ELSEVIER

Journal of Computational and Applied Mathematics 64 (1995) 91–102

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

Embedding of linear programming in a simulated annealing algorithm for solving a mixed integer production planning problem

J. Teghem, M. Pirlot*, C. Antoniadis

Department of Mathematics and Operational Research, Faculté Polytechnique de Mons, 9, rue de Houdain, B-7000 Mons, Belgium

Received 7 October 1993; revised 12 July 1994

Abstract

The present paper is concerned with the grouping of book covers on offset plates in order to minimize the total production cost. The mathematical formulation of the problem involves both binary and continuous variables. As exact methods are unable to provide solutions in reasonable time, a heuristic algorithm of the simulated annealing type is proposed. At each iteration, the values of the current solution binary variables are altered in order to yield a neighboring solution. To compute the corresponding values of the continuous variables and the value of the objective function, a linear programming routine is called at each iteration. This constitutes the main originality of the present approach and is in principle applicable in mixed integer programming problems. The procedure is tested on several examples.

Keywords: Simulated annealing; Linear programming; Combinatorial optimization; Heuristic search

1. Presentation of the problem

The present case study is concerned with the printing of book covers in the publishing industry.[1] More precisely we deal with the "mating" of book covers, i.e., the grouping of different covers on the same offset plate in order to print them simultaneously. In this section, we give a brief outline of the problem context. The interested reader is referred to [3] for more detail of the production process.

In a preliminary step, the covers to be produced, which are similar in size, are grouped in homogeneous subsets w.r.t. the covers main characterstics. These are:

- the type of paper on which they are to be printed and, in particular, its density;
- the number of colors needed both for the recto and the verso of the four page covers.

* Corresponding author.

[1] We are pleased to thank Mrs. M.P. Ladavid-Duboisse from Casterman, S.A., for submitting the problem.

SSDI 0377-0427(95)00009-7

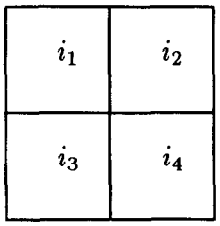

Fig. 1. The "shape" of an offset plate.

The mating problem only concerns such homogeneous subsets. Consider an homogeneous set of I covers. For each of them, a given number N_i, $i = 1, \ldots, I$, of copies has to be printed. With a single offset plate, it is possible to print four covers at a time in one color. After printing, the sheets of paper are cut into four parts. The four covers on a single plate are not necessarily the same and the assignment of a book cover to each compartment of each offset plate is one of the main decisions to be made in this problem. To be more specific, the typical size I of the number of book covers in a production batch is of the order of 12, the number of copies to be printed varying from a few thousands to a few hundred thousands.

Let us call "shape", the specification of the four book covers assigned to a plate. Hence, a shape is defined by a 4-tuple (i_1, i_2, i_3, i_4) with $i_1, i_2, i_3, i_4 \in \{1, \ldots, I\}$ (see Fig. 1).

The cost structure has two main components (see [3] for more details).

- C_f, the unit cost of the plate
- C_t, the unit cost of a sheet of paper whatever the plate used for printing it.

Clearly, C_f is much larger than C_t. Note that some unavoidable waste of paper must be taken into account in the number of covers to be printed. This waste is due to the initial setting up of the machines and to casualties occurring during the printing process. The losses can be a priori estimated to 5% of the total production on average. Hence they are taken into account through multiplication of the cost C_t by the factor 1.05.

The problem of mating book covers consists in deciding the number of plates and their shape in order to produce the needed quantities at minimal cost. This amounts to finding a balance between two "extreme" solutions:

- one can use as many plates as there are book covers (I); in this solution, each cover appears four times on the same plate and on no other plate. In this case, for all ($i = 1, \ldots, I$), there will be $N_i/4$ copies printed of the plate with four times cover i on it. Hence the global production of covers will exactly correspond to the needs (i.e., $\sum_{i=1}^{I} N_i$). On the other hand, the cost associated with the number of used plates is maximal which is the drawback of this solution.
- In the other "extreme" solution, each plate is filled with four different covers i_1, i_2, i_3, i_4, all distinct, and each cover is assigned to only one plate. The number of plates is then minimal: $\lceil I/4 \rceil$, the smallest integer number larger than $I/4$. Hence the cost of the plates is minimal, but the number of printed copies of a plate will be $\max(N_{i_1}, N_{i_2}, N_{i_3}, N_{i_4})$ if i_1, i_2, i_3, i_4 are the covers assigned to the plate. This means that unnecessary copies are produced, unless all four numbers $N_{i_1}, N_{i_2}, N_{i_3}, N_{i_4}$ are equal, and this results in paper losses. Suppose indeed, to be more specific, that

$$N_{i_1} = 1000, \quad N_{i_2} = 4000, \quad N_{i_3} = 9000, \quad N_{i_4} = 7000$$

and that i_3 is present only in this form. The needed number of printed copies of this plate would have to be 9000 in order to satisfy the requirements of book cover i_3 and this will lead to the production of, respectively, 8000, 5000 and 2000 covers of i_1, i_2, i_3, i_4 (resp.) in excess. The number of wasted sheets of papers will hence be 15 000/4 for that single plate.

In Section 2, we present a formulation of this problem in the language of mathematical programming. Due to the number of variables needed and the typical size of the problem instances, it is hopeless to try to solve it by exact methods. An idea for solving this mixed integer programming problem by means of a combination of the simulated annealing technique and linear programming is presented in Section 3. The implementation of this original procedure is also outlined. Preliminary results and conclusions are then exposed in the final section.

2. A mixed integer linear programming formulation

In a first attempt, we try to solve the problem to optimality through a formulation in the language of mathematical programming.

2.1. A mixed integer nonlinear programming model

A nonlinear formulation in both continuous and binary variables is readily obtained. Let i ($i = 1, \ldots, I$) be the index of the different covers to be produced in quantity N_i. On the other hand, the index j ($j = 1, \ldots, I$) labels the plates (or the shapes) that can potentially be used. The maximum number of plates is I, the number of book covers.

Define the variables:

$$x_j = \begin{cases} 1 & \text{if plate } j \text{ is used} \\ 0 & \text{else} \end{cases} \quad (I \text{ binary variables}).$$

$y_{ij} = 0, 1, 2, 3, 4$ represents the number of times cover i is present on the shape chosen for plate j (I^2 integer variables). $z_j \geqslant 0$ represents the number printed of plate j (I continuous variables).

The problem can be formulated as follows:

$$\min C_f\left(\sum_{j=1}^{I} x_j\right) + 1.05\, C_t\left(\sum_{j=1}^{I} z_j\right)$$

subject to

$$(P_1) \qquad \sum_{j=1}^{I} z_j y_{ij} \geqslant N_i, \quad i = 1, \ldots, I,$$

$$\sum_{i=1}^{I} y_{ij} = 4x_j, \quad j = 1, \ldots, I,$$

$$z_i \leqslant M x_j, \quad j = 1, \ldots, I$$

where M is an upper bound of the number of copies of each plate to be produced, e.g., $M = \max_{i=1,\dots,I} N_i$.

The constraints of the first type in (P_1) are unfortunately nonlinear which is a severe drawback. Indeed nonlinear mixed integer programming is not a very well explored field and there is no commercial software which allows to solve such a problem in an efficient way.

2.2. A linear model

Nevertheless it is possible to overcome these difficulties and obtain a linear formulation which is more sophisticated and involves a larger number of variables and constraints. In this formulation, we use the variables x_j and z_j as above but the variables y_{ij} are substituted by the following ones:

$$y_{ijk} = \begin{cases} 1 & \text{if cover } i \text{ lies in the } k\text{th compartment } (k = 1, 2, 3, 4) \text{ of plate } j \\ 0 & \text{otherwise} \end{cases}$$

($4I^2$ binary variables).

$n_{ijk} \geqslant 0$ represents the number of copies of the ith cover produced on the kth compartment of plate j ($4I^2$ binary variables).

In these variables descriptions, the index $k = 1, 2, 3, 4$ refers to the 4 available places on each plate. The problem formulation is then:

$$\min C_f\left(\sum_{j=1}^{I} x_j\right) + 1.05\, C_t\left(\sum_{j=1}^{I} z_j\right)$$

subject to

$$(P_2)\qquad \begin{aligned} &\sum_{i=1}^{I} y_{ijk} \leqslant 1, && i = 1,\dots,I;\ k = 1,\dots,4, \\ &n_{ijk} \leqslant z_j, && i, j = 1,\dots,I;\ k = 1,\dots,4, \\ &n_{ijk} \leqslant N_i\, y_{ijk}, && i, j = 1,\dots,I;\ k = 1,\dots,4, \\ &\sum_{i=1}^{4}\sum_{k=1}^{4} y_{ijk} \leqslant 4x_j, && j = 1,\dots,I, \\ &z_j \leqslant M x_j, && j = 1,\dots,I, \\ &\sum_{j=1}^{I}\sum_{k=1}^{K} n_{ijk} \geqslant N_i, && i = 1,\dots,I. \end{aligned}$$

The number of variables and constraints is significantly increased. There are $8I^2 + 2I$ variables and $8I^2 + 7I$ constraints. For $I = 8$, this makes 528 variables (half of which are binary) and 568 constraints: For $I = 12$, the number of variables raises to 1176 and the number of constraints to 1236. It soon became evident that such problems could not be solved in a reasonable time on a microcomputer by using standard commercial softwares, like, e.g., OMP [4]. Moreover, other attempts using more appropriate formulations and powerful packages on mainframes proved equally unsuccessful. Hence we turn to the use of heuristics.

3. Use of simulated annealing

3.1. Principle of the combination of simulated annealing with linear programming to solve a mixed integer linear program

In recent years the so-called metaheuristics or general heuristics like simulated annealing (SA) – or Tabu Search – have proved to be extremely efficient tools for "solving" hard combinatorial optimization problems. We will not make here another presentation of these techniques which are by now well known. In case of necessity, we refer the reader to a synthesis recently written by one of us on the subject [5].

Usually, heuristics like SA are applied to purely combinatorial problems, i.e., problems entirely in discrete variables. On the contrary, the model described in Section 2 is not of that type as it involves continuous as well as binary variables. Nevertheless, when the determination of the continuous variables values is strongly linked to the binary values and when there is a relatively small number of continuous variables, the following idea seems feasible. At each iteration, we combine the application of

(a) the SA procedure for fixing the values of binary variables only: in the neighborhood $V(X)$ of the current solution X (this neighborhood $V(X)$ is only concerned with variations in the binary variables values), select at random another set of values for the binary variables;

(b) a linear programming routine for determining the best setting of the continuous variables (conditionally to the setting of the binary variables provided in step (a)).

The approach will be feasible only if the linear problem is of small size enough as it has to be solved a huge number of times.

This procedure fully determines a new solution $Y \in V(X)$ and also the corresponding value $F(Y)$ of the objective function. The SA classical test for comparing $F(X)$ and $F(Y)$ can then be performed; it determines which solution from X and Y will be the current solution at the next step. The conclusion of this test depends on the "temperature" at the current stage of the SA procedure and this "temperature" usually decreases in a stepwise geometric manner governed by three parameters:

- the initial temperature,
- the number of iterations performed without changing the temperature,
- the cooling rate, i.e., the decreasing factor applied to the temperature at the end of each "isoterm" series of iterations (see [5]).

The idea of calling an LP routine at each step of a SA process is, as far as we know, original. In principle, it can be applied to any mixed integer linear problem. In the sequel, we show how we implemented and tested this idea on the mating book covers problem. Various attempts have been made which are extensively described in [1]. In Section 3.2, we present what seems the most promising way of dealing with the binary variables (phase (a)) for the application we are concerned with. The binary variables determine the number J ($J \leqslant I$) of plates to be effectively used as well as the shape of each of them. So by the end of phase (a), the values of variables x_j and y_{ij}, $i, j = 1, \ldots, I$ in problem (P_1) are fixed. Problem (P_1) is then reduced to a very simple LP problem with J continuous variables z_j, $j = 1, \ldots, J$ with $J = |\{j : x_j = 1,\ j = 1, \ldots, I\}|$. In the cases of practical importance, I is of the order of 12 as mentioned above which means that it can be hoped to solve such an LP in much less than a second.

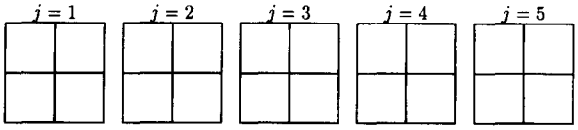

Fig. 2. A five plate example.

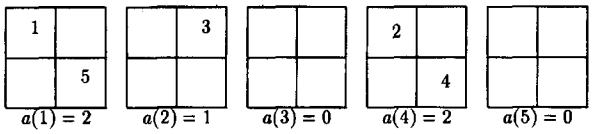

Fig. 3. Random selection of five places of type A.

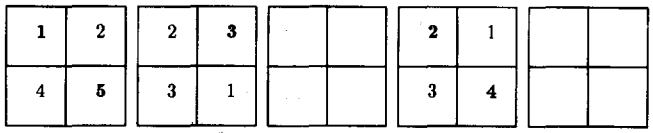

Fig. 4. Initial assignment. Type A places are in bold face.

3.2. *Random choice of new configuration of plates*

We call "configuration of plates" the assignment of a cover to each of the $4I$ "compartments" available on the I plates. We illustrate the procedure on an example with $I = 5$ plates (Fig. 2).

In order to define a configuration, we have to determine the number J of plates that will effectively be used and an assignment of the covers to each of the $4J$ available places. The only constraint on this assignment is that each of the I covers must be assigned to at least one place on one plate.

3.2.1. *Initial feasible configuration*

(a) We first select I places at random among the $4I$ available ones and we assign them the covers $1, 2, \ldots, I$. This procedure guarantees the feasibility of the initial configuration as all covers will appear at least once on the set of plates. These I places will be said to be of "type A". We denote by $a(j)$ (with $0 \leqslant a(j) \leqslant 4$) the number of type A places on plate j. We have $\sum_{j=1}^{I} a(j) = I$.

Example. Take the five plates of Fig. 2 and assume that we randomly selected $I = 5$ places of type A which we assign to the five covers to be produced as illustrated in Fig. 3.

(b) Let J be the number of plates with at least one place of type A, i.e., $J = |\{j: a(j) > 0, j = 1, \ldots, I\}|$.

The $4J - I$ remaining places on the plates with $a(j) > 0$ are called "type B" places and we randomly assign a cover to each of them.

Example. We can see in Fig. 4 a random assignment of covers to the $J = 3$ plates which will be used effectively. The covers assigned to places of type A are in bold face.

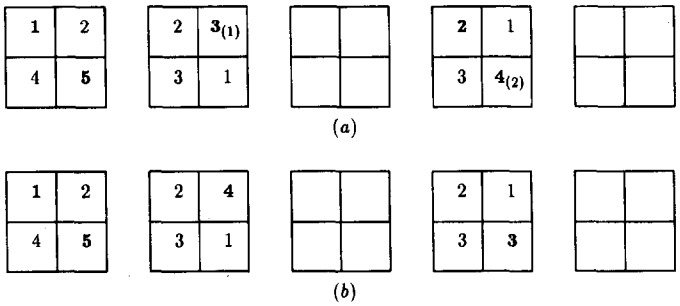

Fig. 5. Exchange of two covers on type A places.

(c) The $I - J$ nonassigned plates will not be used and are said to be of "type C".

Through the procedure described above, we are able to produce any feasible solution using any number of plates (less than or equal to the number of covers). It should be pointed out however that the configurations obtained through this process are not all equally likely.

3.2.2. Random choice of a new feasible configuration

The main constraints in producing new configurations in the "neighborhood" of the current one are the following.

- The new configuration must be feasible, i.e., every cover must appear at least once.
- It should be possible to vary the number of used plates, i.e., to raise J to $J + 1$ or decrease J to $J - 1$ (the latter only if $J > \lceil I/4 \rceil$).

Among the various tested procedures, we have selected the following one which fulfils best the above requirements. The selected procedure involves two steps.

(1) • Choose one place (j_1, k_1) (i.e., the k_1th place on the j_1th plate) among the I places of type A.
  • Choose one place (j_2, k_2) among the $4I - 1$ remaining ones; this place can be indifferently of type A, B or C.

(2) The new configuration depends on the type of place (j_2, k_2) but in all cases, the cover previously assigned to place (j_1, k_1), which we denote by $i(j_1, k_1)$ will now be assigned to (j_2, k_2): $i(j_2, k_2) \leftarrow i(j_1, k_1)$. The A status of (j_1, k_1) is also transferred to (j_2, k_2).

Case a: (j_2, k_2) is of type A. In order not to alter the number of type A places, the covers on (j_1, k_1) and (j_2, k_2) are exchanged: $i(j_2, k_2) \leftarrow i(j_1, k_1)$. The total number of used plates is not altered in this case.

Example. Consider as current configuration the one illustrated in Fig. 4. Assume the selected places are $(j_1, k_1) = (2, 3)$ and $(j_2, k_2) = (4, 4)$. The concerned covers, 3 and 4 are marked by (1) and (2) in Fig. 5(a). In this case covers 3 and 4 are exchanged yielding the new configuration illustrated in Fig. 5(b).

Case b: (j_2, k_2) is of type B.

Case b.1: $a(j_1) > 1$, i.e., the number of type A places on plate j_1 is larger than 1. In this case, we cannot consider the suppression of plate j_1 as it contains other type A places. So (j_1, k_1) is assigned

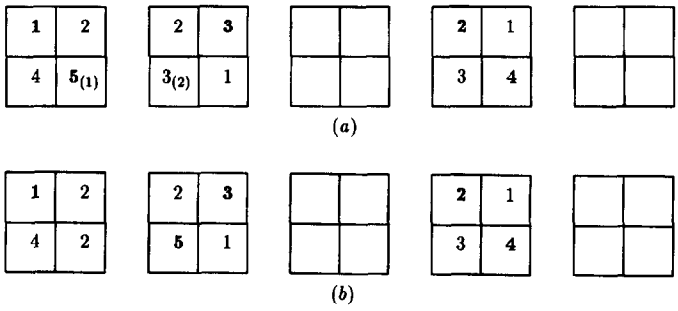

Fig. 6. Case b.1 change.

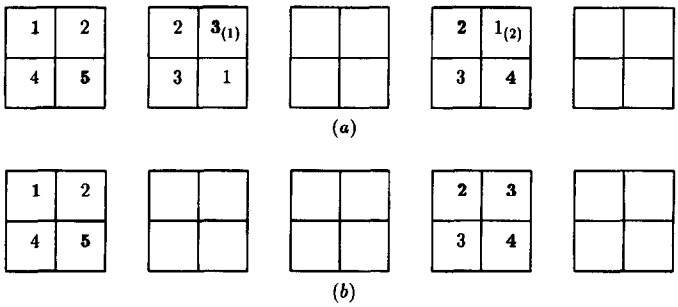

Fig. 7. Case b.2 change.

a cover i^*, randomly chosen among the I covers, and the status of (j_1, k_1) becomes B

$$i(j_1, k_1) \leftarrow i^*; \quad \text{type B.}$$

The number of the used plates J is not altered nor is the number of type A places.

Example. Starting again with the configuration illustrated in Fig. 4 we suppose that $(j_1, k_1) = (1, 4)$ and $(j_2, k_2) = (2, 3)$ (Fig. 6(a)). In the new configuration, place (2, 3) is assigned to cover 5 and receives the A status while the randomly chosen cover 2 will be assigned to place (1, 4) with B status (Fig. 6(b)).

Case b.2: $a(j_1) = 1$. Plate j_1 is emptied; the places of this plate receive the C status and the number of used plates is decreased by one unit: $J \leftarrow J - 1$.

Example. In the current configuration (Fig. 4), $(j_1, k_1) = (2, 2)$ and $(j_2, k_2) = (4, 2)$ (see Fig. 7(a)). As place (2, 2) is the only type A place on plate 2, this plate will not be used in the new configuration (Fig. 7(b)).

Case c: (j_2, k_2) is of type C. In the new configuration, plate j_2 which was not used before includes the cover imported from (j_1, k_1). Place (j_2, k_2) receives the A status. The three remaining type B places are assigned randomly chosen covers i_1^*, i_2^*, i_3^*:

$$i(j_2, k) \leftarrow i_1^*, i_2^*, i_3^*: \quad k \in \{1, 2, 3, 4\} \backslash \{k_2\}.$$

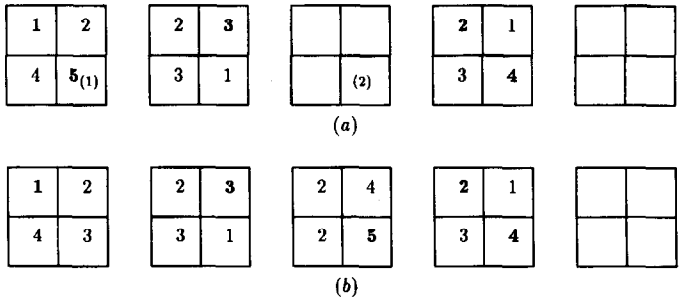

Fig. 8. Case c.1 change.

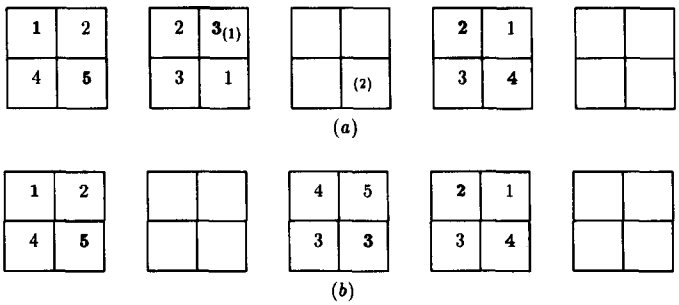

Fig. 9. Case c.2 change.

Plate j_1 is kept in use or not according to the rules described for cases b.1 and b.2.

Case c.1: $a(j_1) > 1$.

$i(j_1, k_1) \leftarrow i^*$; type B

$J \rightarrow J + 1$.

Example. Assume that $(j_1, k_1) = (1, 4)$ and $(j_2, k_2) = (3, 4)$ (Fig. 8(a)). The three covers randomly chosen for filling plate 3 are 2, 4, 2. Cover 3 is chosen to complete plate 1 (Fig. 8(b)).

Case c.2: $a(j_1) = 1$. Plate j_1 is emptied and the corresponding places receive the C status. The total number of plates remains unaltered.

Example. Suppose $(j_1, k_1) = (2, 2)$ and $(j_2, k_2) = (3, 4)$ (Fig. 9(a)). The three covers randomly chosen to fill plate 3 are 4, 5, 3 (Fig. 9(b)).

4. Preliminary results and conclusions

This mating book covers problem is particularly exciting as it is readily presented and looks quite simple but is tremendously difficult to solve. Several student projects were devoted to this

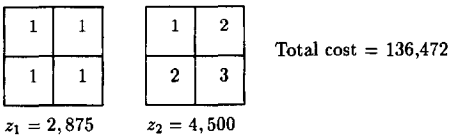

Fig. 10. Optimal solution of Example 1.

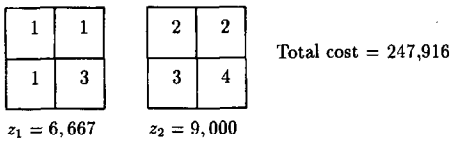

Fig. 11. Best solution found for Example 2.

problem at our university. Two of them are mainly concerned with the approach presented. The first one [3] can be considered as a preliminary exploration. A large part of the final product is a report on the – unfruitful – attempts to solve the mathematical programming formulations described in Section 2. In the second part of this work, a "classical" heuristic algorithm is deviced on the basis of successive resolutions of subproblems involving up to four covers. The great advantage of this approach is the extremely short computer times needed (2 or 3 s on a PC with a 80386 processor). Unfortunately, the solution quality which is quite good or optimal for instances of small size ($I = 3, 4, 5$), rapidly deteriorates when $I > 5$. This is due to the chosen approach and the exploding complexity of the problem.

The second student report [1] is a first attempt to combine SA with linear programming. Three alternative definitions of the neighborhood of the current solutions have been elaborated. The one presented in Section 3 appears to be the most promising. Note however that the numerical experimentation has been quite limited due to the following reason. The student version of the Linear Programming Software OMP (see [4]) is called to solve the LP programs. This results in prohibitive waste of computer time due to continuously loading and unloading OMP as an external routine to the main program. These operations cost about 5 or 6 s for each LP call which is indeed prohibitive, as the application of SA requires hundreds or thousands of calls to the LP routine. These contingencies have strongly limited the extent of the experimentation. We present below the results obtained through the above described method. These results should be considered as preliminary.

Four examples were treated with numerical values of the costs given by $C_f = 18\,676$ and $C_t = 12.8$. We use the notations introduced in Section 2 to present these examples.

Example 1. $I = 3$, $N_1 = 16\,000$; $N_2 = 9000$; $N_3 = 4500$. The best solution found is illustrated in Fig. 10.

Note that this solution is optimal as the exact formulation (P_1) could be solved.

Example 2. $I = 4$, $N_1 = 20\,000$; $N_2 = 18\,000$; $N_3 = 15\,000$; $N_4 = 8500$. The best solution found is illustrated in Fig. 11.

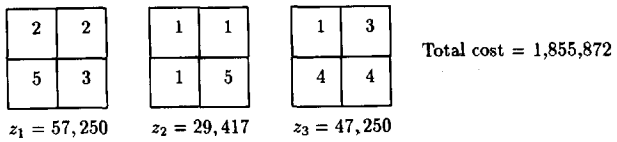

Fig. 12. Best found solution for Example 3.

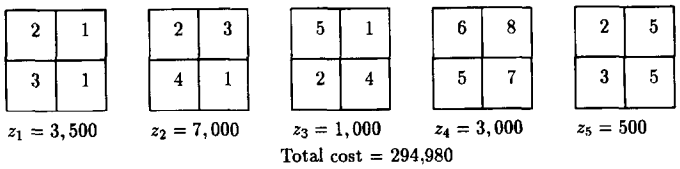

Fig. 13. Best found solution for Example 4.

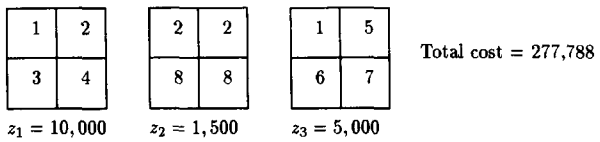

Fig. 14. Best known solution for Example 4.

Example 3. $I = 5$, $N_1 = 13\,500$; $N_2 = 114\,500$; $N_3 = 103\,500$; $N_4 = 94\,500$; $N_5 = 84\,500$. The best solution found is illustrated in Fig. 12.

Example 4. $I = 8$, $N_1 = 15\,000$; $N_2 = 12\,000$; $N_3 = 10\,000$; $N_4 = 8000$; $N_5 = 5000$; $N_6 = N_7 = N_8 = 3000$. The best solution found is illustrated in Fig. 13.

This solution is not optimal. For this example, lower cost solutions were obtained through the use of one of the alternative methods alluded to in Section 3 (see [1]). The best solution we know for this example is illustrated on Fig. 14.

Another student project is currently being elaborated on the same subject. It aims at realizing a deeper experimentation on the same principle of combination of SA (or Tabu Search) and LP. In this work however, a specially tailored LP routine will be integrated in the main program to avoid any disk transfer times. We hope that this will allow to test more in depth the feasibility and efficiency of the present approach, by solving larger examples and experimenting on different neighborhood structures.

References

[1] C. Antoniadis, Problème du mariage des couvertures: Résolution par la méthode du recuit similé, Graduate Thesis, Université de Mons-Hainaut, Belgium, 1992.

[2] C. Batta and J. Teghem, Optimization of production scheduling in the plastics processing industry, *JORBEL* **34** (2) (1994) 55–79.
[3] M. Naya, Problème du mariage des couvertures posé par la S.A. Casterman, Graduate Thesis, Faculté Polytechnique de Mons, Belgium, 1986.
[4] OMP optimization (version 3.0), User Manual, Beyers et Partners, Brasschaat, Belgium, 1987.
[5] M. Pirlot, General local search heuristics in combinatorial optimization: a tutorial, *JORBEL* **32** (1–2) (1992) 7–67.