Strojniški vestnik - Journal of Mechanical Engineering 51(2005)5, 234-239
UDK - UDC 519.85:621.9
Pregledni znanstveni članek - Preview scientific paper (1.02)

Reševanje problemov zlaganja z uporabo simuliranega izničenja in genetskih algoritmov

The Resolution of Packing Problems Using Simulated Annealing and Genetic Algorithms

Alberto Gomez - David de la Fuente - Javier Puente - José Parreño

S tem prispevkom želimo predstaviti dva algoritma, ki sta načrtovana za optimiranje postopka rezanja na odrezalu oblike L ter zmanjšanje števila kosov, potrebnih za izdelavo določenega števila pravokotnih kosov. Predlagamo dva algoritma, prvega na osnovi genetskih algoritmov in drugega na osnovi simuliranega izničenja. Primerjali smo ju s pomočjo baze primerov. Oba algoritma sta dala zelo dobre rezultate, čeprav imata oba tudi posebnosti, ki so tudi predstavljene v tem prispevku.

(Ključne besede: algoritmi genetski, zlaganje, postopki rezanja, optimiranje postopkov)

The aim of this paper is to present two algorithms that are designed to optimise the cutting process of an L-type guillotine and to minimise the number of sheets used to manufacture a number of rectangular pieces. Two algorithms are proposed, one based on genetic algorithms and the other on simulated annealing. They are compared with the help of a bank of examples. Both algorithms provide very good results, although each of them has its peculiarities, which are described in this paper.

(Keywords: genetic algorithms, packing, cutting processes, process optimization)

0 INTRODUCTION

This paper describes an application that our work team made for a company in the metalwork industry. The aim was to make a computer programme to minimise the number of sheets of steel used by the company in the course of its production process.

The application that was designed focuses in particular on optimising the use of the L-type guillotine that the company owns. In contrast to traditional guillotines, which can only cut vertically or horizontally, the L-type guillotine can cut in both these directions simultaneously, which lends greater flexibility to the cutting and also leads to more efficient use of the steel sheets.

The company operates in the following way: orders for supplying rectangular pieces are received from the clients, and the order of production and numeric control of the guillotine are decided on the basis of these orders. The parts are cut from a fixed-size, rectangular base surface (which will generally henceforth be referred to as the metal sheet) measuring 2995 × 1250 mm.

Based on the above premises, the programme to be implemented must situate the different pieces that the client has ordered on the metal sheet, bearing in mind that orders are generally made for more than one unit of a particularly dimensioned piece, and that more than one metal sheet is generally required to complete any given order. The positioning should be such as to make the maximum use of the sheet material, which is equivalent to minimising the number of metal sheets that are used.

Once positioned, the programme provides the guillotine's numeric control with the cutting sequence of the sheets (the order the pieces are to be cut in). The guillotine begins cutting in the top, right-hand corner of the metal sheet, and successive cuts leave the pieces and the leftover material. To do this, an algorithm was designed based on positioning the pieces within the base surface and generating the corresponding cutting sequence for numeric control.

1 AN APPROACH TO THE PACKING PROBLEM

The cause of the problem lies in the fact that the raw materials that industry uses are available in certain standard sizes that usually need cutting up before they can be used in the industrial process. The obvious aim of this cutting phase is to make maximum use of the raw material. Outstanding early work in this field was done by Gilmore and Gomory ([1] and [2]) in resolving single-dimension problems; the scope of this paper was widened in 1965 to two-dimensional problems [3]. Analyses of these problems has since spread rapidly, though no global solution has been offered for all of them because of their complexity.

One way of solving this type of problem is to divide it into several sub-problems [4], and try to solve each of them separately. This paper will focus in particular on one such sub-problem, the packing problem, which fits perfectly into the kind of industrial process that a solution is needed for.

Various authors have confronted this particular problem, though none have found a method that provides the optimum solution for every case. From amongst the different approaches that have been put forward, mention should be made of the heuristic methods designed by Coffman [5], and Jakobs [6], which have provided different methods, none of which have solved the problem completely.

2 DESCRIPTION OF THE PROBLEM

Before analysing the solution that we have proposed, a more in-depth comment should first be made on the characteristics of the problem to be solved.

The first point to be noted is that the dimensions of the metal sheets are always the same. This simplifies the problem a great deal, as pieces are always placed on the same type of sheet. Furthermore, as customers generally order pieces that are of large dimensions, the number of pieces that fit onto a single metal sheet generally ranges between five and twenty.

Each order is generally made up of several hundred pieces, and as they are usually large, more than fifty metal sheets are generally involved in each order.

Furthermore, the particular conditions of the L-type guillotine used by the company entails space on the metal sheets being lost, which is an important point to be kept in mind. On its first approximation to the metal sheets the guillotine makes two 'approximation cuts' of 5 mm, one vertically and the other horizontally, in addition to which the clamping system used to hold the metal causes a further loss of 20 mm horizontally. Figure 1 shows the material loss that using this particular L-type guillotine leads to.

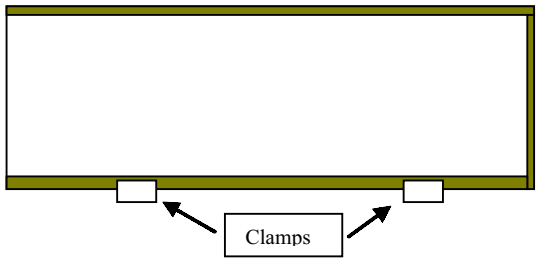

Fig. 1. *Leftovers*

The above factors mean less useable surface; in fact, the dimensions of the metal sheets are: [(2995-5) mm × (1250-20-5) mm].

Having commented on the particular nature of the problem that was analysed, two independent algorithms will next be developed, the position algorithm (section 3) and the cutting-sequence algorithm (section 4).

3 THE SOLUTION PROPOSED FOR CUTTING

This section will analyse the genetic algorithm designed to minimise the number of steel sheets that were used. The steps that were followed to do this will be described next. The codification used in the algorithm is based on integer numbers; each piece to be placed on the metal sheet is assigned a number, and that individual is formed with a string of numbers (a string of parts). The order in which the numbers appear in the string represents the positions of the pieces on the metal sheet.

An example will be worked through so as to explain the codification clearly: imagine you have 4 rectangular pieces to be placed on the metal sheet, and that each of these pieces is assigned a correlative number, the first piece is assigned the number 1, the second is number 2 and so forth. The individual (genotype) (1,3,4,2) could be a solution to the problem, which would mean that the first piece is first onto the metal sheet, followed by 3, then 4, and finally by 2.

In the case being analysed in this paper, this codification needs some modification, because, as has already been mentioned, not all the pieces fit

onto the same metal sheet, and several sheets must be used to make all the pieces ordered by the client.

A piece is assigned to a metal sheet when the "chromosome" is analysed. When the first position of the individual (the first piece) is read, a metal sheet is 'activated', and the piece is placed on the sheet; the individual's second position is then read, and an analysis determines whether it can be fitted onto the sheet or not. If it can, then that piece is assigned to the metal sheet and the third position is read off. This reading process continues until one of the pieces fails to fit onto the metal sheet. On detecting that a piece cannot be assigned to a metal sheet because the space available is not large enough, a second metal sheet is 'activated' and the piece is assigned to this second sheet. The process is iterated until all the pieces have been placed.

The next problem to be solved is the positioning of the pieces on the sheets. So far, pieces have been assigned to sheets, but their positions have yet to be specified. To solve this problem a modification of the 'Bottom-left algorithm' [6] was used. The reason for using this modification lies in the fact that Jakob's algorithm does not permit certain positions of pieces on the sheet [7].

This algorithm, called the 'Free Fall with Replacement algorithm', works as follows: the first piece assigned to the metal sheet is placed in the bottom, left-hand corner (piece 1 in Figure 2), and the others are placed as low down in the space as they can possibly fit. Once the piece has been placed, a check is made as to whether there is a space below it. If there is a space, then a search ensures to see if one of the rectangles still to be placed fits into that space. If such a rectangle shows up, it is slotted into the space.

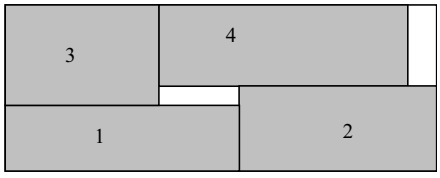

Fig. 2. *Free fall with replacement*

Having established the methods used to codify the placement of the pieces and to calculate their fit, the type of crossover and mutation to be used must be decided on next. In the light of earlier work that we have carried out [8], a Partial Matching type crossover was opted for, along with an order-based mutation. The Partial Matching Crossover "PMX" [9] is a crossover type that is widely used in published work on how to resolve the travelling salesman problem by using genetic algorithms with decimal coding. Adapting this crossover to packing problems is simple, and will now be described. Given two "parent chromosomes", the operator copies a substring of one of the parents directly into the same positions in the offspring. The remaining positions are then filled with the values that have yet to be used, in the same order as they occur in each of the parents.

The mutation, called Order-based Mutation, is based on the work of Davis [10] and consists of interchanging the positions of two rectangles of the same individual.

4 CUTTING THE PIECES

As has already been pointed out, a second algorithm had to be applied in order to instruct the guillotine's numeric control on how to cut the metal sheets in order to obtain the pieces. It is useful to explain the way the guillotine approaches the steel sheets in order to understand how the algorithm works. The guillotine starts at the top, right-hand corner of the sheet, and after making the initial two cuts that have already been mentioned in section 3, it proceeds to cut the pieces.

The algorithm responsible for carrying out this task acquires the distribution of the pieces in the metal sheet from the genetic algorithm (GA), and generates an output file that tells the guillotine's numeric control what order to cut the pieces in. In order to indicate this sequence the 'X' and 'Y' coordinates of the different cuts need to be known. Thus, for example, to cut pieces 1 and 2 of Figure 3, the coordinates of the bottom, left-land corner of both pieces should be given. The guillotine first cuts up to coordinates 'X' and 'Y' of the second piece and then cuts the first piece.

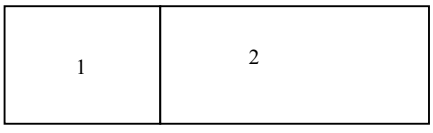

Fig. 3. *An example of the cutting sequence*

The problem of this cut lies in the areas where there is no piece assigned, which also have to be cut if the pieces ordered by the client are to be produced. Figure 4 highlights the problems that cutting these pieces involves; if the useful pieces are the rectangles (1,2,3,4), (A,B) represent the wasted

parts of the metal sheet, then piece B cannot be obtained from a single cut, and must be divided in two because of its non-rectangular shape.

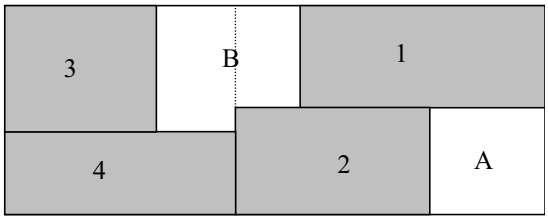

Fig. 4. *Cutting sequence*

The task of the sequencing algorithm that was designed consists of detecting these problematic parts and of deciding how to divide them. To do this, the algorithm simulates how the guillotine works. First, it acquires the positions of the pieces to be cut from the genetic algorithm, and then it fictitiously places the guillotine in the top, right-hand corner of the metal sheet and attempts to start cutting the different pieces one by one. When it detects that a piece cannot be cut because there is a waste piece in the guillotine's cutting area, it tells numeric control to eliminate this leftover piece before proceeding to cut the customer's piece.

Figure 4 serves to illustrate this process. First, an attempt is made to cut piece 1. To see if this piece can be cut, the guillotine is 'placed' in the lower, left-hand corner of this same piece, and the metal sheet is 'cut' in this position. If the piece is produced cleanly, the following piece is tried; if it not, then the extra material on piece 1 is eliminated. In this example, the piece is obtained without extra material and the cut is accepted; once liberated, the simulator attempts to cut the next piece, which in this particular case is piece 2. However, a problem arises, because if the guillotine is placed in the bottom, left-hand corner of this piece and the cut is made, the piece that is produced is L-shaped, and is made up of pieces A, 2, and part of B. Once this problem is detected the algorithm must determine how to obtain piece 2 without extra material. In this case the algorithm tells numeric control that first piece A should be cut, and in order for this to occur it provides numeric control system with the relevant coordinates for the bottom, left-hand corner; then one part of piece B must be cut, so the algorithm must provide numeric control system with the coordinates. When these cuts have been made, piece 2 can easily be cut out. Piece 3 is the next to be cut, but before doing so, the part of piece B that has yet to be cut must be cut; finally, piece 4 is cut out.

5 RESULTS

The computer programme that was already being used by the company in question was used in the study this paper describes. As this programme was also involved in trying to minimise the number of metal sheets being used, it is useful to compare the solutions provided by this programme with the solutions that our research came up with.

A number of experiments have been carried out using real data provided by the company, and our algorithms and the one previously used by the company were contrasted. Table 1 provides the results of this comparison.

The parameters applied during the different tests by the genetic algorithm are: crossover probability, 0.7; mutation probability, 0.3; special mutation probability, 0.4; number of generations, 3; and population size, 10 (only 30 calculations will be made using these parameters). Similarly, the Simulated Annealing (SA) parameters were selected for 30 calculations. The low number of generations and the small population size are both a result of the type of solution that was demanded. The company required a 'quick fix', where within three minutes the algorithms would generate a solution that was similar to the solution provided by the original system as shown in Table 1 for solution 1.

Table 1 highlights how the heuristics proposed provide better than or at least equal results to those offered by the guillotine's original programme (the old algorithm) in most cases. It also indicates how both the SA and GA solutions get better as the number of pieces involved gets higher.

Column 2 is the number of pieces in the experiment, and the other columns indicate the number of sheets required to manufacture these pieces using each of the algorithms.

Tests were run to generate random solutions (column 3 of Table 1) to check whether GA and SA were really necessary. Generating 30 random solutions and using the free fall with replacement heuristic to analyse the number of sheets needed to place and cut all the rectangles reached the random solution. The results shown in this column confirm the need for GAs or SAs.

Table 1 also highlights how it is impossible to decide which of the two heuristics is better. In

Table 1. *Experiment results*

Example	Pieces	Old Algorithm	Random	Simulated Annealing (SA)	Generic Algorithm (GA)
1	547	69	71	61	67
2	229	45	48	47	45
3	52	21	21	21	21
4	16	5	4	5	4
5	228	42	45	42	42
6	722	110	107	91	103
7	683	151	149	144	149
8	732	176	175	174	173
9	798	114	115	105	111
10	754	136	133	129	131
11	580	82	79	79	79
12	1346	223	219	208	208
13	192	42	42	44	42
14	573	53	57	51	52
15	529	92	89	91	87

some cases the GA provides better solutions, whilst SA is better for other examples.

6 CONCLUSIONS

This paper describes two hybrid systems to solve a real-life industrial problem: that of automating an L-type guillotine. Such an automation involves designing a programme to distribute pieces ordered by clients onto a number of rectangular sheets, minimising the number of sheets required. The programme should also indicate the guillotine's numeric control, the sequence the pieces should be cut in. A series of examples were generated, and the quality of the solutions provided by the algorithms was compared with those generated by the programme supplied by the manufacturer of the machinery. The algorithms were shown to work better in most cases. The two algorithms were also compared to each other to decide which of the two was more suited to this kind of problem. No conclusions could be drawn in this respect, as whether one algorithm is superior to another depends on the particular example that is used.

7 REFERENCES

[1] Gilmore P.C., Gomory R.E. (1961) A linear programming approach to the cutting-stock problem. *Operations Research* 9:724-746.

[2] Gilmore P.C., Gomory R.E (1963) A linear programming approach to the cutting stock problem. *Operations Research* 11:863-888.

[3] Gilmore P.C., Gomory R.E. (1965) Multistage cutting stock problems of two and more dimensions. *Operations Research* 13:94-1120.

[4] Dyckhoff H. (1990) A typology of cutting and packing problems. *European Journal of Operational Research* 44:145-159.

[5] Coffman E.G., Shor P.W. (1990) Average-case analysis of cutting and packing in two dimensions. *European Journal of Operational Research* 44:134-144.

[6] Jakobs S. (1996) On genetic algorithms for the packing of polygons. *European Journal of Operational Research* 88:165-181.

[7] Gómez A., De la Fuente D., Priore P. (2000) Resolución del problema de strip-packing mediante la metaheurística algoritmos genéticos. *Boletín de la SEIO* 12-16.

[8] Gómez, A., De la Fuente D. (2000) Resolution of strip-packing problems with genetic algorithms. *Journal of the Operational Research Society*, 51:1289-1295.

[9] Golberg, D.E., Lingle R. (1985) Alleles, Loci, and the TSP. In Proceedings of the first international conference on genetic algorithms, *Lawrence Erlbaum Associates*, Hillsdale, NJ 154-159.
[10] Davis L. (1991) Handbook of genetic algorithms. *Van Nostrand Reinhold*, New York.

Authors' Address: Prof.Dr. Alberto Gomez
David de la Fuente
Prof.Dr. Javier Puente
José Parreño
University of Oviedo
Campus de Viesques, s/n
33204 Gijon Asturias
Spain
agomez@epsig.uniovi.es
david@epsig.uniovi.es
jpuente@epsig.uniovi.es
parreno@epsig.uniovi.es

Prejeto: 3.12.2004
Received:
Sprejeto: 25.5.2005
Accepted:
Odprto za diskusijo: 1 leto
Open for discussion: 1 year