OPERATIONS RESEARCH
Vol. 58, No. 4, Part 1 of 2, July–August 2010, pp. 948–970
ISSN 0030-364X | EISSN 1526-5463 | 10 | 5804 | 0948

informs

DOI 10.1287/opre.1090.0770

Irregular Packing Using the Line and Arc No-Fit Polygon

E. K. Burke, R. S. R. Hellier, G. Kendall, G. Whitwell
School of Computer Science, University of Nottingham, Nottingham NG8 1BB, United Kingdom
{ekb@cs.nott.ac.uk, rxh@cs.nott.ac.uk, gxk@cs.nott.ac.uk, gxw@cs.nott.ac.uk}

The no-fit polygon is a geometric construct that can offer faster and more efficient handling of geometry between pairs of shapes than traditional line-by-line intersection. The detection of intersections is a critical operation within the irregular two-dimensional stock-cutting problem (also known as "nesting"), which aims to place shapes onto sheets of material so that the material is utilised as efficiently as possible and the waste (or trim loss) is reduced. The problem forms an important process within many real-world manufacturing industries such as metalworking, automotive production, aerospace, clothing and conservatory manufacture, and others. If manufacturers can reduce their costs by utilising raw materials more effectively, this can directly translate into increased profit margins or greater competitiveness within the marketplace. Moreover, there are significant environmental benefits to be gained. Several methods have been proposed to calculate no-fit polygons, but most, if not all, can only operate on geometry that consists of line segments. This paper extends the orbital sliding method of calculating no-fit polygons to enable it to handle arcs and then shows the resultant no-fit polygons being utilised successfully on the two-dimensional irregular packing problem. As far as the authors are aware, this is the first time that a no-fit polygon algorithm has been able to handle arcs robustly without decomposing to their line approximations. The modification of the authors' previously published packing algorithm to utilise the proposed no-fit polygon approach yields solutions of excellent quality (including several best-known) on well-established literature benchmark problems after only a few minutes. The authors believe that the success of the packing strategy and the line and arc no-fit polygon algorithm make this approach a serious candidate for use in real-world production environments.

Subject classifications: search production/scheduling; cutting-stock/trim; production/scheduling; approximations/heuristic; computers/computer science; artificial intelligence.
Area of review: Optimization.
History: Received February 2008; revision received October 2008; accepted April 2009. Published online in *Articles in Advance* April 9, 2010.

1. Introduction

The irregular two-dimensional variant of the cutting and packing problem impacts upon several important manufacturing industries such as textiles, plastics, metal cutting, and others. These problems usually consist of a number of irregular pieces that are to be placed onto one or more sheets of material in the most efficient layout possible, so that all pieces are assigned and do not overlap. Problems of this type are categorised as 2D irregular open-dimensional problems (Wäscher et al. 2007). Additionally, there are usually rotational constraints enforced on the pieces due to the physical properties of the problem, such as a grain on the material, patterns on textiles, and the cutting technology being employed. Sometimes rotational constraints may be used for nonphysical reasons such as to restrict pieces to a finite set of rotations, thus simplifying layout construction procedures and allowing faster solutions to be obtained. The two-dimensional stock-cutting problem has been shown to be NP-hard and is therefore intrinsically difficult to solve (Garey and Johnson 1979). There have been many different strategies for producing solutions to the irregular stock-cutting problem. These include linear programming approaches, heuristic placement methods, metaheuristic guided search techniques, and other novel approaches, such as the iterative jostling of pieces (Dowsland et al. 1998).

The no-fit polygon (NFP) is itself a polygon. It enables us to detect the relationship between two given polygons and to detect if two polygons overlap, just touch, or are completely separated. One of the many advantages in using the no-fit polygon is computational efficiency, and the detection of overlap is reduced to a simple point-inclusion test. The no-fit polygon has applications in a diverse range of domains. For example, it has uses in stock cutting, where items must be configured on stock sheets of material in order to minimise waste, and robot navigation, where the no-fit polygon is often referred to as the configuration space obstacle.

When the two polygons of interest are convex, calculating the NFP is trivial (Cunninghame-Green 1989, Cunninghame-Green and Davis 1992). However, calculating the NFP for nonconvex shapes provides significant complexities. Previous research has considered various approaches, including orbiting (Burke et al. 2007, Mahadevan 1984), Minkowski sums (Ghosh 1991, Bennell et al. 2001, Bennell and Song 2008), and Φ-functions (Bennell et al. 2008).

In Burke et al. (2007), a robust orbital method is described for deriving NFPs. The approach resolves known degenerate cases from previous approaches (for example, Mahadevan 1984, Kendall 2000), including coping with holes, dealing with interlocking concavities, and enabling jigsaw-type pieces to be dealt with. The algorithm only uses two simple, easily implemented geometric stages. The authors provide the generation times for 32 irregular packing benchmark problems from the literature, including real-world data sets.

Bennell and Song (2008) utilise a Minkowski-sum approach to calculate the NFP. This represents a significant improvement over the previous work from 2001 (Bennell et al. 2001). The procedure is capable of handling all previously known degenerate cases.

Bennell et al. (2008) investigate Φ-functions and their relationship with Minkowski sums and the NFP. They outline the advantages of Φ-functions over other approaches, providing a clear definition for the set of objects for which Φ-functions may be derived. A procedure for deriving Φ-functions, together with examples, is also presented.

Huyao et al. (2007) present a new approach for generating the NFP that is simple, intuitive, and computationally efficient. It is based on the novel concept of trace line segments that are derived from the interaction of two-component polygons. The complete set of the trace line segments contains all the boundary edges of the NFP and internal points that need to be discarded. Algorithms for deriving the trace line segments, efficiently determining those segments that form the boundary of the NFP and the identification of holes and degenerate cases, are described.

Bennell and Oliveira (2008) provide an excellent tutorial for many of the fundamental geometric methodologies that are required to implement NFP algorithms. It includes basic trigonometry procedures, including *D*-functions, that enable the positions of points and lines, in relation to one another, to be determined. The NFP is also discussed, together with the various ways in which it can be implemented.

2. Overview of the Orbital No-Fit Polygon Approach

In Burke et al. (2007), we presented an approach for generating no-fit polygons based on the orbital method proposed by Mahadevan (1984). The approach was able to produce no-fit polygons robustly for line-based shapes, and we demonstrated the algorithm by tackling well-known degenerate cases that had been identified as causing difficulties for previously published approaches. The approach was based on the principle that two shapes in any touching, nonintersecting position will have one or more touch points each involving an edge from shape A and an edge from shape B. Each of these touching edge pair structures can be used to calculate a possible vector of movement. However, these movement vectors were only guaranteed to be feasible for the particular touching edge pair from which they were derived, and not necessarily for the movement of shape B as a whole. Therefore, the movement vectors were tested for feasibility with each of the other touching edge pairs until one was found to be feasible for *all* of the touching edge pairs. The feasible move was then trimmed, if necessary, to prevent the shapes from moving into an intersecting position. This was achieved by casting the feasible move from each vertex on shape B and testing for intersection (and trimming) with shape A, and casting the move in reverse from shape A and testing for intersection (and trimming) with shape B. Finally, shape B can be translated by the trimmed feasible movement vector. The touching edge pairs are identified again and the procedure repeats until shape B returns to its original starting position. The concatenation of all of the movement vectors yielded the no-fit polygon NFP_{AB}.

In addition to this, a procedure was also identified for ensuring that narrow concavities and holes were not omitted from the NFP. Firstly, shape B was translated such that its maximal y vertex was aligned to the minimal y vertex of shape A. Once in this position, the outer loop of the NFP was created by utilising the sliding procedure outlined above. By flagging each edge that was visited, we identified if any edges had not been seen and which may contain valid touching and nonintersection starting positions for the two shapes. If such positions were found, the sliding procedure was again used to identify the NFP. These additional paths formed the internal loops to the outer loop of the NFP that was calculated in the first sliding procedure. Figure 1 shows some of the degenerate cases that were handled using this procedure. We refer the reader to Burke et al. (2007) for full implementation details and a computational comparison of the approach.

3. A Simplified Case Involving Circles

In order to describe the arc modifications needed for no-fit polygon generation, it is beneficial to examine a case involving two circles. These concepts are then generalised to arcs in the remainder of this section. The no-fit polygon of two circles, A and B (the convention of shape B moving around shape A is maintained), is defined as the path followed by the reference point of B (the circle's centre point) when circle B orbits circle A. The no-fit polygon, NFB_{AB}, of two circles is a circle of radius equal to the sum

Figure 1. No-fit polygons generated using the approach presented by Burke et al. (2007).

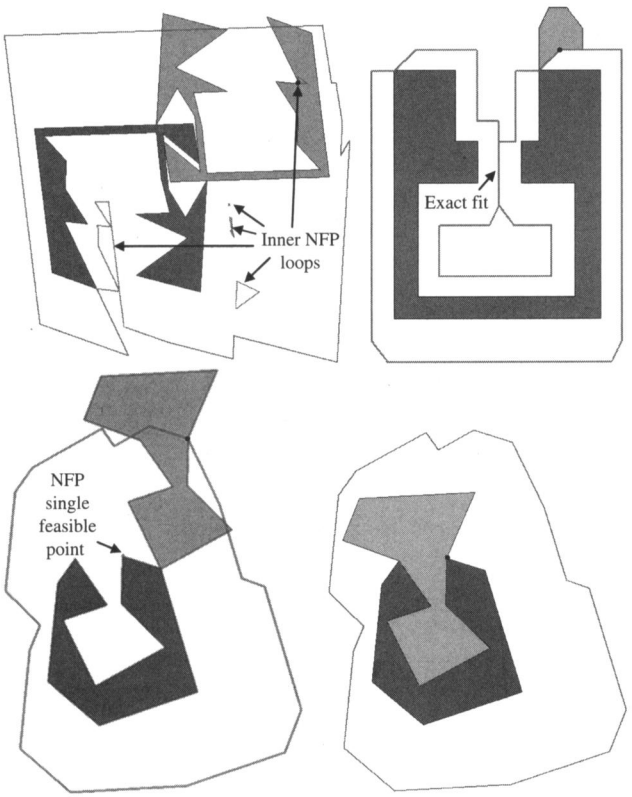
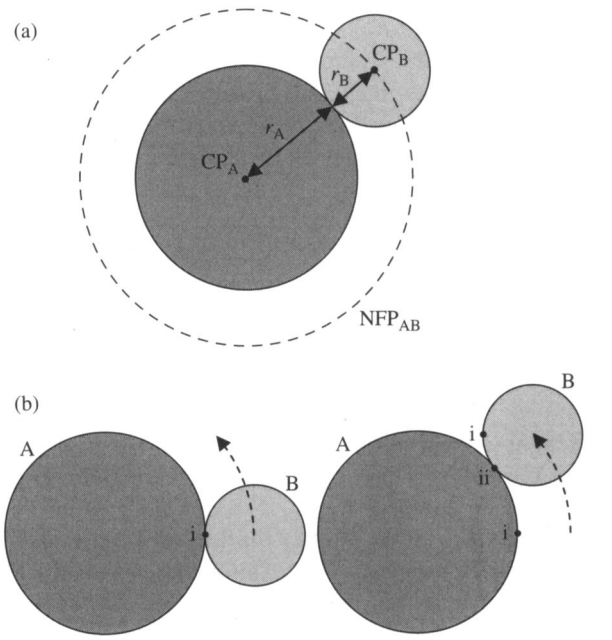


Figure 2. (a) No-fit polygon of two circles; (b) movement of the touch position.



of A and B's radii sharing the same centre point as circle A (see Figure 2(a)). We term the NFP of two circles the *circular no-fit polygon.*

The edges within line-only no-fit polygons are derived from the same edges as the two input shapes A and B. However, this is no longer the case with arcs because the no-fit polygon does not take the form of the shape edges from which it was derived, but rather it assumes properties from *both* interacting edges (namely the radii). This is because the touching position moves on *both* arcs as B moves around A (see Figure 2(b)).

4. Generalisation to Arcs

The circular no-fit polygon example presented in Figure 2 can be generalised to convex arcs by eliminating the segment of the no-fit polygon for which the arcs will not touch on their radial extents. This can be created by examining the start and end angles of the two arcs (see Figure 3(a)).

From Figure 3(b), the angular position of tangent point tp_A on arc B's parent circle is identified as the angle represented by the opposite angle, $\pi + \theta_{AStart}$. If this angle exists on arc B, then it is a valid tangent. Likewise, tangent point tp_B's angular position on parent circle A is given by $\pi + \theta_{BEnd}$. The partial no-fit polygon is then defined as the arc of radius $r_A + r_B$ through the angular range of tp_A through to tp_B. Figure 4 shows the start position whereby arc B touches arc A at tp_A and, after completing the partial no-fit polygon, the arcs touch at tp_B.

If there are no tangent points, then the two arcs cannot interact to create any of the circular no-fit polygon (see Figure 5).

A further example must be examined whereby one of the circles exists in its concave form (i.e., a hole) because this will allow for generalisations to be made for concave arcs. Figure 6 shows an example of the no-fit polygon produced when a circle travels around the inside edge of another circle. In Figure 6(a), the no-fit polygon is a circle once again, but this time it has a radius equal to $r_A - r_B$. Its centre

Figure 3. Finding circle tangents (A and B are convex).

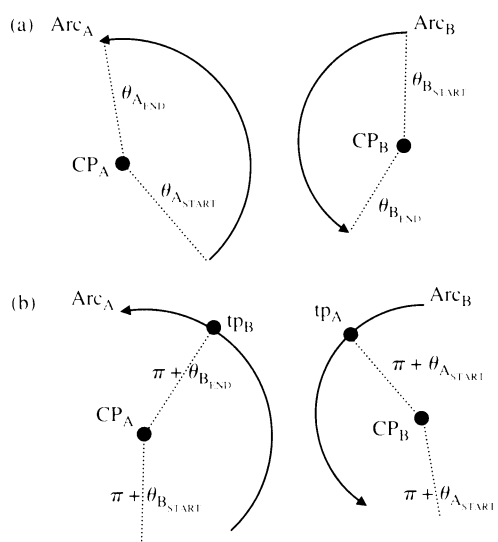
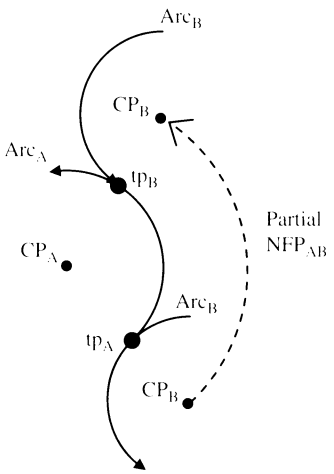


Figure 4. Creating the partial no-fit polygon circle from the arc tangents.



point is the same as A's centre point. To create the partial circular no-fit polygon in this instance, the same tangent point procedure can be repeated except that the start/end angles are not modified by π. The partial circular no-fit polygon is defined by the arc of radius $r_A - r_B$ and the start and end angles defined by the tangent point angles, tp_B and tp_A, respectively (see Figure 6(b)).

In the case where arc A is convex and B is concave, the partial circular no-fit polygon can be obtained by the same procedure as above. For this, arcs A and B must be switched so that arc B is now the convex arc and A is concave. Now, by rotating the resultant partial circular no-fit polygon by 180°, the correct arc section for the no-fit polygon is produced. This is possible because as B moves in one direction, A can also be thought of as moving relatively in the opposite direction. If one concave and one convex arc are involved, then a partial circular no-fit polygon is only possible if the concave arc has a larger radius than the convex arc.

It is not physically possible for two concave arcs to orbit each other via the circular no-fit polygon because the shapes with which they belong would have to be intersecting.

Figure 5. No arc tangent points available.

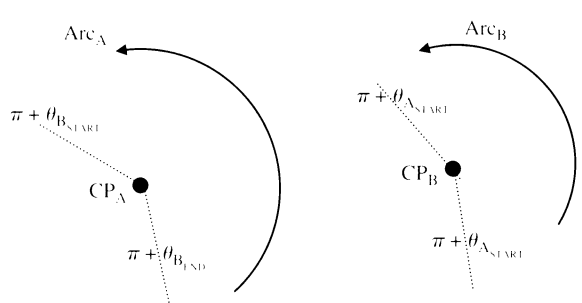
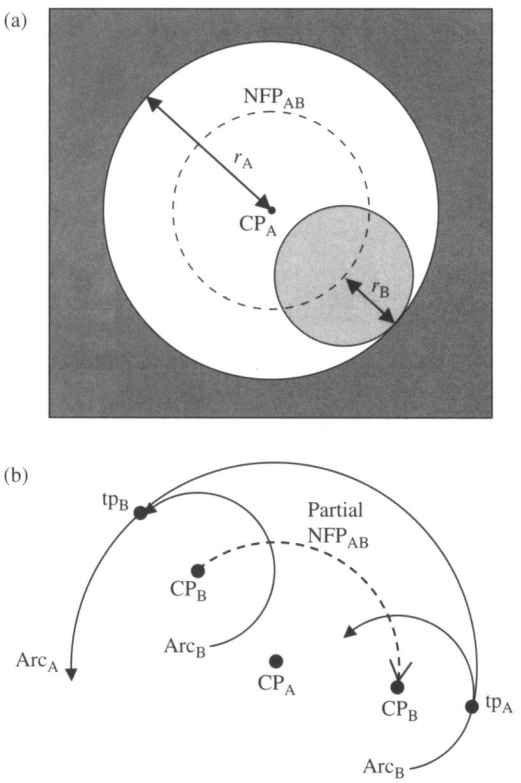


Figure 6. (a) No-fit polygon of two circles in the concave case; (b) creating the partial no-fit polygon where arc A is concave and arc B is convex.



5. Modifying the Orbital No-Fit Polygon Approach

Now that the concept of the circular no-fit polygon has been described together with generalisation to arcs, the modifications that are required to extend the orbital no-fit polygon generation algorithm are presented. The arc modifications are described using the algorithmic steps defined in Burke et al. (2007) and discussed in §2 of this paper. The steps are as follows:

Step 1. Detection of touching edges (see §5.1)

Step 2. Creation of potential translation vectors (see §5.2)

Step 3. Finding a feasible translation (see §5.3)

Step 4. Trimming the feasible translation (see §5.4)

Step 5. Applying the feasible translation (see §5.5)

In the following subsections, we give details of the modifications required for each step to facilitate the correct handling of arcs when deriving the no-fit polygon. Whilst steps 1 and 5 require no modification to handle arcs, they have been included for completeness and to provide the reader with a greater understanding of the overall orbital procedure.

Figure 7. Touching edges with two distinct touch positions (edge a_2 and b_3).

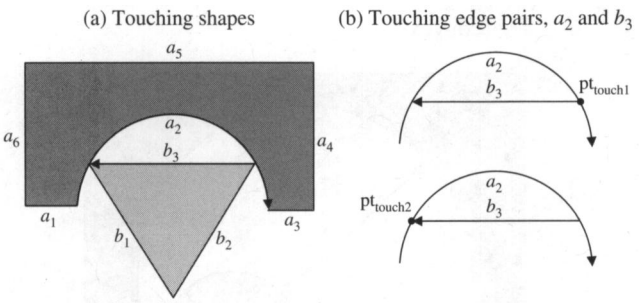
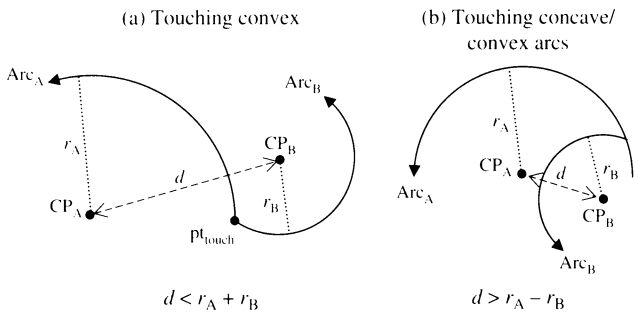


Figure 8. Nontangential touching arcs.



5.1. Detection of Touching Edges

In this step, we use standard trigonometry intersection routines to identify each pair of touching edges from shape A and B and their touch points. The detection of touching edges does not require modification of the line-based no-fit polygon algorithm except that the geometry library must be able to detect that lines are touching arcs or that arcs are touching other arcs. When concave arcs are involved, there is a possibility of two edges touching at two separate points. However, this complication can be eliminated by storing two separate touching edge pair structures for each of the touching points, as shown in Figure 7.

In Figure 7(a), both the start and end points of line b_3 touch the concave arc edge, a_2. Therefore, two touching entries are stored in the touching edge pair list to reflect this, and are shown by Figure 7(b). No further modifications are required for this stage of the algorithm.

5.2. Creation of Potential Translation Vectors

Step 2 involves the creation of potential translation vectors using the touching edge pairs that we identified in step 1. In this stage, the inclusion of arc edges requires a number of modifications based on the notion of the partial circular no-fit polygon (described in §3). In addition to this, we must also create potential translation vectors for touching edge pairs consisting of a line and an arc.

5.2.1. Potential Translation Vectors for Touching Pairs Involving Two Arcs. As already discussed, the circular no-fit polygon can easily be created when convex arcs are touching at their tangent points (see §4). This is when the distance between the two arc centres is equal to the sum of the two radii (the maximal distance where the arcs can touch). In the case of touching concave and convex arcs, the concave radius must be greater than the convex radius and the distance between the two arc centres must be equal to the convex radius subtracted from the concave radius. In these situations, the potential translation is simply the partial circular no-fit polygon. However, the arcs may also touch so that the distance between the centre points does not allow the partial circular no-fit polygon to be used. Examples of this situation are shown in Figure 8(a) with touching convex arcs and Figure 8(b) with touching concave and convex arcs.

In such a situation, two potential translations can be derived from, firstly, the stationary arc A, and then the orbiting arc B. There are three possible cases: (i) the two arcs both touch on start/end points, (ii) arc B's start/end point touches the middle of arc A, or (iii) arc A's start/end point touches the middle of arc B.

Cases (ii) and (iii) will be examined first, because these are the most simple within the line case. In §8.2.2, given two touching lines, A and B, the potential translation vector for case (ii) was the line defined by the touching point to the end point of line A (stationary line). The same principle is used for the arc case (ii), the potential translation vector is the arc defined by the touch point to the end point of arc A (stationary arc), which also has the same centre point and radius of arc A. Figure 9 shows an example of a potential translation vector for case (ii).

However, as shown in Figure 9(b), this potential translation could result in arc B intersecting with arc A. On further examination, this can be explained because a tangent point exists before the end point of arc A (see Figure 10(a)). Therefore, the potential translation should be trimmed to any tangent point that exists along the potential translation (Figure 10(b)). Assuming that the translation of Figure 10 is performed in full, the next potential translation can now be derived from the circular no-fit polygon of the two arcs (i.e., the distance between the arc centre points is equal

Figure 9. Potential translation vector derived in case (ii): arc B's start/end point touches the middle of arc A.

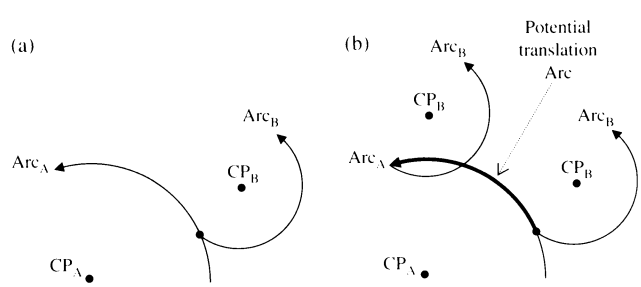
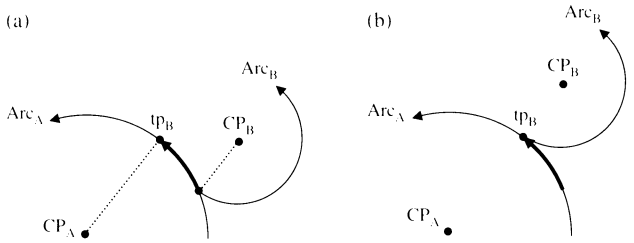


Figure 10. Potential translation vector derived in case (ii) when a tangent point is present: arc B's start/end point touches the middle of arc A.



to $r_A + r_B$ in the convex case and equal to $r_A - r_B$ in the concave case).

The same principles found in case (ii) also apply for case (iii) where the start/end point of arc A touches the middle of arc B (see Figure 11(a)). Once again, we must firstly examine whether the two arcs are tangential about their touch point. If this is the case, then the partial circular no-fit polygon can be derived. If not, then an arc is created from the touch point to the end point of arc B (with the same centre point and radius of B). However, this defines the potential translation vector with which arc A moves (relatively) so the translation vector must be rotated by 180° to obtain the correct translation vector for arc B (see Figure 11(b)).

From Figure 11(b), it can be seen that when arc B is translated by the potential vector, the two arcs remain in contact, without intersecting, throughout arc B's translation. Once again, this only holds true if there are no tangential positions along the translation. If one exists, the translation should be trimmed (as with the previous case). Although only convex cases have been presented within the figures, the concave cases follow the same procedure except for the calculation of the circular no-fit polygon. The final case is case (i), where both arcs touch on their start or end points. We must examine whether the two arcs are tangential using the centre point distance calculation. If they are tangential, then the partial circular no-fit polygon can be used as

Figure 11. Potential translation vector derived in case (iii): arc A's start/end point touches the middle of arc B.

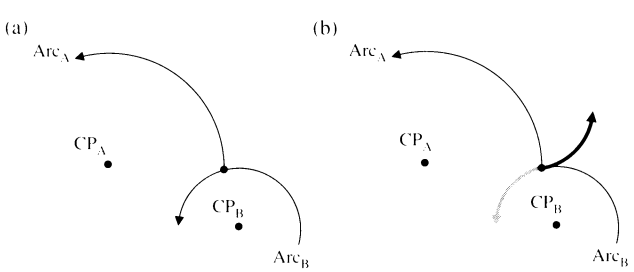
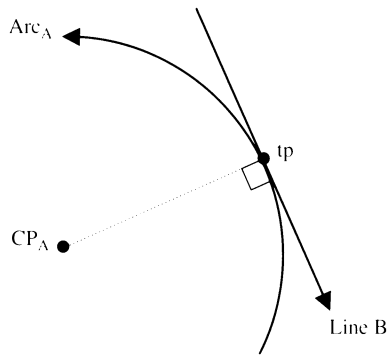


Figure 12. Tangents with lines and arcs.



before. If not, then potential translation vectors are derived from both of the arcs based on cases (ii) and (iii), described previously. If the arc's touching point is the end point, then no potential translation can be created (i.e., a zero-length arc translation would be created because the end point is also the touch point). Also, if both arcs touch on their start points, then two potential translation vectors will be produced (one from arc$_A$ and one from arc$_B$).

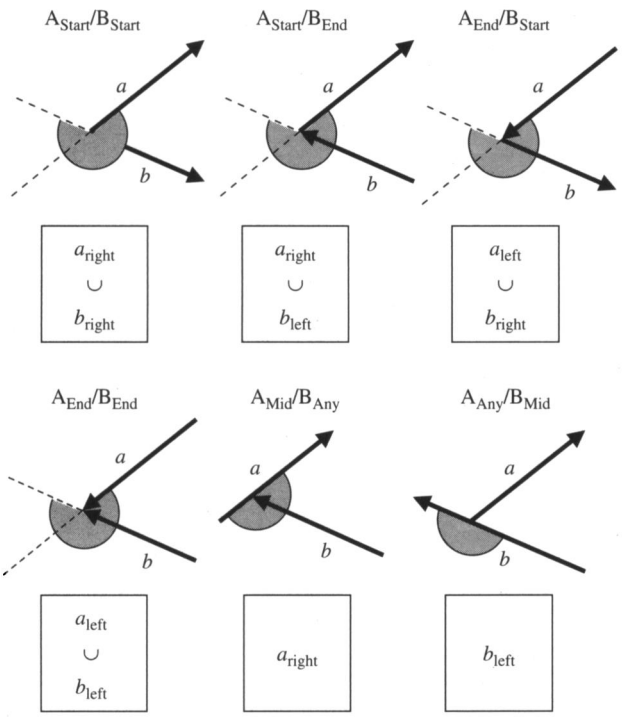
5.2.2. Potential Translation Vectors for Touching Pairs Involving a Line and an Arc. Touching lines and arcs are less complicated than the touching arcs case, but tangential points must still be identified (see Figure 12).

The same three cases will be briefly summarised for touching lines and arcs (because the principles are the same as the above). A potential translation arc is created from the touch point to the end point of arc A in case (ii) where no tangent point exists, and from the touch point to the tangent point where one does exist. It is important to detect whether there is a tangent point to avoid translations that result in intersections. In case (iii), the potential translation vector is defined by the end point of line B to the touch point (tangent points are not required here). Finally, case (i) is handled by the techniques from cases (ii) and (iii). Once again, two potential translation vectors are produced (one derived from the arc and one from the line). Once again, no potential translation can be derived from a primitive where its end point is also the touching point.

5.3. Finding a Feasible Translation

The next stage is to find a feasible translation from the set of potential translations. A potential translation is only feasible if it is feasible for each of the touching edge pairs. In Burke et al. (2007), a set of rules was developed to indicate translation feasibility through the use of left/right line tests. Examples are reproduced in Figure 13 because they will also be used when handling arcs. The touching point can either be at the start, end, or in the middle of an edge (the touching point for each edge is identified above each diagram in Figure 13).

Figure 13. Identifying the feasible angular range of translations (indicated by the arc) for different touch categories.



Now that arcs are involved, it initially seems that the problem is more complicated. However, the line tests shown in Figure 13 can be used providing that any arcs within the touching edge pairs and potential translations are reduced to a tangential line at the touch point. Figure 14 shows how the tangential line is created depending on the position of the touching point on the arc: (i) at a touching start point, the tangential line must start at the touch point, (ii) a midarc touch results in a tangential line with its midpoint on the touch point, and (iii) a touching end point requires that the tangential line also ends at the arc's end point.

Figure 14. Conversion to tangential lines at the touch point of an arc.

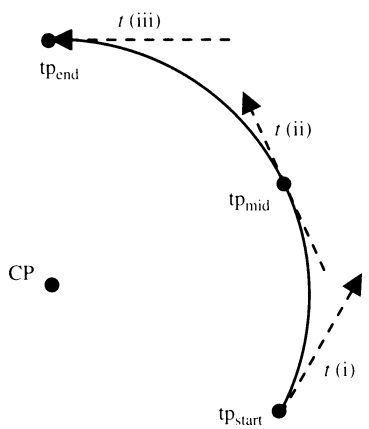
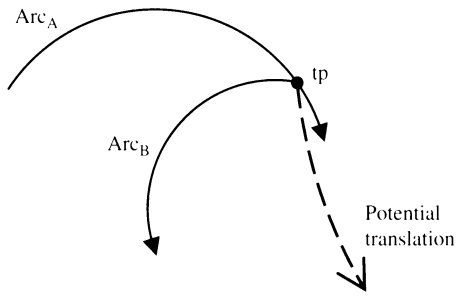


Figure 15. Two touching arcs and a potential arc translation.



It is important that the directionality of the arc is maintained when creating the tangential line for the method of §8.2.3 to be applicable (the length of the tangential line does not matter because it is only used for left/right tests). Figures 15, 16, and 17 demonstrate the procedure.

Figure 15 shows an example of two touching arcs and a potential arc translation (arc_A is concave, arc_B is convex and the touch point is tp). In Figure 16, tangential lines are created for the two touching arcs and the potential translation arc. In Figure 17(a), the feasibility test from Burke et al. (2007) for touching line edges is used to define the feasible region, and Figure 17(b) shows that the potential translation arc is feasible for this pair of touching edges.

On further examination of why the tangential lines can be used, we must examine what is being calculated. In order to show that a potential translation is feasible for a particular edge pair, it only needs to be shown that the translation will not *immediately* result in an intersection. The same

Figure 16. Creating the tangential lines.

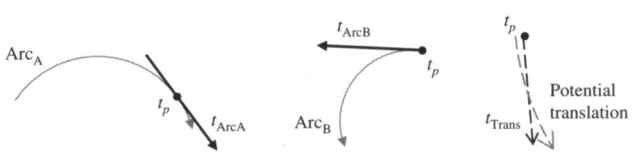
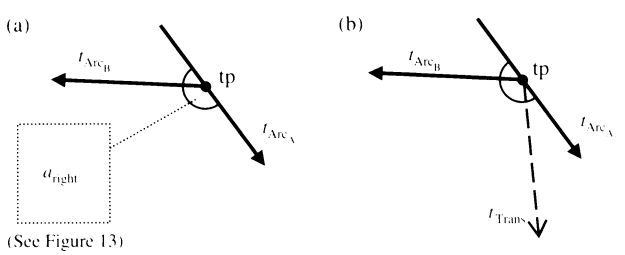


Figure 17. Revisiting the line method using tangential lines.



routines can be used from the line-only case because the tangential lines of arc primitives define the vector path of the arc at a particular position. The feasibility of any potential translation and combination of touching edge pairs can be identified provided that the tangential lines of the arcs are used.

5.4. Trimming the Feasible Translation

The last step before polygon B can be translated is to trim the translation vector. This is important because there may be other edges that could interfere with the translation of the orbiting polygon. In the line-only algorithm of Burke et al. (2007), this was achieved by projecting the translation vector through each vertex of shape B and testing for its intersection with lines of shape A. The translation vector was also projected back from the vertices of shape A to test for intersection with shape B's primitives. In order to correctly handle arcs, the trimming procedure requires only a simple modification in order to deal with: (i) arc-to-arc trims, (ii) line-to-arc trims, and (iii) arc-to-line trims. For each of these, the same approach is taken as with the line case (projecting the translation through each start/end vertex etc.). However, an extra projection must also be conducted from the tangent point of any arcs. Given an arc and a line, the tangent point of the line and arc must be detected. This tangent point is then treated as if it were another vertex of the arc (i.e., the translation is also projected from the point). The line/arc trim procedure is demonstrated in Figure 18.

Similarly, when trimming is performed on two arcs, we not only need to project from each start/end point, but also from the centre point of either of the arcs to test for any intersection with the partial circular no-fit polygon of the two arcs. If there is an intersection, then the translation is trimmed as normal. The process for an arc/arc trim is demonstrated in Figure 19.

In the arc/arc case, if no partial circular no-fit polygon exists for two arcs, then the start/end point projections will result in the correct trim.

5.5. Applying the Feasible Translation

In the final step of the algorithm, the trimmed translation is added to the NFP edge list, and shape B is translated to its new position. However, shape B may need to be translated along the path of an arc because we now have the arc edges, but there is no difference between translating shape B along a line or arc edge, because we can just translate to the end point of the edge. It is from this new position that we repeat the procedure of finding the touching edges (step 1) until the shape returns to its original location (indicating that a no-fit polygon loop has been created).

5.6. Summary

In this section, we identified the modifications required to allow the orbital approach of Burke et al. (2007) to be able to generate no-fit polygons for shapes that also contain arc edges. Figure 20 shows examples of no-fit polygons that have been generated using the proposed line and arc no-fit polygon algorithm. Included in Figure 20 are annotations that explain what procedures were used to create a particular edge on the no-fit polygon.

In the remainder of this paper, the line and arc no-fit polygon algorithm will be applied to the two-dimensional irregular packing problem.

Figure 18. Tangential trimming of the feasible translation from an arc and a line, a_4 and b_4.

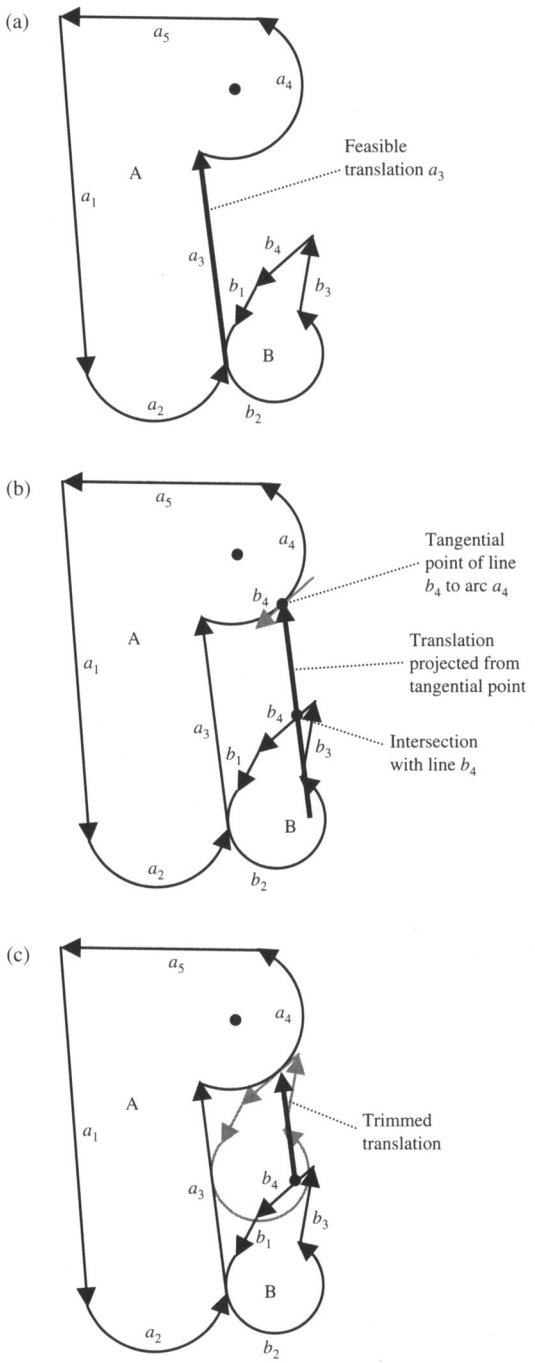
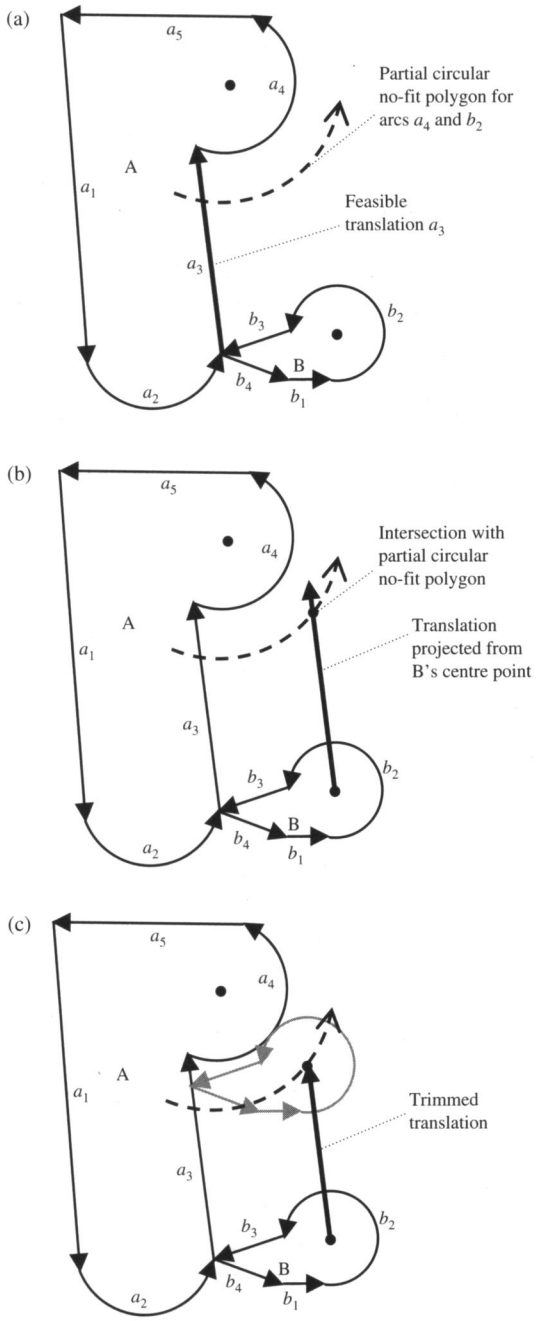


Figure 19. Tangential trimming of the feasible translation from two arcs.



6. An Irregular Packing Approach Using the Line and Arc No-Fit Polygon

In our previous irregular packing approach, presented in *Operations Research* (Burke et al. 2006), we proposed a new bottom-left-fill algorithm that had produced several best-known solutions on 26 well-established literature benchmark problems. In addition to this, 10 new benchmark problems consisting of shapes with arcs and holes were added because they had not been represented within the literature. The algorithm was modified for the experiments presented in this section, and the steps are summarised below.

(1) Place the first shape in the bottom-left corner of the sheet (i.e., a nonintersecting position), and add it to the sheet.

(2) Put the next shape in the bottom-left corner of the sheet.

(3) Test the current shape for intersection with any already-placed shapes. If shape intersects, go to (4). If the shape is not intersecting, go to (6).

(4) Resolve the intersection between the two intersecting shapes by translating the current shape vertically until it no longer intersects with the placed shape.

(5) Go to (3).

(6) If the current shape has moved off the top of the sheet, then go to (7), else go to (8).

(7) Translate to the bottom of the sheet and translate right by some x increment. Go to (3).

(8) A valid placement position is found, so add the shape to the sheet.

(9) If there are more shapes to place, go to (2).

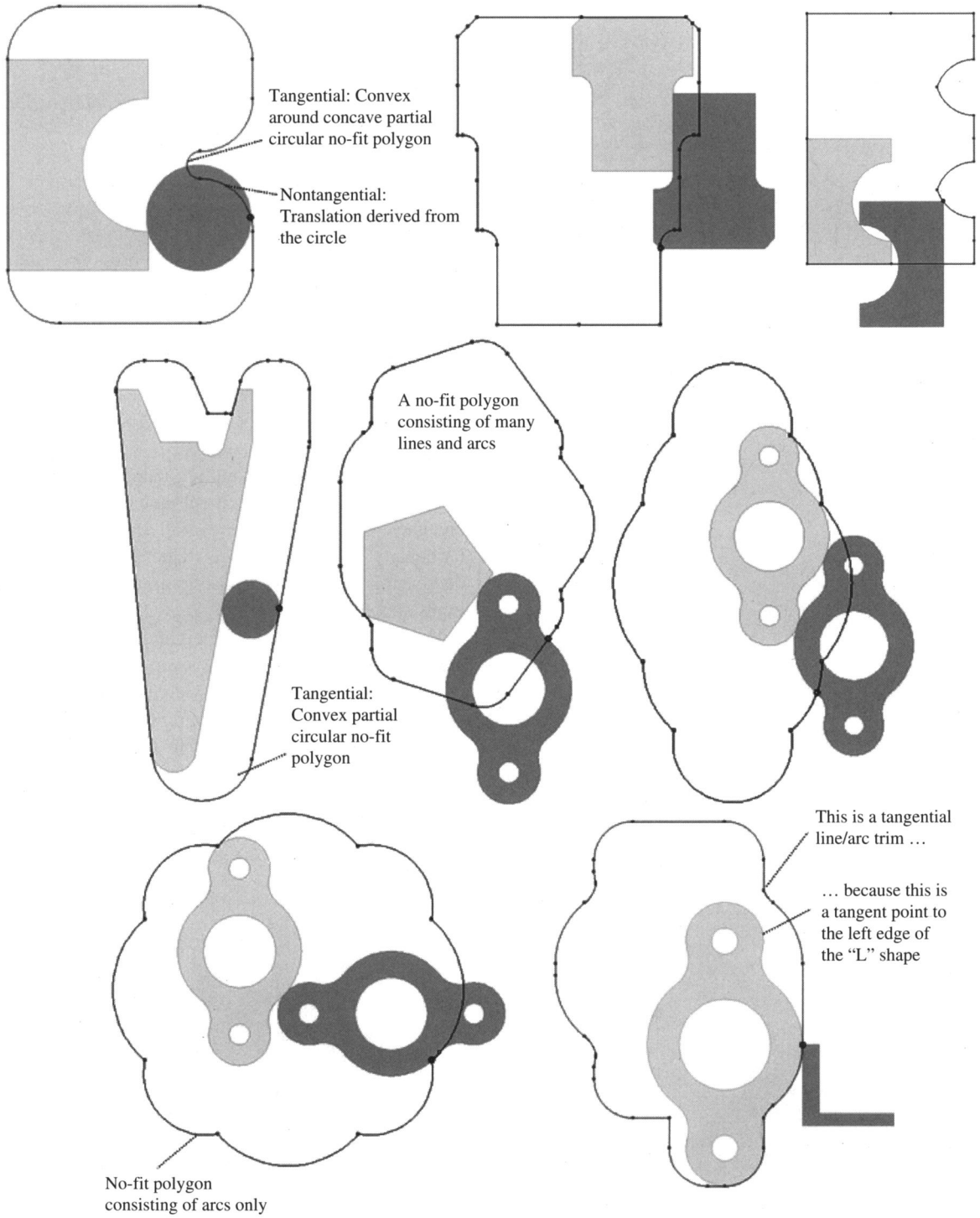
(10) Otherwise, return layout.

The main principle of the approach was that feasible placement positions could be obtained by resolving intersecting shapes vertically using standard trigonometry-based intersection. Figure 21 shows how two intersecting shapes were resolved by progressively resolving the intersections between their respective edges. Figure 21(a) shows the initial positions of the two shapes (we are trying to find a feasible placement location for shape B, shape A is already assigned to the sheet).

In Figure 21(a), there are two pairs of intersecting edges—two arc edges and two line edges. In previous work (Burke et al. (2006)), it was identified that when there is a choice, it is computationally more efficient to resolve the line intersections. Figure 21(b) shows the resulting position of shape B after resolving the intersection of the two lines. The two shapes are still overlapping, but there are no edge intersections, so a vertical line is cast from the bottommost point of shape B and tested for intersection with shape A. Shape B is translated vertically by the distance between the nearest intersection point and the bottommost point. Figure 21(c) shows that this results in a new pair of edges intersecting (a line from shape A and an arc from shape B). Figure 21(d) shows the final arc-arc intersection and Figure 20(e) gives the final nonintersecting configuration for these two shapes. For specific implementation details and pseudo-code for this procedure, we refer the reader to Burke et al. (2006).

In contrast to the standard trigonometry-based intersection detection, no-fit polygons can be used to quickly identify the intersection state of two shapes by a simple point-in-polygon test. If the reference point of shape B is inside the no-fit polygon NFP_{AB}, then the two shapes are known to be intersecting. In Burke et al. (2006), standard

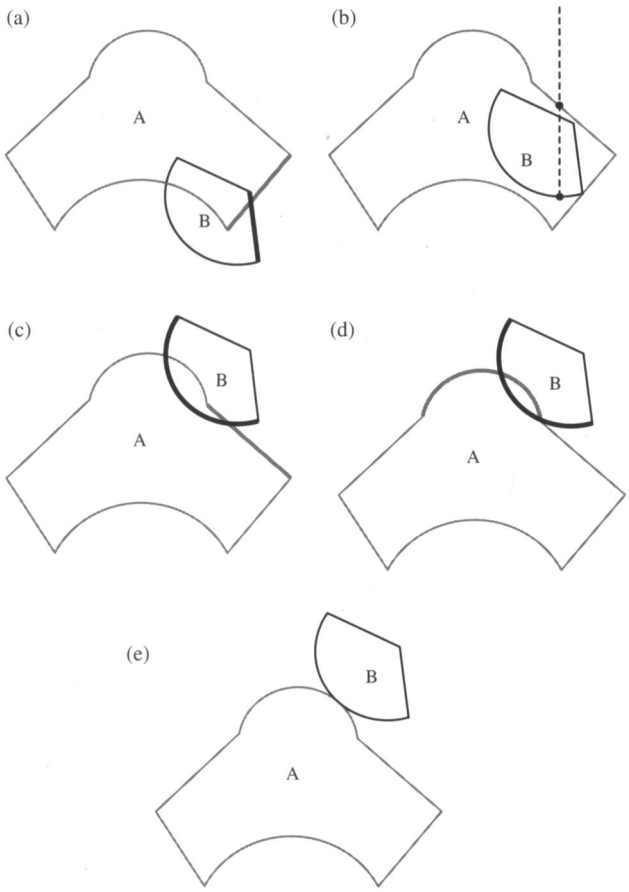
Figure 20. Examples of no-fit polygons generated by the proposed approach.



Note. Orbiting shape shown in dark grey.

trigonometry intersection detection was used because, at that time, there were no approaches that could generate no-fit polygons robustly for all cases and be capable of handling shapes consisting of circular arcs without decomposing to their line approximations. Robustness and accuracy were both critical features for the algorithm to be applicable for use in industry. With the modified no-fit polygon generation approach proposed in this paper, we no longer have concerns with the use of the no-fit polygon. We now modify the intersection detection and vertical resolution approach to utilise no-fit polygons. Figure 22 shows that intersections can be resolved by casting a vertical line from

Figure 21. Resolving intersecting shapes vertically (Burke et al. 2006).



the reference point of shape B (i.e., the locus of the no-fit polygon) and intersecting with the NFP_{AB}. The intersection between the two shapes is guaranteed to be resolved in one step by translating shape B using the following:

$$\textit{Vertical translation} = \textit{Nearest vertical intersection point} - \textit{Ref Point B}.$$

7. Experiments on Literature Benchmark Problems

In this section, we evaluate the proposed approach by generating new solutions for the previously published literature benchmark problems (see Tables 1 and 2) using the new no-fit polygon bottom-left-fill implementation and hill-climbing (HC) and tabu (TABU) local search procedures where a solution is represented by a specific sequence/ordering of the shapes. To provide a numerical evaluation of the shape sequence, we place each shape in the order given by the shape sequence using the no-fit polygon bottom-left-fill placement heuristic discussed in §6, and then we calculate the overall length of the layout and its density. The initial shape sequences are provided by the decreasing length or decreasing area sorted ordering of the input shapes.

During the local search, neighbouring solutions are created by applying one of five swap operators to the current shape input sequence. These were defined as 1, 2, 3, 4, and N opt operators where 1 opt randomly selects one shape in the sequence and moves it to another random location, 2 opt randomly chooses two shapes and swaps their positions, 3 opt randomly chooses three shapes (not necessarily

Figure 22. Resolving intersecting shapes with the arc and line no-fit polygon.

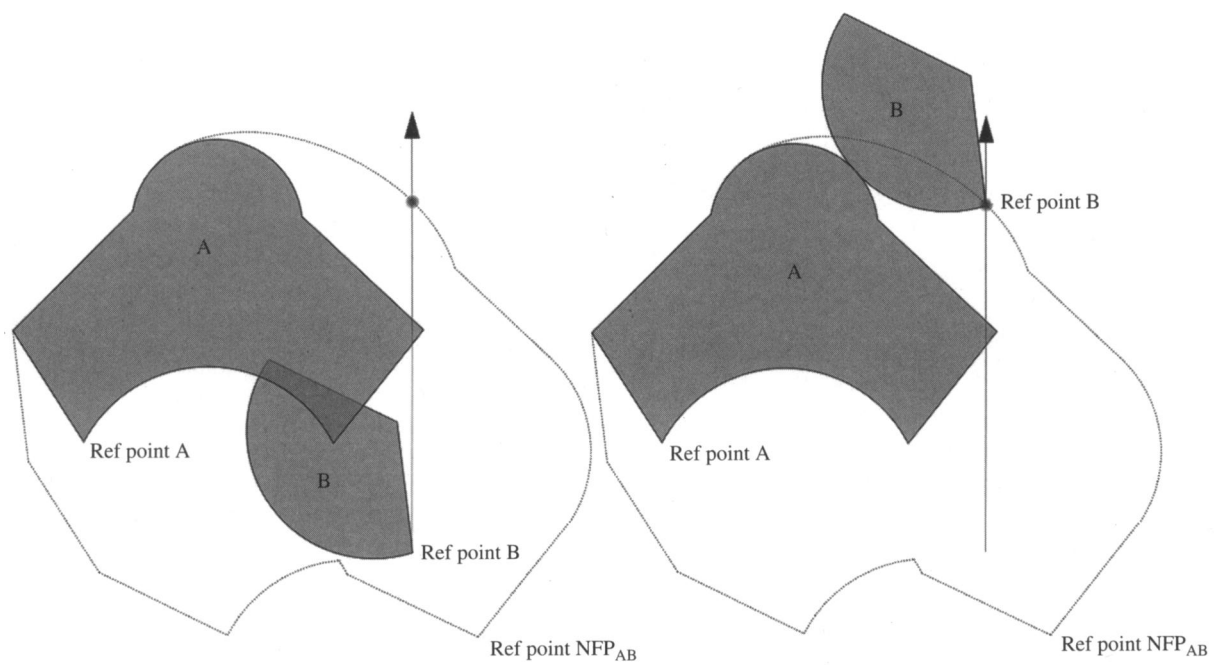


Table 1. Length evaluated irregular benchmark problems from the literature.

Original author	Problem name	Shapes	Rotational constraints	Sheet width	Best length	Time to best (s)	Best result reference
Blazewicz et al. (1993)	Blasz1	28	0, 180 absolute	15	26.57	5,603	Bennell and Song (2008) [called SHAPES2]
Ratanapan and Dagli (1997)	Dagli	30	90 incremental	60	57.64	17,331	Bennell and Song (2008)
Fujita et al. (1993)	Fu	12	90 incremental	38	31.57	1,192	Bennell and Song (2008)
Jakobs (1996)	Jakobs1	25	90 incremental	40	11.4	2,193	Bennell and Song (2008)
Jakobs (1996)	Jakobs2	25	90 incremental	70	24.97	4,540	Gomes and Oliveira (2006)
Marques et al. (1991)	Marques	24	90 incremental	104	77.79	10,692	Bennell and Song (2008)
Hopper (2000)	Poly1A	15	90 incremental	40	14.00	12	Burke et al. (2006)
Hopper (2000)	Poly2A	30	90 incremental	40	28.17	121	Burke et al. (2006)
Hopper (2000)	Poly3A	45	90 incremental	40	40.33	1,515	Burke et al. (2006)
Hopper (2000)	Poly4A	60	90 incremental	40	54.93	203	Burke et al. (2006)
Hopper (2000)	Poly5A	75	90 incremental	40	69.37	476	Burke et al. (2006)
Hopper (2000)	Poly2B	30	90 incremental	40	30.00	180	Burke et al. (2006)
Hopper (2000)	Poly3B	45	90 incremental	40	40.74	418	Burke et al. (2006)
Hopper (2000)	Poly4B	60	90 incremental	40	51.73	96	Burke et al. (2006)
Hopper (2000)	Poly5B	75	90 incremental	40	57.53	52,514	Bennell and Song (2008)
Hopper (2000)	SHAPES	43	90 incremental	40	59	31	Burke et al. (2006)
Oliveira and Ferreira (1993)	SHAPES0	43	0 absolute	40	60	3,914	Gomes and Oliveira (2006)
Oliveira and Ferreira (1993)	SHAPES1	43	0, 180 absolute	40	55	398	Bennell and Song (2008)
Oliveira and Ferreira (1993)	SHIRTS	99	0, 180 absolute	40	61.33	7,033	Bennell and Song (2008)
Oliveira et al. (2000)	SWIM	48	0, 180 absolute	5,752	5,895.17	15,721	Bennell and Song (2008)
Oliveira et al. (2000)	TROUSERS	64	0, 180 absolute	79	241	5,988	Bennell and Song (2008)
Burke et al. (2006)	Profiles1	32	90 incremental	2,000	1,377.74	189	Burke et al. (2006)
Burke et al. (2006)	Profiles2	50	90 incremental	2,500	3,216.10	1,264	Burke et al. (2006)
Burke et al. (2006)	Profiles3	46	45 incremental	2,500	8,193.89	1,759	Burke et al. (2006)
Burke et al. (2006)	Profiles4	54	90 incremental	500	2,453.12	1,131	Burke et al. (2006)
Burke et al. (2006)	Profiles5	50	15 incremental	4,000	3,332.70	1,317	Burke et al. (2006)
Burke et al. (2006)	Profiles6	69	90 incremental	5,000	3,097.86	813	Burke et al. (2006)
Burke et al. (2006)	Profiles7	9	90 incremental	500	1,296.30	680	Burke et al. (2006)
Burke et al. (2006)	Profiles8	18	90 incremental	1,000	1,318.70	354	Burke et al. (2006)
Burke et al. (2006)	Profiles9	57	90 incremental	1,500	1,290.67	1,215	Burke et al. (2006)
Burke et al. (2006)	Profiles10	91	0 absolute	3,000	11,160	111	Burke et al. (2006)

adjacent in the sequence) and randomly reassigns them back to the positions (i.e., a shuffle), 4 opt and N opt perform the same as 3 opt except with 4 and N shapes respectively. Each operator has a different chance of selection, from 1 Opt, which has the largest chance of being selected, down to N Opt, which has a much lower chance of being selected. This is because the less radical operators allow for concentration of the search, and the more radical operators, e.g., N Opt, enable the search to escape from local optima. See Burke et al. (2006) for additional discussion of the search procedure and for a presentation of the pseudo-code. All of the experiments have been conducted on a 2 GHz Intel Pentium 4 processor with 256 MB RAM, identical equipment to that used for the experiments in Burke et al. (2006).

In Tables 3 and 4, we report the best and average layout lengths obtained by the proposed approach for the length and density evaluated literature benchmark problems. We

Table 2. Density evaluated irregular benchmark problems from the literature.

Original author	Problem name	Shapes	Rotational constraints	Sheet width	Best density (%)	Time to best (s)	Best result reference
Albano and Sapuppo (1980)	Albano	24	90 incremental	4,900	87.88	5,460	Bennell and Song (2008)
Blazewicz et al. (1993)	Blasz2	20	90 incremental	15	83.61	2,257	Gomes and Oliveira (2006)
Dighe and Jakiela (1996)	Dighe1	16	90 incremental	100	100	1.4	Gomes and Oliveira (2006), Bennell and Song (2008)
Dighe and Jakiela (1996)	Dighe2	10	90 incremental	100	100	0.3	Gomes and Oliveira (2006), Bennell and Song (2008)
Bounsaythip and Maouche (1997)	Mao	20	90 incremental	2,550	81.01	667	Gomes and Oliveira (2006)

Table 3. Experiments on length evaluated literature benchmark problems.

	Hill climbing								Tabu search							
	Length ordered				Area ordered				Length ordered				Area ordered			
		Best result				Best result				Best result				Best result		
Problem	Average length	Length	Density1 (%)	Density2 (%)	Average length	Length	Density1 (%)	Density2 (%)	Average length	Length	Density1 (%)	Density2 (%)	Average length	Length	Density1 (%)	Density2 (%)
Blasz1	27.54	**26.80**	**80.6**	**82.8**	27.73	27.20	79.4	81.2	27.33	27.10	79.7	81.0	27.51	27.00	80.0	81.3
Dagli	60.81	**59.94**	**84.6**	**85.8**	60.87	60.36	84.0	85.1	61.03	60.24	84.2	84.8	61.15	60.38	84.0	84.8
Fu	32.98	32.89	86.7	89.1	32.33	**31.57**	**90.2**	**92.1**	32.94	32.70	87.2	87.7	32.92	**31.57**	**90.2**	**92.1**
Jakobs1	11.90	**11.50**	**85.2**	88.5	12.00	12.00	81.7	86.0	11.97	11.86	82.6	88.9	11.79	**11.50**	**85.2**	**90.3**
Jakobs2	26.00	26.00	74.2	77.5	26.00	26.00	74.2	78.4	26.00	26.00	74.2	80.8	25.41	**24.70**	**78.1**	**82.5**
Marques	78.80	**78.00**	**88.7**	89.6	79.59	79.00	87.6	89.1	79.71	78.93	87.6	**89.9**	79.60	79.00	87.6	88.0
Poly1A	13.67	**13.30**	**77.1**	**82.8**	13.81	13.70	74.8	78.5	13.78	13.69	74.8	79.3	13.97	13.69	74.9	77.1
Poly2A	27.53	**27.09**	**75.7**	77.7	27.79	27.58	74.3	75.2	27.19	**27.09**	**75.7**	**78.8**	27.55	27.13	75.6	77.3
Poly3A	41.35	**41.07**	**74.9**	76.5	41.75	41.53	74.0	75.7	41.55	41.25	74.5	**77.7**	41.77	41.67	73.8	75.4
Poly4A	55.78	55.14	74.4	76.0	55.73	55.53	73.8	75.7	55.75	**54.60**	**75.1**	**76.7**	55.73	55.10	74.4	75.7
Poly5A	69.98	69.84	73.4	74.6	70.20	69.56	73.7	74.2	70.06	69.13	74.1	75.4	69.79	**68.84**	**74.4**	**75.5**
Poly2B	30.37	30.11	75.1	77.4	30.44	**29.63**	**76.3**	**77.5**	30.54	30.40	74.4	76.6	30.52	30.28	74.7	76.1
Poly3B	41.00	40.66	75.0	**76.5**	40.97	40.63	75.1	76.0	41.20	41.06	74.3	75.8	41.02	**40.50**	**75.3**	76.4
Poly4B	52.66	52.33	73.9	74.7	52.32	51.84	74.6	75.3	52.10	51.72	74.8	75.7	51.67	**51.18**	**75.6**	**76.9**
Poly5B	61.68	61.14	75.1	76.1	61.62	61.31	74.9	**76.4**	61.61	**60.86**	**75.4**	75.9	61.81	61.71	74.4	75.8
SHAPES	57.40	**56.00**	**71.2**	**73.7**	57.80	57.00	70.0	71.1	57.90	57.50	69.4	70.7	58.20	**56.00**	**71.2**	72.5
SHAPES0	62.00	**60.00**	**66.5**	**70.2**	62.40	62.00	64.4	66.5	62.30	61.00	65.4	67.3	64.22	62.50	63.8	65.5
SHAPES1	56.20	**55.00**	**72.5**	**74.3**	57.00	57.00	70.0	71.6	58.46	58.00	68.8	71.1	58.20	58.00	68.8	70.4
SHIRTS	63.71	**63.40**	**85.2**	**86.9**	64.10	63.98	84.4	86.3	64.04	63.93	84.5	86.5	64.09	63.85	84.6	86.2
SWIM	6,464.73	6,311.28	70.1	71.3	6,531.24	6,305.94	70.1	**71.5**	6,416.59	**6,270.88**	**70.5**	71.4	6,566.58	6,499.91	68.1	70.0
TROUSERS	248.56	245.95	88.7	89.5	247.75	246.80	88.4	89.4	246.60	**245.28**	**88.9**	**89.8**	247.98	247.17	88.2	89.1
Profiles1	1,389.14	1,380.10	81.6	83.0	1,391.40	1,386.00	81.3	82.5	1,412.88	1,404.80	80.2	81.4	1,394.54	**1,359.90**	**82.8**	**84.5**
Profiles2	3,262.10	3,216.06	50.0	50.8	3,230.98	**3,194.19**	**50.3**	**51.5**	3,264.40	3,252.70	49.4	50.3	3,267.78	3,223.30	49.8	50.8
Profiles3	8,073.88	**7,881.13**	**52.9**	**54.0**	8,230.33	8,189.66	50.9	51.7	8,074.34	7,999.74	52.1	53.0	8,177.60	8,045.84	51.8	53.0
Profiles4	2,476.46	2,452.42	75.2	75.7	2,475.58	2,464.35	74.8	75.4	2,466.55	**2,425.26**	**76.0**	**76.5**	2,486.49	2,482.72	74.2	74.8
Profiles5	3,394.32	3,367.88	69.4	70.6	3,401.17	**3,351.94**	**69.8**	**70.8**	3,385.32	3,364.35	69.5	70.5	3,399.48	3,384.90	69.1	70.4
Profiles6	3,134.01	**3,121.36**	**75.0**	**78.6**	3,172.97	3,156.02	74.2	75.6	3,161.51	3,146.56	74.4	76.8	3,176.52	3,161.22	74.1	76.3
Profiles7	1,305.78	**1,292.30**	**77.4**	**79.9**	1,307.44	**1,292.30**	**77.4**	**79.9**	1,316.32	1,296.30	77.1	77.7	1,313.44	1,309.10	76.4	79.6
Profiles8	1,303.17	1,268.98	78.8	79.8	1,283.19	1,268.98	78.8	79.8	1,308.61	1,293.88	77.3	77.5	1,285.44	**1,263.11**	**79.1**	**82.1**
Profiles9	1,314.73	**1,278.21**	**52.7**	**54.2**	1,298.42	1,290.00	52.2	53.2	1,308.93	1,298.62	51.9	52.9	1,303.43	1,292.63	52.1	53.3
Profiles10	11,373.40	**11,219.60**	**65.8**	**66.6**	11,515.03	11,403.99	64.8	65.4	11,392.36	11,302.50	65.3	66.0	11,653.40	11,585.81	63.7	64.7

Table 4. Experiments on density evaluated literature benchmark problems.

	Hill climbing							
	Length ordered				Area ordered			
		Best result				Best result		
Problem	Average length	Length	Density1 (%)	Density2 (%)	Average length	Length	Density1 (%)	Density2 (%)
Albano	10,203.84	**9,980.86**	**87.2**	**88.3**	10,196.87	10,169.47	85.6	86.4
Blasz2	24.94	24.80	75.9	79.9	24.92	24.90	75.6	79.9
Dighe1	1,257.96	1,239.60	80.7	81.1	1,260.38	1,250.00	80.0	81.9
Dighe2	1,224.66	1,215.70	82.3	83.0	1,205.66	**1,180.00**	**84.7**	**86.5**
Mao	1,857.70	1,847.20	79.8	83.1	1,841.52	1,821.70	80.9	**83.9**
	Tabu search							
	Length ordered				Area ordered			
		Best result				Best result		
Problem	Average length	Length	Density1 (%)	Density2 (%)	Average length	Length	Density1 (%)	Density2 (%)
Albano	10,130.41	10,010.73	87.0	87.4	10,174.15	10,037.38	86.7	87.5
Blasz2	24.96	**24.80**	**75.9**	**80.4**	25.06	24.90	75.6	79.0
Dighe1	1,289.62	1,270.00	78.7	80.1	1,257.72	**1,210.00**	**82.6**	**83.8**
Dighe2	1,221.12	1,190.00	81.1	84.7	1,229.32	1,226.60	81.5	83.6
Mao	1,853.90	1,821.20	80.9	83.1	1,838.78	**1,811.50**	**81.4**	83.6

also show two different density measures: The first is a simple straight-line density (Density1), whereas the second density measure, used by Hopper (2000), is based on the union of all individual shape-bounding rectangles. This allows us to use a nonrectangular final density measurement (Density2). For each data set, we perform 10 runs using each of the four different combinations of initial sorting and local search procedure: (i) HC + Area, (ii) HC + Length, (iii) TABU + Area, and (iv) TABU + Length. For clarity, we present additional statistical analysis giving minimum, maximum, average layout length, and standard deviation for each of the four combinations of sort and local search procedure separately in Table 5. From Tables 3, 4, and 5, it is clear that no sort/search combination dominates across the entire set of benchmarks. This may indicate that the combination of sort and search is somewhat dependent on the input data. For example, the SWIM data set performs better when using a length-sorted initial ordering for *both* hill-climbing and tabu local searches.

Table 6 provides a summary of the results and a comparison to both best-known solutions and the solutions obtained in Burke et al. (2006). We identify the best result in bold type, and we also underline results where the proposed approach achieved less than 1% worse solutions, but did so in a faster time. It is useful to reiterate that we are using a packing approach identical to that of Burke et al. (2006), except that we implement intersection detection using the no-fit polygon instead of standard trigonometry.

Comparing the "Time/Nest" columns in Table 6, it can be seen that the use of the no-fit polygon has considerably reduced the amount of time required to produce layouts compared to our previous trigonometric approach, presented in Burke et al. (2006). This difference is perhaps most evident in the problems "SWIM" and "Profiles9," which both involve shapes consisting of numerous small edges. For "SWIM," the no-fit polygon implementation is approximately 10 times quicker, and for "Profiles9," layouts can be produced about 22 times faster. With some other problems, such as "Fu" and the "Poly" problems, there are only small improvements in layout generation time. On closer examination, these data sets contain shapes with relatively few edges (4, 5, or 6), and therefore the benefit of resolving intersections in one step via the NFP does not provide as great an advantage over the edge-by-edge intersection resolution using trigonometry as presented in Burke et al. (2006).

In comparison to the best solutions from the literature on the 36 benchmark problems, the proposed approach obtains 16 new outright best solutions, 3 equal best solutions in a significantly faster time, and 7 solutions that are within 1% of the best literature solution, but found in a significantly faster time. These three groups are highlighted in Table 7.

Table 5. Min, max, average length, and standard deviation statistics for the proposed method on the literature benchmark problems.

	Hill climbing								Tabu search							
	Length ordered				Area ordered				Length ordered				Area ordered			
Problem	Min length	Max length	Average length	Std Dev	Min length	Max length	Average length	Std dev	Min length	Max length	Average length	Std dev	Min length	Max length	Average length	Std dev
Albano	9,980.86	10,380.57	10,203.84	143.52	10,169.47	10,276.34	10,196.87	44.98	10,010.73	10,197.14	10,130.41	74.87	10,037.38	10,258.82	10,174.15	92.46
Blasz1	26.80	27.90	27.54	0.44	27.20	27.90	27.73	0.30	27.10	27.50	27.33	0.18	27.00	27.80	27.51	0.30
Blasz2	24.80	25.00	24.94	0.09	24.90	25.00	24.92	0.04	24.80	25.10	24.96	0.11	24.90	25.20	25.06	0.13
Dagli	59.94	61.16	60.81	0.49	60.36	61.27	60.87	0.36	60.24	61.49	61.03	0.55	60.38	61.73	61.15	0.54
Dighe1	1,239.60	1,280.00	1,257.96	16.06	1,250.00	1,270.00	1,260.38	9.11	1,270.00	1,300.00	1,289.62	11.88	1,210.00	1,280.80	1,257.72	29.26
Dighe2	1,215.70	1,226.90	1,224.66	5.01	1,180.00	1,226.90	1,205.66	19.73	1,190.00	1,230.00	1,221.12	17.45	1,226.60	1,230.00	1,229.32	1.52
Fu	32.89	33.00	32.98	0.05	31.57	33.00	32.33	0.70	32.70	33.00	32.94	0.13	31.57	33.80	32.92	0.81
Jakobs1	11.50	12.00	11.90	0.22	12.00	12.00	12.00	0.00	11.86	12.00	11.97	0.06	11.50	12.00	11.79	0.23
Jakobs2	26.00	26.00	26.00	0.00	26.00	26.00	26.00	0.00	26.00	26.00	26.00	0.00	24.70	26.00	25.41	0.63
Mao	1,847.20	1,865.80	1,857.70	8.14	1,821.70	1,869.00	1,841.52	19.23	1,821.20	1,875.00	1,853.90	20.51	1,811.50	1,860.80	1,838.78	18.99
Marques	78.00	79.00	78.80	0.45	79.00	80.00	79.59	0.54	78.93	80.00	79.71	0.47	79.00	80.00	79.60	0.55
Poly1A	13.30	13.83	13.67	0.21	13.70	13.86	13.81	0.07	13.69	13.83	13.78	0.05	13.69	14.10	13.97	0.17
Poly2A	27.09	27.77	27.53	0.26	27.58	27.98	27.79	0.15	27.09	27.32	27.19	0.08	27.13	27.82	27.55	0.29
Poly3A	41.07	41.68	41.35	0.22	41.53	41.96	41.75	0.19	41.25	41.81	41.55	0.26	41.67	41.94	41.77	0.13
Poly4A	55.14	56.24	55.78	0.50	55.53	55.90	55.73	0.18	54.60	56.86	55.75	0.80	55.10	56.22	55.73	0.42
Poly5A	69.84	70.11	69.98	0.11	69.56	70.53	70.20	0.40	69.13	70.57	70.06	0.63	68.84	70.36	69.79	0.69
Poly2B	30.11	30.78	30.37	0.28	29.63	30.87	30.44	0.46	30.40	30.69	30.54	0.11	30.28	30.76	30.52	0.21
Poly3B	40.66	41.26	41.00	0.24	40.63	41.26	40.97	0.25	41.06	41.30	41.20	0.09	40.50	41.31	41.02	0.32
Poly4B	52.33	52.88	52.66	0.24	51.84	52.56	52.32	0.29	51.72	52.42	52.10	0.28	51.18	51.98	51.67	0.33
Poly5B	61.14	62.14	61.68	0.42	61.31	62.01	61.62	0.29	60.86	61.98	61.61	0.45	61.71	61.98	61.81	0.12
SHAPES	56.00	59.00	57.40	1.14	57.00	58.00	57.80	0.45	57.50	58.00	57.90	0.22	56.00	59.00	58.20	1.30
SHAPES0	60.00	63.00	62.00	1.22	62.00	63.00	62.40	0.55	61.00	64.00	62.30	1.14	62.50	65.00	64.22	1.04
SHAPES1	55.00	57.00	56.20	0.76	57.00	57.00	57.00	0.00	58.00	59.00	58.46	0.51	58.00	59.00	58.20	0.45
SHIRTS	63.40	64.09	63.71	0.30	63.98	64.20	64.10	0.09	63.93	64.11	64.04	0.08	63.85	64.41	64.09	0.26
SWIM	6,311.28	6,590.38	6,464.73	116.16	6,305.94	6,608.86	6,531.24	127.43	6,270.88	6,535.02	6,416.59	114.10	6,499.91	6,614.67	6,566.58	45.30
TROUSERS	245.95	249.48	248.56	1.47	246.80	248.30	247.75	0.59	245.28	247.56	246.60	0.93	247.17	248.82	247.98	0.68
Profiles1	1,380.10	1,392.10	1,389.14	5.12	1,386.00	1,397.50	1,391.40	4.14	1,404.80	1,417.40	1,412.88	5.61	1,359.90	1,414.20	1,394.54	22.72
Profiles2	3,216.06	3,302.21	3,262.10	41.59	3,194.19	3,258.04	3,230.98	29.64	3,252.70	3,280.60	3,264.40	14.48	3,223.30	3,301.74	3,267.78	29.87
Profiles3	7,881.13	8,155.11	8,073.88	111.70	8,189.66	8,258.72	8,230.33	25.61	7,999.74	8,133.98	8,074.34	57.91	8,045.84	8,227.48	8,177.60	78.39
Profiles4	2,452.42	2,486.76	2,476.46	14.04	2,464.35	2,482.38	2,475.58	7.48	2,425.26	2,488.26	2,466.55	23.54	2,482.72	2,489.30	2,486.49	2.43
Profiles5	3,367.88	3,415.66	3,394.32	19.81	3,351.94	3,459.24	3,401.17	38.43	3,364.35	3,398.02	3,385.32	14.67	3,384.90	3,413.20	3,399.48	11.45
Profiles6	3,121.36	3,149.66	3,134.01	12.79	3,156.02	3,183.51	3,172.97	12.41	3,146.56	3,171.17	3,161.51	12.04	3,161.22	3,184.02	3,176.52	9.29
Profiles7	1,292.30	1,317.60	1,305.78	12.79	1,292.30	1,317.60	1,307.44	9.23	1,296.30	1,325.40	1,316.32	13.23	1,309.10	1,322.30	1,313.44	6.17
Profiles8	1,268.98	1,321.78	1,303.17	20.09	1,268.98	1,293.60	1,283.19	11.24	1,293.88	1,322.10	1,308.61	10.03	1,263.11	1,297.11	1,285.44	14.09
Profiles9	1,278.21	1,341.88	1,314.73	23.55	1,290.00	1,305.60	1,298.42	7.18	1,298.62	1,324.45	1,308.93	11.56	1,292.63	1,316.22	1,303.43	8.97
Profiles10	11,219.60	11,484.88	11,373.40	96.37	11,403.99	11,601.51	11,515.03	94.89	11,302.50	11,477.91	11,392.36	65.73	11,585.81	11,699.05	11,653.40	54.82

Table 6. A comparison of the best results of the proposed no-fit polygon packing approach to that of Burke et al. (2006) and the best results from the literature.

			Burke et al. (2006)					Proposed no-fit polygon packing best					% improvement over	
Problem	Best literature	Best literature time (s)	Length	Density1 (%)	Density2 (%)	Time/ nest (s)	Time to best (s)	Length	Density1 (%)	Density2 (%)	Time/ nest (s)	Time to best (s)	Best literature (%)	Burke et al. (2006) (%)
Blaszl	**26.57**	5,603	27.80	77.7	81.0	0.32	21	26.80	80.6	82.8	0.09	**281**	−0.87	**3.60**
Dagli	**57.64**	17,331	60.57	83.7	89.0	2.04	189	59.94	84.6	85.8	0.42	252	−3.99	**1.05**
Fu	31.57	1,192	32.80	86.9	90.8	0.24	21	**31.57**	90.2	92.1	0.17	**139**	**0.00**	**3.75**
Jakobs1	**11.40**	2,193	11.86	82.6	92.6	0.74	43	11.50	85.2	90.3	0.19	**29**	−0.88	**3.02**
Jakobs2	**24.97**	4,540	25.80	74.8	83.3	2.13	81	**24.70**	78.1	82.5	0.64	**51**	**1.08**	**4.26**
Marques	**77.97**	10,692	80.00	86.5	89.3	0.25	5	78.00	88.7	89.6	0.07	**21**	−0.04	**2.50**
Poly1A	14.00	12	14.00	73.2	78.2	0.36	12	**13.30**	77.1	82.8	0.11	**254**	**5.03**	**5.03**
Poly2A	28.17	121	28.17	72.8	77.5	1.24	121	**27.09**	75.7	78.8	0.50	**239**	**3.84**	**3.84**
Poly3A	**40.33**	1,515	41.65	73.8	75.5	2.01	210	41.07	74.9	76.5	1.00	159	−1.83	**1.40**
Poly4A	54.93	**203**	54.93	74.6	75.9	2.43	203	**54.60**	75.1	76.7	1.71	**224**	**0.60**	**0.59**
Poly5A	69.37	476	69.37	73.9	75.7	5.04	476	**68.84**	74.4	75.5	1.76	**300**	**0.76**	**0.76**
Poly2B	30.00	180	30.00	75.4	77.5	2.50	180	**29.63**	76.3	77.5	0.90	**189**	**1.23**	**1.24**
Poly3B	40.74	418	40.74	74.9	77.1	4.26	418	**40.50**	75.3	76.4	2.22	**114**	**0.58**	**0.58**
Poly4B	51.73	96	51.73	74.8	77.4	8.24	96	**51.18**	75.6	76.9	6.00	**176**	**1.07**	**1.07**
Poly5B	**57.53**	52,514	60.54	75.8	77.2	14.70	677	60.86	75.4	75.9	12.62	299	−5.78	−0.52
SHAPES	59.00	31	59.00	67.6	69.1	0.60	31	**56.00**	71.2	73.7	0.27	**226**	**5.08**	**5.09**
SHAPES0	**60.00**	3,914	66.00	60.5	62.6	0.93	21	**60.00**	66.5	70.2	0.14	**274**	**0.00**	**9.09**
SHAPES1	**55.00**	398	60.00	66.5	68.9	0.82	2	**55.00**	72.5	74.3	0.22	**239**	**0.00**	**8.33**
SHIRTS	**61.33**	7,033	63.80	84.6	87.3	4.99	58	63.40	85.2	86.9	0.77	194	−3.38	**0.62**
SWIM	**5,895.17**	15,721	6,462.40	68.4	71.6	12.39	607	6,270.88	70.5	71.4	1.24	141	−6.37	**2.96**
TROUSERS	**241.00**	5,988	246.57	88.5	90.1	7.89	756	245.28	88.9	89.8	1.02	253	−1.77	**0.52**
Albano	**87.9%**	5,460	10,292.90	84.6	86.5	1.18	93	9,980.86	87.23	88.25	0.25	**299**	−0.75	**3.03**
Blasz2	**83.6%**	2,257	25.28	74.5	79.9	0.16	11	24.80	75.94	80.41	0.07	14	−10.10	**1.88**
Dighe1	**100.0%**	1.2	1,292.30	77.4	78.9	0.22	9	1,210.00	82.65	83.84	0.15	3	−21.00	**6.37**
Dighe2	**100.0%**	0.3	1,260.00	79.4	84.3	0.10	7	1,180.00	84.75	86.50	0.70	148	−18.00	**6.35**
Mao	**81.0%**	667	1,854.30	79.5	82.9	0.38	30	1,821.70	80.91	83.86	0.13	**152**	−0.12	**1.76**
Profiles1	1,377.74	189	1,377.74	82.0	85.2	0.83	189	**1,359.90**	82.8	85.4	0.10	**15**	**1.29**	**1.29**
Profiles2	3,216.1	1,264	3,216.10	50.0	51.3	31.15	264	**3,194.19**	50.3	51.5	2.88	**295**	**0.68**	**0.68**
Profiles3	8,193.89	1,759	8,193.89	50.9	52.6	7.68	1,759	**7,881.13**	52.9	54.0	0.80	**283**	**3.82**	**3.82**
Profiles4	2,453.12	1,131	2,453.12	75.1	75.7	1.04	1,131	**2,425.26**	76.0	76.5	0.16	**256**	**1.14**	**1.14**
Profiles5	**3,332.7**	1,317	3,332.70	70.2	73.6	65.92	1,317	3,351.94	69.8	70.8	7.39	**300**	−0.58	−0.58
Profiles6	**3,097.86**	813	3,097.86	75.6	77.8	2.44	813	3,121.36	75.0	78.6	0.67	**171**	−0.76	−0.76
Profiles7	1,296.3	680	1,296.30	77.1	80.2	0.06	680	**1,292.30**	77.4	79.9	0.02	**211**	**0.31**	**0.31**
Profiles8	1,318.7	354	1,318.70	75.8	77.2	0.55	354	**1,263.11**	79.1	82.1	0.08	**279**	**4.22**	**4.22**
Profiles9	1,290.67	1,215	1,290.67	53.1	54.9	8.8	1,215	**1,278.21**	52.7	54.2	0.39	**98**	**0.97**	**0.97**
Profiles10	**11,160**	**111**	11,160.10	66.2	66.8	1.89	111	11,219.60	65.8	66.6	0.42	247	−0.53	−0.53

Table 7. Summary of problems for which the proposed approach yields better solution quality and/or time than the best known solutions from the literature.

		Best literature		Proposed approach	
Group	Problem	Evaluation	Time taken (s)	Evaluation	Time taken (s)
(1) Better evaluation	Jakobs2	24.97	4,540	**24.70**	**51**
	Poly1A	14.00	**12**	**13.30**	254
	Poly2A	28.17	**121**	**27.09**	239
	Poly4A	54.93	**203**	**54.60**	224
	Poly5A	69.37	476	**68.84**	**300**
	Poly2B	30.00	**180**	**29.63**	189
	Poly3B	40.74	418	**40.50**	**114**
	Poly4B	51.73	**96**	**51.18**	176

Note. Table continues next page.

Table 7. (Continued)

Group	Problem	Best literature		Proposed approach	
		Evaluation	Time taken (s)	Evaluation	Time taken (s)
	SHAPES	59.00	**31**	**56.00**	226
	Profiles1	1,377.74	189	**1,359.90**	**15**
	Profiles2	3,216.1	1,264	**3,194.19**	**295**
	Profiles3	8,193.89	1,759	**7,881.13**	**283**
	Profiles4	2,453.12	1,131	**2,425.26**	**256**
	Profiles7	1,296.3	680	**1,292.30**	**211**
	Profiles8	1,318.7	354	**1,263.11**	**279**
	Profiles9	1,290.67	1,215	**1,278.21**	**98**
(2) Equal evaluation and better time	Fu	**31.57**	1,192	**31.57**	**139**
	SHAPES0	**60.00**	3,914	**60.00**	**274**
	SHAPES1	**55.00**	398	**55.00**	**239**
(3) Slightly worse evaluation (<1%) and better time	Blasz1	**26.57**	5,603	26.80	**281**
	Jakobs1	**11.40**	2,193	11.50	**29**
	Marques	**77.97**	10,692	78.00	**21**
	Profiles5	**3,332.7**	1,317	3,351.94	**300**
	Profiles6	**3,097.86**	813	3,121.36	**171**
	Albano	**87.9%**	5,460	87.23%	**299**
	Mao	**81.0%**	667	80.91%	**152**

To compare the time frames with which solutions can be produced with the proposed approach we will further examine groups 2 and 3. In group 2, the proposed packing approach achieves equal solutions to the best reported for "Fu," "Shapes0" and "Shapes1." However, with the presented approach, the solutions are found 8, 14, and 1.5 times more quickly, respectively. In the final group, we report solutions where the proposed approach found a slightly worse solution (within 1%) than the literature best but in a significantly quicker time. The most extreme example of this is demonstrated by the "Marques" data set where the best literature solution was found after around 3 hours of computation whilst an almost equivalent solution was found by the proposed approach well within the 5 minutes allowed (over 500 times faster). These results have been included to demonstrate that very good solutions of comparative quality can be achieved within time frames that would be acceptable for use in industry.

8. Summary

Until now, no-fit polygon algorithms only operated on straight edge representations. If arcs were present, the polygons had to be represented by straight line segments in order for the no-fit polygon to be used. This led to computational inefficiencies, especially if a low resolution was used, so that the arcs could be more accurately represented.

The main contribution of this paper is the presentation of a robust no-fit polygon algorithm that is able to cope with arcs. In order to demonstrate its effectiveness, we show that we are able to produce better results on 16 (of 36) benchmark problems. On another 3 instances, we find results that are equal to the best-known solution. Perhaps of more interest is the fact that our algorithm is much faster than other approaches. The major change from our previous work is the use of the arc no-fit polygon. This demonstrates that not only is the arc no-fit polygon able to produce superior solutions, but it can also be done more efficiently.

Having access to an arc no-fit polygon, we believe, allows more adoption of this methodology both by the academic and user communities in order to produce higher-quality solutions, in much less time, than is possible with current approaches.

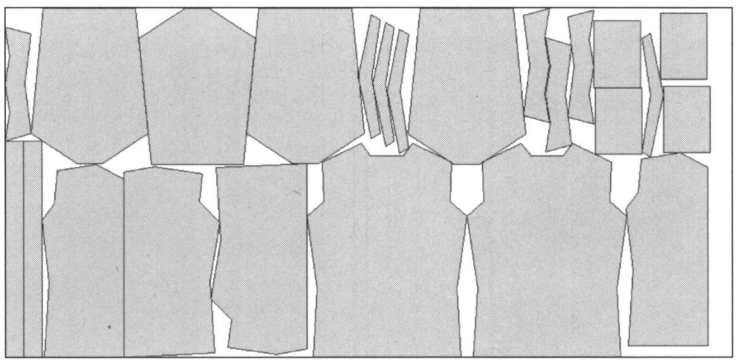
Acknowledgments

The authors thank the Engineering and Physical Sciences Research Council (EPSRC) for funding the project (ref. GR/S52414/01).

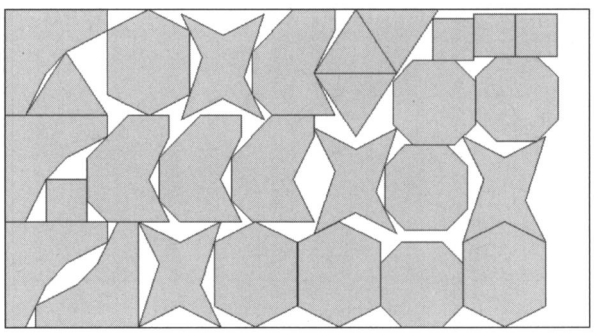
Appendix. Best Solutions Produced on Literature Benchmark Problems

In the plates of this appendix we present best solutions based on benchmark problems from the literature.

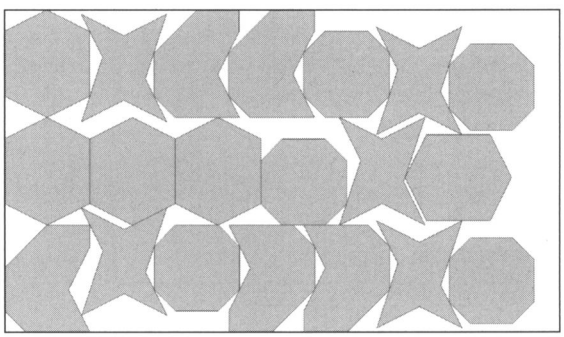
Plate 1



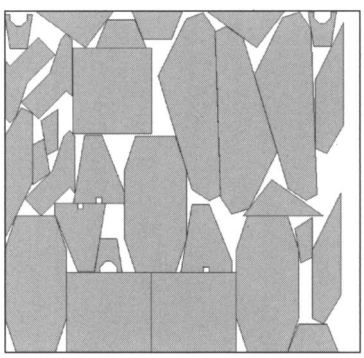
Albano: 9,980.86 units (299s)



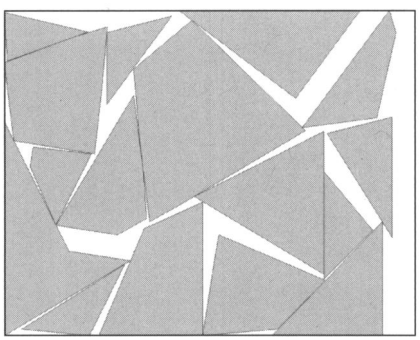
Blasz1: 26.80 units (281s)



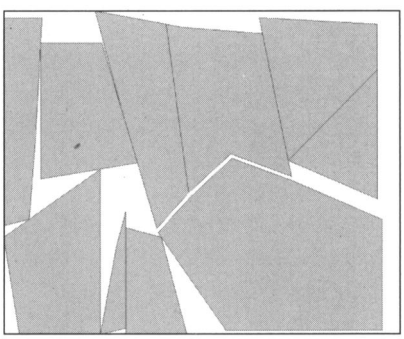
Blasz2: 24.80 units (14s)



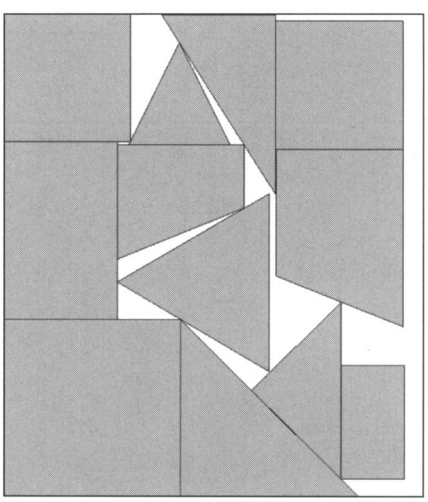
Dagli: 59.94 units (252s)



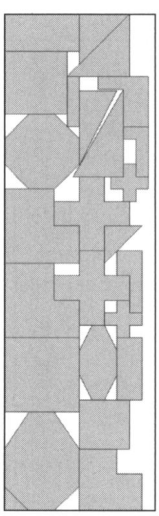
Dighe1: 1,210.00 units (3s)



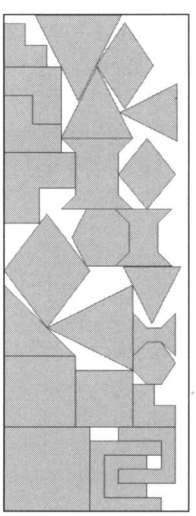
Dighe2: 1,180.00 units (148s)



Fu: 31.57 units (139s)

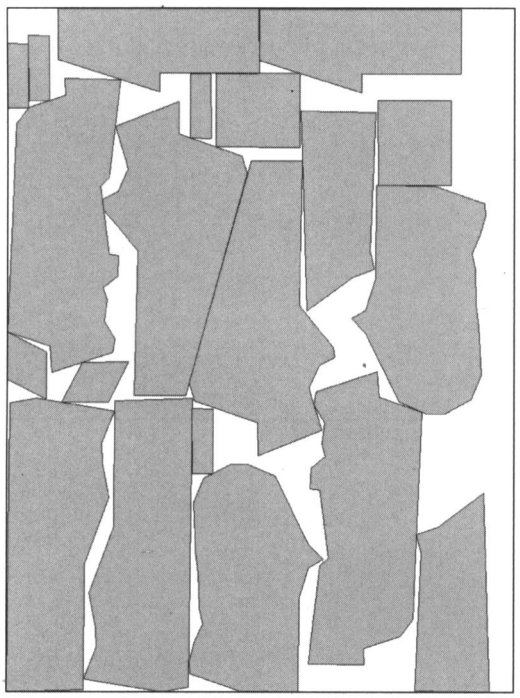


Jakobs1: 11.50 units (29s)

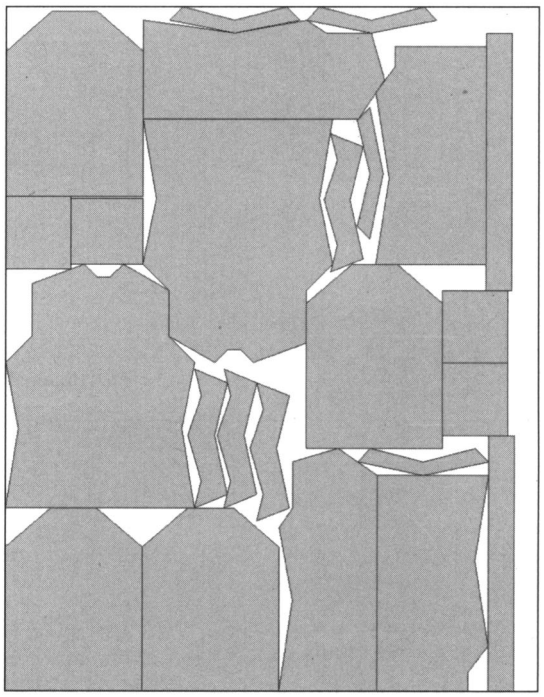


Jakobs2: 24.70 units (51s)

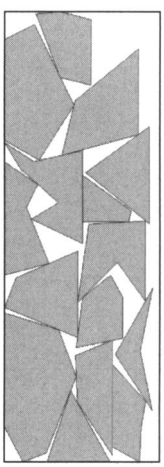
Plate 2



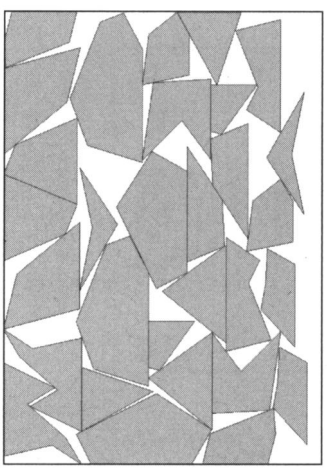
Mao: 1,821.70 units (152s)



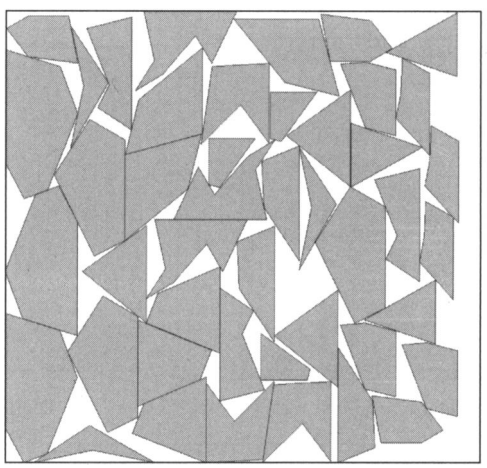
Marques: 78.00 units (21s)



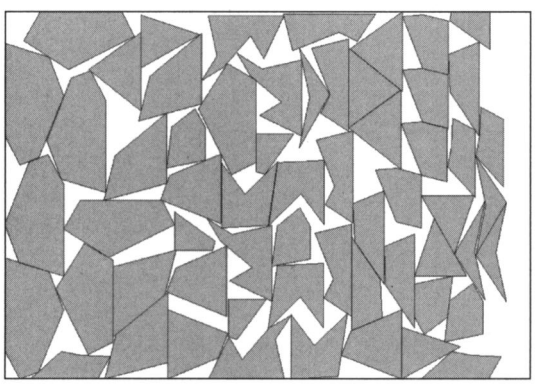
Poly1A: 13.30 units (254s)



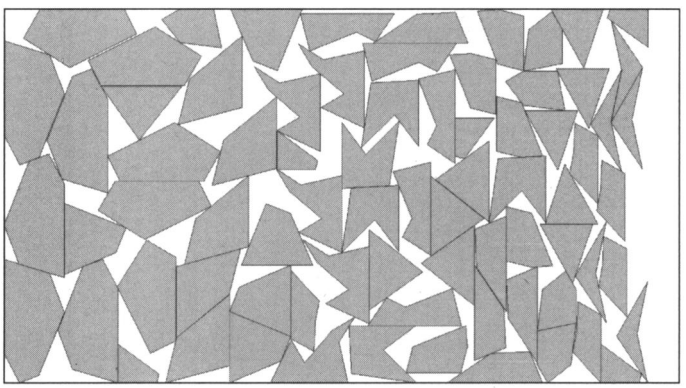
Poly2A: 27.09 units (239s)



Poly3A: 40.45 units (429s)

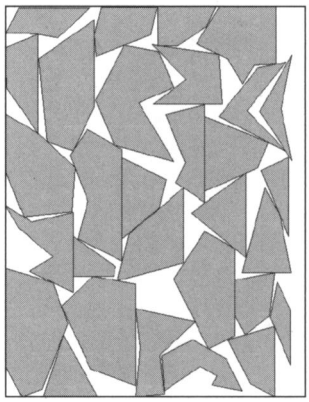


Poly4A: 54.60 units (224s)

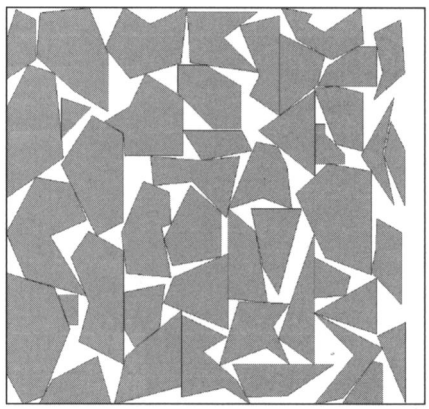


Poly5A: 68.84 units (300s)

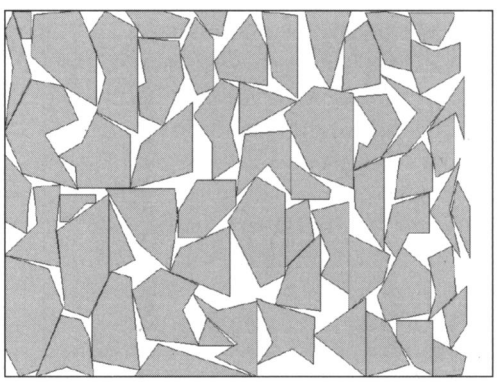
Plate 3



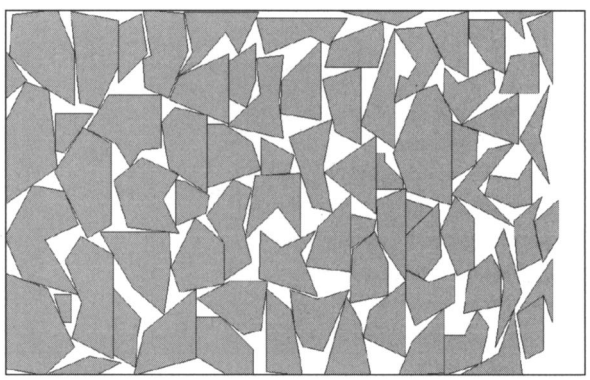
Poly2B: 29.63 units (189s)



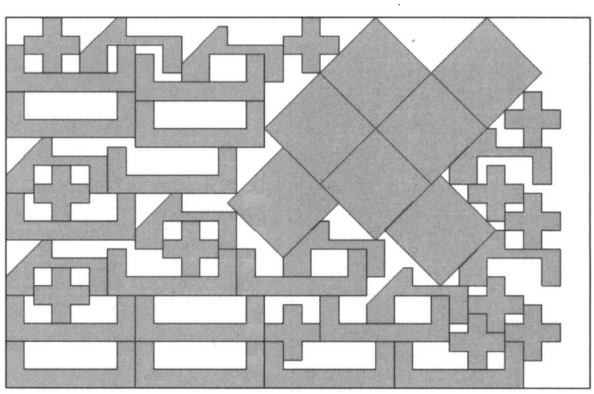
Poly3B: 40.50 units (114s)



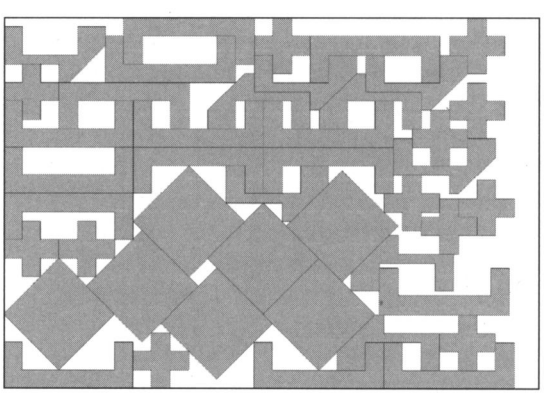
Poly4B: 51.18 units (176s)



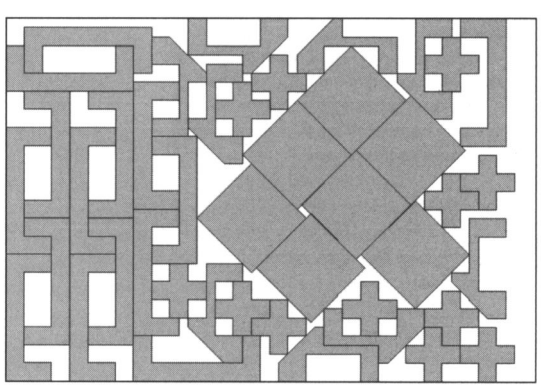
Poly5B: 60.86 units (299s)



SHAPES0: 60.00 units (274s)

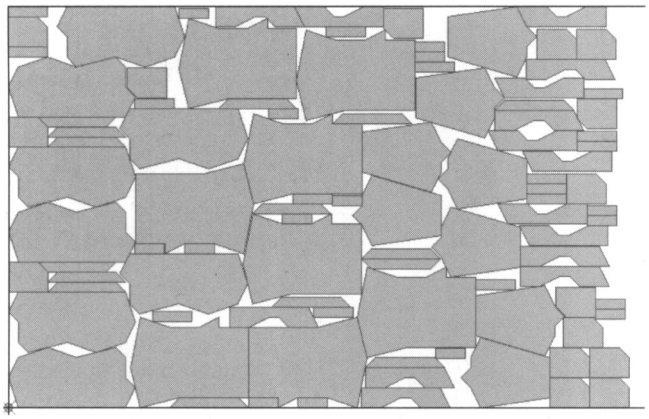


SHAPES1: 55.00 units (166s)

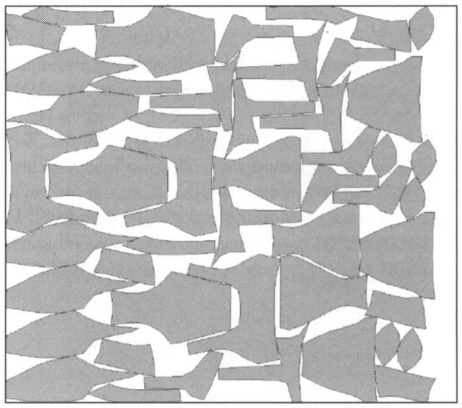


SHAPES: 56.00 units (226s)

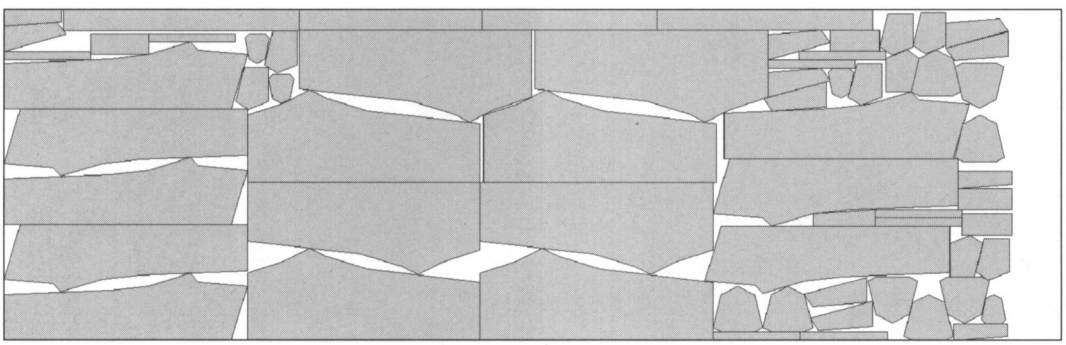
Plate 4



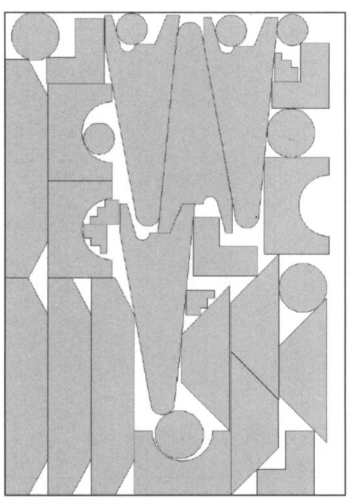
SHIRTS: 63.16 units (806s)



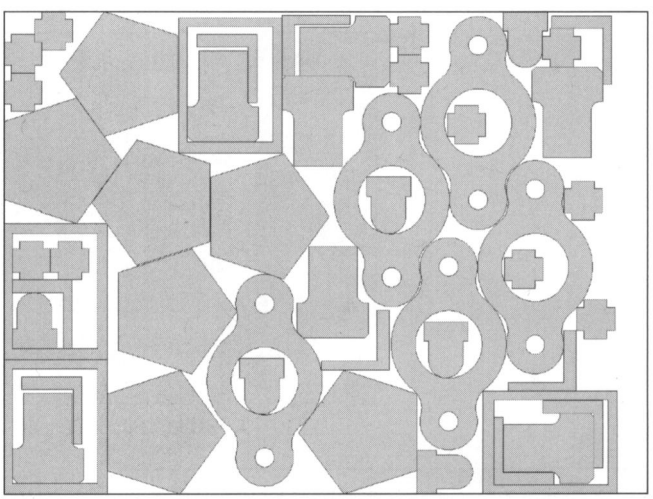
SWIM: 6,270.88 units (141s)



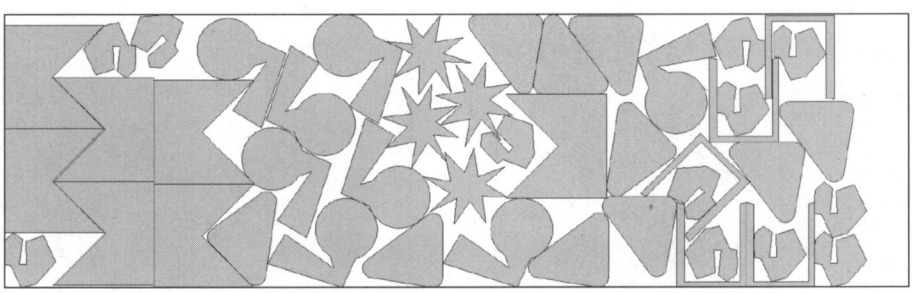
TROUSERS: 243.00 units (12,484s)



Profiles1: 1,359.90 units

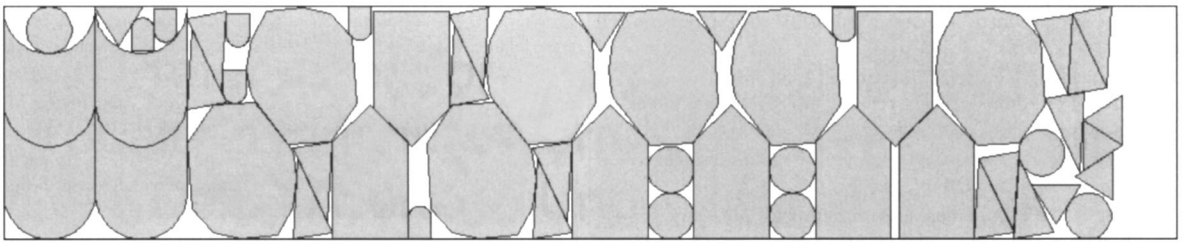


Profiles2: 3,194.19 units

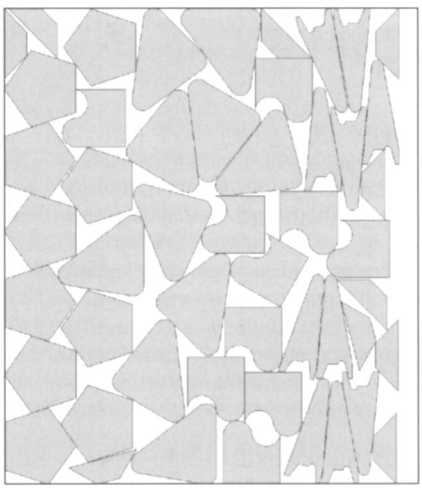


Profiles3: 7,881.13 units

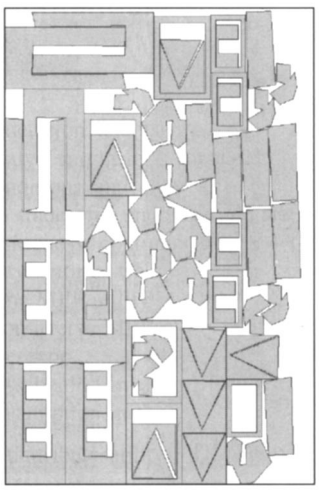
Plate 5



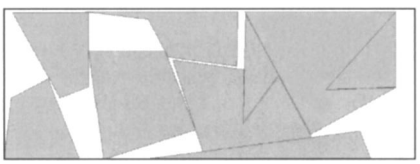
Profiles4: 2,425.26 units



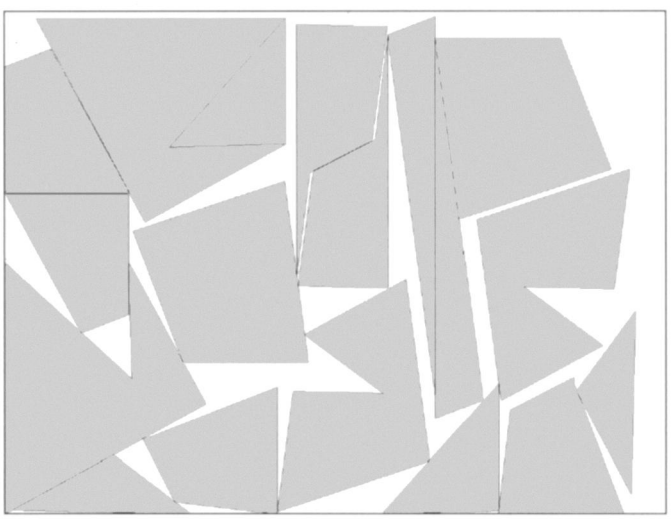
Profiles5: 3,351.94 units



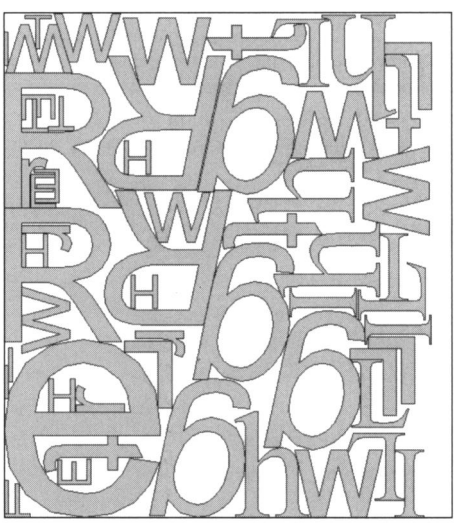
Profiles6: 3,121.36 units



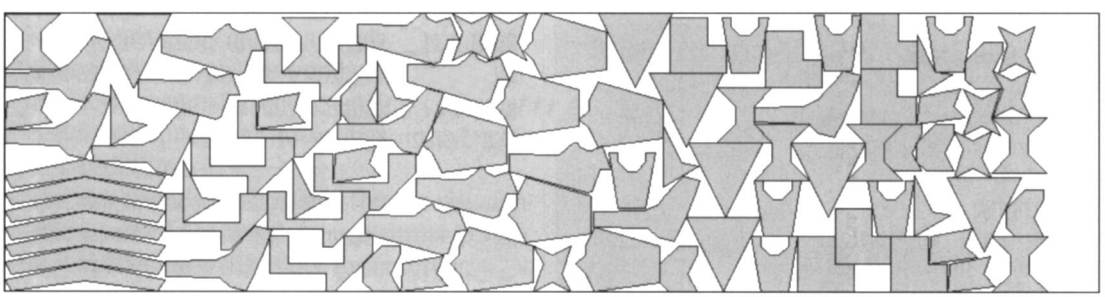
Profiles7: 1,292.30 units



Profiles8: 1,263.11 units



Profiles9: 1,278.21 units



Profiles10: 11,219.60 units

References

Albano, A., G. Sapuppo. 1980. Optimal allocation of two-dimensional irregular shapes using heuristic search methods. *IEEE Trans. Systems, Man and Cybernetics* **SMC-10** 242–248.

Bennell, J., G. Scheithauer, Y. Stoyan, T. Romanova. 2008. Tools of mathematical modeling of arbitrary object packing problems. *Ann. Oper. Res.*, ePub ahead of print November 9.

Bennell, J. A., J. F. Oliveira. 2008. The geometry of nesting problems: A tutorial. *Eur. J. Oper. Res.* **184** 397–415.

Bennell, J. A., X. Song. 2008. A comprehensive and robust procedure for obtaining the nofit polygon using Minkowski sums. *Comput. Oper. Res.* **35**(1) 267–281.

Bennell, J. A., K. A. Dowsland, W. B. Dowsland. 2001. The irregular cutting-stock problem—A new procedure for deriving the no-fit polygon. *Comput. Oper. Res.* **28**(3) 271–287.

Blazewicz, J., P. Hawryluk, R. Walkowiak. 1993. Using a tabu search approach for solving the two-dimensional irregular cutting problem. F. Glover, M. Laguna, T. E. de Werra, eds. *Tabu Search. Annals of Operations Research*, Vol. 41. J. C. Baltzer AG, Science Publishers, Red Bank, NJ, 313–325.

Bounsaythip, C., S. Maouche. 1997. Irregular shape nesting and placing with evolutionary approach. *Proc. IEEE Internat. Conf. Systems, Man and Cybernetics*, Vol. 4. 3425–3430.

Burke, E. K., R. S. R. Hellier, G. Kendall, G. Whitwell. 2006. A new bottom-left-fill heuristic algorithm for the two-dimensional irregular packing problem. *Oper. Res.* **54**(3) 587–601.

Burke, E. K., R. S. R. Hellier, G. Kendall, G. Whitwell. 2007. Complete and robust no-fit polygon generation for the irregular stock cutting problem. *Eur. J. Oper. Res.* **179**(1) 27–49.

Cunninghame-Green, R. 1989. Geometry, shoemaking and the milk tray problem. *New Scientist* **1677** 50–53.

Cunninghame-Green, R., L. S. Davis. 1992. Cut out waste! *O.R. Insight* **5**(3) 4–7.

Dighe, R., M. J. Jakiela. 1996. Solving pattern nesting problems with genetic algorithms employing task decomposition and contact detection. *Evolutionary Comput.* **3** 239–266.

Dowsland, K. A., W. B. Dowsland, J. A. Bennell. 1998. Jostling for position—Local improvement for irregular cutting patterns. *J. Oper. Res. Soc.* **49**(6) 647–658.

Fujita, K., S. Akagji, N. Kirokawa. 1993. Hybrid approach for optimal nesting using a genetic algorithm and a local minimisation algorithm. *Proc. 19th Annual ASME Design Automation Conf.*, Albuquerque, NM, Vol. 65. Part 1. ASME, New York, 477–484.

Garey, M., D. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York.

Ghosh, P. 1991. An algebra of polygons through the notion of negative shapes. *CVGIP: Image Understanding* **54**(1) 119–144.

Gomes, A. M., J. F. Oliveira. 2006. Solving irregular strip packing problems by hybridising simulated annealing and linear programming. *Eur. J. Oper. Res.* **171**(3) 811–829.

Hopper, E. 2000. Two dimensional packing utilising evolutionary algorithms and other meta-heuristic methods. Ph.D. thesis, University of Wales, Cardiff.

Huyao, L., H. Yuanjun, J. A. Bennell. 2007. The irregular nesting problem: A new approach for no-fit polygon calculation. *J. Oper. Res. Soc.* **58** 1235–1245.

Jakobs, S. 1996. On genetic algorithms for the packing of polygons. *Eur. J. Oper. Res.* **88** 165–181.

Kendall, G. 2000. Applying meta-heuristic algorithms to the nesting problem utilising the no-fit polygon. Ph.D. thesis, School of Computer Science and Information Technology, University of Nottingham, Nottingham, UK.

Mahadevan, A. 1984. Optimisation in computer aided pattern packing. Ph.D. thesis, North Carolina State University, Raleigh, NC.

Marques, V. M. M., C. F. G. Bispo, J. J. S. Sentieiro. 1991. A system for the compaction of two-dimensional irregular shapes based on simulated annealing. *Proc. 1991 Internat. Conf. Indust. Electronics, Control and Instrumentation—IECON'91*, Kobe, Japan, Vol. 99. 1911–1916.

Oliveira, J. F., J. S. Ferreira. 1993, Algorithms for nesting problems. R. V. V. Vidal, ed. *Applied Simulated Annealing. Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, Berlin, 255–273.

Oliveira, J. F., A. M. Gomes, J. S. Ferreira. 2000. A new constructive algorithm for nesting problems. *OR Spektrum* **22** 263–284.

Ratanapan, K., C. H. Dagli. 1997. An object-based evolutionary algorithm for solving irregular nesting problems. *Proc. Artificial Neural Networks Engrg. Conf. (ANNIE'97)*, Vol. 7. ASME Press, New York, 383–388.

Wäscher, G., H. Haußner, H. Schumann. 2007, An improved typology of cutting and packing problems. *Eur. J. Oper. Res.* **183**(3) 1109–1130.

C