Nesting two-dimensional shapes in rectangular modules

MICHAEL ADAMOWICZ
(Dept. of Electrical Engineering, Polytechnic Institute of Brooklyn, 333 Jay St., Brooklyn, New York 11201, USA)
ANTONIO ALBANO
(Istituto di Scienze dell'Informazione, Università di Pisa, Corso Italia 40, 56100 Pisa, Italy)

A problem of relevant interest to some industries is that of optimum two-dimensional layout. In this problem, one is given a number of rectangular sheets and an order for a specified number of each of certain types of two-dimensional regular and irregular shapes. The aim is to cut the shapes out of the sheets in such a way as to minimize the amount of waste produced. A two-stage solution is proposed in which the problem is converted from one of placing irregularly shaped pieces to one of allocating rectangular modules. The clustering algorithm used in the first stage to produce rectangular modules is presented and the results obtained when it was applied to some typical layout problems are described.

(Received in revised form on June 13th 1975)

A problem encountered in some industries is that of optimum two-dimensional layout. An operator is given a number of two-dimensional sheets, an order for a specified number of each of certain types of two-dimensional shapes and a set of constraints. The objective is to cut the shapes out of the sheets in such a way as to minimize the amount of waste produced and to satisfy all the constraints[1–4]. Previous investigations concerned with rectangular sheets and shapes are reported in References 5–11. One example of the general two-dimensional allocation problem is encountered in the shipbuilding industry. In this example the operator is provided with:

1. A set ot standard rectangular sheets of steel
2. An order to produce a large number of various types of shapes which include irregular as well rectangular-like shapes
3. The requirements that no two shapes may overlap and that only the exact types and numbers of shapes ordered are to be produced
4. Minimum waste.

Because of the large number of pieces involved in this type of problem, an algorithm that seeks for an optimal layout working directly with the irregular shapes sounds infeasible. The approach proposed in Reference 3 is a two-stage solution in which the problem is converted from one of placing irregular shapes to one of allocating rectangular modules. The first stage is concerned with producing 'optimal' rectangular enclosures, called *modules*, for the shapes and for various clusterings of the shapes, and the second stage finds 'optimal' layouts of the modules on the rectangular sheets.

In Reference 11 we provide a complete description of the rectangular layout stage of the algorithm. The results obtained indicate that it is an effective tool in designing minimal waste layouts for problems that involve:

1. Rectangular sheets
2. Orders for a large (or small) number of pieces having a wide range in dimensions
3. Non-overlapping restrictions
4. The requirements of producing only the exact number and types of pieces ordered.

This paper describes the algorithm for clustering irregular pieces in rectangular modules. Each piece is marked by a processing option. The set of options available to the user permits him to determine the types of clustering produced and to control the amount of processing time required per shape type.

SOLUTION APPROACH

Basic approach

The principal task of the algorithm is to accept an order for a specified number of various types of shapes (which may include irregular as well as rectangular shapes) and to produce a list of rectangular modules. These modules will, in general, be obtained from:

1. Rectangular shapes
2. Rectangular enclosures for irregular shapes
3. Rectangular enclosures for pairs of irregular shapes
4. Rectangular enclosures for various clusterings of irregular and rectangular shapes.

The overall procedure is shown in Figure 1.

To make the algorithm as flexible as possible and to provide the possibility of user interaction, it was organized as a set of seven shape processing options:

1. Input and process a rectangular shape type.
2. Input an irregular shape type and produce the 'best' rectangular enclosure for the shape type.
3. Perform option 2. If the enclosure is good enough, accept it. Otherwise, determine the 'optimal' clustering of two shapes of that type when the orientations of the shapes differ by 180°. Select either the rectangular enclosure of the single shape or the 'optimal' clustering of the two shapes depending on which is better.

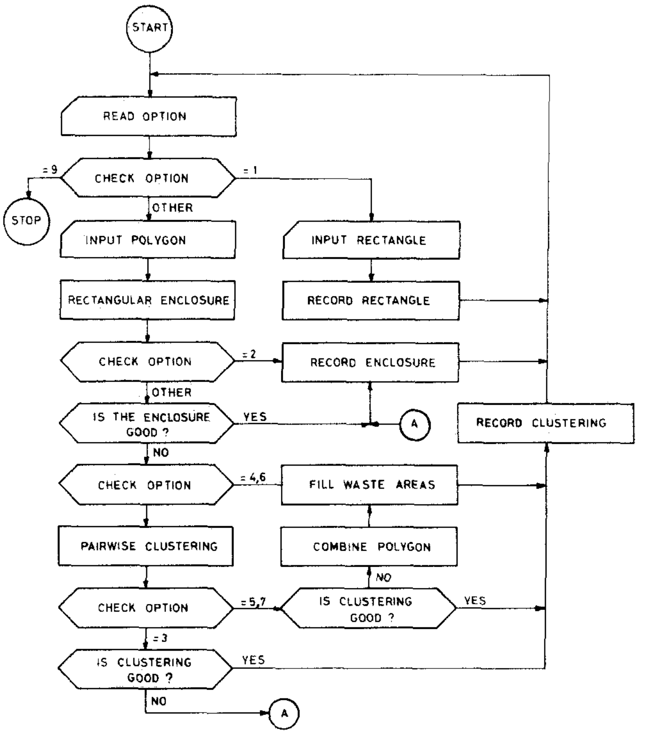

FIGURE 1. Overall approach

4, Perform option 2. If the enclosure is good enough,
5, accept it. Otherwise try to fill in the waste areas with rectangles†.

6, Perform option 2. If the enclosure is good enough,
7. accept it. Otherwise determine the 'optimal' clustering of two shapes of the given type when the orientations of the shapes differ by 180°. If the clustering is good enough, accept it. Otherwise, try to fill in the waste areas of the clustering with rectangles†.

Figure 2 provides examples showing the types of results one could obtain using the various clustering options.

† The difference between these options is described later in the paper.

Shape representation

The algorithm assumes that each shape is either a rectangle or a polygon. A rectangle is described by giving its length and width. The reference point of a rectangle is originally assumed to be its lower left-hand corner (see Figure 3a). A polygon is described by providing a list of the points (vertices) of the polygon given in a counter-clockwise (CCW) direction (see Figure 3b). The co-ordinates of each point of the polygon are given relative to the polygon reference point, which may or may not be a point on the original polygon.

Using these representational schemes one can completely describe the placing of a shape (rectangle or polygon) by specifying the position of the reference point and the orientation (CCW) of the shape about the reference point.

An alternative method for completely describing the placing of a shape, often useful at various stages of the processing of polygons, involves translating and rotating each point of the polygon. Suppose that we wish to rotate a given polygon by θ degrees CCW about the reference point (x_0, y_0) and then translate it by $(\Delta x, \Delta y)$. Then for each point (x_i, y_i) we can compute the corresponding point (x_i', y_i') of the repositioned polygon as follows:

$$x_i' = x_i \cos\theta - y_i \sin\theta + \Delta x$$

$$y_i' = x_i \sin\theta + y_i \cos\theta + \Delta y$$

The position of the reference point remains the same.

Multiple-connected shapes, that is, shapes with holes, must be coded so that they will be handled as a simply-connected one. Figure 3c provides an example of how such a shape can be coded. Of course, the description of Figure 3d is used to draw the layout.

Rectangular enclosure

The procedure for providing 'optimal rectangular enclosures' for irregular, simply-connected polygons is a combination

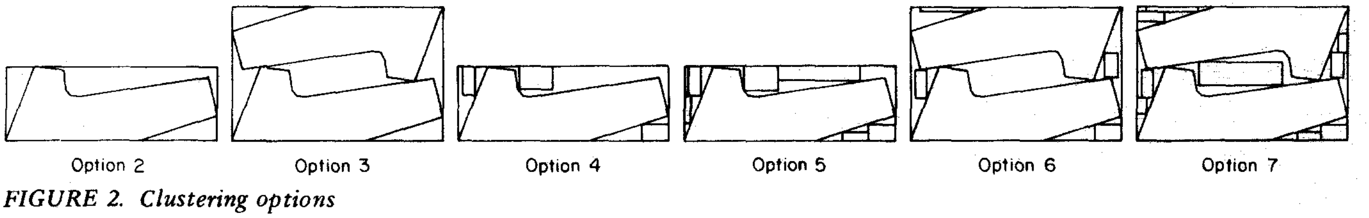

FIGURE 2. Clustering options

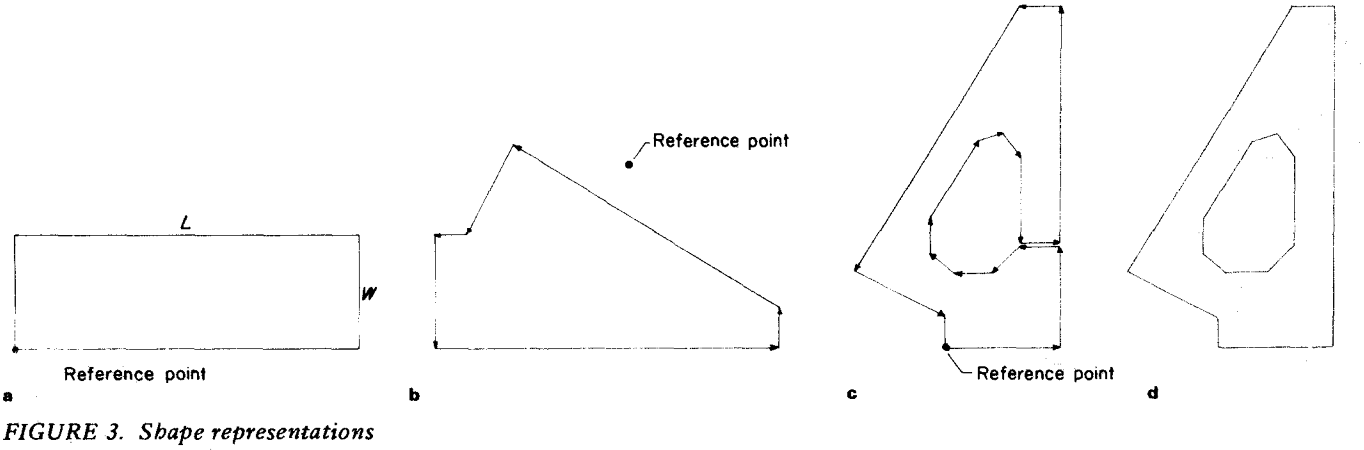

FIGURE 3. Shape representations

of heuristic and algorithmic techniques. Given an arbitrary finite polygon, a finite set of 'potentially good absolute orientations' are determined. Since the number and values of these orientations are determined by the shape itself, we can avoid being restricted to a predetermined set of angles with fixed increments. Because of this, processing is performed quickly and often yields the absolute minimum area rectangular enclosure for an arbitrary shape type. Before describing the procedure for selecting a set of orientations, two useful definitions are presented. These are:

Definition. Given a vertex (x_i, y_i) of an arbitrary, simply-connected polygon, the 'incoming vector' is the directed line from vertex (x_{i-1}, y_{i-1}) to (x_i, y_i). The 'outgoing vector' is the directed line from vertex (x_i, y_i) to (x_{i+1}, y_{i+1}).

Definition. A vertex of a given polygon is a point of 'convexity' if the cross product of the incoming vector with the outgoing vector is not negative. The magnitude of the cross product of the incoming vector with the outgoing at vertex i of a given polygon is:

$$(x_i - x_{i-1})(y_{i+1} - y_i) - (y_i - y_{i-1})(x_{i+1} - x_i).$$

The procedure for determining a finite set of potentially good absolute orientations for a given shape type makes use of a technique often used by experienced men. The technique involves determining the angles of rotation such that one or more of the sides of the shape become parallel with either the horizontal or the vertical axis. The procedure used here considers only those sides that connect two points of convexity of the polygon. An example of the use of this technique is given in Figure 4. Since vertex 5 is not a point of convexity, lines 4–5 and 5–1 are not considered in determining the potentially good absolute orientations.

Once the set of potentially good absolute orientations is obtained, the procedure examines each of the angles in turn. For each orientation, the smallest rectangular enclosure with sides parallel to the x and y axes is determined for the rotated polygon. The 'optimal' rectangular enclosure is chosen as the one with the minimum area.

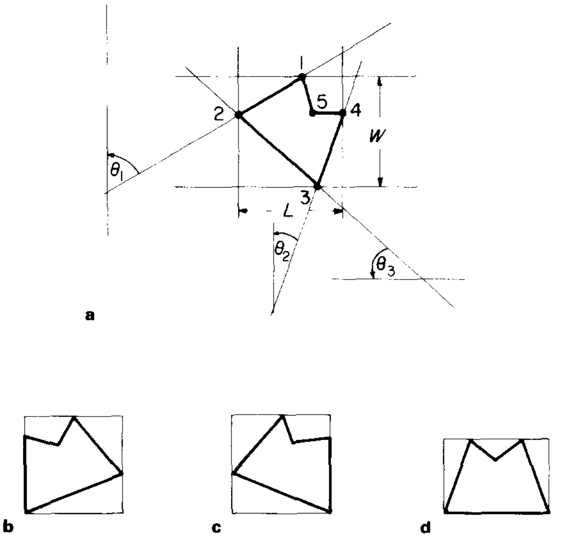

FIGURE 4. Rectangular enclosure at potentially good absolute orientations. (a) Original orientation; Rotation by (b) θ_1; (c) θ_2; (d) θ_3

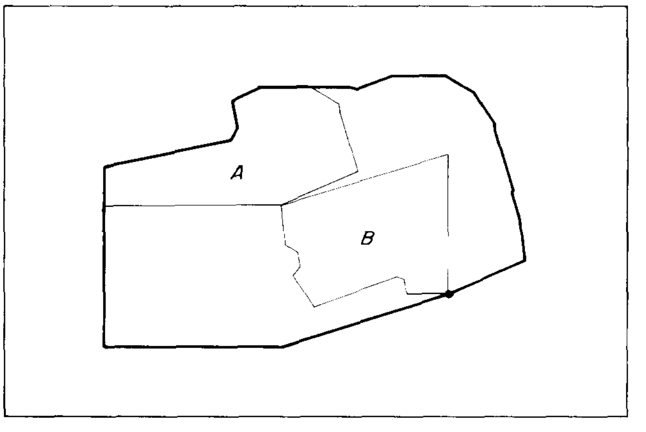

FIGURE 5. The no fit-polygon

The no-fit-polygon

An algorithm has been developed for completely determining all the arrangements that two arbitrary polygons may assume such that the shapes do not overlap. The polygons are assumed to have fixed absolute orientations but are free to move anywhere on the 2-D plane. A simplified version of the algorithm was originally presented in Reference 1.

The description of all the arrangements that two polygons may assume without overlapping is specified in terms of a no-fit-polygon (NFP). Given two polygons A and B such that the position of A, the orientation of A, and the orientation of B are all fixed, then the NFP of B relative to A completely describes all those positions where the reference point of B may be placed in order to have B touching A without overlapping it. An example of NFP is given in Figure 5.

With multiple-connected shapes, the NFP can be composed of a set of polygons. This can also happen with shapes individually simply-connected.

Definition. Given two polygons A and B, if the NFP of B relative to A is composed of a single polygon, we say that A and B are relatively simply-connected. Otherwise they are relatively multiple-connected.

The NFP is one of the most important features of the overall clustering algorithm and, as we will see in the following, it is fundamental for the clustering process.

Pairwise clustering

There are several aspects of the pairwise clustering problem that make it potentially quite difficult to handle. One source of difficulty is the potentially unmanageable combinatorial problem associated with selecting, (1) a pair of shapes; (2) an absolute orientation for one shape; and (3) a relative orientation for the second shape with respect to the first. A second source of difficulty derives from the problem of finding the 'optimum' clustering of two shapes with given orientations.

Suppose that we were given two polygons A and B with fixed orientations and we wanted to find the position of B relative to A such that the shapes do not overlap and also such that the area of the rectangular enclosure of the pair of polygons is minimized. It can show that we need examine only the points on the NFP in order to find the optimum position of B relative to A. However, there are still infinitely many such points to examine. The procedure included in

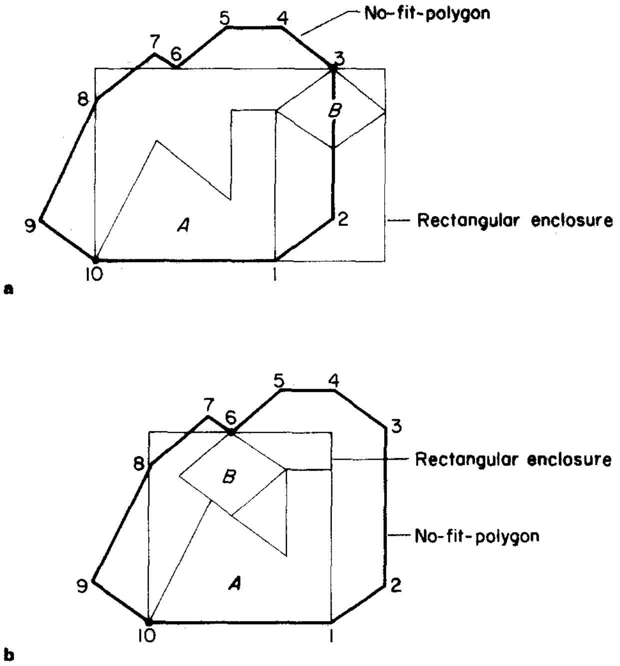

FIGURE 6. Clustering using the no-fit-polygon

the overall clustering algorithm gets around this difficulty by examining only the vertices of the NFP in the above manner.

Figure 6 provides an example of the use of the NFP in finding the minimum area rectangular enclosure for two relatively simply-connected polygons. Figure 6a portrays a rather poor clustering with the reference point of *B* placed at vertex 3 of the NFP. The minimum area rectangular enclosure for the shapes with the given orientations is obtained as shown in Figure 6b with the reference point of *B* placed at vertex 6. With the example in Figure 6 it was possible to find the optimum clustering by considering only the vertices of the NFP. However, this will not always hold true. The clustering algorithm presented here includes a special procedure to be used at each vertex in order to determine if it is possible to improve the clustering by moving the reference point part of the way along either of the two lines of the NFP connected to the vertex. This special procedure is referred to as the 'extended search'.

Definition. The 'spans' of a shape or set of shapes are the minimum and maximum x and y values for all the points contained in the shape or set of shapes.

Definition. The 'span control region' corresponding to a vertex of the NFP for polygon *B* relative to polygon *A*, is the region where placing the reference point of *B* the spans of the clustered shapes will not be extended.

Figure 7 provides a graphical representation of the application of the extended search. Figure 7a shows a clustering of two shapes obtained by placing the reference point of *B* on vertex 3 of the NFP of *B* relative to *A*. Figure 7b shows the 'span control region' corresponding to vertex 3 of the NFP. As shown in Figure 7a, the spans determine the minimum area rectangular enclosure for a positioned shape or set of shapes. The span control region ensures that the maximum x and y values will not be increased and the minimum x and y values will not be decreased.

Figure 7b also shows the 'extended search points', the intersections of the incoming and outgoing vectors associated with vertex 3 of the NFP with the boundary of the span control region. In degenerate cases, an extended search point is chosen as the one furthest from the vertex. Since these points are on the NFP, the two shapes will still be in contact but will not overlap. The extended search procedure examines both these intersection points to determine if an improved rectangular enclosure can be obtained by using either of them to position the shapes. In Figure 7b one of these intersection points is coincident with vertex 3 of the NFP, while the other lies along line 3–4 and is labelled *t*. The clustering resulting from this positioning has the absolute minimum area rectangular enclosure for the pair of shapes with the given orientations.

Using a clustering procedure that for each vertex of the NFP first examines the vertex and then employs the extended search, provides a technique for handling the second source of difficulty associated with the pairwise clustering problem: the difficulty of finding the 'optimum' clustering of two shapes with given orientations. However, there still remain the problems of deciding what two shapes to cluster and of determining which orientations to try.

An examination of some typical types of irregular shapes used in different application areas shows that with a large percentage of the types one can obtain very good clusterings by using two shapes of the same type and a relative orientation of 180°. The overall clustering algorithm makes use

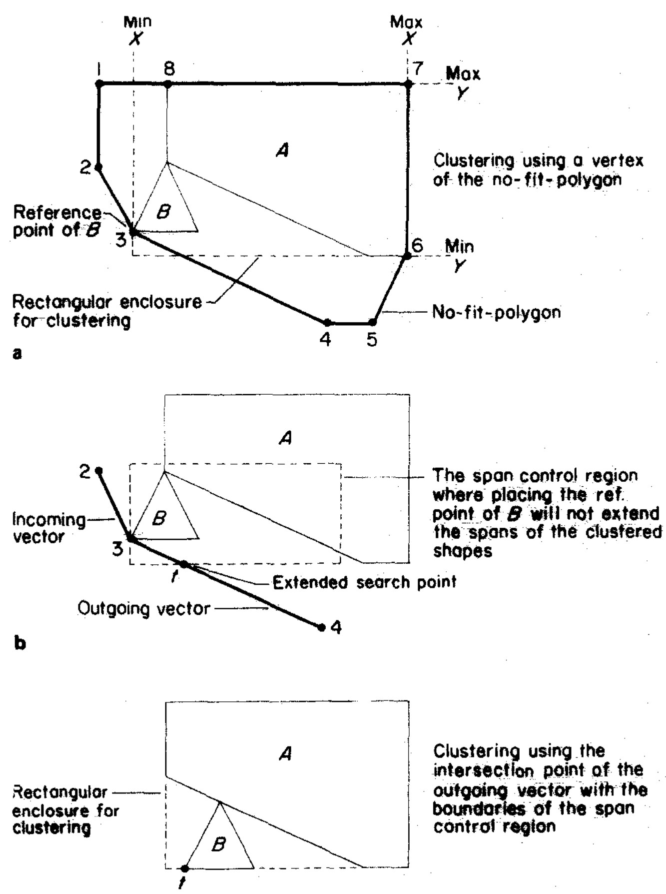

FIGURE 7. The extended search. (a) Clustering using a vertex of the no-fit-polygon. (b) The span control region where placing the reference point of B will not extend the spans of the clustered shapes. (c) Clustering using the intersection point of the outgoing vector with the boundaries of the span control region

of this observation by providing a special option (n° 3) which will perform pairwise clustering with two pieces of the same type of shape when the relative orientation is 180°. The algorithm evaluates the 'optimal' clustering using all the potentially good absolute orientations and applying the extended search at each vertex examined.

Multistage clustering

The procedure presented above can be used to produce clusters of more than two shapes by approaching the multiple shape clustering problem as a multistage pairwise clustering problem. One could first obtain the optimal clustering of two shapes and combine the two shapes to form one composite shape. Then one would obtain the optimal clustering of the composite shape with another shape and form the new composite shape. The procedure could be repeated as many times as one wishes. Since the shapes are clustered in stages and there is no provision for backtracking to reposition shapes, the procedure cannot guarantee that the 'optimum' clustering of the shapes will be attained.

The difficulties associated with using multistage clustering derive from the same sources as those associated with pairwise clustering. One must provide procedures for selecting the shapes to cluster, the orientations of the shapes, and the order in which the shapes are to be clustered. There will, in general, be little to be gained by using only one type of shape with only one relative orientation. Consequently, the techniques that were used to simplify the pairwise clustering problem will be of little value in the more general multistage clustering problem. New techniques are needed.

There are heuristic techniques that can be used to handle the problems of deciding what shapes to cluster, the orientations of the shapes, and the order in which the shapes are to be clustered. Unfortunately, there is another aspect of the multistage clustering problem that makes it even more difficult to obtain a practical algorithm. This aspect is concerned with developing a workable data-structure. In order for a multistage clustering algorithm to be practical for irregular shapes, one must be able to efficiently record and retrieve descriptions of arbitrarily complex polygons and clusterings of polygons. The efficiency must be with respect to the storage requirement and the computation time. In as much as the computations concerned with the other aspects of the algorithm will be rather complex and lengthy, an efficient data-structure becomes a major requirement.

The multistage clustering approach presented here is a simplified approach because it makes use only of rectangles which are the modules obtained from the shapes or various clustering of the shapes as it is described in the following. The user of the algorithm can specify four types of multistage clustering options, numbers 4, 5, 6 and 7. The number 4 option provides a clustering of up to four rectangles with the given shape; the number 6 option provides a clustering of up to four rectangles with pairwise clustering of two shapes of the same type with 180° relative orientation; and the number 5 and 7 options provide for a clustering of any number of rectangles respectively with the given shape and with pairwise clustering. Figure 2 gives examples of the types of results that one could obtain using these clustering options. Options 4, 5 and 6, 7 differ only in their starting points. Options 4 and 5 begin with the given shape positioned so that the optimum rectangular enclosure is attained. Options 6 and 7 begin with the 'minimum area encompassing polygon' in the optimum pairwise clustering. Figure 8a provides an example of a minimum area encompassing polygon. Option 7 differs from 6 in the evaluation of the minimum area encompassing polygon. When the ratio between the sum of the areas of the clustered pieces and the minimum area encompassing polygon is lower than a fixed threshold, the polygon shown in Figure 8b will be considered in the multistage clustering, in order to make the interior waste potentially usable.

Using all these options, the next step determines all the waste regions between the polygon (or the minimum area encompassing polygon) and the rectangular enclosure. Once these regions have been found, the four largest are placed in order of decreasing areas. Then, starting with the largest waste region, the algorithm seeks to fit one rectangle in each of the four regions.

The techniques used to fit a rectangle in a waste region are modifications of the pairwise clustering techniques. First, there are a set of tests to determine which rectangle to try. Then, one of the principal orientations of the selected rectangle is chosen and the NFP of the rectangle relative to the original shape (or the minimum area encompassing polygon if there are two shapes clustered together) is determined. The vertex examination and extended search procedures described previously in the pairwise clustering section are then followed with one added feature. The new feature is a test to see if the clustering position being considered has the reference point inside the waste region currently being filled. If it has, then the clustering is considered as before. If it has not, then the clustering position is rejected. In this way one can avoid computing the NFP for the second rectangle relative to the combination of the first rectangle and the original polygon, the NFP for the third rectangle relative to the combination of the first and second rectangles and the original polygon, and etc. With the added feature it is sufficient to compute the NFP of each rectangle relative to only the given polygon.

Figure 9 demonstrates the restricted pairwise clustering of a rectangle with a minimum area encompassing polygon. Since we are trying to fill waste region 1, only that part of the NFP that lies along lines *a*-2 or 2-*b* will be considered.

The final step in the pairwise clustering of a rectangle with a given polygon to fill a particular waste region involves determining if the entire rectangle clustered with the 'optimal' position fits entirely within the waste region. If it does,

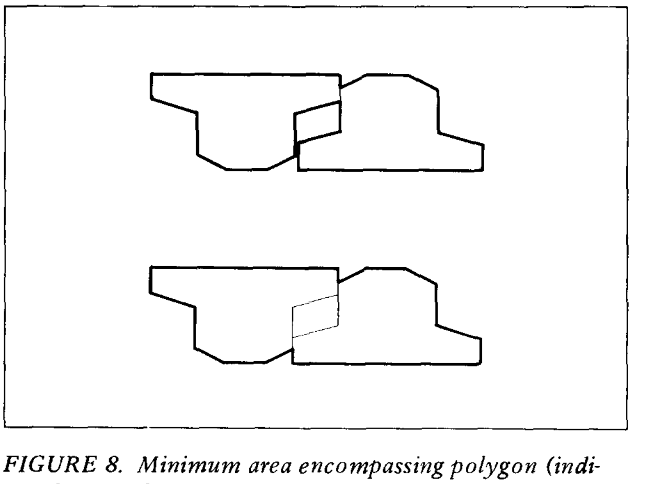

FIGURE 8. Minimum area encompassing polygon (indicated by thicker lines)

the clustering is accepted. If it does not, the clustering is rejected and another rectangle must be found to fill in the waste region.

The above procedure describes how rectangles are used to fill waste regions. However, we must still describe the procedure to be used to determine what rectangles and which orientations to try. The algorithm searches for the largest rectangle with

1. An area less than or equal to the area of the waste region
2. An order for a number of pieces greater than or equal to the number ordered of the given shape
3. A set of spans within those of the waste area.

It considers the clustering of the rectangle in only the two principal orientations. An example of this procedure appears in Figure 9. The rectangle satisfies all preliminary tests and is therefore considered. However, since the 'optimal' clustering of the rectangle does not lie completely within waste region 1, the clustering portrayed in the figure will be rejected.

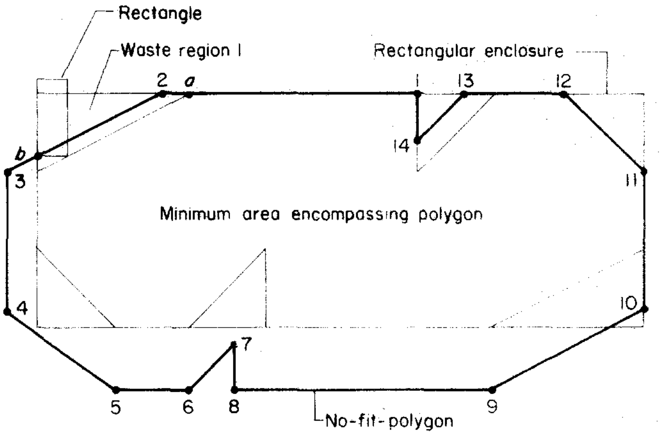

FIGURE 9. Restricted pairwise clustering

APPLICATIONS

An experimental version of the clustering algorithm has been implemented in FORTRAN on the IBM 360/67. The storage requirement is 45 k bytes. With a set of shapes appearing in a typical hand-made layout involving 25 rectangular types of shapes, 29 shapes that ranged from close to rectangular to highly irregular types, and a total of 983 pieces of all types, the computation time for the clustering stage was 70 s. The average usage coefficient of the produced modules, that is, the average ratio between the area of the clustered pieces and the minimum area enclosing rectangle, was 95%. We have made use of a CalComp incremental plotter in order to produce drawings.

By developing the clustering algorithm as a set of shape processing options we intended to make the algorithm more efficient and useful, but not more difficult to use. If the user has some knowledge of the shapes being entered, he can take advantage of the various options to control the nature of the processing performed on the shapes. But if he has no such knowledge, then he can simply specify the 5 or 7 options for all shapes not described as rectangles.

The suggested order of using the options is first 1, then 2, 3, 4, 5, 6 and 7. One or more of the options could, of course, be omitted. By following this sequence, rectangles resulting from the use of options 1, 2, and 3 could be made available for use when options 4, 5, 6 and 7 are invoked. Because of the way the algorithm is designed, options 3, 6 and 7 should be used only if the number of pieces of the shape is even. Options 1, 2, 4 and 5 can be used for both odd and even number of pieces. Figure 10 shows an example of layout that can be obtained when the rectangular modules produced by the clustering algorithm are allocated by the rectangular layout procedure described in Reference 11

SUMMARY AND CONCLUSIONS

We have described an algorithm for clustering two-dimensional shapes in rectangular modules. Our experience with the experimental version of the program has shown that computational time and total memory requirements are quite reasonable for realistic problems and that the algorithm is an effective tool for attacking large size template layout problems. The discussion has been oriented toward the production of rectangular modules because we had in mind large template layout problems such as the ones encountered in the shipbuilding industry. If this is the case, we think that a two-stage approach is the most promising one. In the first stage, the pieces are encased in minimum area rectangular modules either singly or in combination with other pieces. Then these modules are used in the second stage to produce optimal layouts on the rectangular sheets. However,

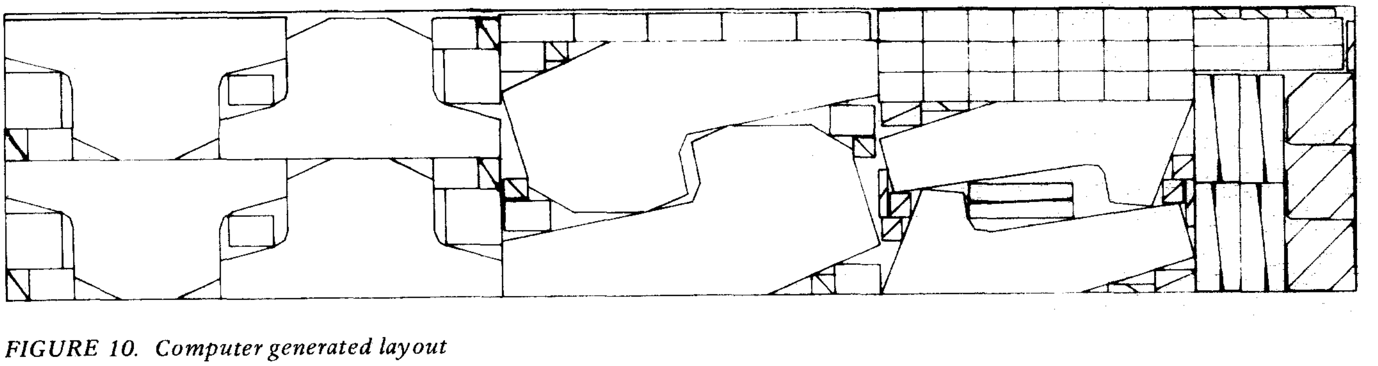

FIGURE 10. Computer generated layout

in those applications where it is reasonable, a procedure that seeks for an optimal layout by working directly with the irregular shapes, the clustering techniques presented here, can play a dominant role in the solution.

REFERENCES

1 Adamowicz, M. 'The optimum two-dimensional allocation of irregular, multiple-connected shapes with linear, logical and geometric constraints, Ph.D. Thesis, Dept. of Elect. Eng., New York University, (1969)

2 Haims, M. J., and H. Freeman, 'A multistage solution of the template layout problem', *IEEE Transactions on System Science and Cybernetics*, Vol.SSC-6, No 4 (1970)

3 Adamowicz, M., and A. Albano, 'A two-stage solution of the cutting stock problem'. *Information Processing 71* (Proc. IFIP Congress 71) Freiman, North-Holland Publishing Co., Amsterdam (1972), pp.1086–1091

4 Freeman, H. 'Computer processing of line drawing images', *Comput. Surveys*, Vol.6, No 1 (1974)

5 Gilmore, P. C., and R. E. Gomory. 'A linear programming approach to the cutting-stock problem', *Oper. Res.*, Vol.9, (1961)

6 Gilmore, P. C., and R. E. Gomory. 'A linear programming approach to the cutting-stock problem: part II', *Oper. Res.*, Vol.11, (1963)

7 Gilmore, P. C., and R. E. Gomory. 'Multistage cutting-stock problems of two and more dimensions', *Oper. Res.*, Vol.13, (1965)

8 Gilmore, P. C., and R. E. Gomory. 'The theory and computation of knapsack functions', *Oper. Res.*, Vol.14 (1966)

9 Christofides, N. 'Optimum Cutting of 2-D Rectangular Shapes', *CAD 74: International Conference on Computers in Engineering and Building Design*, Published by IPC Science & Technology Press Ltd, Guildford, Surrey, England (microfiche only) 1974

10 Adamowicz, M., and A. Albano. 'A solution of the rectangular layout problem', Consiglio Nazionale delle Ricerche, Istituto di Elaborazione della Informazione, Nota Interna B70/14, (June 1970)

11 Adamowicz, M., and A. Albano. 'A solution of the rectangular cutting stock problem', to be published in *IEEE Transactions on System Science and Cybernetics*