
Algorithmic aspects of plan generation and enumeration

R Krishnamurti, P H O'N Roe

Department of Systems Design, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada

Received 24 October 1978

Abstract. Plans composed from elements of rectangular grids are considered. A general approach towards the generation and enumeration of nonequivalent plans is presented. It is shown that for these plans there is a minimum colouring which permits easy detection of isomorphs without the need for external storage devices. The notions of threading patterns and colour rules are introduced. Four specific algorithms for different types of plans involving rectangular elements are developed.

In this paper we present a general approach to the problem of generating and enumerating architectural plans based on rectangular grids. The general approach makes it possible to examine various classes of plans within a consistent algorithmic framework. The classes of architectural plans we have chosen for illustrative purposes include arrangements of rectangular rooms within a rectangular envelope (the rectangular-dissection problem); bilaterally symmetric Palladian schemes in which the central room is not necessarily rectangular; rectangular dissections of the square which display either four-fold cyclic (C_4) or dihedral (D_4) symmetries; and traditional Japanese room plans based on the arrangements of the tatami or standard mat. It is noted in our conclusions that this general approach may be extended to three-dimensional arrangements, and has already been used successfully to enumerate assemblies of cubes and the packing of boxes.

Plans

A *plan* is an arrangement on the *unit grid* (Bloch and Krishnamurti, 1978) of non-overlapping regions whose boundaries form polygons whose edges lie on the grid lines. The edges are combined to form *maximal lines* (Stiny, 1979). Intersecting maximal lines meet in an L, T, or + shape, known respectively as *2-way*, *3-way*, and *4-way* points. Plans are *trivalent* if they do not contain a 4-way point; *nonaligned* if each grid line contains at most one maximal line; and *standard* if each grid line contains at least one maximal line. Plans are *equivalent* if they are identical after a sequence of rotations and reflections. Our aim is the enumeration of nonequivalent plans via the counting of the representatives (called *canonical plans*) of the equivalence classes.

The polygons of a plan are either *shaded* or *unshaded*, where shading may be interpreted as corresponding to some special feature in the plan such as a courtyard or central hall. The polygons are referred to as the *plan elements*. A plan is denoted by P , and the number of its unshaded elements is its *content*, p .

The grid cells of an (l, m) unit grid (Bloch and Krishnamurti, 1978) are associated with integer coordinates (x, y) , or (column, row), where $x \in \{0, 1, 2, \dots, m-1\}$ and $y \in \{0, 1, 2, \dots, l-1\}$. The horizontal and vertical grid lines respectively lie on the lines $Y = y$, $y \in \{0, 1, \dots, l\}$, and $X = x$, $x \in \{0, 1, 2, \dots, m\}$. Let $h_y(P)$ and $v_x(P)$ each denote a relationship between plan P and the grid lines, measured in as yet undetermined units. For instance h_y may be Boolean-valued, indicating whether or not there is a maximal line on $Y = y$. Or it may be integer-valued, denoting the number of lines (distinct edges of the polygons touching the grid line) on $Y = y$.

The *threading pattern*, Γ , defines a bijection of the grid cells onto the integers 1, 2, ..., lm , and is denoted by the ordered set

$$\langle (x_1, y_1), (x_2, y_2), \dots, (x_{lm}, y_{lm}) \rangle,$$

where $\Gamma[(x_j, y_j)] = j$. The choice of a threading pattern for a unit grid depends upon the set of production rules in the construction of the plans. It is assumed, however, that, for a given class of plans, Γ is fixed.

Other grid parameters may be defined, such as the *cell type*, but these appear to be dependent upon the particular requirements of the generating algorithm.

Plan representation

To represent any plan P with content p , p distinct 'colours' are allocated to the unshaded plan elements in such a manner that grid cells in the same element have the same colour. The colours are identified with integers. The shaded elements are given the special colour $\hat{0}$. Such an assignment of colours is termed a *proper colouring* of the plan. A proper colouring C defines a map between the cells and the set of colours.

$C(q)$ denotes the colour of (x, y) , where $\Gamma(x, y) = q$.

There are of course many possible proper colourings for a plan. Define the ϕ -colouring, $\phi(P)$, of P as follows. Let Γ be the threading pattern. Label the cells, taken in their Γ -number order, with increasing numbers starting from 1, such that grid cells in the same unshaded element have the same number. Label all cells in the shaded elements with $\hat{0}$.

Let $\phi(P) = \langle \phi(1), \phi(2), \dots, \phi(lm) \rangle$, where $\phi(q)$ is the colour associated with the cell whose Γ -number is q . Since $\phi(P_1) = \phi(P_2)$ if and only if $P_1 = P_2$, $\phi(P)$ may be used as a representation of P .

Canonical plans

Recall that one plan is equivalent to another if they are identical after a sequence of rotations and reflections. Consider the group of such transformations, which map plans onto fellow members of their equivalence class. In order to find a representative of each equivalence class, without loss of generality we may restrict our attention to those plans with $l \leq m$, and thus to those transformations that leave the grid invariant. Let G denote the group of such transformations; it can easily be checked that G is a subgroup of the dihedral group of order eight, D_4 , if $l = m$ and the Klein four-group, K_4 , if $l < m$. The elements of the group are internally represented as cell-to-cell mappings (Krishnamurti and Roe, 1979). These grid-invariant transformations of plan P are denoted by gP , for each $g \in G$.

A plan P is defined to be *canonical* if and only if

$$\phi(P) \leq_L \phi(gP) \quad \text{for all } g \in G,$$

where \leq_L denotes less than or equal to in the lexicographical sense. A canonical plan is well defined, for if $\phi(P) = \phi(gP)$ then $P = gP$. Each canonical plan is taken to be the representative of its equivalence class of plans under G .

The internal representation of a plan P is given by $\phi(P)$, which is defined in such a way that $\phi(gP)$, $g \in G$, cannot be computed directly from it. Instead $\phi(gP)$ is calculated as follows. First a colouring $g\phi(P)$ is determined, where

$$g\phi(P) = \langle \phi[g^{-1}(1)], \phi[g^{-1}(2)], \dots, \phi[g^{-1}(lm)] \rangle,$$

where $g^{-1}(q)$, $q = 1, 2, \dots, lm$, is the Γ -number of the cell that maps to (x_q, y_q) under $g \in G$. Then there exists a permutation π of the colours $\hat{0}, 1, 2, \dots, p$ such that $\pi g\phi(P) = \phi(gP)$. The permutation π is constructed as follows.

Construct $\langle g^{-1}(1), g^{-1}(2), \dots, g^{-1}(lm) \rangle$. For all i , mark $\pi(i)$ as 'undefined'. Mark the numbers from 1 to p as 'unused'. Set $\pi(\hat{0}) = \hat{0}$. Select cells q' from

$$\langle g^{-1}(1), g^{-1}(2), \dots, g^{-1}(lm) \rangle$$

in order. Three cases arise.

Case 1: $\phi(q') = \hat{0}$. Assign $\hat{0}$ as the new colour of $g(q')$.

Case 2: $\phi(q') \neq \hat{0}$ and $\pi[\phi(q')]$ is defined. Assign this as the new colour of $g(q')$.

Case 3: $\phi(q') \neq \hat{0}$ and $\pi[\phi(q')]$ is undefined. Let t be smallest 'unused' number.

Mark t as 'used'. Assign t as the value of $\pi[\phi(q')]$ and as the new colour of $g(q')$.

It is clear that π is a permutation. Since the new colours are associated with the cells in increasing order, it follows that $\pi g \phi(P)$ is a ϕ -colouring of the transformed plan gP : that is, $\pi g \phi(P) = \phi(gP)$.

This construction of π forms the basis of the algorithm for detecting canonical plans (algorithm 1). The algorithm is described in an ALGOL-like language. The notation is adapted from that of Reingold et al (1977).

The algorithm may be explained as follows. Statement A chooses the current transform g from G and proceeds to test for isomorphs provided *flag* remains **true**. The Boolean-valued variable *flag* signifies whether the plan is canonical with respect to the previous set of transforms that have been applied, and is initially set equal to **true**. Statement B selects the cells q in order, and the rules for the construction of π are applied (statement C). The smallest 'unused' colour is contained in *newcolour*; *transformed colour* is the colour associated with q under a ϕ -colouring of gP and is equal to the permutation of the colour associated with $g^{-1}(q)$ under a ϕ -colouring of P . The loop defined by statement B is continued as long as $\phi(P)$ and $\phi(gP)$ correspond component by component. Statement D is the actual symmetry test [when $\phi(P) \neq \phi(gP)$] which determines whether or not P is canonical with respect to g . The algorithm returns the value **true** for *flag* if and only if P is canonical.

The algorithm indicates that, for any generating algorithm for plans, isomorphs can be detected without resorting to any external storage device. The worst case complexity occurs when P has all the symmetries of G (that is, $P = gP$ for all $g \in G$); in this case, $|G|lm$ comparisons have to be made.

Algorithm CANONICAL(P)

¶ P and π are represented by the arrays *colour*[1, ..., lm] and $\pi[\hat{0}, \dots, p]$ ¶

flag ← **true**

A: for $g \in G$ while *flag* do

 for all i set $\pi[i] \leftarrow$ 'undefined'

$\pi[\hat{0}] \leftarrow \hat{0}$

newcolour ← 0

 B: for $g \in \langle 1, 2, \dots, lm \rangle$ while *flag* do

transformed cell ← $g^{-1}(q)$.

 C: if $\pi[\text{colour}[\text{transformed cell}]]$ is undefined

 then { *newcolour* + ← 1

transformed colour ← $\pi[\text{colour}[\text{transformed cell}]] \leftarrow$ *newcolour*

 else *transformed colour* ← $\pi[\text{colour}[\text{transformed cell}]]$

flag ← *transformed colour* = *colour*[q]

 D: if *flag* is false

 then *flag* ← *colour*[q] < *transformed colour*

return (*flag*)

Algorithm 1.

Shape and colouring rules

Shape grammars invented by Stiny (1975; 1977) and Gips (1975) for the recursive specification of shapes form the natural basis for describing generating algorithms for plans. Shape grammars for plans consist of rules that act on lines on the underlying grid according to some local criteria. For computational purposes these rules may be recast in terms of cell colouring rules. The shape rules for plans assert the presence or absence of lines on the grid lines; the corresponding colour rules belong to one of three types.

R1: *Same-colour rule*. A cell is given the same colour as one of its neighbouring (edge-adjacent) cells. This means that the two cells are in the same element, and there is no line in the plan between these cells.

R2: *New-colour rule*. A cell is given a colour different from its neighbouring cell(s). This introduces a line or lines in the plan between the cell and its neighbour(s).

R3: *Null-colour rule*. This colours the cell with the special colour $\hat{0}$. This rule is separately stated, though in many instances it can take the form of rules R1 or R2. The effect of this rule is to assert that the cell is in a shaded plan element, or 'courtyard'.

Notice that these rules do not require the exact colours of the neighbouring cells to be explicitly stated—only the spatial colour relationship is required. This allows us to specify colouring rules in terms of parametrized colours, with the actual substitution taking place when the rule is applied.

Shape rules often use markers to determine the process of generation. For plans these markers can be regarded as reflecting the values associated with certain grid parameters and can be expressed as Boolean-valued predicates involving these parameters. For example, the existence of a line on a grid line can be determined by testing whether or not the appropriate h or v parameter is nonzero. That is, the application of colours is controlled by the satisfaction of Boolean predicates.

To summarize, a colouring grammar consists of a set of grid parameters, a set of colours, a set of predicates, a set of colouring rules, and a set of initial colouring rules, where the rules take the form

$$\text{uncoloured shape} \xrightarrow{\text{predicate}} S_1; \text{coloured shape}; S_2 .$$

The *uncoloured shape* is a collection of cells with at least one uncoloured cell, that results in the *coloured shape* after application of the rule. S_1 and S_2 are sets of assignments involving grid parameters before and after the rule has been applied. The predicate is a Boolean-valued predicate which must hold in order that the entire rule may apply.

The abstract version of the generating algorithm for plans may be described as follows. Proceed recursively over the grid, choosing cells in order (defined by Γ). All possible colour rules that apply to the current cell are selected and for each rule the appropriate colour is allocated to the cell.

Algorithm GENERATE

```

if all the cells have been coloured
then if CANONICAL('generated plan') then plan is canonical
else { choose the next cell(s) from the threading pattern
      { for all possible colour rules that apply to this (these) cell(s) do
        { colour cell(s)
        { GENERATE
  
```

Algorithm 2.

In ALGOL-like notation, the algorithm takes the form shown in algorithm 2. The exact mechanism of the colour rules becomes apparent when considering specific problems.

Example 1: $[p, 2]$ -rectangulations

A $[p, 2]$ -rectangulation is a plan each of whose p plan elements is a rectangle. For a given p , the set of dimensions of (l, m) unit grids ($l \leq m$) for which there exists at least one standard $[p, 2]$ -rectangulation is (Bloch, 1976)

$$\left\{ (l, m): 1 \leq l \leq \left\lceil \frac{p}{2} \right\rceil, \max \left(\left\lceil \frac{p}{i} \right\rceil, l \right) \leq m \leq p+1-l \right\},$$

where, for any real number N , $[N]$ denotes the least integer greater than or equal to N .

Biggs (1969) and Earl (1978) have shown that

- (1) a standard $[p, 2]$ -rectangulation on an (l, m) unit grid is trivalent and nonaligned if and only if $l+m-1 = p$; and
- (2) a standard trivalent (nonaligned) $[p, 2]$ -rectangulation on an (l, m) unit grid such that $l+m-1 = p$ is nonaligned (trivalent).

A shape grammar for rectangulations is given in figure 1. [Compare this with Earl's (1977) grammar for standard trivalent nonaligned rectangulations.] Rule (2) states that the generation of any rectangle must begin with a single 1×1 square. Rule (3) extends a rectangle horizontally by adding a square at a time. Rule (4) extends a rectangle vertically by a unit height at a time. The grammar generates any rectangulation (not necessarily standard) by a unique sequence of shape rules.

This shape grammar may be translated into colouring rules in the following manner. First note that the shape rules construct the rectangulations row by row from the bottom row up. Second, every new rectangle begins with a single square; that is, every new rectangle starts with a square coloured differently from its previously coloured neighbours. Third, the extension of a rectangle horizontally or vertically requires that the new cells have the same colours as their fellow members of the rectangle.

The equivalent colouring rules are given in figure 2. Rules 1 to 7 are of type R1, and rules 8 to 13 are of type R2. The former set extends existing rectangles both horizontally and vertically depending upon local colour conditions. The latter set terminates the current rectangle(s) and creates the start of a new rectangle. The grid parameter μ indicates the number of colours used; r, s , and t are parametrized colours. The threading pattern (implicitly defined in this grammar) is the ordered set of cells from left to right, bottom to top, in that order.

Suppose we want to generate standard $[p, 2]$ -rectangulations on the (l, m) unit grid. Consider the predicates

$$F_1 \equiv (lm - q \geq p - \mu),$$

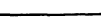
$$F_2 \equiv (p > \mu),$$

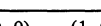
$$F_3 \equiv (x_q \neq m - 1) \vee [(x_q = m - 1) \wedge (h_{y_q} \geq 1)],$$

and

$$F_4 \equiv (y_q \neq l - 1) \vee [(y_q = l - 1) \wedge (v_{x_q} \geq 1)],$$

where q is the current cell number from the threading pattern, $(x_q, y_q) = \Gamma^{-1}(q)$, and where h_{y_q} and v_{x_q} denote the number of rectangles touching the grid lines $Y = y_q$ and $X = x_q$ from above and from the right respectively. Insertion of these predicates into figure 2 gives the colouring rules for standard $[p, 2]$ -rectangulations (figure 3).

S :  L : $\{(0, 0): A\}, \{(0, 0): A'\}, \{(0, 0): C\}, \{(0, 0): \hat{C}\}$

I :  T : translation and reflection in the Y -axis

R : (1) 

A' A C
 $(-1, 1)$ $(0, 1)$ $(m, 1)$

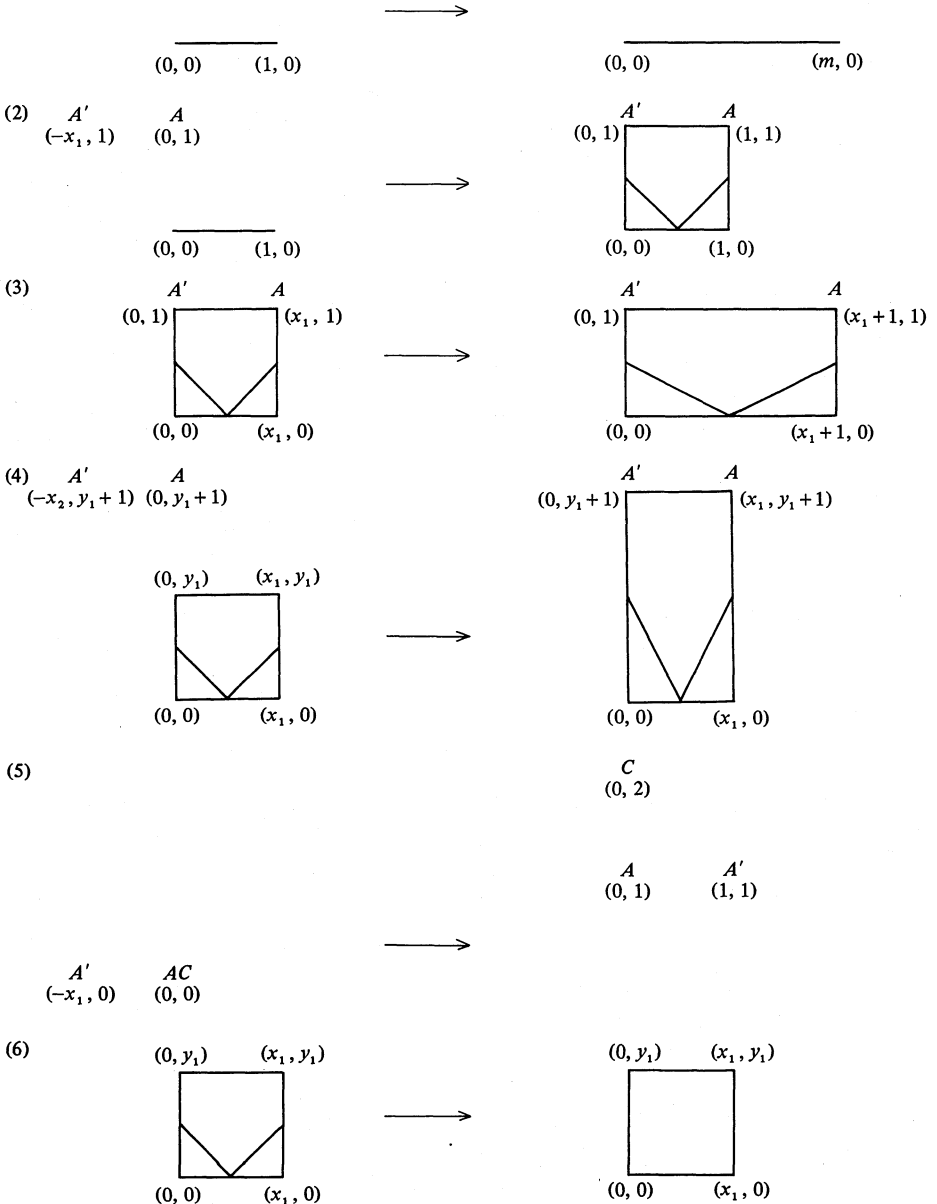


Figure 1. Parametric shape grammar for rectangulations.

(7) $\begin{matrix} C \\ (0, 1) \end{matrix}$

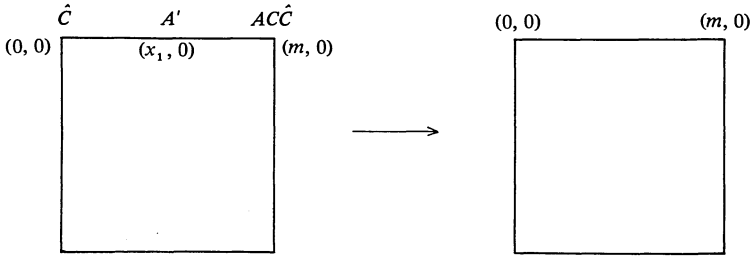


Figure 1 (continued).

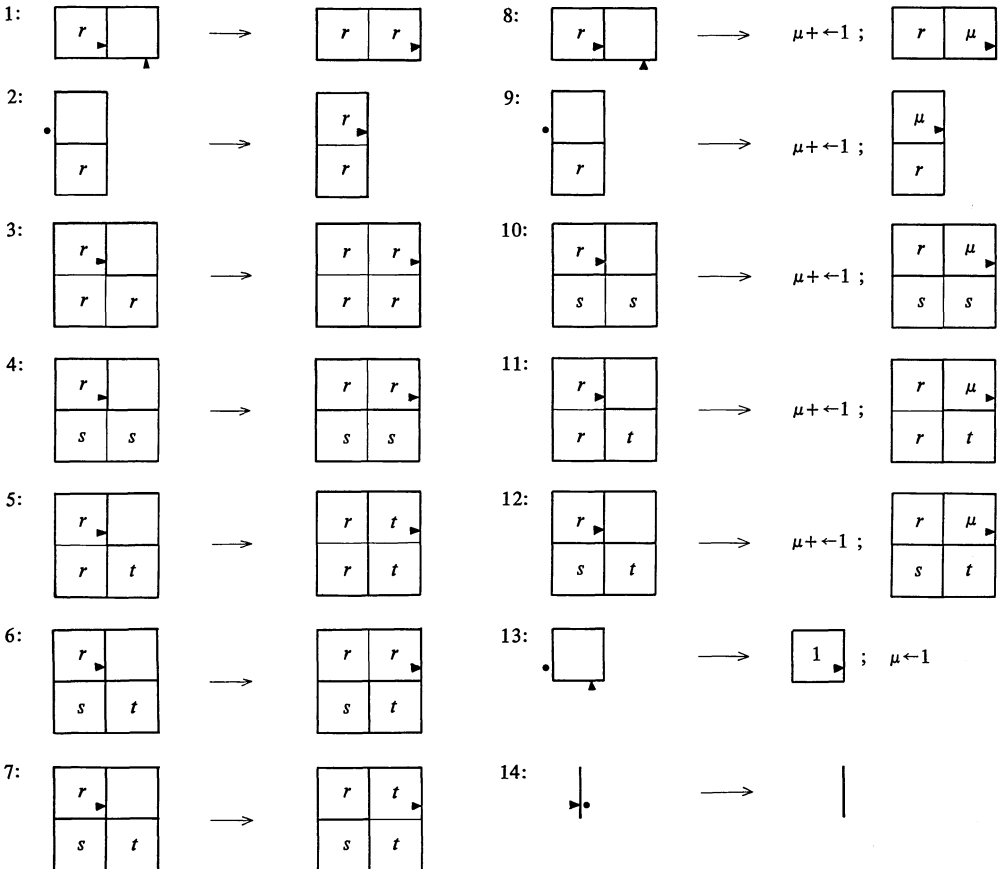
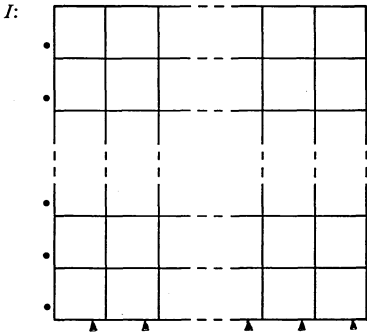


Figure 2. Colouring rules for rectangulations.

Notice that rule 12 in figure 2, and the corresponding rule in figure 3, is the only rule that generates a 4-way point. Omission of this rule ensures that trivalency is never violated.

For nonaligned rectangulations, consider the predicates

$$F_5 \equiv (v_{x_q} = 0) \quad \text{and} \quad F_6 \equiv (h_{y_q} = 0),$$

and replace the appropriate rules in figure 3 by the rules shown in figure 4.

The colouring rules in figures 2, 3, and 4 yield ϕ -colourings. All the rectangulations on the grid are uniquely and exhaustively generated. (This assertion may be demonstrated by an induction argument.)

These rules are incorporated into algorithm GENERATE for $[p, 2]$ -rectangulations. Γ is the mapping $\Gamma(x, y) = my + x + 1$. For any q ,

$$\Gamma^{-1}(q) = (x_q, y_q) = \left(q - 1 \bmod m, \left\lfloor \frac{q-1}{m} \right\rfloor \right),$$

where, for any real number N , $[N]$ is the greatest integer less than or equal to N . The (l, m) unit grid is the input. The rules are simplified by partitioning them into four categories: (1) rules that apply to cell 1, which is given the initial colour 1; (2) rules that apply to cells in the bottom row, excluding cell 1; (3) rules that apply to cells in the leftmost column, excluding cell 1; and (4) rules that apply to all other cells. A *cell type* is designated to each cell in order to specify which set of rules apply:

- type 0: $x = 0, y = 0$;
- type 1: $x > 0, y = 0$;
- type 2: $x = 0, y > 0$;
- type 3: $x > 0, y > 0$.

q : current cell from the threading pattern

$$F_1: lm - q \geq p - \mu$$

$$F_2: p > \mu$$

$$F_3: (x_q \neq m-1) \vee [(x_q = m-1) \wedge (h_{y_q} \geq 1)]$$

$$F_4: (y_q \neq l-1) \vee [(y_q = l-1) \wedge (v_{x_q} \geq 1)]$$

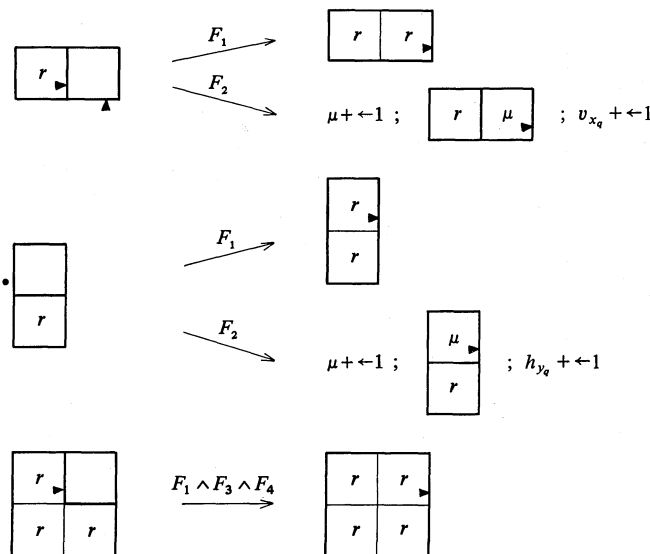


Figure 3. Colouring rules for standard $[p, 2]$ -rectangulations.

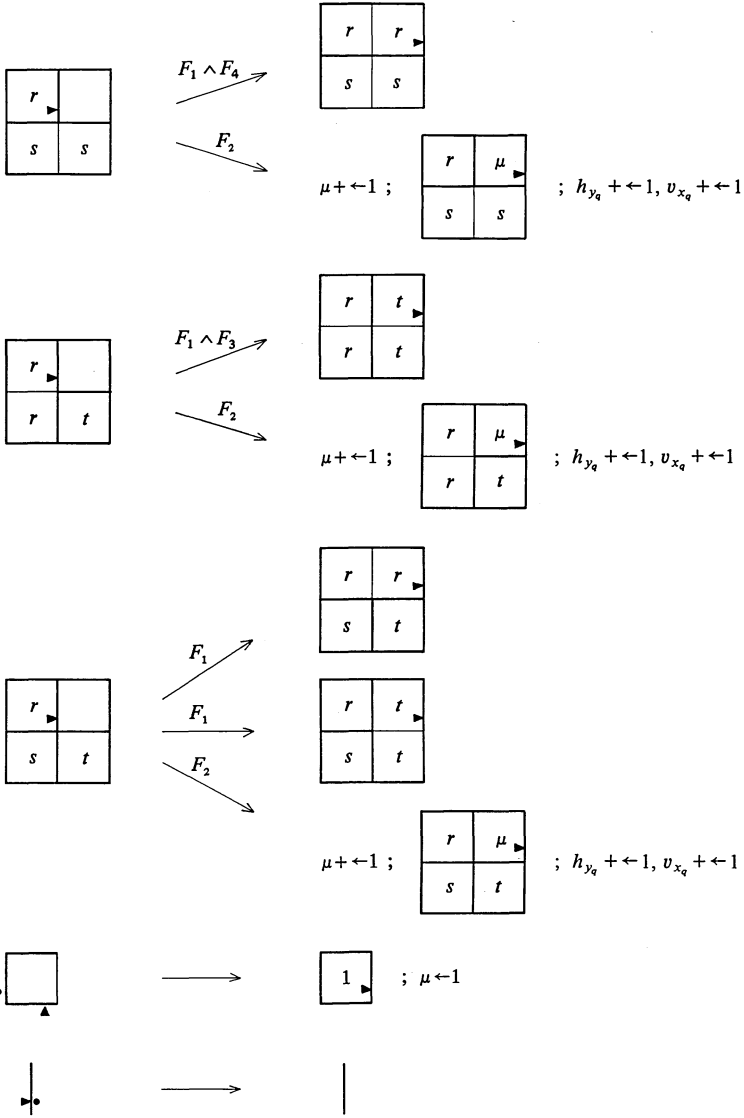


Figure 3 (continued).

$F_5: v_{x_q} = 0$
 $F_6: h_{y_q} = 0$

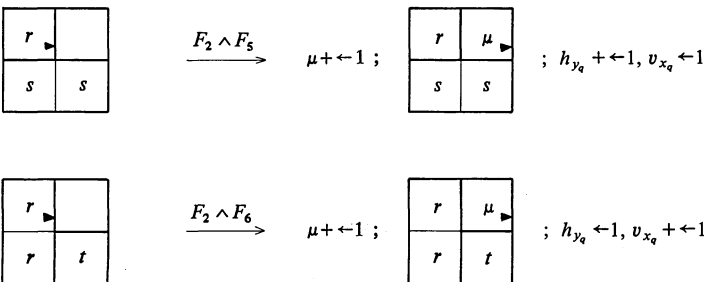


Figure 4. Modification of the colour rules in figure 3 to ensure nonalignment.

Let q denote the current cell number. What colour q takes depends upon its previously coloured neighbour(s). For type 1 cells the only such neighbour is $q-1$; hence extending the rectangle is equivalent to setting $\phi(q) = \phi(q-1)$, and creating the start of a new rectangle is equivalent to setting $\phi(q) = \mu$. For type 2 cells the only previously coloured neighbour is the cell immediately below q , namely, $q-m$. The colouring rules are $\phi(q) = \phi(q-m)$ or $\phi(q) = \mu$. For type 3 cells the neighbours that play a role are $q-1$, $q-1-m$, and $q-m$. Their respective ϕ -colours are r , s , and t . Four cases arise:

$r = t$, which implies $r = s = t$: only one rule applies, namely, $\phi(q) = r$;

$r \neq s = t$: two rules apply, $\phi(q) = r$ or μ ;

$r = s \neq t$: two rules apply, $\phi(q) = t$ or μ ;

$r \neq s \neq t$, which implies $r \neq t$: three rules apply, $\phi(q) = r, t$, or μ .

From this it follows that a rule may be of type R2, that is $\phi(q) = \mu$, if and only if cell type = 3 and $r = t$ do not hold simultaneously. Note that rules of type R2 apply only if the number of colours used so far does not exceed p ; that is, if $p > \mu$. Also rules of type R1 apply only if the number of cells to be coloured exceeds the number of new colours required; that is, if $lm - q \geq p - \mu$.

Let h_y and v_x denote the number of lines on grid lines $Y = y$ and $X = x$. At least one line is created in the plan whenever a rule of type R2 has been applied. For cells of type 1, a line on $X = x_q$ is created (increment v_{x_q}); for cells of type 2, a line on $Y = y_q$ is created (increment h_{y_q}); for cells of type 3, lines on both $Y = y_q$ and $X = x_q$ are created (increment v_{x_q} and h_{y_q}). In order to ensure that the rectangulation is standard, type 3 cells whose x value is $m-1$ and/or whose y value is $l-1$ are tested. For rules of type R1 to apply there must be at least one line in the plan on the grid lines touching these cells (which grid line is considered is apparent from the context). To ensure trivalent plans, the rule of type R2 for type 3 cells with $r \neq s \neq t$ does not apply. For nonaligned rectangulations the following rules of type R2 cannot be applied: when the cell is of type 3 and either (1) $s = t$ and there is a maximal line on the grid line $X = x_q$, or (2) $r = s$ and there is a maximal line on the grid line $Y = y_q$.

From the results stated earlier, when $l+m-1 = p$ only those rules that ensure trivalency and nonalignment need be applied.

The algorithm based on this outline was implemented in ALGOL68C and tested for values of p up to 11. The rule applications were further pruned by use of the symmetry properties associated with $\phi(P)$. The results of the generation up to $p = 10$ are reported in Bloch and Krishnamurti (1978).

Example 2: Palladian plans

Palladian plans are standard arrangements of rectangular, T-shaped, I-shaped, and +-shaped regions laid out with respect to a single axis on a $(l, 2m+1)$ grid (Stiny and Mitchell, 1978a). In this example the axis of symmetry is the Y -axis. Note that in any Palladian plan there can be at most one nonrectangular element, and this must be located so that it is bisected by the axis of symmetry (Stiny and Mitchell, 1978a).

Owing to the symmetry of Palladian plans, it is only necessary to consider the generation of half-plans on the $(l, m+1)$ unit grid. The central figures, namely, the T-shaped, I-shaped, and +-shaped regions, respectively become the leftmost (or rightmost, depending upon the half-plan chosen) Γ -shaped, $\bar{\Gamma}$ -shaped, and \dagger -shaped figures.

Consider two cases: (1) plans with no central figures, and (2) plans with a central figure. Case (1) is equivalent to a rectangulation on the $(l, m+1)$ grid, and the colouring rules in figure 2 can be applied. Case (2) is further divided into three subcases, cases (2a), (2b), and (2c), depending upon whether the plans have T-, I-, or +-shaped central figures. Consider case (2a). Notice that any horizontal reflection of

satisfied:

$$\alpha_1, \alpha_2 \geq 1, \tag{1}$$

$$\alpha_1 + \alpha_2 \leq m + 1, \tag{2}$$

$$1 \leq \beta_2 < \beta_1 \leq l, \tag{3}$$

$$\alpha_1 > 1 \Rightarrow \beta_1 < l, \tag{4}$$

$$\beta_2 > 1 \Rightarrow \alpha_1 + \alpha_2 \leq m, \tag{5}$$

and

$$0 \leq \delta \leq l - \beta_1. \tag{6}$$

Conditions (4) and (5) are to ensure that the plan is standard. These conditions are mapped into an algorithm (algorithm 3).

For the [-shaped figure, conditions (1) to (5) are needed together with the following four conditions:

$$\beta_3 \geq 1, \tag{7}$$

$$\beta_2 + \beta_3 < \beta_1, \tag{8}$$

$$0 \leq \delta \leq \frac{l - \beta_1}{2}, \tag{9}$$

and

$$\delta = \frac{l - \beta_1}{2} \Rightarrow \beta_3 \leq \beta_2. \tag{10}$$

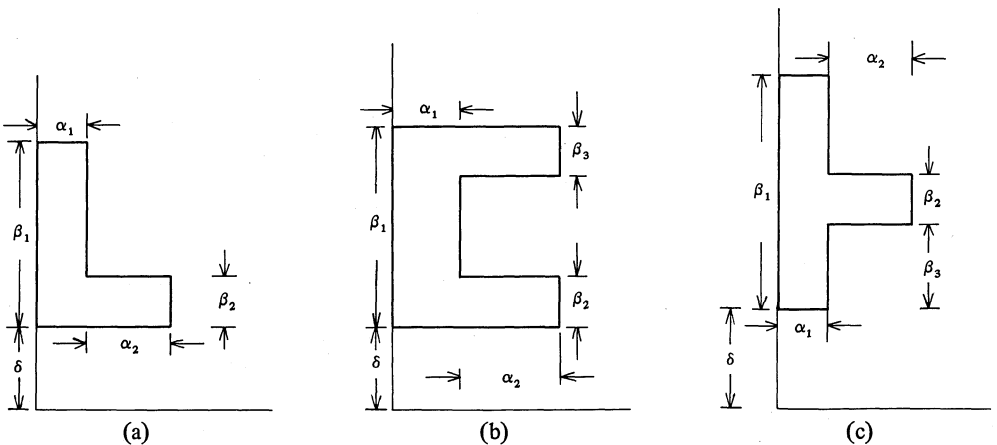


Figure 6. Outlines of the central figures for Palladian plans.

Algorithm STARTING T-PLANS

for $\alpha_1 \leftarrow 1$ until m do

 for $\beta_1 \leftarrow 2$ until (if $\alpha_1 = 1$ then l else $l - 1$) do

 for $\beta_2 \leftarrow 1$ until (if $\alpha_1 = m$ then 1 else $\beta_1 - 1$) do

 for $\alpha_2 \leftarrow 1$ until (if $\beta_2 = 1$ then $m + 1 - \alpha_1$ else $m - \alpha_1$) do

 for $\delta \leftarrow 0$ until $l - \beta_1$ do

$\alpha_1, \alpha_2, \beta_1, \beta_2,$ and δ specify a starting central T as shown outlined in half-figure form in figure 6(a)

Algorithm 3.

For T-shaped figures, conditions (1) to (5) and (7) to (9) are needed together with the following condition:

$$\delta = \frac{l - \beta_1}{2} \Rightarrow \beta_3 \leq \frac{\beta_1 - \beta_2}{2} \tag{11}$$

It is left as an exercise to the readers to specify the algorithms that construct the starting patterns for the T- and + -shaped central figures. It can be easily seen that the symmetrically placed T-shaped and + -shaped figures correspond to the condition $\delta = (l - \beta_1)/2$, and $\beta_3 = \beta_2$ or $\beta_3 = (\beta_1 - \beta_2)/2$ respectively.

The numbers of starting patterns for Palladian plans with the different central figures is given in table 1. Figure 7 illustrates the set of starting patterns for the (3, 5) grid. The algorithm based on the colouring rules in figures 2 and 5 was implemented in ALGOL68C. The results for the plan generation on various (l, 2m + 1) grids are reported in Stiny and Mitchell (1978b).

Table 1. Numbers of starting patterns, with different central figures, for Palladian plans based on various grids.

Central figure	Grid								
	(3, 3)	(3, 5)	(4, 5)	(5, 5)	(6, 5)	(7, 5)	(3, 7)	(4, 7)	(5, 7)
T	3	9	21	39	64	97	17	43	84
T	1	2	6	16	30	55	3	11	33
+	1	2	6	19	37	73	3	11	37

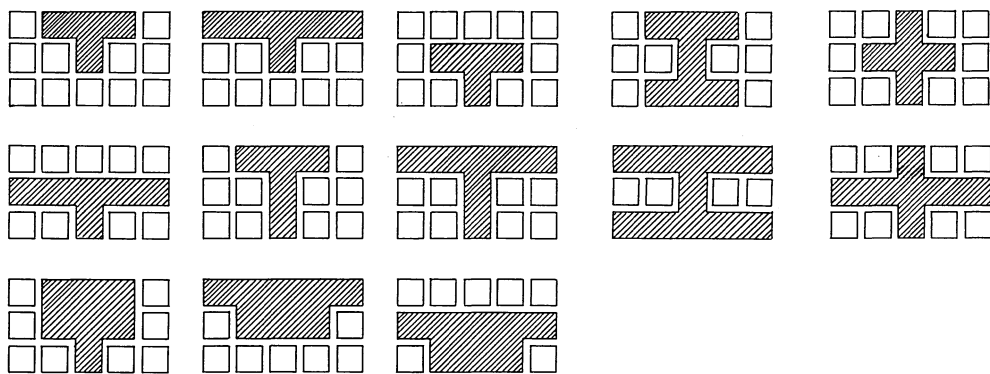


Figure 7. Starting patterns for Palladian plans on the (3, 5) grid.

Example 3: C₄- and D₄-plans

A C₄-plan is a rectangulation that possesses all the rotational symmetries. Clearly C₄-plans only exist on (l, l) unit grids. Two plans are equivalent if one is a reflection of the other. A D₄-plan is a C₄-plan which possesses in addition a reflectional symmetry. The remainder of the discussion is directed towards C₄-plans.

For the purposes of generating C₄-plans, the unit grid is viewed as being composed of necklaces of square cells. These are defined in the following manner. If l is odd, the (l, l) unit grid can be constructed from a single (1, 1) square by surrounding this square by a boundary of (1, 1) square cells to form the (3, 3) unit grid, and then successively surrounding this and each other new grid by another boundary of (1, 1) square cells until the (l, l) unit grid is arrived at. Each of these boundaries is a necklace of the (l, l) unit grid. The initial (central) (1, 1) square cell is also considered

to be a necklace. For example, the (5, 5) unit grid can be constructed in two steps from a single (1, 1) square, one to form the (3, 3) unit grid and the second to form the (5, 5) unit grid, and has three necklaces, the central (1, 1) square cell, the boundary of the (3, 3) grid, and the boundary of the (5, 5) grid. If l is even, the (l, l) unit grid can be similarly constructed from the (2, 2) unit grid, and the necklaces are defined in the obvious way.

Each plan is considered to unfold, necklace by necklace, from the centre outwards. Consider two successive necklaces with $4(m-1)$ and $4(m+1)$ square cells respectively. Any colouring of the cells in the outer necklace depends upon the colours associated with the cells in the inner necklace. The colours of the corner cells in the outer necklace also depend upon the colours of all neighbouring cells in the outer necklace, and hence are coloured last. Furthermore, when any cell is coloured, all cells corresponding to this cell through a cyclic rotation must bear the same colour relationship with their neighbouring cells. These considerations lead to a recursive specification for the threading pattern for C_4 -plans. A threading pattern, specified in terms of one necklace, is illustrated in figure 8.

The set of colouring rules are shown in figure 9. It is left as an exercise to the readers to convince themselves that the rules yield unique generations of the plans and that all plans are exhaustively generated. It should be noted that, in rules 3 to 5, where four cells are coloured at once, q only refers to the cell in the threading pattern which occurs first out of these four, namely the one whose neighbouring cells have parametrized colour(s) with subscript 1.

An ALGOL-like translation for the rules in figure 9 is given in algorithm 4. The threading pattern is as usual denoted by the ordered set $\langle (x_1, y_1), (x_2, y_2), \dots \rangle$, where $\Gamma[(x_j, y_j)] = j$, and is represented by the arrays x and y . The colours are applied in groups of four cyclically equivalent cells denoted by their cell numbers $q, q+1, q+2$, and $q+3$. All cyclically equivalent cells are stored in a circular list whose nodes consist of two fields *link* and *cell*: *link* contains a pointer to the next node in the list; *cell* contains the cell number of the next cell taken either clockwise or counterclockwise depending upon the programmers fancy and the threading pattern chosen. The appropriate node in this list is pointed to by *ptr*. The rules are selected and placed on the *stack*, and are applied recursively. The number of colours used so far is contained in μ .

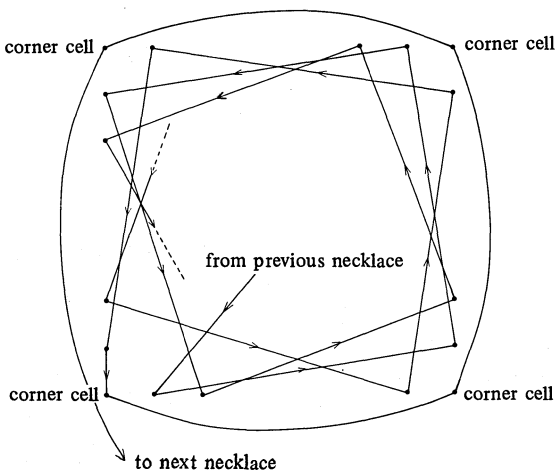


Figure 8. Recursive specification of the threading pattern. The dots represent grid cells.

- q : first of the four current cells
from the threading pattern
- F_1 : $x_q = y_q + 1$
 - F_2 : $x_q > y_q + 1$
 - F_3 : $x_q = y_q$
 - F_4 : $(r_1 = t_1) \vee (r_1 \neq s_1)$
 - F_5 : $(r_1 \neq t_1) \wedge (t_1 \neq s_1)$
 - F_6 : $r_1 \neq t_1$

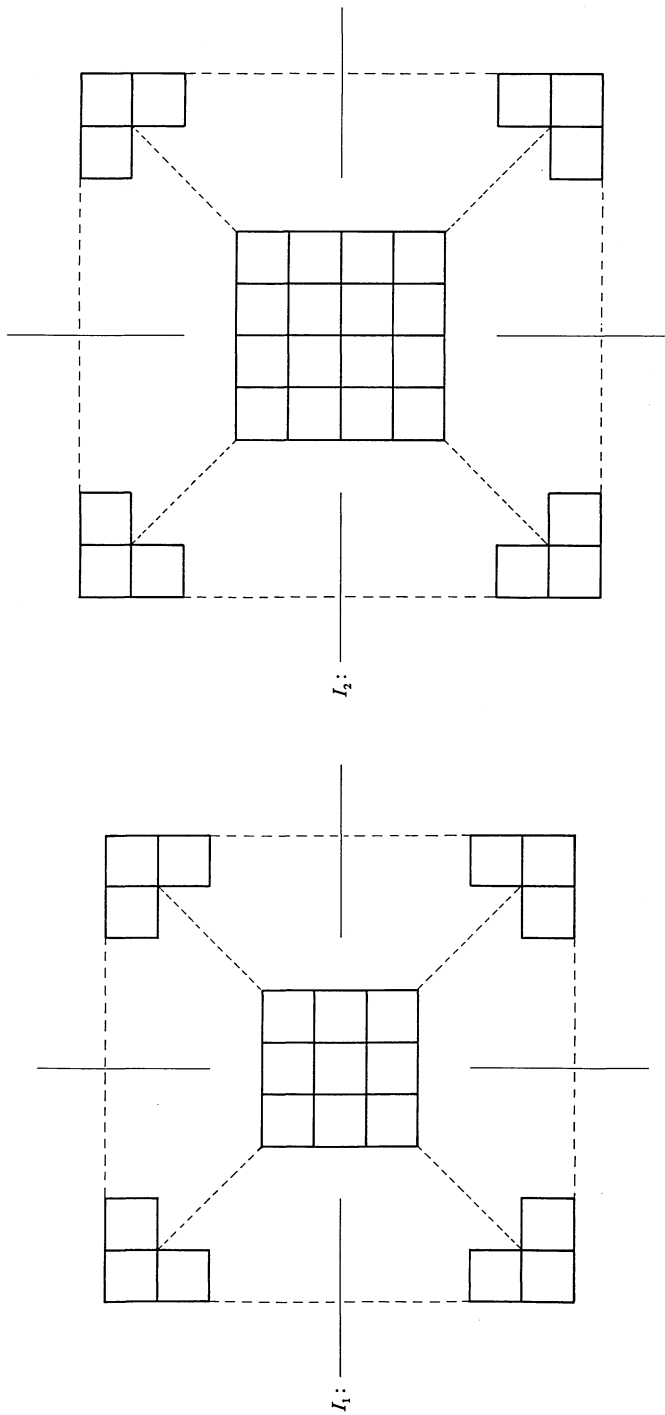


Figure 9. Colouring rules for C_4 -plans. In I_1 and I_2 , the dashed lines and the sets of three corner cells merely indicate the full extent of the grid.

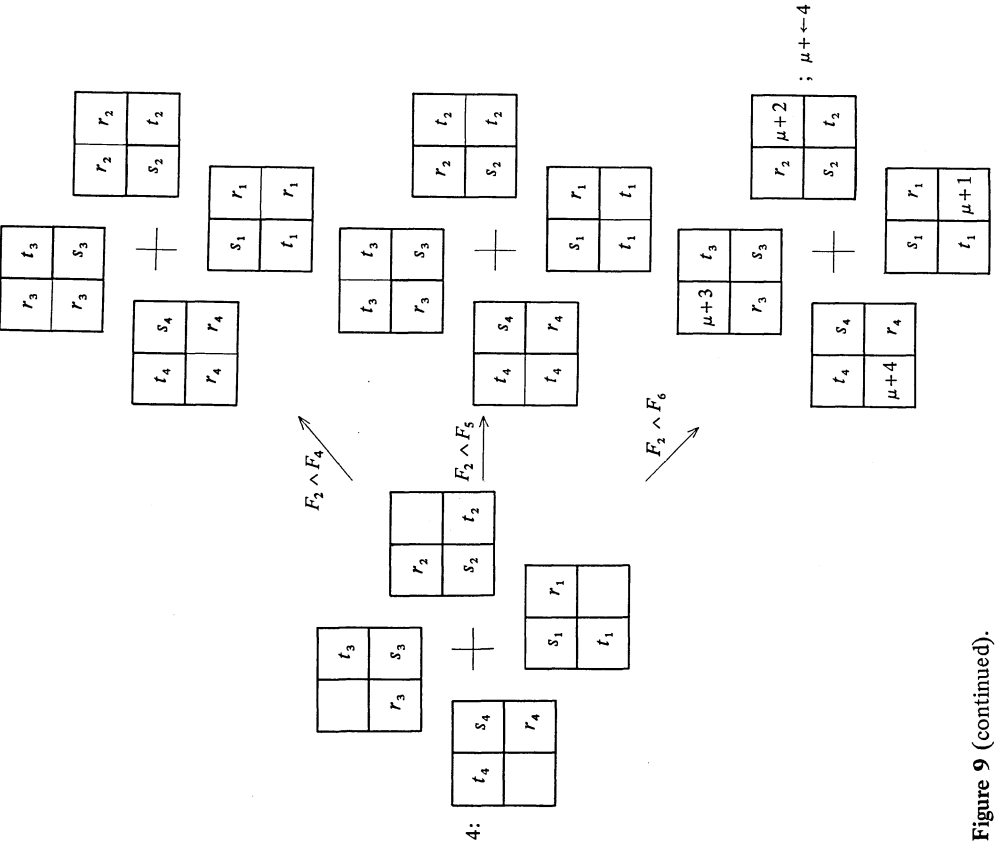
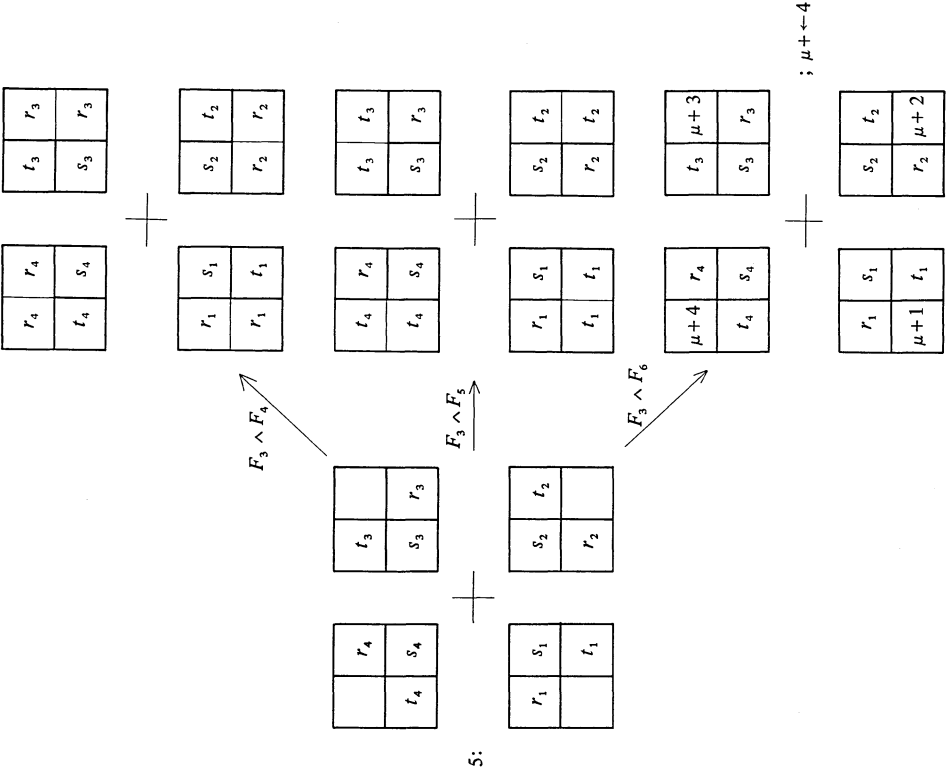


Figure 9 (continued).

To facilitate rule application cell types (denoted merely by *type* in the algorithm) are associated with the cells. For any cell with coordinates (x, y) , its cell type is specified as follows:

type 1: $x = y + 1$;

type 2: $x > y + 1$;

type 3: $x = y$.

The rules are simplified in a manner similar to that done for rectangulations.

Standard plans are ensured by keeping track of the number of distinct lines on the grid lines, and, when the outermost necklace is reached (that is, when the y -value of the first cell in each set of four cyclically equivalent cells is equal to zero), tests are

Algorithm C_4 -PLANS

¶ q points to the current cell in the threading pattern ¶

if $q > l^2$

then if CANONICAL('generated plan') then plan is canonical

```

else
  if case type( $q$ ) of
    (= 1) true
      (= 2)  $\phi[\Gamma(x_q - 1, y_q)] \neq \phi[\Gamma(x_q, y_q + 1)]$ 
      (= 3)  $\phi[\Gamma(x_q + 1, y_q)] \neq \phi[\Gamma(x_q, y_q + 1)]$ 
    then ¶ rules of type R2 ¶
      add lines on the appropriate grid lines
      to 4 do
        {  $\phi(q) \leftarrow (\mu + 1)$ 
           $q \leftarrow 1$ 
        }
       $C_4$ -PLANS
       $q \leftarrow 4$ 
       $\mu \leftarrow 4$ 
      remove the added lines from the appropriate grid lines
    if
      if  $y(q) = 0$  then { test for standard plans depending upon the type and  $x$ -value
                        } of  $q$  (the test returns value of either true or false)
      else true
    then ¶ declare local variable save initially set to stack ptr ¶
      ¶ rules of type R1 ¶
      if type( $q$ ) = 1
      then  $stack(stack\ ptr + 1) \leftarrow \Gamma(x_q, y_q + 1)$ 
      else case type( $q$ ) of
        (= 2) {  $r \leftarrow \Gamma(x_q, y_q + 1)$ 
              {  $s \leftarrow \Gamma(x_q - 1, y_q + 1)$ 
                {  $t \leftarrow \Gamma(x_q - 1, y_q)$ 
              }
            }
        (= 3) {  $r \leftarrow \Gamma(x_q, y_q + 1)$ 
              {  $s \leftarrow \Gamma(x_q + 1, y_q + 1)$ 
                {  $t \leftarrow \Gamma(x_q + 1, y_q)$ 
              }
            }
        if  $[\phi(r) = \phi(t)] \vee [\phi(r) \neq \phi(s)]$  then  $stack(stack\ ptr + 1) \leftarrow r$ 
        if  $[\phi(t) \neq \phi(r)] \wedge [\phi(t) \neq \phi(s)]$  then  $stack(stack\ ptr + 1) \leftarrow t$ 
      while  $stack\ ptr > save$  do
        {  $neighbour \leftarrow stack(stack\ ptr)$ 
          {  $node \leftarrow ptr(neighbour)$ 
            to 4 do
              {  $\phi(q) \leftarrow \phi(neighbour)$ 
                {  $neighbour \leftarrow cell(neighbour)$ 
                  {  $node \leftarrow link(node)$ 
                }
              }
            }
          }
        }
         $stack\ ptr \leftarrow 1$ 
       $C_4$ -PLANS
       $q \leftarrow 4$ 

```

Algorithm 4.

made to check whether any lines need to be added. These tests only need be applied to cells of type 2 whose x -value is not less than $\lceil l/2 \rceil$, and to corner cells of type 3. Lines are created on the grid lines when colour rules of type R2 are applied.

The algorithm was implemented in ALGOL68C. The plans were generated on various (l, l) unit grids and classified according to their symmetry groups. The total numbers of the C_4 - and D_4 -plans for values of l up to 7 are shown in table 2, and the plans for $l = 3$ and $l = 4$ are illustrated in figure 10.

Table 2. The numbers of C_4 - and D_4 -plans for $l = 1, \dots, 7$.

l	1	2	3	4	5	6	7
C_4 -plans	1	1	2	8	39	411	4584
D_4 -plans	1	1	1	3	5	27	57

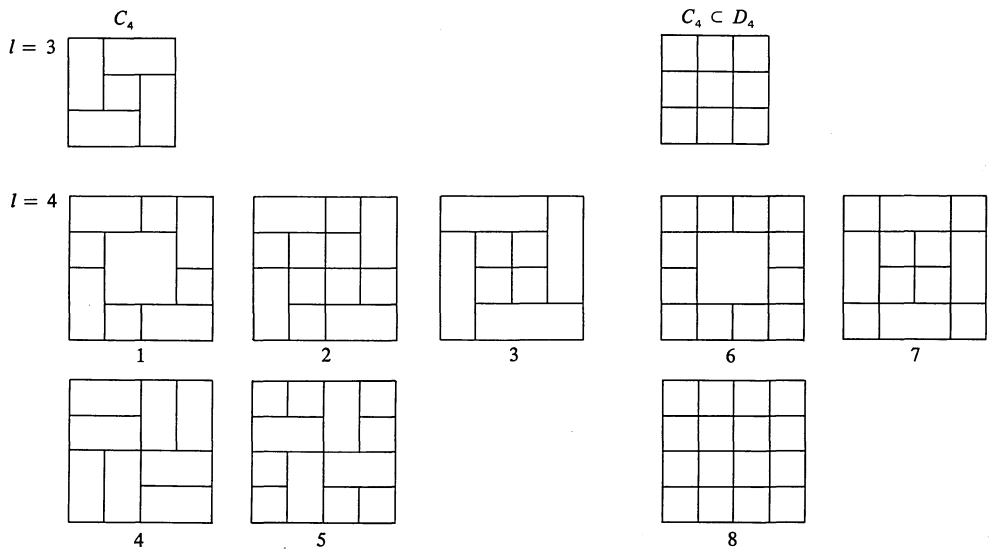


Figure 10. C_4 - and D_4 -plans for $l = 3$ and $l = 4$.

Example 4: tatami mat designs

A tatami is a mat made of rice straw sewn together and which measures 6×3 shaku (Yoshida, 1955). According to Morse (1961) a tatami design is an arrangement of tatami mats that are laid in a rectangular room and in which no four mats meet at a point. This restriction seems peculiar to Morse (see Kirby, 1962; Gropius et al, 1960). In exceptional circumstances the use of a half-mat measuring 3×3 shaku is permitted. For our purposes the mats are given computational aliases—a mat becomes a 1×2 rectangle and the half-mat becomes a 1×1 square. A p -mat plan is an arrangement of p 1×2 rectangles that pack a unit grid. A $(p + \frac{1}{2})$ -mat plan is an arrangement of p 1×2 rectangles and one 1×1 square on a unit grid. Note that these designs need not be standard in the sense defined earlier.

The colouring rules are similar to those for generating rectangulations. The reader is urged to derive a possible set of rules which uniquely and exhaustively generate the plans. Our own algorithm (which we do not give here) is based on a set of colouring rules and allows for tatami plans with and without trivalency. In table 3 are given the numbers of nonequivalent plans for traditional Japanese values for p . These numbers include nonequivalent arrangements on the trivial grid, namely, the $(1, 2p)$

grid for p -mat designs and the $(1, 2p+1)$ unit grid for $(p+\frac{1}{2})$ -mat designs. There is only one possible arrangement on the $(1, 2p)$ grid, and there are $\lfloor (p+1)/2 \rfloor$ arrangements on the $(1, 2p+1)$ grid. If these are discounted, the number of nontrivial tatami plans results. All tatami plans for $p = 3, 4,$ and $4\frac{1}{2}$ are shown in figure 11.

Table 3. Numbers of nonequivalent tatami plans.

p	3	4	$4\frac{1}{2}$	6	8	10	12	15
Tatami plans	3	5	6	15	31	85	272	990
Trivalent tatami plans	3	4	5	9	14	27	56	152

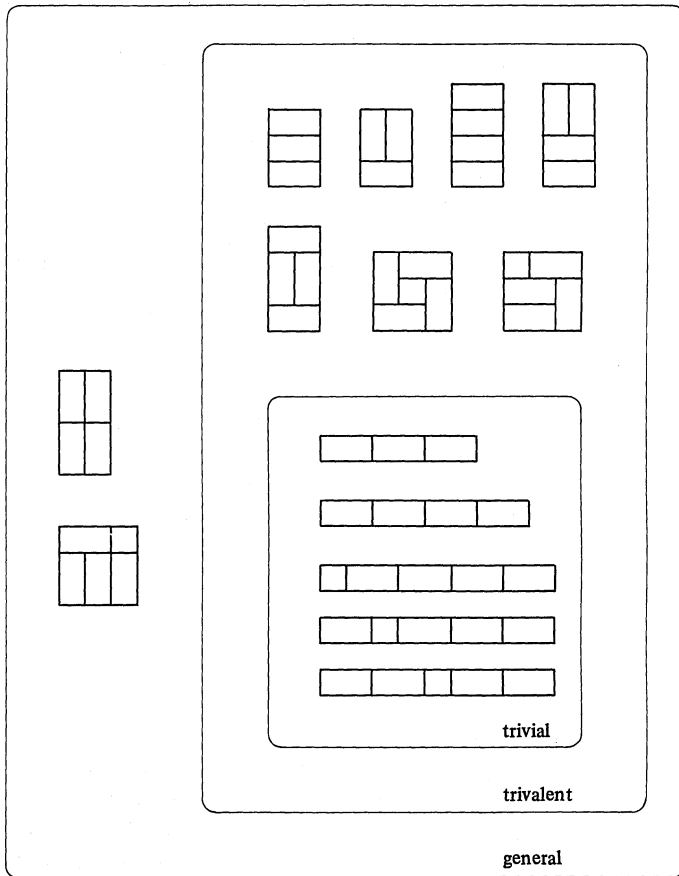


Figure 11. Tatami plans for $p = 3, 4,$ and $4\frac{1}{2}$.

Conclusions

In this paper we have presented the following.

1. A minimum lexicographic colouring for any arrangement of polygons on unit grids, which allows for the detection of symmetry isomorphs without having to search a list of generated objects or, indeed, without requiring any external storage device.
2. The notion of a threading pattern which describes the natural unfolding of a plan and its relation to the minimum colouring and to the definition of canonical plans.
3. The recursive specification of plans via predicate-controlled colouring rules as computational expressions of formal shape grammars.

4. The outline of the generation algorithm based on these colouring rules. The production of duplicates does, however, depend on the uniqueness of all possible sequences of rule application.
5. Four specific plan problems involving rectangular elements, which are representatives of larger classes of arrangements on gratings:
 - (1) rectangulations—as examples of arbitrary arrangements;
 - (2) Palladian plans—as examples of arrangements with predefined empty spaces or courtyards;
 - (3) C_4 - and D_4 -plans—as examples of arrangements possessing certain symmetry properties and as an illustration of a nontrivial threading pattern; and
 - (4) tatami plans—as examples of arrangements which implicitly employ the physical attributes of the plan elements.

The proof of any computational strategy does not lie so much in the elegance of the ideas but rather in the correctness of its algorithms and in its computational complexity. As a measure of the technique's efficiency for generating plans, we present some run times on the IBM 370/165 computer (approximate to the nearest minute or second): three minutes for the $[9, 2]$ -rectangulations, a total 58072 nonequivalent arrangements; eight minutes for the Palladian plans on the $(7, 5)$ unit grid, of which there are 2604791; and under ten seconds for the $(7, 7)$ C_4 -plans and 15-mat tatami designs.

Finally we claim that the ideas in this paper are easily adapted to higher-dimensional plans. In fact we have developed an algorithm based on these concepts for enumerating solid rectangular packings of cuboids, namely, the $[p, 3]$ -rectangulations. The results of that enumeration will be published in a forthcoming paper.

Acknowledgements. The first author (RK) is grateful to the University of Waterloo for financial aid in the form of a graduate studentship, and he would like to thank Dr Ho and Professor March for their encouragement and support.

References

- Biggs N, 1969 "Rectangulations" *Proceedings of the Cambridge Philosophical Society* 65 399–408
- Bloch C J, 1976 "On the set and number of minimal gratings for rectangular dissections" *Environment and Planning B* 3 71–74
- Bloch C J, Krishnamurti R, 1978 "The counting of rectangular dissections" *Environment and Planning B* 5 207–214
- Earl C F, 1977 "A note on the generation of rectangular dissections" *Environment and Planning B* 4 241–246
- Earl C F, 1978 "Joints in two- and three-dimensional rectangular dissections" *Environment and Planning B* 5 179–187
- Gips J, 1975 *Shape Grammars and Their Uses: Artificial Perception, Shape Generation and Computer Generation and Computer Aesthetics* (Birkhäuser, Basel)
- Gropius W, Tange K, Ishimoto Y, 1960 *Katsura Tradition and Creation in Japanese Architecture* (Yale University Press, New Haven, Conn.)
- Kirby J B, 1962 *From Castle to Teahouse* (Charles E Tuttle, Tokyo)
- Krishnamurti R, Roe P H O'N, 1979 "On the generation and enumeration of tessellation designs" (in preparation)
- Morse E S, 1961 *Japanese Homes and Their Surroundings* (Dover, New York); originally published by Ticknor and Company in 1886
- Reingold E M, Neivergelt J, Deo N, 1977 *Combinatorial Algorithms: Theory and Practice* (Prentice-Hall, Englewood Cliffs, NJ)
- Stiny G, 1975 *Pictorial and Formal Aspects of Shape and Shape Grammars: On Computer Generation of Aesthetic Objects* (Birkhäuser, Basel)
- Stiny G, 1977 "Ice-ray: a note on the generation of Chinese lattice designs" *Environment and Planning B* 4 89–98
- Stiny G, 1979 *The Grammar of Form* (Pion, London) forthcoming
- Stiny G, Mitchell W J, 1978a "The Palladian grammar" *Environment and Planning B* 5 5–18
- Stiny G, Mitchell W J, 1978b "Counting Palladian plans" *Environment and Planning B* 5 189–198
- Yoshida T, 1955 *The Japanese House and Garden* (Praeger, New York)