ELSEVIER

Int. J. Production Economics 46-47 (1996) 13-26

international journal of
production
economics

# Using metaheuristics for solving a production scheduling problem in a chemical firm. A case study

Ph. Fortemps*, Ch. Ost, M. Pirlot, J. Teghem, D. Tuyttens

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

**Abstract**

The problem is raised by a workshop management team in a chemical industry producing materials of different types. The workshop contains several equipments in parallel and in series which can be used for the production of each job. The equipments in series are connected by a piping system forming a network by which the material runs out. Some connections are shared by several pairs of equipments so that the use of such a connection for one pair of equipments makes this connection unavailable at the same time for other pairs. Several particular constraints must be satisfied and deadlines exist for some jobs. The objectives are to minimize the makespan and the tardiness.

A method is proposed to heuristically obtain a satisfying production schedule. The procedure involves two stages. In a first step, given a fixed order of the jobs, a conventional heuristic is proposed to successively assign the jobs to the available equipments. In a second step, the simulated annealing or tabu metaheuristics are applied to optimize the order of the jobs. The method was implemented on a PC and applied to an instance of the problem. Some comparisons are made to analyse and compare the efficiency of the heuristics.

*Keywords*: Production scheduling; Simulated annealing; Tabu search

## 1. Introduction

The case we are dealing with in the present paper was raised by the managers of a workshop in a chemical plant producing materials of different types. In a future prospect of an automatic schedule, the managers of the workshop asked to our research team to define a method for optimizing a feasible production schedule of an order-book. The content of the contract was to implement the method on a PC, the resulting software having to be user-friendly and able to output a production schedule within a few minutes.

Many production scheduling problems are known to be hard from a computational viewpoint. Even simple "theoretical models such as Flowshop or Jobshop scheduling are NP-hard [1, pp. 241–242]. In practical problems, a lot of specific constraints and pecularities of the workshop are added; one can cite for instance ready dates of the components to be processed, due dates for the final products, availability periods of the production equipments,.... This often makes the classical problems and the algorithms developed for solving them of little value for practical purposes. Due to

* Corresponding author.


*SSDI* 0925-5273(95)00168-9

the intricate structure of the constraints and the various uncertainties involved in the data, finding a feasible schedule is sometimes considered as satisfactory. However, explicit objectives can be formulated such as minimizing makespan or maximal tardiness,.... If some constraints are modelled as soft constraints, the objective may be written as a linear combination including terms penalizing the violation of constraints. For an overview of the subject, the reader is referred to [2].

In view of this, real life production scheduling problems are often "solved" by means of adhoc heuristics specially tailored to a given situation. In recent years however, general heuristic search principles and methods have been proposed and extensively applied to a large variety of combinatorial optimization problems including sequencing and scheduling. These methods among which the best known are Simulated Annealing, Tabu Search and Genetic Algorithms, are often called "metaheuristics" as they provide general search schemes that have to be adapted to particular problems. A specific reference for each of the three above-mentioned methods is [3–5] (respectively). A general introduction to all those methods can be found in [6].

The case treated here does not fall into one of the categories of classical scheduling problems, but it presents similarities with some of them and there are many workshops in real factories where similar scheduling problem structures are met. Roughly speaking, we are dealing with a multi-stage production process, like in a Flowshop, with the same sequence of operations for all jobs. There are several machines of each type and each machine is devoted to a single type of operation. Among the – not uncommon – peculiarities of the present case is the fact that the paths leading from one type of machine to the next one belong to a transportation network which may be relatively complicated. The flowing of products through a node or an arc of the network takes some time and forbids any other product to use this node or arc. This means that it is necessary to schedule the network occupancy as well. Another pecularity of the present case is the fact that a single job can eventually be processed simultaneously on several machines of the same type. This makes the management of the transfer network still more crucial and delicate since the access of a job to the type of machines downstream may be impossible when the network is "crowded".

Due to the complexity of the problem, exact approaches were excluded from the start and in view of the large variety of constraints, a flexible optimization method was highly commendable, in particular for being able to adapt to the evolution of the constraints and of the workshop layout. The architecture of our algorithm is relatively classical. It relies on the notion of a "priority list" of jobs which is simply here any (arbitrary) ordering of the jobs in a list. The selection of (non-arbitrary) priority lists (or dispatching rules) designed in view of yielding good schedules has been discussed extensively in the literature (see e.g. [2], Chapter 10, for an overview). Priority lists, in the sense we use them here, have been considered in Genetic Algorithms as a way of encoding schedules in a compact manner [7, 8]. The main ingredient, called "the scheduler" is a simple algorithm which is able to build a scheduling of the jobs on the basis of any priority list that could be considered. The "scheduler" takes the tasks one by one according to the order defined in the priority list and schedules them at their earliest possible date given the processing periods of the tasks which come before in the list and have already been scheduled. So, the problem reduces to finding a good (the best) schedule among those that a "scheduler" is able to generate. The search for a good (the best) priority list which yields a good (the best) priority list is performed by a master algorithm, called the "optimizer" which sequentially perturbs the current priority list and evaluates the new list by calling the "scheduler" and evaluating the schedule produced on the basis of the new list. Our "optimizer" is an implementation of a metaheuristic (Simulated Annealing or Tabu Search).

As already mentioned, this kind of algorithmic architecture is not uncommon for solving scheduling problems. Of course, the quality of the "scheduler" is crucial. By "quality" we mean the ability of the "scheduler" to produce optimal or near optimal schedules for some appropriate choices of the input priority list. Ideally, it would be desirable to prove that a "scheduler" is able to generate a class of schedules which contains at least one optimal schedule w.r.t. the chosen criterion (minimal makespan, ... ). Due to the complexity of the

constraints, we are unable to do it in the present case study.

Two main requirements have guided the conception of our "scheduler". It had to be fast (as it is called a huge number of times by the optimizer) and should yield schedules which are "as good as possible". The quality of the "scheduler" has been evaluated in an informal manner, by examining the produced schedules on a variety of test problems forwarded by the Company. The results were considered highly satisfactory as compared with current performances in the factory. Close examination of the schedules for the different types of equipments showed very little idle periods for some equipments and led to the conclusion that one could hardly do better. Note however that the "scheduler" is closely tailored to the case. It works upstream to downstream. The jobs are always scheduled to begin at their earliest possible date except in one case: there is a special type of constraints which says that the processing periods of a job on two particular types of equipments must be consecutive, without discontinuity. This requirement may force to delay the beginning of the processing period on the upstream equipment in order to wait for an availability period for the downstream equipment.

In the sequel we give more detail on the problem at hand and on the algorithm we have implemented to solve it. Finally, we sketch the results obtained on a test problem.

### 1.1. Description of the workshop

A simplified description is given in this section; additional comments are provided in Section 4.

The workshop is equipped to achieve some jobs which correspond to a fixed quantity – identical for each job – of material of different types. To be

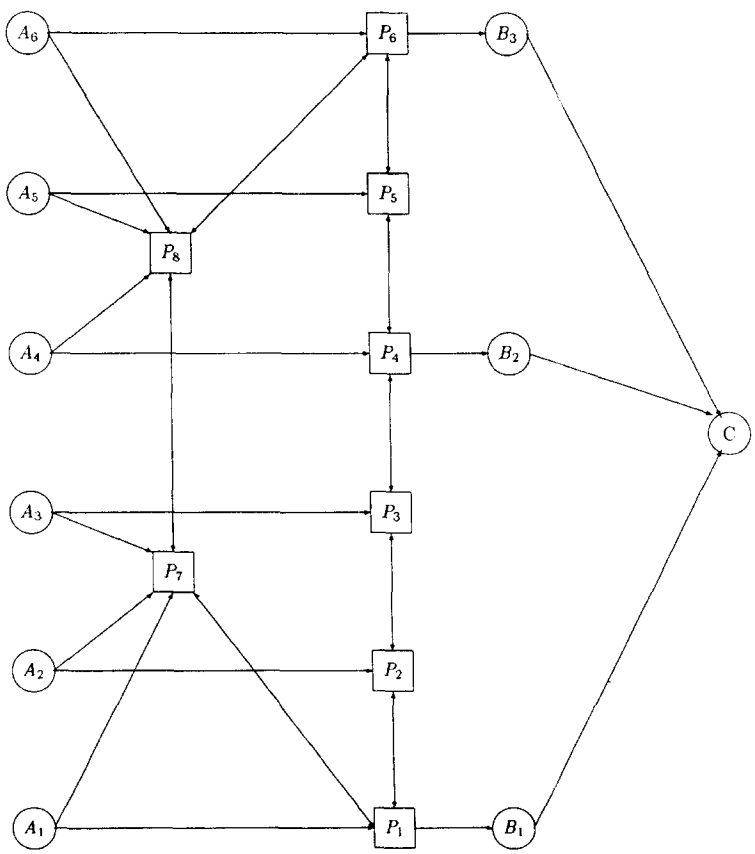


Fig. 1. Workshop graph.

processed, a job must be assigned successively to three types of equipments.

(a) The equipments of the first type are denoted by A; there are six identical equipments of this type $(A_1, \ldots, A_6)$ in the workshop (see Fig. 1). Each job can be treated by any equipment A and the processing time of a job depends only of the type of the job. Before a process can start, there is a constant set-up time corresponding to the loading of the equipment. This initial operation is performed by a special team of workers so that at each instant, it is impossible to simultaneously prepare more than one equipment. After a job has been processed, the equipment does not become immediately available. It is necessary to wait for the end of the flowing of the material out of the equipment towards an equipment of type B (see (b) below). By this time, the equipment A is completely empty and another job can be loaded on it.

(b) The second equipments are denoted by B; there are three identical equipments of this type $(B_1, B_2, B_3)$ (see Fig. 1). They are connected to the equipments A by a piping system which forms a network. Each job flows out through this network from an equipment A to one or two equipments B following one of the possible paths of the piping system. Several constraints have to be taken into account:

- No waiting time can exist between the end of the treatment of a job by an equipment A and the beginning of the flowing of this job to the chosen equipment(s) B. So if waiting time cannot be avoided, it must be scheduled before the processing period of a job on an equipment A – before or after the loading period – so that the flowing period towards the equipment(s) B will immediately follow this processing period.
- The flowing period is independent of the type of the job and is constant for each equipment B. Nevertheless if several equipments B are used for the same job, the running time is a decreasing function of the number of used equipments.
- Obviously, when an edge of the network is occupied for the flowing of a job, this edge is blocked and unavailable for other jobs.

So if several equipments B are used simultaneously for the flowing of a job, the flowing duration is reduced but during this shorter time period a larger part of the network is blocked.

(c) The third stage of the production process consists in putting the material into boxes: this is made by an unique machine – of type C – which can be assigned to each equipment B (see Fig. 1). The required time to empty an equipment B is constant. Of course only one equipment can be treated at a time. When an equipment B is empty, it becomes available to receive another job and the machine C can be assigned to another equipment B.

This workshop – more precisely the piping system between equipments A and B – has been modelized by a graph (see Fig. 1). Nodes $A_i (i = 1, \ldots, 6)$, $B_j (j = 1, \ldots, 3)$ and C denote the different equipments of the workshop. To modelize the impossibility of simultaneously using some paths of the piping system for the flowing of several jobs, some additional nodes – denoted by $P_k (k = 1, \ldots, 8)$ have been added to represent particular connection points of the network. The possible paths to flow the jobs from the equipments A to B are represented by the arrows of Fig. 1. Through the analysis of this graph it is then possible, at each instant, to determine the feasible paths for the flowing of a new job.

To illustrate this, let us suppose that on instant $t$, a job is running from equipment $A_2$ to equipment $B_2$ through path $A_2P_2P_3P_4(\rightarrow B_2)$. If on this instant, the processing of a job on $A_4$ is finishing, two feasible paths to run the latter job are for instance:

$A_4P_8P_7P_1$ to reach $B_1$,
$A_4P_8P_6$ to reach $B_3$.

### 1.2. The problem

An order-book of jobs is given, corresponding approximately to a production period of one week. Each job is characterized by

- its type and thus, the duration of processing on a type A equipment,
- a possible ready date, corresponding to the first instant at which the job can be loaded on an equipment A and by a possible due date, corresponding to the expected instant at which it will be ready into the boxes.

Some particular constraints must also be taken into account:

- A ready date can be associated to each equipment corresponding for instance to the instant at which this equipment becomes available by the end of the preceding scheduling period.
- A period of unavailability for any equipment A may be defined by its initial and final instants; it corresponds for instance to a scheduled maintenance operation.

The most important objective formulated by the Management is to enhance productivity by a more intensive utilization of the machines (reduction of idle periods). This objective was translated into makespan minimization. Optimal schedules according to this objective are usually characterized by a well-balanced and intensive utilization of the production lines. Given a list of jobs, it is thus necessary to produce a scheduling of the jobs on the different equipments taking into account:

- the ready dates of the jobs and of the equipments,
- the no-waiting time requirement between the successive processing periods of a job on equipments A and B.

In a second step, the managers want to take into account:

- the possible due dates of the jobs,
- the possible unavailability periods of the equipments.

The first set of constraints will be strictly satisfied; the second will be considered in a more flexible way.

## 2. The algorithm

The method proposed to obtain a satisfactory production schedule is formed of two embedded heuristics:

The first one is a reasonable rule to assign the jobs, in a fixed sequential order to the equipments of types A and B; this scheduling rule is described in Section 2.1.

The second is a metaheuristic like Simulated Annealing or Tabu Search which iteratively alters the ordering of the jobs in the list in view of building a good or optimal schedule w.r.t. makespan; some comments on the application of these procedures are made in Section 2.2.

In the following presentation, we drop some details of the procedure concentrating on the main lines. Some additional remarks are presented in Section 4.

### *2.1. The scheduler*

Let us first suppose that all the jobs are ranked, in a fixed order, into a list L. Each job is characterized by its processing time and possibly by a ready date and/or a due date. At each iteration, another job will be scheduled. The jobs already scheduled have been eliminated from the list L and we know the availability data i.e. the actualized ready date of each equipment (A, B and C) and of the loading team taking into account the current partial schedule. The scheduling of the first not yet scheduled job in the list L is determined in five steps.

#### *2.1.1. Selection of an equipment A*

The equipment A with the earliest availability date is selected.

#### *2.1.2. Determination of the possible processing period*

We consider the first instant, after the earliest availability date of the selected equipment A, at which the loading team is available and we add the loading time of the job to it, obtaining the first instant at which a processing period on equipment A can be initiated for the job under consideration.

#### *2.1.3. Selection of a job*

It is the first job in the list L whose ready date allows for a loading at the beginning of the possible production period just determined.

#### *2.1.4. Rough test of availability of equipment A*

A lower bound for the next availability date of the selected type A equipment is computed. It corresponds to an ideal situation where by the end of the processing period of the selected job, two equipments B are immediately available to flow the product. Nevertheless, at this stage, no check of such availability is made. If this lower bound

(which is also an upper bound for a busy period of the selected equipment if the selected job is assigned to it) is not compatible with a possible unavailability period of the equipment, then this pair "equipment A-job" is rejected: steps (a)(c) are resumed to select another job and/or another equipment A. Otherwise this pair is definitely accepted for this iteration.

**Remark.** It is important to note that this procedure does not insure that an unavailability period of the equipment A will be strictly respected. Indeed it is possible that two equipments B will not be immediately available so that the ideal situation described above will not occur and the flowing period will be delayed. Nevertheless (see 2.2) some penalty will be introduced in the objective function in case of violation of an unavailability period of an equipment; so that this constraint will be taken into consideration in an indirect manner through the objective function.

*2.1.5. Selection of the flowing paths of the material*

An analysis of the graph is performed to determine the possible paths for the flowing of the job (for more details see [10]).

All the solutions using one or two equipments B are compared, taking into account their availability in function of the jobs already scheduled. For each solution, the new date of availability of the selected equipment A is determined. Different selection rules have been implemented and tested. The best one seems to be to select the solution yielding the earliest availability date of the equipment A and in case of tie, to retain the one using the smallest number of equipments B. Then, the availability dates of each selected equipment A, B, P and machine C which empties busy equipments B as soon as possible in function of its availability are updated before a new iteration is performed. The procedure continues until all jobs of the list have been scheduled.

**Remark.** At this stage, there is no verification of the satisfaction of the due date of the job. Again some penalties proportional to the possible tardiness will be considered in the objective function (see Section 2.2) to take into account this particular constraint.

*2.2. The "optimizer"*

The role of the second heuristic is to optimize the order of the jobs in the list L. In view of this, both Simulated Annealing (SA) and Tabu Search (TS) have been implemented. These procedures are now well known and their principles will not be recalled in this paper. The reader is referred to the recent tutorial published by one of us [6] for a detailed presentation of these methods and a selected bibliography.

Given a "scheduler", an ordered list of jobs may be considered a coded version of a schedule which can be restored by using the scheduler. So, once again conditionally to a given scheduler, we may call an ordered list of jobs, a solution and work with the set of ordered lists of jobs as solution space. It is indeed far more convenient to build a SA or TS algorithm on such a space than dealing directly with schedules. For both SA and TS, it is necessary to define a neighbourhood of a solution; in our case, this neighbourhood is the set of all solutions which can be obtained by permutation of two jobs in the current list L. The objective function is the sum of three terms:

- the makespan i.e. the completion time of the latest job,
- a penalty reflecting the violation of the availability period of the equipments A,
- a penalty proportional to the tardiness of the jobs with respect to their due date.

For each new solution (i.e. each new ordered list of jobs), the scheduler – see Section 2.1 – must be applied to obtain the value of the objective function. It is the reason why the scheduling rule must be a very efficient routine, consuming little computing time, because in these metaheuristics, hundreds or thousands of solutions are considered and for each of them the scheduler is called and run.

To apply the Simulated Annealing technique, at each iteration a new solution is chosen randomly in the neighborhood of the current solution, i.e. two jobs are chosen randomly and their positions in the list are exchanged. The parameters of the procedure – the initial temperature, the cooling factor, the number of iterations at fixed temperature, the stopping criterion (a final temperature or a number of successive temperatures with very slow improvement

of the objective function) – must be fixed after some experimentation in a "trial and error" manner; these parameters depend on the size of the problem.

In applying the Tabu Search technique, we define a subneighbourhood of the current solution: it is formed of a fixed number of solutions chosen randomly in the neighbourhood. A Tabu list stores characteristics of the last current solutions; it is used to avoid cycling. At each iteration, the best non-tabu solution in the subneighbourhood is determined and it becomes the new current solution. Note that the tabu status of a solution may be overwritten and the solution accepted if it passes some aspiration level, in fact if the solution furnishes a value of the objective function which is better than those observed since the beginning of the search. The main parameters of the Tabu method are the size of the sub-neighborhood, the length of the tabu list, the aspiration criterion and the stopping criterion, i.e., the number of iterations without any improvement of the objective function. Again, these parameters must be fixed after some experimentations. Usually, and it is the case here, SA as well as TS are not too sensitive to parameter setting; it is thus relatively easy to find parameters which yield good results.

## 3. An instance of the problem and some results

In this section we present
- the description of an instance of the problem and the corresponding data structure (see Section 3.1),
- some results which can be obtained by the method described in the previous section (see Section 3.2).

### 3.1. Describing an instance of the problem

It is necessary to feed the software with two kinds of data:

(1) Some "permanent" information containing technical and stable data:
- a "*configuration file*" describing the workshop: structure of the network, number of equipments A, B, C; connection nodes P; this file also contains the flowing times of a job from an equipment A to the equipments B in the cases where one or two of them are used simultaneously.
- a "*technological file*" giving the different existing types of products and their corresponding processing times.

(2) The data needed to solve a particular problem:
- a "*job file*" describing an order-book.
- a "*status file for the workshop*" specifying the availability of the equipments.

A brief description of the second kind of data is given below. A user-friendly interface under "Windows" helps the production manager to enter the informations about the workshop and the order-book.

#### 3.1.1. The "job file"

The instance of an order-book composed of twenty products is described in Table 1. For each of them, one specifies:

column 1: A name; e.g. prod 1.

Table 1
The "job file"

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| # NRPRODUCTS | | | | | | | |
| 20 | | | | | | | |
| # PRODUCTS | | | | | | | |
| Prod 1 | n.s. | $\alpha$ 20 | 3 | | | | |
| Prod 2 | n.s. | $\alpha$ 37 | 3 | | | | |
| Prod 3 | n.s. | $\alpha$ 34 | 3 | | | | |
| Prod 4 | n.s. | $\alpha$ 31 | 3 | | | | |
| Prod 5 | n.s. | $\beta$ 02 | 3 | | | | |
| Prod 6 | n.s. | $\alpha$ 31 | 3 | | | | |
| Prod 7 | n.s. | $\beta$ 98 | 3 | r | 18/03:05 h | d | 20/03:05 h |
| Prod 8 | n.s. | $\beta$ 32 | 3 | | | | |
| Prod 9 | n.s. | $\alpha$ 34 | 3 | | | | |
| Prod 10 | n.s. | $\alpha$ 31 | 3 | r | 20/03:20 h | d | 22/03:20 h |
| Prod 11 | n.s. | $\beta$ 76 | 3 | | | | |
| Prod 12 | n.s. | $\alpha$ 31 | 3 | r | 19/03:10 h | | |
| Prod 13 | n.s. | $\beta$ 02 | 3 | | | | |
| Prod 14 | n.s. | $\beta$ 76 | 3 | | | | |
| Prod 15 | n.s. | $\beta$ 52 | 3 | | | | |
| Prod 16 | n.s. | $\beta$ 52 | 3 | | | | |
| Prod 17 | n.s. | $\alpha$ 30 | 3 | | | | |
| Prod 18 | n.s. | $\beta$ 33 | 3 | | | | |
| Prod 19 | n.s. | $\beta$ 52 | 3 | | | | |
| Prod 20 | n.s. | $\alpha$ 30 | 3 | | | | |

"n.s." is for "non specified"; "r" for "ready date" and "d" for "due date".

column 2: A possible equipment A on which the product has to be processed (see Section 4). The abbreviation "n.s." stands for "not specified" and means that any equipment A is allowed.

column 3: The type of the product according to a classification of the chemical plant. Knowing product type (e.g. $\alpha$ 20) the duration of the process is furnished by the "nomenclature file".

column 4: The maximum number of equipments B in which the product can flow (see Section 4); for this instance, this number is 3 for all products.

column 5, 6: Possible ready date (preceded by "r") and due date (preceded by "d"); in this instance, there are three ready dates and three due dates and two jobs have both a ready date and a due date.

*3.1.2. The "workshop status file"*

This file is used to specify initial conditions of the workshop and possible programmed periods of maintenance (see Table 2).

1. First of all, the current initial date is intr-oduced. By the way, notice that all dates are given in a rather complete format "day/month: hour" unlike usual format in software which is normalized unit of time starting from "0". Asking the manager to enter the initial date allows to initialize the natural time-scale which is displayed on all graphics and is much more user-friendly for monitoring the process.

2. Then each equipment A is specified:

column 1: The initial date at which the equipment could be used.

column 2, 3: The start and the end of a possible unavailability period for this equipment (e.g. maintenance period of $A_2$ is from 19/03: 17 h till 21/03: 06 h). The

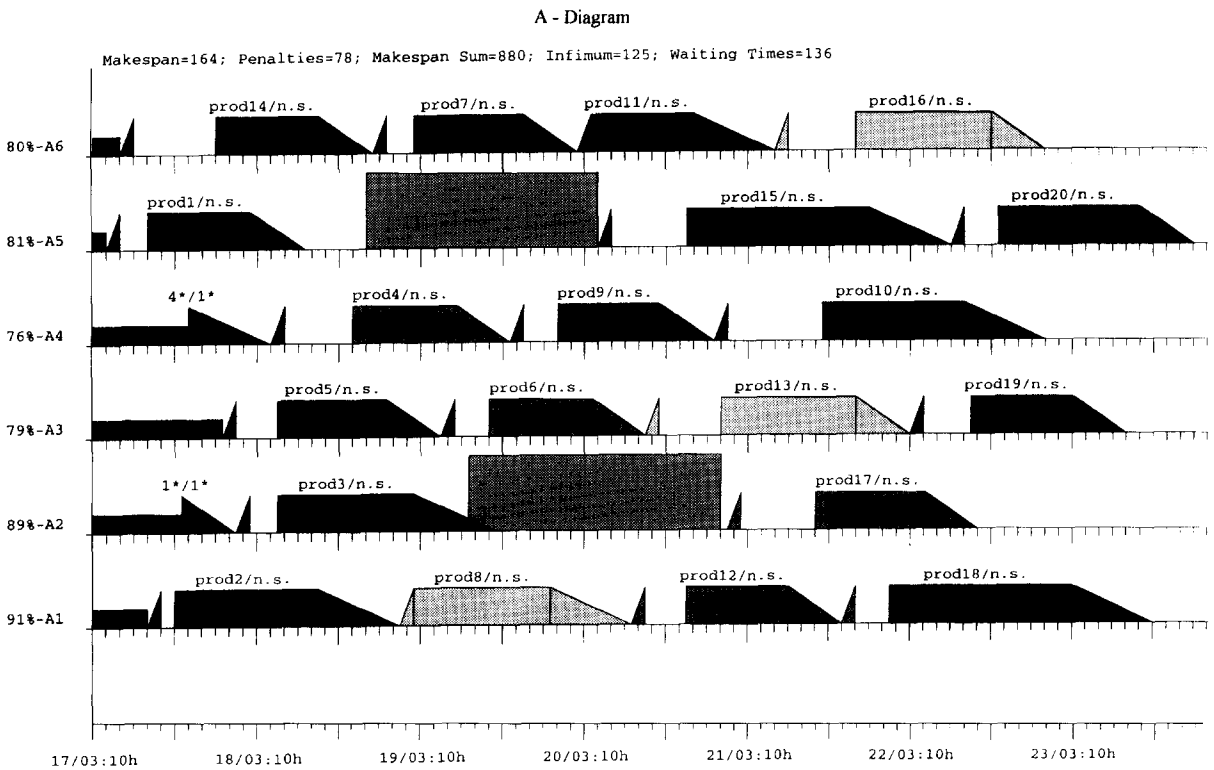


Fig. 2. Initial schedule for the equipments of type A.

Table 2
The "workshop status file"

```
#DATEINIT
17/03/1993:10 h

#CONDINIT A
A1  17/03:18 h                                  e
A2  17/03:23 h   19/03:17 h   21/03:06 h   l  2  prod 11
A3  18/03:05 h                                  e
A4  18/03:10 h                                  l  1  prod 12
A5  17/03:12 h   19/03:02 h   20/03:12 h   e
A6  17/03:14 h                                  e

#CONDINIT B
B1  17/03:12 h
B2  17/03:15 h   21/03:08 h   22/03:04 h
B3  17/03:14 h

#CONDINIT C
C   17/03:12 h
```

"e" is for "empty" and "l" for "loaded".

following columns are only used for equipments of type A: they give information on the initial state of each of them.

column 4: Status of the equipment at the initial date (column 1): e ↦ the equipment is empty; the following columns may be ignored 1 ↦ the equipment is loaded; column 1 must then be interpreted as a "end of processing date".

column 5: Number of equipments B previously computed to which the product is flowing.

column 6: The name of the product loaded on the equipment A.

3. Similar information is given for the different equipments B.

4. Similar information is given for the machine C. Let us remark that it is possible to introduce some unavailability periods for such tools (see Section 4).

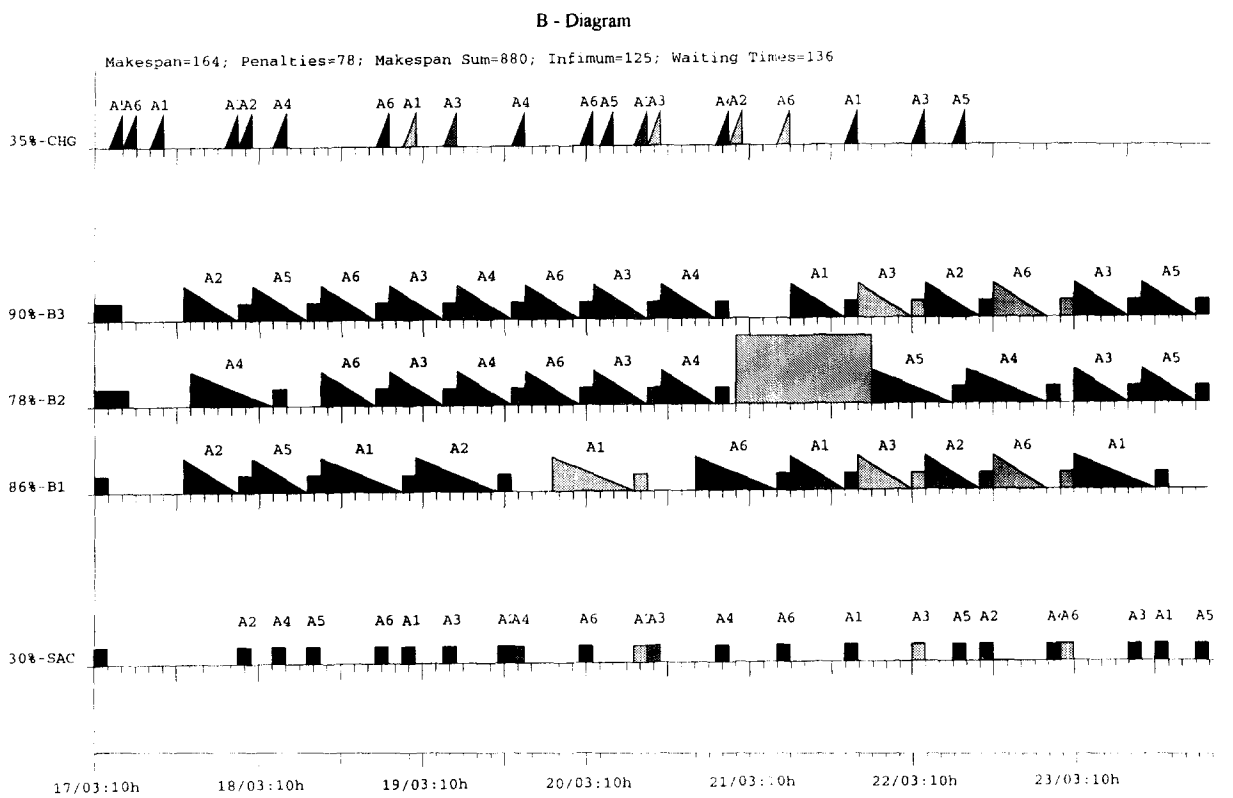


Fig. 3. Initial schedule for all equipments but type A.

### 3.2. The results

As explained in Section 2, the objective function to minimize is composed of two terms:
- the makespan: the unit of time is an hour
- the penalties: they are proportional to the duration of the violation of the unavailability periods of the equipment (in this instance, this penalty coefficient is chosen equal to ten) and to the lateness of the jobs with respect to their due date (in this instance this penalty coefficient is chosen equal to one).

For comparison purposes, we first show the schedule obtained by applying the scheduler to the initial ordering of the jobs (as given in the "job file"): then, an example of a final schedule yielded by the optimizer allows to observe the improvement obtained by using it.

#### 3.2.1. The scheduler

The results of the initial application of this dispatching heuristic are presented in the diagrams of Figs. 2 and 3. As indicated at the top of these diagrams:
- the makespan is equal to 164 (hours),
- the total penalty is equal to 78,
- the sum of the completion time of the six equipments A is equal to 880 (hours),
- the "waiting time" – i.e. the global time of non busy periods of the equipments A – is equal to 136 (hours).

Let us now present the keys needed to read these two diagrams.

"*A-diagram*" *(fig. 2).* Each line corresponds to one of the six equipments A:
- Large-height rectangles represent the maintenance periods (see equipments $A_2$ and $A_5$).
- Small-height rectangles, at the beginning of each line, correspond to the initial conditions as described in the "status file of the workshop".
- A job is represented by the three successive symbols:
  - a "left triangle" stands for the constant loading time (loading of the material into the equipment A),

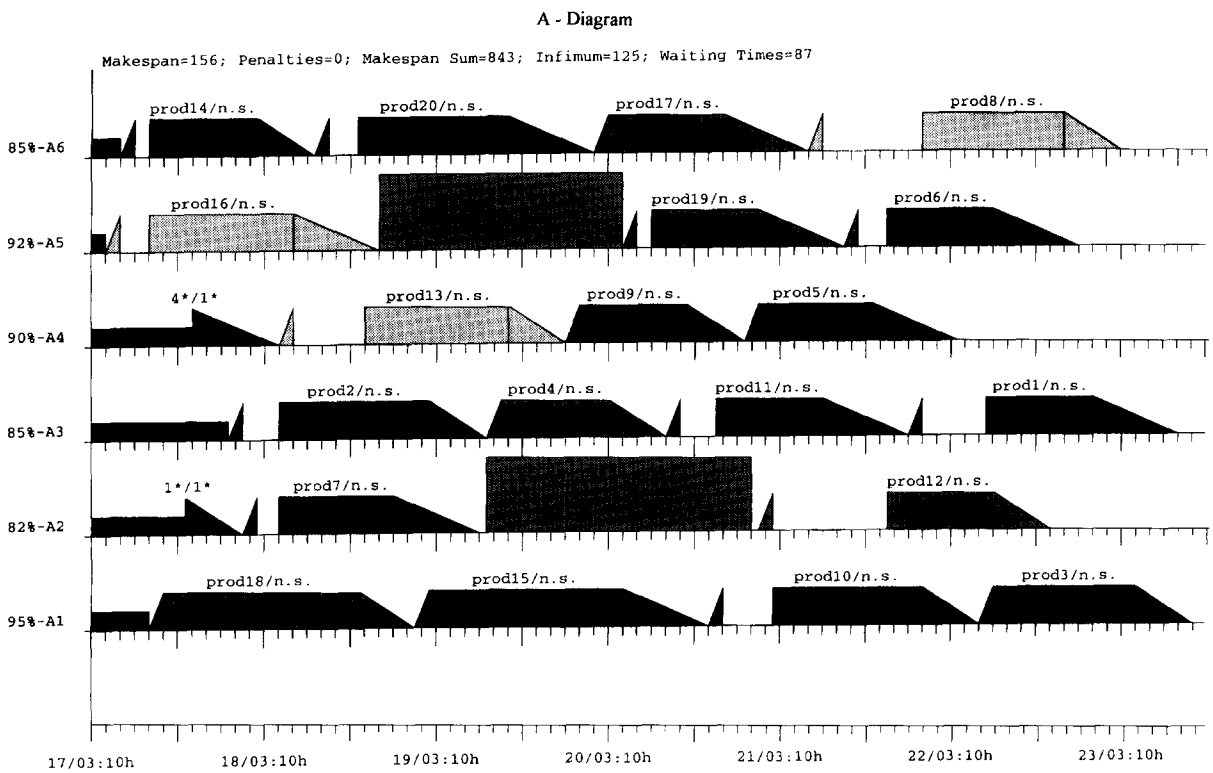


Fig. 4. Schedule for equipments of type A optimized by Simulated Annealing.

- a medium-height rectangle stands for the processing time of the job (the name of the job is written above this rectangle),
- a "right triangle" is for the flowing process into the equipments B. Note that the basis length of this triangle indicates indirectly the number of equipments B used for this emptying process (decreasing time with the number of equipments – 1, 2 or 3 - used).

*B-diagram (fig. 3)*

- The first line is concerned with the special team of workers preparing the equipments A: the different loading periods are labelled by the name of the concerned equipment.
- Lines 2 till 5 correspond to the use of the three equipments B. The symbols displayed are the same as those in Fig. 2; small squares are used to represent the constant periods needed by machine C to empty the equipments B.
- Line 6 describes the use of machine C.

In the left margin of both diagrams, the "utilization rate" of each equipment (equipments A in the first diagram; loading team, equipments B and C in the second) are displayed: these rates are computed by:

$$\left(1-\frac{\text{waiting time of equipment}}{\text{completion time of the equipment}}\right)\times 100.$$

Clearly, this schedule can be improved:

- the global penalty is rather high because
  - job "prod 3" is processed on $A_2$ during the beginning of the maintenance period of this equipment,
  - the two jobs with due date (see the "job file") induced penalty due to some lateness,
- the waiting time is important,
- the difference between the makespan and the minimal completion time seems too large. The work charge is quite unbalanced between the equipments A:
  - "$A_2$" is empty a long time before "$A_5$".

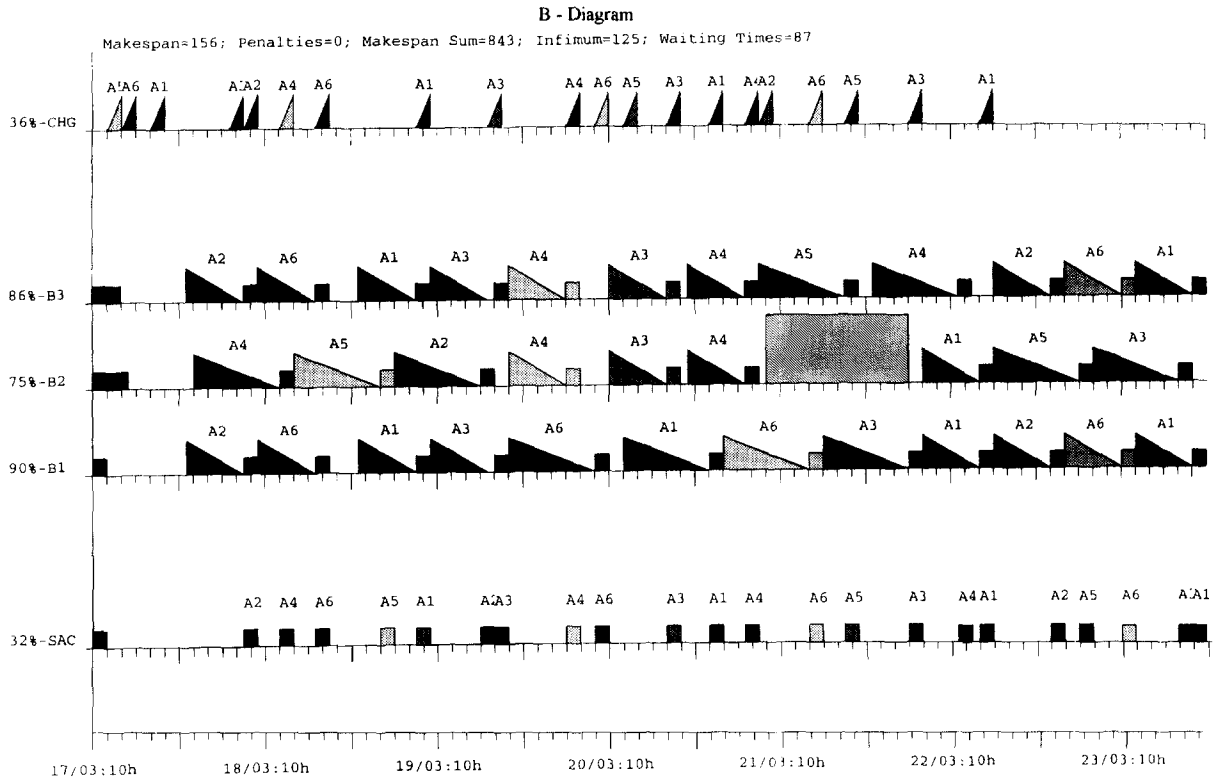


Fig. 5. Schedule for all equipments but type A optimized by Simulated Annealing.

*3.2.2. The optimizer*

As explained in Section 2, either Simulated Annealing (SA) or Tabu Search (TS) are used to optimize the order in which the list of jobs is considered by the scheduler. For both methods, it is necessary to do several experiments to fix the best values of the parameters. These values must be adapted to the size of the treated instance. We do not present in this paper the complete analysis realized to implement these two metaheuristics and to compare their performance. This extensive study will be reported in a specific paper in preparation [9].

We just present here some partial results in order to illustrate the principle of the method. The diagrams of Figs. 4 and 5 on one hand, Figs. 6 and 7 on the other hand, represent results obtained, respectively, with SA and TS. As shown in these figures, if the schedules obtained are different, the performances are equivalent. Both the makespan and the total penalty are drastically reduced:

- the makespan is equal to 156 both with SA and with TS,
- the penalty is equal to 0 both with SA and with TS.

The unavailability periods of the equipments and the due dates of the jobs are now respected. Consequently, the "waiting time" and the sum of the completion times are also improved. It results from deeper comparison – on different instances – presented in [9] that the performances of both metaheuristics can be considered as equivalent. However, TS appears to be less time consuming for this problem. For this instance, the results were obtained after 2 min for TS, 3 min for SA on a 486 Intel processor.

## 4. Conclusions

The flexible job-shop problem described in Section 1 is clearly a $\mathcal{NP}$-hard problem. Attempts

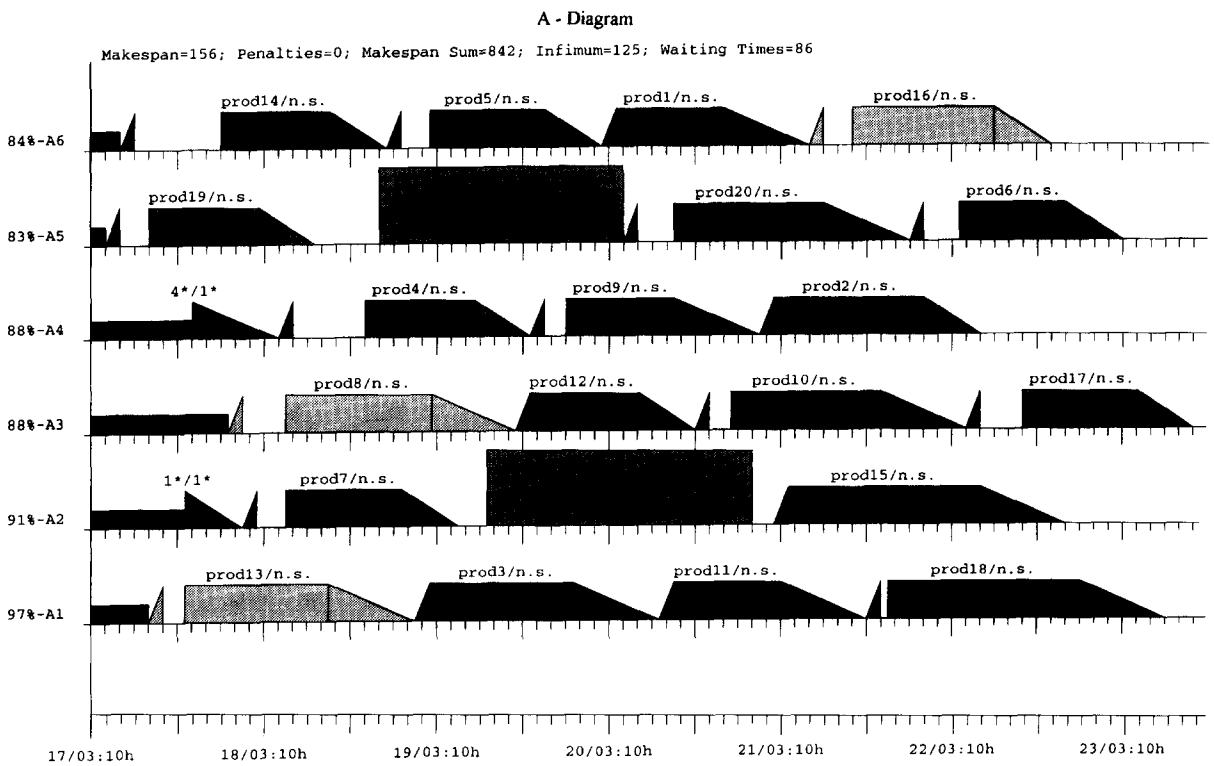


Fig. 6. Schedule for equipments of type A optimized by Tabu Search.

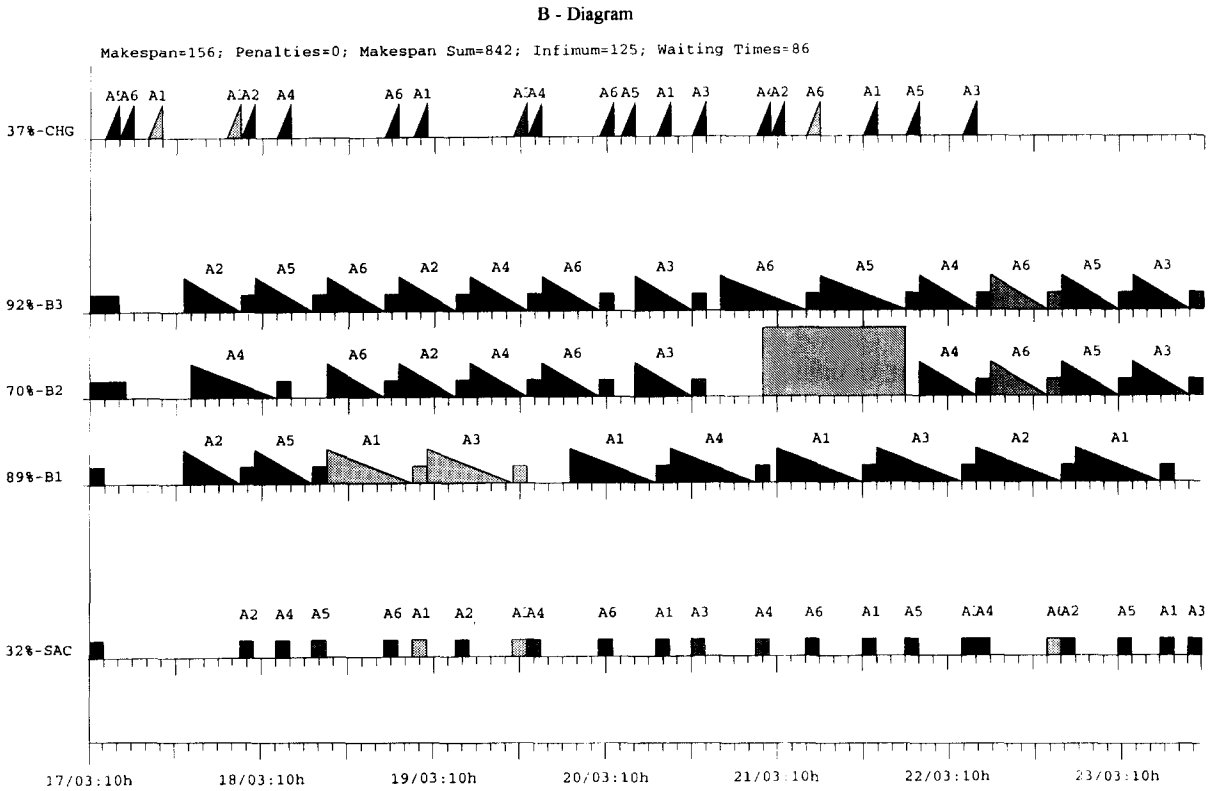


Fig. 7. Schedule for all equipments but type A optimized by Tabu Search.

could be made to write mathematical formulations into combinatorial optimization models. This could probably be done in different manners, all leading to very complex mathematical models: it is even doubtful that all subtle constraints could be modelled. In any case such models would almost certainly be intractable by classical methods like Branch and Bound or even polyhedral approaches.

The results presented in Section 3 prove clearly that our 2-level heuristic – a combination of a scheduler and an optimizer based on metaheuristics – is a powerful approach from the point of view of either quality of the solutions or computation time.

Moreover this method can easily be adapted to many different situations of complex flexible jobshop. In particular, the 2-level architecture and the use of "priority lists" offer many possibilities: only the scheduler has to be re-designed in order to build schedules which both satisfy the specific constraints and are of good quality w.r.t. the managers objectives.

The problem described in Section 1 is in fact a simplified version of the real chemical workshop; several additional particularities are taken into consideration in the software designed in the framework of the contract. In particular, let us mention some of them:

- For technical reasons it can be specified that a job must necessarily be loaded on a particular equipment A.
- Sometimes it is also specified that a job must flow to less than a given number of equipments B or to a fixed number of them.
- Some unavailability periods can be introduced for all the equipments, not only of type A but also of type B (see the instance of Section 3) or C.

- In certain circumstances, simultaneous use of some equipments B can be forced for the flowing of the jobs.

The software also offers several facilities to the user; in particular:

- The scheduler can be used alone like a simulation tool. In fact, several heuristics – based on different logical rules – are developed in the scheduler and the user can test any of them.
- An updating module has been designed: it offers to the user the possibility to introduce new jobs in the order-book without changing the whole schedule. Similarly, a schedule can be "frozen" till a certain date and a new optimization performed – possibly with additional jobs – for the schedule from this date on.

Moreover, the "Windows" environment gives the opportunity to integrate all these facilities in a very flexible user-friendly package, perfectly adapted to the specificities of the workshop, which is certainly not the case for the common commercial packages proposed for these types of scheduling problems.

From the point of view of the company, this software is a precious auxiliary tool both for the operational management of the workshop and also for higher level management purposes. It does not only allow to increase productivity by reducing idle times (which was the initial motivation of the firm); it also offers simulation capabilities. It allows indeed to evaluate the gains in productivity which would result of changes in the topology of the workshop. These capabilities may be extremely precious to document a more strategic decision process and this was highly appreciated by the firm.

In conclusion, it appears once again that metaheuristics are a powerful tool. In particular this study shows their capabilities to optimize complex flexible job-shop problems. They really open new perspectives in the field of computerized and automatic production scheduling.

## Acknowledgements

We warmly thank three anonymous referees for their constructive suggestions which helped us to prepare the revised version of the paper.

## References

[1] Garey, M.R. and Johnson, D.S., 1979. Computers and Intractability. A Guide to the Theory of NP-Completeness. Freeman and Co., New York.

[2] French, S., 1982. Sequencing and Scheduling. Wiley, New York.

[3] van Laarhoven P.J.M. and Aarts E.H.L., 1987. Simulated Annealing: Theory and Applications. Reidel, Dordrecht.

[4] Glover, F., Laguna, M., Taillard, E. and de Werra, D. (Eds.), 1993. Tabu Search, Ann. Oper. Res., 41.

[5] Goldberg, D., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, New York.

[6] Pirlot, M., 1992. General local search heuristics in combinatorial optimization: A tutorial, JORBEL, 32: 7–67.

[7] Falkenauer, E and Bouffouix, S. 1991. A genetic algorithm for jobshop. Proc. of the 1991 IEEE Internat. Conf. on Robotics and Automation.

[8] Della Croce. F., Tadei, R. and Volta, G., 1995. A genetic algorithm for the jobshop problem, Comput. Oper. Res., 22: 15–24.

[9] Fortemps, Ph., Ost, Ch., Pirlot, M. Teghem, J. and Tuyttens, D., Comparison of simulated annealing and tabu search on a flexible job-shop problem, Technical Report in preparation. Laboratory of mathematics and operational research. F.P.Ms. Belgium.

[10] Ost, Ch., 1993. Hierarchical planning in a two stages production process. Technical Report 93.01. Laboratory of mathematics and operational research. F.P.Ms. Belgium.