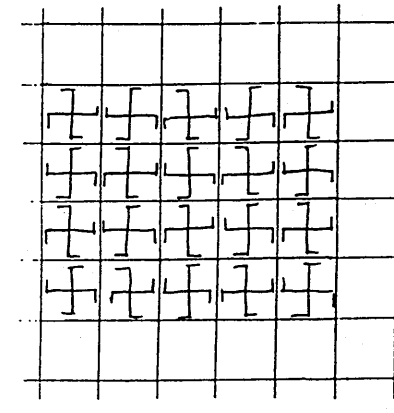
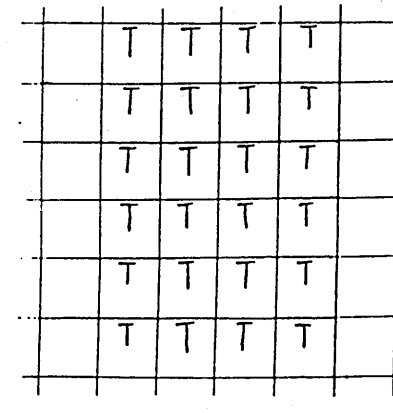
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
56	$[a^+b^+c^+d^+;\ b^+a^+c^-d^-]$	p4m e	
57	$[a^+b^+a^+b^+;\ a^+b^+]$	p2 c2	
58	$[a^+b^+a^+b^+;\ a^-b^+]$	pmg c2	

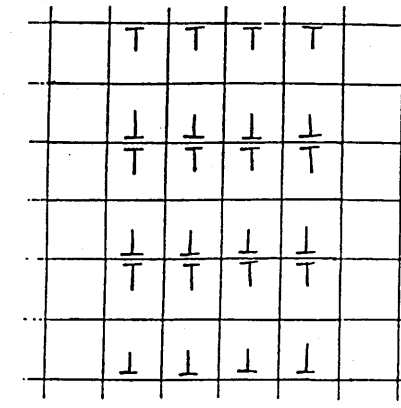
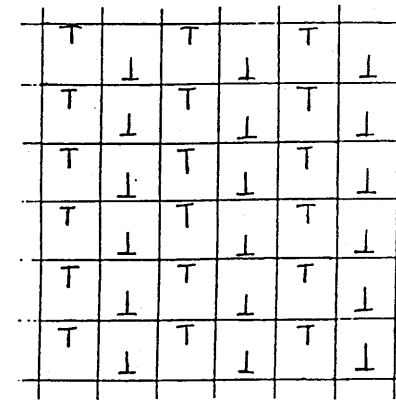
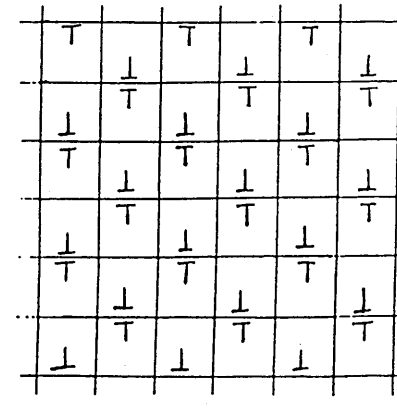
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
59	$[a^+b^+a^+b^+;\ b^-a^-]$	pgg c2	
60	$[a^+b^+a^+b^+;\ a^-b^-]$	cmm c2	
61	$[a^+b^+a^+b^+;\ b^+a^+]$	p4 c2	

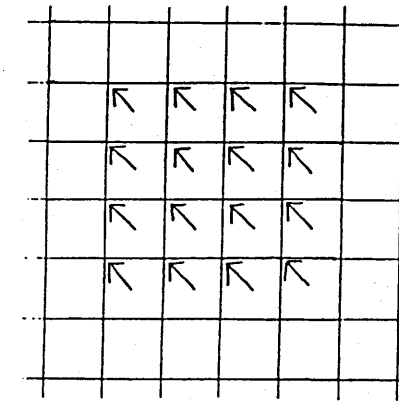
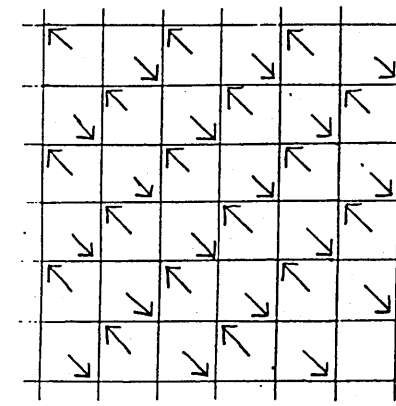
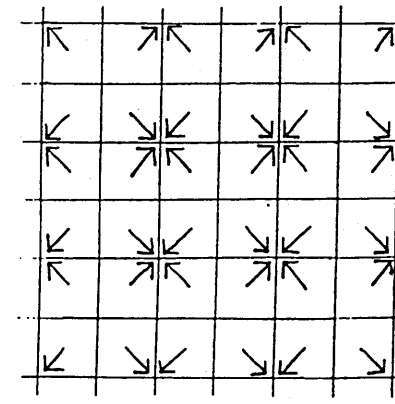
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
62	$[a^+a^+a^+a^+ ; a^+]$	p4 c4	
63	$[a^+a^+a^+a^+ ; a^-]$	p4g c4	
64	$[ab^+cb^- ; cb^-a]$	pm d1	

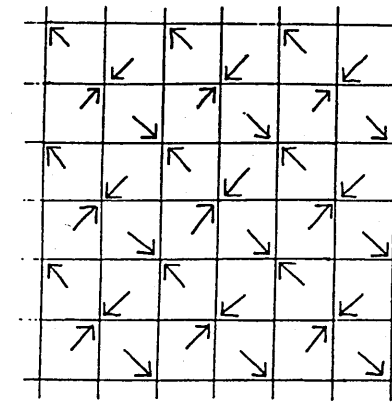
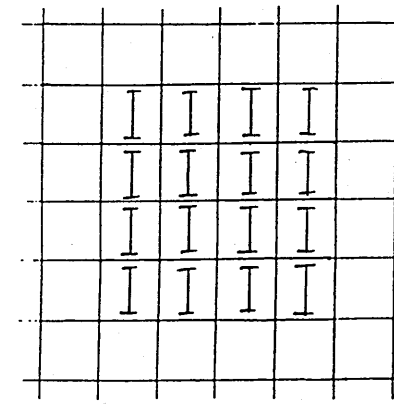
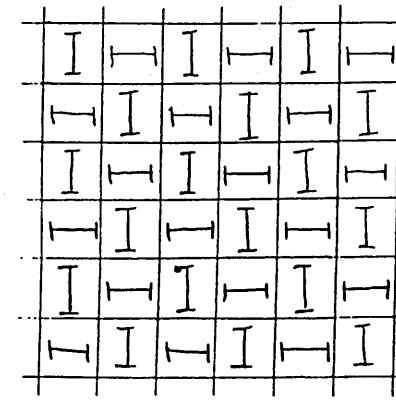
topological type : $[4^A]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
65	$[ab^+cb^-;\ ab^-c]$	pmm d1	
66	$[ab^+cb^-;\ cb^+a]$	pmg d1	
67	$[ab^+cb^-;\ ab^+c]$	cmm d1	

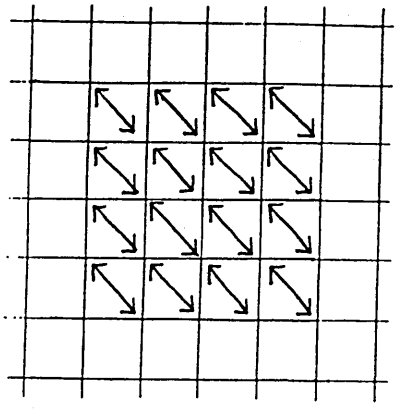
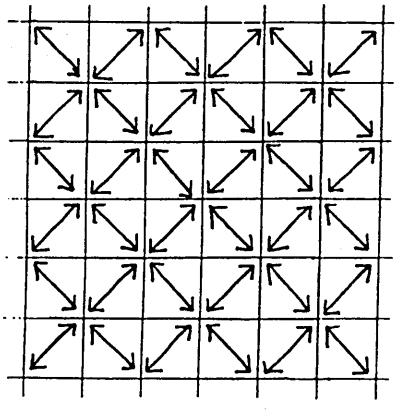
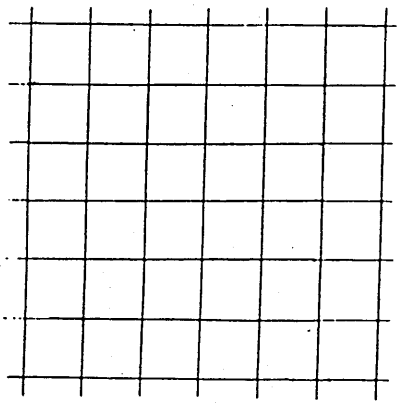
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
68	$[a^+b^+b^-a^-;\ b^-a^-]$	cm d1	
69	$[a^+b^+b^-a^-;\ a^+b^+]$	pmg d1	
70	$[a^+b^+b^-a^-;\ a^-b^-]$	p4m d1	

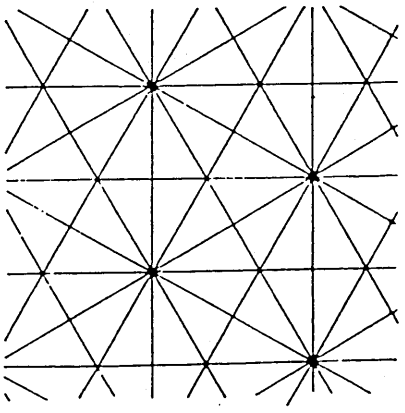
topological type: $[4^4]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
71	$[a^+b^+b^-a^-;\ b^+a^+]$	p4g / d1	
72	$[abab;\ ab]$	pmm / d2	
73	$[abab;\ ba]$	p4g / d2	

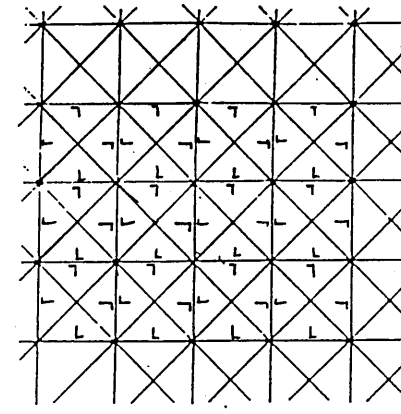
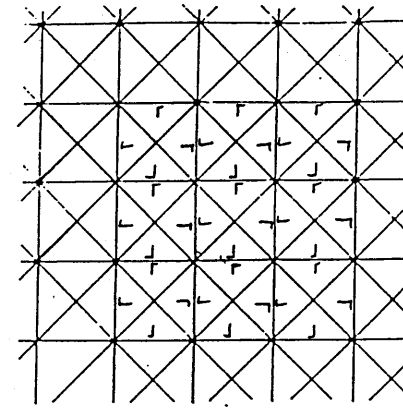
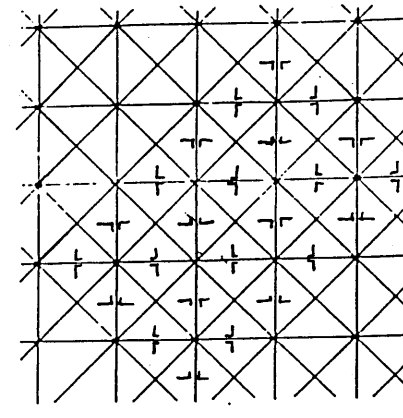
topological type : $[4^4]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
74	$[a^+a^-a^+a^-\,;\,a^+]$	cmm $d\bar{\alpha}$	
75	$[a^+a^-a^+a^-\,;\,a^-]$	p4m $d\alpha$	
76	$[aaaa\,;\,a]$	p4m d4	

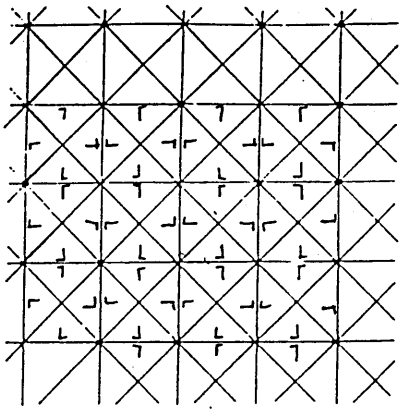
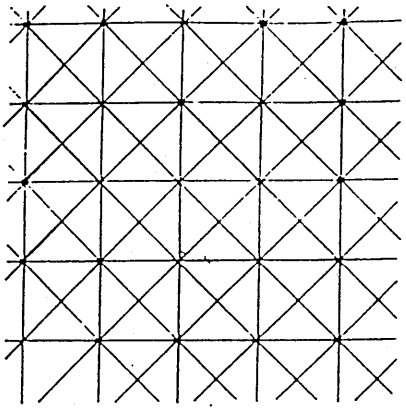
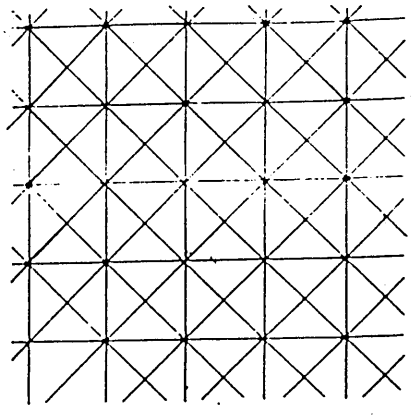
topological type : [4.6.12]

IH type	incidence symbol	symmetry group stabiliser	marked representation
77	$[a^+b^+c^+;\ a^-b^-c^-]$	p6m e	

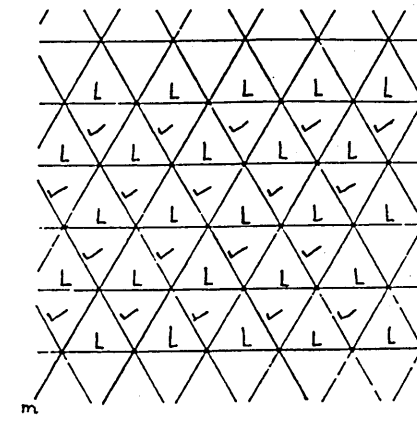
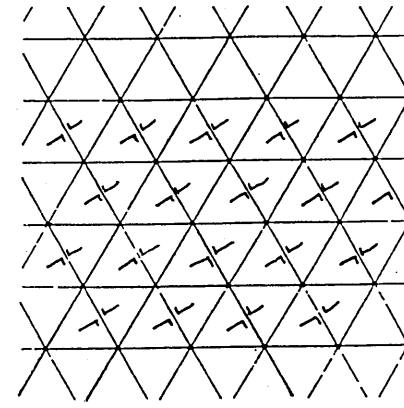
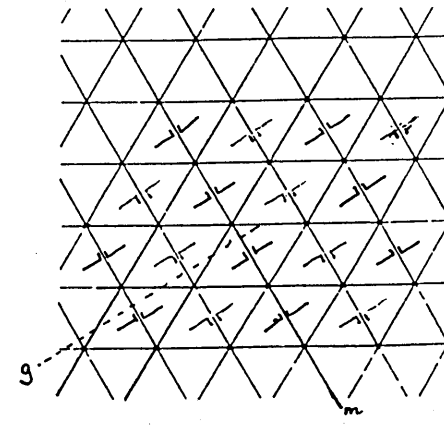
topological type: $[4.8^2]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
78	$[a^+b^+c^+;\ a^+b^-c^-]$	cmm e	
79	$[a^+b^+c^+;\ a^+c^+b^+]$	p4 e	
80	$[a^+b^+c^+;\ a^-b^=c^-]$	p4m e	

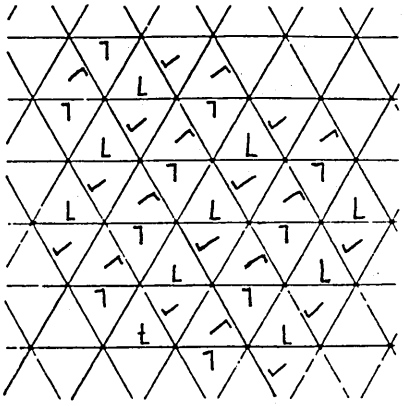
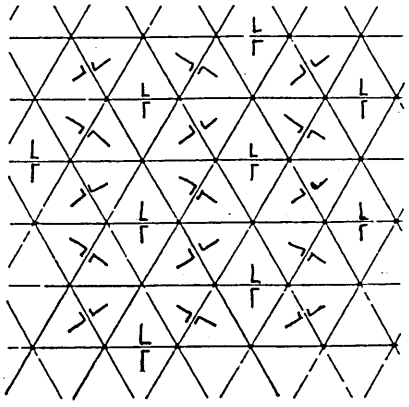
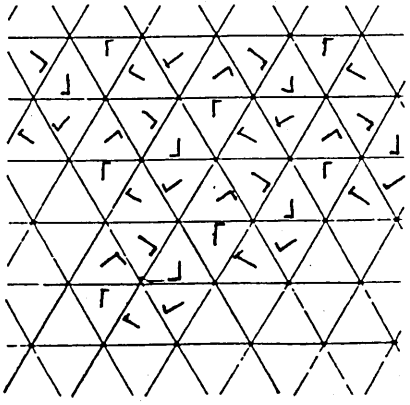
topological type: $[4.8^2]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
81	$[a^+b^+c^+;\ a^-c^+b^+]$	p4g e	
82	$[ab^+b^-;\ ab^-]$	p4m d1	
			

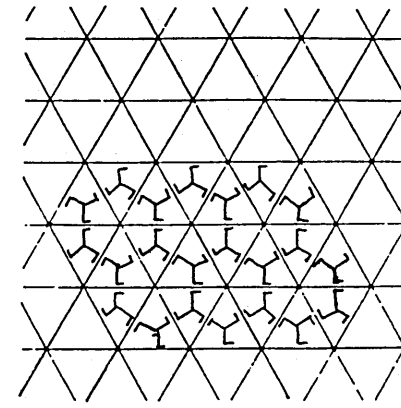
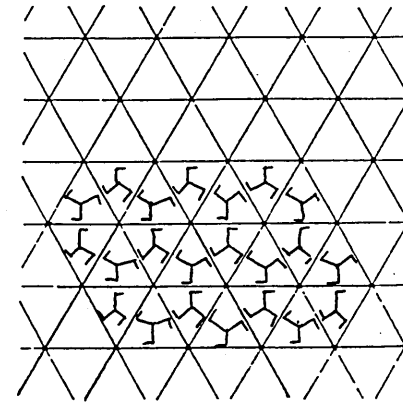
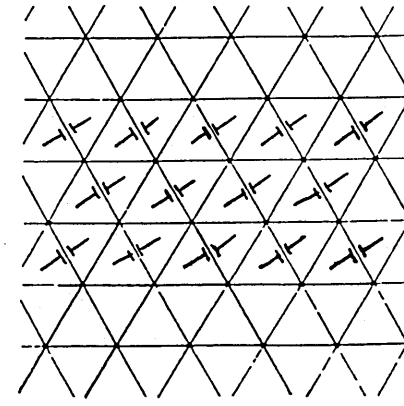
topological type: $[6^2]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
83	$[a^+b^+c^+;\ b^-a^-c^-]$	cm e	
84	$[a^+b^+c^+;\ a^+b^+c^+]$	p2 e	
85	$[a^+b^+c^+;\ a^-b^+c^+]$	pmg e	

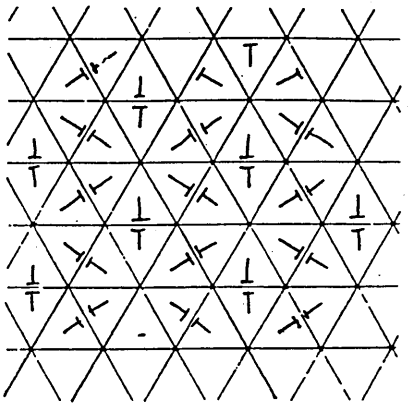
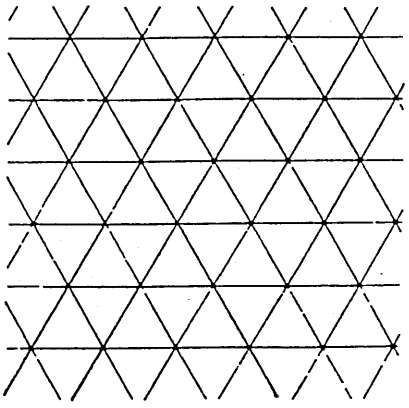
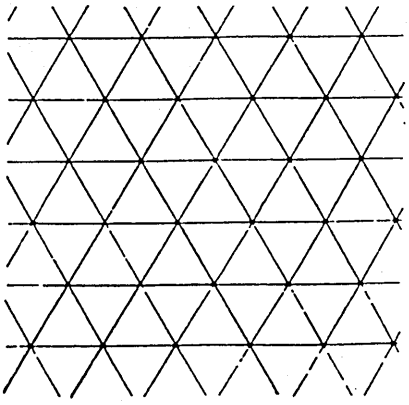
topological type: $[6^2]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
86	$[a^+b^+c^+;\ b^-a^-c^+]$	pgg e	
87	$[a^+b^+c^+;\ a^-b^-c^-]$	p3m1 e	
88	$[a^+b^+c^+;\ b^+a^+c^+]$	p6 e	

topological type: $[6^3]$

IH type	incidence symbol	symmetry group / stabiliser	marked representation
89	$[a^+a^+a^+ ; a^-]$	p31m C_3	
90	$[a^+a^+a^+ ; a^+]$	p6 C_3	
91	$[a\,b^+b^- ; ab^+]$	cmm d_1	

topological type: $[6^2]$

IH type	incidence symbol	symmetry group stabiliser	marked representation
92	$[ab^+b^-;\ ab^-]$	p6m d1	
93	$[aaa;\ a]$	p6m d3	
			

Appendix B

Source code for amalgamation generating program

This appendix contains the Pascal source code for the computer program used in chapter 5 for generating strongly isohedral amalgamations of translational tilings.

```
(* <R.WINGATE>MOLGN4.PAS.4, 27-Feb-87 14:59:41, Edit by R.WINGATE *)
PROGRAM molgen(output);

    (* This program generates, for each translational subgroup   *)
    (* of index s of the overall symmetry group, all molecules   *)
    (* whose amalgamted structure is isomorphic to the original. *)

CONST
    s = 8; (* index of subgroup *)
    offset = 16; (* 2 * s: to make all indices positive *)
    arraysize = 33; (* 4 * s + 1 *)
    valency = 4; (* i.e. number of adjacency vectors *)

TYPE
    orbnum = 0..63; (* to save space *)
    window = ARRAY[0..arraysize,0..arraysize] OF orbnum;
    tile = ARRAY[1..2] OF 0..arraysize;
    vector = ARRAY[1..2] OF integer;
    link = ^molecule;
    tileset = ARRAY[1..s] OF tile; (* i.e. a molecule *)
    molecule = RECORD
                    next : link;
                    data : tileset;
            END;
    vecs = ARRAY[1..6,1..2] OF integer;

VAR
    points : window; (* we record the orbits here *)
    n,p,q,t : integer;
    top,middle,rear : link;
    i,j : integer; (* loop variables if necessary *)
    adjvec : vecs; (* adjacency vectors *)

PROCEDURE zeroarray(VAR points : window);

    (* zeros the array *)

    VAR
      x,y : integer;

    BEGIN (* zeroarray *)
      FOR x:=0 TO 4*s+1 DO
          FOR y:=0 TO 4*s+1 DO points[x,y]:=0
    END; (* zeroarray *)

PROCEDURE zero_orbit(p,q,t : integer;
                     VAR points : window);

    (* calculates the zero orbit of the subgroup determined by  *)
    (* p,q,t by determining images of the origin under elements *)
    (* of the subgroup.                                         *)

    VAR
      x,y,x1,y1,xlim,ylim : integer;

    BEGIN (* zero_orbit *)
      xlim := 2*s;
      ylim := 2*s;
      FOR x := -xlim TO xlim DO
          FOR y:= -ylim TO ylim DO
```

```
                BEGIN
                    x1 := p*x + t*y;
                    y1 := q*y;
                    IF (abs(x1) <= offset) AND (abs(y1) <= offset)
                    THEN
                            points[x1+offset,y1+offset] :=1;
                END; (* for *)
    END; (* zero_orbit *)

PROCEDURE nextpoint(VAR x,y : integer);

    (* Subroutine used by otherorbits to cycle through points *)

    BEGIN (* nextpoint *)
      y:= y + 1;
      IF y>s THEN
          BEGIN
                y := 0;
                x := x+1;
          END (* if *)
    END; (* nextpoint *)

PROCEDURE otherorbits(VAR points : window);

    (* Determines the other orbits (1...s-1, labelled 2..s) as  *)
    (* follows: find a point near the origin that isn't in an   *)
    (* orbit we have already determined - so it must be in      *)
    (* another orbit. Now we can determine the displacement     *)
    (* between this new orbit and the zero orbit, and so we can *)
    (* determine as much of the new one as we need.        *)

    VAR
      x,y,x1,y1,orbit : integer;

    BEGIN (* otherorbits *)
      orbit := 1;
      x := 0;
      y := 0;
      WHILE orbit < s DO
          BEGIN
                WHILE points[x+offset,y+offset] > 0 DO nextpoint(x,y);
                orbit := orbit + 1;
                FOR x1 := -offset TO offset DO
                    FOR y1 := -offset TO offset DO
                            IF points[x1+offset,y1+offset] = 1 THEN
                                IF (abs(x1 + x) <= offset) AND (abs(y1 + y) <= offset)
                                THEN
                                        points[x1+x+offset,y1+y+offset] := orbit
          END; (* while *)
    END; (* otherorbits *)

PROCEDURE printorbits(points : window);

    (* prints out the orbits. *)

    VAR
      x,y,i : integer;

    BEGIN (* printorbits *)
      writeln;
```

```
      writeln('Orbits...');
      writeln;
      FOR y:= offset DOWNTO 0 DO
          BEGIN
                FOR x:= -offset TO -1 DO
                    write( points[x+offset,y+offset] : 3 );
                write('  |');
                FOR x:= 0 TO offset DO
                    write( points[x+offset,y+offset] : 3 );
                writeln
          END; (* for *)
      FOR i:=-3*offset TO 3*(offset+2) DO write('-');
      writeln;
      FOR y:= -1 DOWNTO -offset DO
          BEGIN
                FOR x:= -offset TO -1 DO
                    write( points[x+offset,y+offset] : 3 );
                write('  |');
                FOR x:= 0 TO offset DO
                    write( points[x+offset,y+offset] : 3 );
                writeln
          END; (* for *)
      writeln
    END; (* printorbits *)

PROCEDURE nextnorm(index : integer;
                   VAR p,q,t : integer);

    (* Determines the 'next' normal matrix after [pt/0q], or *)
    (* sets p=q=0 if there are no more.                      *)

    BEGIN (* nextnorm *)
      IF t < p-1 THEN
          t:= t + 1
      ELSE
          IF p < index THEN
                BEGIN
                    REPEAT
                          p:= p + 1
                    UNTIL index MOD p = 0;
                    q := index DIV p;
                    t := 0
                END
          ELSE
                BEGIN
                    p := 0;
                    q := 0;
                    t := 0;
                END; (* ELSE *)

    END; (* nextnorm *)

FUNCTION lessthan(a,b : tileset) : boolean;

    (* Provides a total ordering of molecules (lexicographical) *)

    VAR
      flag,done : boolean;
      i,j : integer;
```

```
    BEGIN
      flag := false;
      done := false;
      FOR i:=1 TO s DO
          for j:=1 to 2 DO
                IF (NOT done) THEN
                    IF a[i,j]<>b[i,j] THEN
                          BEGIN
                              done:=true;
                              flag:=a[i,j]<b[i,j];
                          END;
      lessthan:=flag;
    END;


FUNCTION equal(a,b : tileset) : boolean;

    (* true if molecules a,b, are equal *)

    VAR
      i,j : integer;
      flag : boolean;

    BEGIN
      flag := true;
      FOR i:=1 TO s DO FOR j:=1 TO 2 DO
          IF a[i,j]<>b[i,j] THEN flag:=FALSE;
      equal := flag;
    END;

PROCEDURE findstruct(mol : tileset;
                     VAR count : integer;
                     VAR vectors : vecs);

    (* Determines the structure which the molecule mol has when  *)
    (* tessellated as follows: make a copy of 'points' and set   *)
    (* those entries corresponding to mol to zero. Now use the   *)
    (* adjacency vectors of the underlying grid to find point    *)
    (* adjacent to the molecule: when we find one whose entry is *)
    (* still non-zero we know that we have found an atom in a    *)
    (* molecule adjacent to mol. Since we can find which orbit   *)
    (* this molecule is in we can go on to determine the whole   *)
    (* molecule and 'blank it out'.                              *)

    VAR
      atoms   : window;
      i,j,k   : integer;
      newtile : tile;
      orb     : orbnum;

    BEGIN
      atoms := points;
      count := 0;
      FOR i:= 1 TO s DO atoms[mol[i,1],mol[i,2]]:=0;
      FOR i:=1 TO valency DO
          FOR j:=1 TO s DO
                BEGIN
                    newtile[1]:=mol[j,1]+adjvec[i,1];
                    newtile[2]:=mol[j,2]+adjvec[i,2];
                    orb:=atoms[newtile[1],newtile[2]];
```

```
                    IF orb<>0 THEN
                          BEGIN
                              count:=count+1;
                              vectors[count,1]:=newtile[1]-mol[orb,1];
                              vectors[count,2]:=newtile[2]-mol[orb,2];
                              FOR k:=1 TO s DO
                                    atoms[mol[k,1]+vectors[count,1],
                                          mol[k,2]+vectors[count,2]]:=0;
                          END;
                END;
    END;


PROCEDURE printlist;

    (* prints those molecules with correct amalgamted structure *)

    VAR
      p : link;
      currmol : tileset;
      i,count,adjno : integer;
      vectors : vecs;

    BEGIN
      writeln;
      writeln('Molecules....');
      writeln;
      p := top;
      count := 0;
      WHILE p<> NIL DO
          BEGIN
                currmol := p^.data;
                findstruct(currmol,adjno,vectors);
                IF adjno=valency THEN
                    BEGIN
                          count:= count + 1;
                          FOR i:=1 TO s DO
                              BEGIN
                                    write('(',currmol[i,1]-offset : 1
                                          ,', ',currmol[i,2]-offset : 1,') ');
                                    IF i<>s THEN
                                        write(' ; ')
                                    ELSE
                                        writeln
                              END;
                          {write('... a.v.''s: ');
                          for i:=1 to valency do
                          begin
                          write('(',vectors[i,1] : 1,
                          ', ',vectors[i,2] : 1,') ');
                          if i<>valency then write(' ; ') else writeln;
                          end;}
                    END;
                p := p^.next;
          END;
      writeln;
      writeln(count : 1,' molecules counted');
      writeln;
    END;
```

```
PROCEDURE initlist;

    (* initialises the list of molecules *)

    VAR
      i : integer;
      p : link;

    BEGIN
      new(p);
      top := p;
      middle := NIL;
      rear := NIL;
      top^.data[1,1] := offset;
      top^.data[1,2] := offset;
      FOR i:=2 TO s DO
          BEGIN
                top^.data[i,1] := 0;
                top^.data[i,2] := 0;
          END;
      top^.next := NIL;
    END;

PROCEDURE addlist(newmol : tileset);

    (* Puts newmol in its place *)

    VAR
      place,p : link;
      i :integer;
      placed : boolean;

    BEGIN
      p := middle;
      placed := false;
      IF p=NIL THEN
          BEGIN
                new(p);
                p^.data := newmol;
                p^.next := NIL;
                middle := p;
                rear :=p;
                placed:=true;
          END;
      IF (NOT placed) AND lessthan(newmol,p^.data) THEN
          BEGIN
                place:=p;
                new(p);
                p^.data := newmol;
                p^.next := middle;
                middle :=p;
                placed := true;
          END;
      IF (NOT placed) AND equal(newmol,p^.data) THEN
          placed:=true;
      WHILE NOT placed DO
          BEGIN
                IF p^.next=NIL THEN
                    BEGIN
```

```
                        place := p;
                        new(p);
                        p^.data := newmol;
                        p^.next := NIL;
                        place^.next := p;
                        rear := p;
                        placed := true
                    END (* if *)
                ELSE
                    IF lessthan(p^.next^.data,newmol) THEN
                        p:=p^.next
                    ELSE
                        BEGIN
                            IF NOT equal(newmol, p^.next^.data) THEN
                                BEGIN
                                    place:=p;
                                    new(p);
                                    p^.data := newmol;
                                    p^.next := place^.next;
                                    place^.next := p;
                                END; (* if *)
                            placed := true;
                        END; (* else *)
            END; (* while *)
        END;

    PROCEDURE expand; (* expand mol. at top *)

        (* 'Expands' the molecule at the top of the list: i.e. finds *)
        (* all molecules with one more tile in them which can be     *)
        (* obtained by sticking one more tile on to the existing     *)
        (* molecule. Cf. findstruct for method.                      *)

        VAR
          currmol,newmol : tileset;
          i,j,k : integer;
          scratch : link;
          neworb : orbnum;
          newtile : tile;

        BEGIN
          scratch := top;
          currmol := top^.data;
          top := top^.next; (* take next mol. from list *)
          dispose(scratch); (* free list entry *)
          FOR i:= 1 TO s DO
              FOR j:= 1 TO valency DO
                  BEGIN
                      IF (currmol[i,1]<>0) AND (currmol[i,2]<>0) THEN
                          BEGIN
                              newtile[1] := currmol[i,1]+adjvec[j,1];
                              newtile[2] := currmol[i,2]+adjvec[j,2];
                              neworb := points[newtile[1],newtile[2]];
                              IF (currmol[neworb,1]=0) AND (currmol[neworb,2]=0)
                              THEN
                                  BEGIN
                                      FOR k:=1 TO s DO
                                          IF k=neworb THEN
                                              newmol[k]:=newtile
                                          ELSE
```

```
                                                    newmol[k]:=currmol[k];
                                              (* newmol now contains data to be added:
                                               now we scan the list
                                               for the correct place to put it *)
                                              addlist(newmol);
                                        END;
                              END;
                    END;
          END;


PROCEDURE checkdups(p : link);

    (* Determines the translates of the molecule pointed to by p *)
    (* and removes them from the list.                            *)

    VAR
      current,translate : tileset;
      displacement : vector;
      ttile : tile;
      q,r : link;
      i,j : integer;

    BEGIN
      current:=p^.data;
      FOR i:=2 TO s DO
          BEGIN
                displacement[1]:=current[1,1]-current[i,1];
                displacement[2]:=current[1,2]-current[i,2];
                FOR j:=1 TO s DO
                    BEGIN
                          ttile[1]:=current[j,1]+displacement[1];
                          ttile[2]:=current[j,2]+displacement[2];
                          translate[points[ttile[1],ttile[2]]]:=ttile;
                    END;
                q:=p;
                WHILE NOT equal(translate, q^.next^.data) DO q:=q^.next;
                r:=q^.next;
                q^.next:=q^.next^.next;
                dispose(r);
          END;
    END;


PROCEDURE duplicates;

    (* eliminates translational equivalents. *)

    VAR
      p:link;

    BEGIN
      p:=top;
      REPEAT
          BEGIN
                checkdups(p);
                p:=p^.next
          END
      UNTIL p=NIL;
    END;
```

```
BEGIN (* testit *)
    adjvec[1,1] := 1 ;
    adjvec[1,2] := 0 ;
    adjvec[2,1] := 0 ;
    adjvec[2,2] := 1;
    adjvec[3,1] := -1 ;
    adjvec[3,2] := 0;
    adjvec[4,1] := 0 ;
    adjvec[4,2] := -1;
    p := 1;
    q := s;
    t := 0;
    WHILE p <> 0 DO
      BEGIN
          zeroarray(points);
          zero_orbit(p,q,t,points);
          otherorbits(points);
          writeln;
          writeln('Subgroup H, A(H) = (',p : 3,t : 3,' )');
          writeln('                   (',0 : 3,q : 3,' )');
          printorbits(points);
          initlist;
          FOR n:=1 TO s-1 DO
                BEGIN
                    WHILE top<>NIL DO expand;
                    top:=middle;
                    middle:=NIL;
                    rear:=NIL;
                END;
          duplicates;
          printlist;
          nextnorm(s,p,q,t)
      END; (* while *)
END.
(* testit *)
```