Evolutionary Design of Time-Way Charts for Plating Machines

Georges E. Matile
STS Industrie S.A.
Le Mont-sur-Lausanne
Switzerland
gmatile@vtx.ch

Andrea G. B. Tettamanzi
Università degli Studi di Milano, Polo
Didattico e di Ricerca di Crema
Via Bramante 65, I-26013 Crema (CR), Italy
tettaman@dsi.unimi.it

Marco Tomassini
Université de Lausanne
Institut d'informatique
1015 Lausanne, Switzerland
marco.tomassini@iismail.unil.ch

Abstract- Scheduling the production for plating machines is a tedious and difficult task of critical importance for the economic exploitation of this equipment. This paper describes a promising approach to solving a simple version of this problem, namely cyclical hoist scheduling, based on evolutionary algorithms. The issues of solution encoding and specialised genetic operators are discussed and some preliminary results are presented.

1 Introduction

A plating machine is one which is designed to cover parts with a layer of a precious or base material, using chemical or electrochemical processes. Typical examples are zinc, for the protection of iron parts, copper + tin for printed circuits boards, copper, nickel and precious metals in the decorative industry.

The equipment consists of a set of tanks containing the different chemical or electrochemical solutions (as well as rinsing positions), rectifiers for electrolytic treatments, and hoists (conveyors) to transfer the parts to be treated from one tank to another (see figure 1).

The parts must follow a given chemical sequence, which is generally not the same as the physical position of the tanks in the machine. In this work the tank positions are considered as fixed. However, they could also be co-optimised together with the time-way chart. This is a difficult problem, as the chemical production constraints must be respected and will not be dealt with in this paper. The immersion time in every position is also defined, within certain tolerances.

If the production always follows the same program (a program is a chemical sequence associated with process times), the best way to obtain the best performances regarding the productivity of the machine, with a strict repetition of all immersion times (especially important with respect to the ISO9000 standards) is to determine a cyclical hoist scheduling, called "time-way chart". To our knowledge there is very little published work in this field. References [1, 3] are concerned with electroplating of printed circuit boards but the Authors employ different methodologies. References [2, 4, 6, 7] give a treatment of the present problem by using constraint logic programming.

This subject presents some correlation with the "job shop scheduling problem", but the constraints are much more critical:

- greater importance of transport time, as well as greater restrictions due to the transport tool
- no intermediate buffer possible
- the "resting time" of parts in a given place must strictly remain inside given tolerances.

The problem is difficult to solve in a mathematical way, as most of the rules concerning the immersion times are not Boolean, but often follow fuzzy constraints.

The theoretical number of possibilities is the factorial of the number of steps in the sequence; this means that a 20 step sequence corresponds to $2.43 * 10^{18}$ possibilities.

The method generally used to create a time-way chart consists of drawing different propositions on a sheet of paper, with or without the help of a computer, and modifying them to obtain more or less satisfactory propositions. The experience of the designer is the most important factor of success.

Genetic algorithms [5,8] have been shown to be effective when the search space is very large and there are no easy deterministic algorithms to explore it. Also, with many mathematical methods there is the risk of being stuck in a local minimum while stochastic approaches such as genetic algorithms or simulated annealing are able to escape from such local minima.

The interest of using genetic algorithms in such a case is the following:

- the user is not looking for the optimal solution, but only for one among the best solutions
- to be useful, a good solution must be found in reasonable time (in a matter of minutes)
- it is very easy to introduce any fuzzy or expert rules in the calculation of the fitness, as it is easy to analyse the quality of a proposition

Figure 1. Example of a typical plating machine for the electronic industry

This paper is organised in the following way: The next section describes the transferring process by using a graphical tool called a *Time-way chart*. The following section formally introduces the optimisation problem and the genetic algorithm representation and setting. Next we describe the simulation experiments and the results we obtained. Finally, we draw our conclusions and hint at further work that should be done in the future.

2 Time-Way Charts

Figure 2 shows a typical example of a time-way chart using one hoist. The following information can be retrieved from this diagram where the horizontal axis represents time in minutes, and the vertical axis represents distance and machine positions.

Oblique lines show the displacement of the hoist. The slope is a function of the distance to cover and the speed of the hoist.

Short horizontal lines represent vertical movements to extract or insert the load from or into the tank.

A thin horizontal line represents a waiting time for the hoist. Such a time following an "UP" movement, means a drainage time to reduce the drag-out.

A thick horizontal line shows the immersion time in a position. At any time, the observation of the thick lines shows the occupation of positions.

Two types of movements can be defined:

a) Active movements, allowing the transfer of a load from one position to another. During the transfer, the lift system of the hoist is in the upper position.

b) Idle movements, when the hoist must move to the next position requiring a transfer. The lift system of the hoist is in the lower position. Idle movements are drawn as dotted lines.

The conveyor and the plating line have to be designed so as to allow a loaded hoist, in upper position, to transfer a load over already occupied positions. A hoist must also be able to move, in its lower position, over occupied tanks.

2.1 Multiple positions

If an immersion time must be larger than the expected cycle time, more than one position will be required for the treatment. During every cycle, a different position will be unloaded, then reloaded. In the diagram shown in Fig. 2, the copper tank includes 2 positions (24 + 25). This allows an immersion time of up to 1750 seconds for a cycle time of 900 seconds.

It is important to understand that, for a given sequence, all active movements will be executed at each cycle. The immersion times in the different positions are the consequence of the order in which these movements are executed, as well as possible waiting times between transfers. This means that, if we give a symbol for every transfer (1 or the first transfer, 2 for the next one etc.), the time-way chart can be represented by a sequence of numbers, from 1 to the number defining the last transfer, placed in an order to be defined.

Once a sequence is defined, it is not difficult to analyse the corresponding time-way chart, introduce waiting times to the best possible places and calculate a number defining the quality of the proposition. (Fitness)

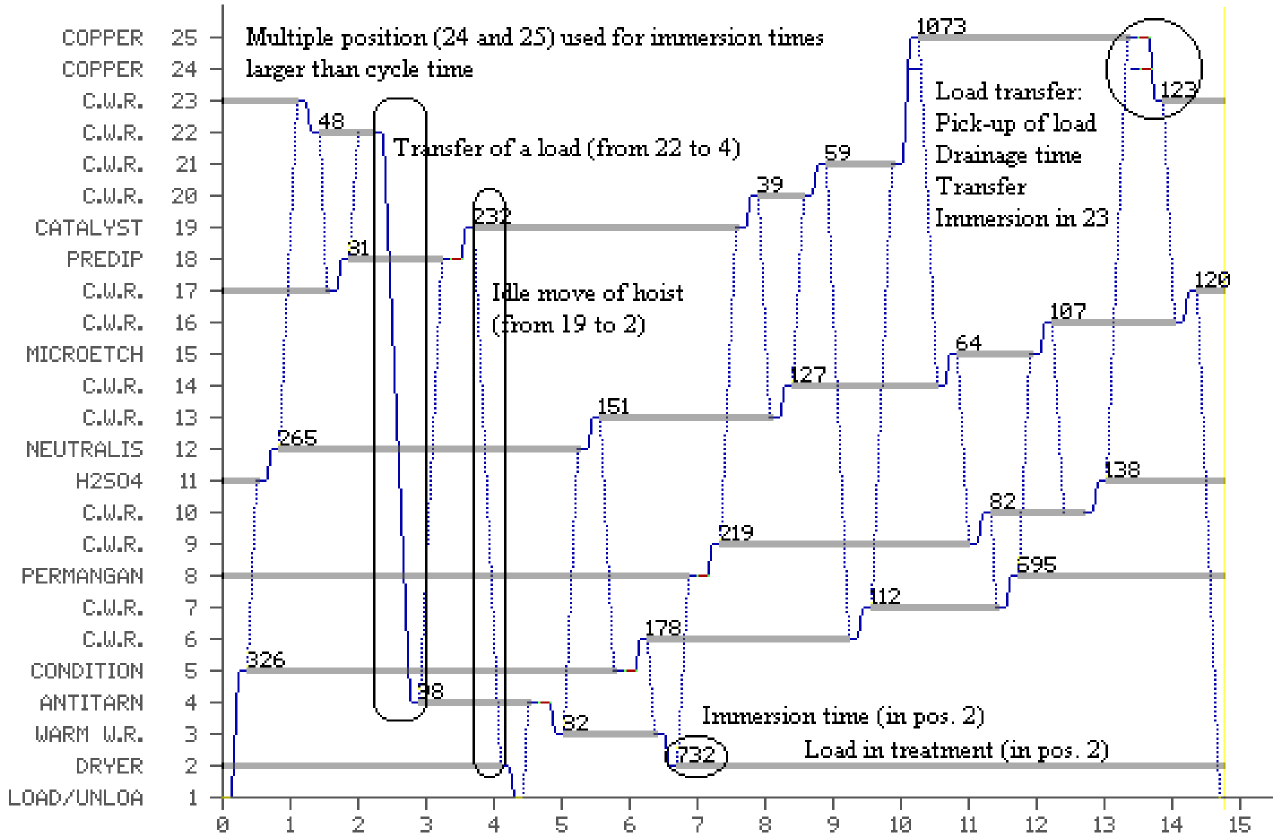

Figure 2. A typical time-way chart for a chemical copper plate for printed circuit boards.

3 Problem Definition

Given:

- A plating machine with M tanks (or sets of equivalent tanks called *groups*) and one or more hoists
- A well defined chemical sequence. Parts have to be treated successively in different tanks. A chemical sequence can be represented as a permutation of tanks (or groups) $\{c_1, c_2, ..., c_n\}$; correspondingly, the first transfer will be from the buffer (or from the loading station) to c_1, the i-th transfer, with $1 < i <= n$, is from tank $c_{\{i-1\}}$ to tank c_i, and the $(n + 1)$-th transfer is from c_n to the buffer (or to the unloading station), for a total of $m = n + 1$ transfers.
- Desired immersion times with more or less fuzzy tolerance constraints: for the purpose of this work, it is assumed that a minimum and a maximum immersion time, t_i^{min} and t_i^{max} are given for each treatment.

Objective: The best possible production of the machine respecting the constraints.

3.1 Example

The following example is an extremely simplified one and is included for the sake of illustration and clarity.

Plating machine : one load / unload position and 5 process tanks, organised in a way to respect the chemical constraints of proximity and drainage. The lay-out of the machine is as follow:

Position 1	Load / Unload
Position 2	Treatment 5
Position 3	Treatment 4
Position 4	Treatment 1
Position 5	Treatment 2
Position 6	Treatment 3

Chemical sequence and immersion times are defined in table 1.

Step Nb	Position	Drainage time	Treatment	Min imm. time	Max imm. Time	Rem
1	Load / unload	0	Loading	120 s	not limited	*
2	4	0	Treatment 1	80 s	100 s	
3	5	0	Treatment 2	10 s	10 s	
4	6	10 s	Treatment 3	80 s	100 s	
5	3	0	Treatment 4	100 s	120 s	
6	2	0	Treatment 5	120 s	140 s	
7	Load/Unload	0	Unloading			*

Table 1. Chemical sequence and immersion times used in the example.
*: 120 s represents the minimum time necessary to unload the treated parts and to load the next untreated parts on a carrier.

The following 3 time-way charts show different propositions:

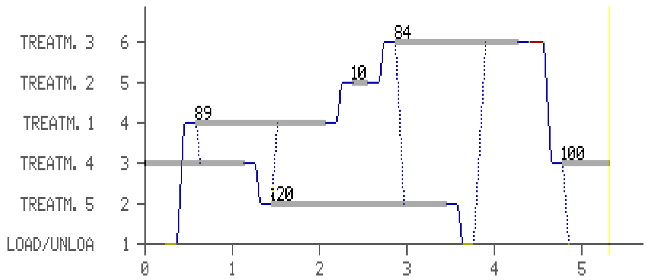

Figure 3. Long cycle

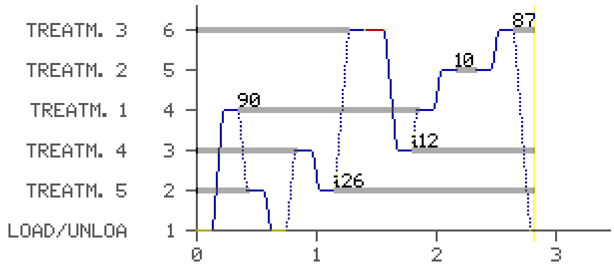

Figure 5. Good time-way chart

Figure 3 shows a time-way chart respecting all immersion times, but with a very poor output. The hoist is waiting more than 50 % of the time.

Figure 5 shows a good time-way chart. All immersion time constraints are respected and the output is good, but is it not possible to find a better proposition?

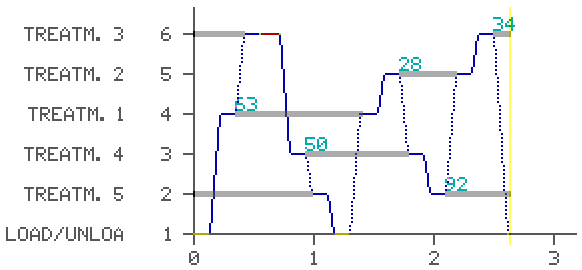

Figure 4. Incorrect immersion times

Figure 4 shows a diagram where the hoists are continuously in operation. The production is very high, but the immersion time constraints are not respected.

3.2 Limit of Production

The production can be limited by two factors:

1.- Tank occupation: the cycle time T cannot be shorter than the longest immersion time of a single position tank, plus the necessary time to load and to unload the position.

$$T > \max_{1<=i<=n} \{t_i^{min}\}$$

2.- Hoist constraints: the addition of all active movements, plus some time for the necessary idle moves and waiting times, gives also a production limit.

If the hoist constraints are few compared to the tank constraints, it is generally easy to find a good time-way chart, as a lot of different possibilities will allow the immersion time constraints to be respected.

In the other case, only a few possibilities (if any) will do it; if it is not possible to find a solution, either a hoist must be added to the machine, or some changes have to be made to immersion times, or the throughput has to be reduced.

In such a case, the development of a time-way chart can be very time consuming.

4 Evolutionary algorithm setting

4.1 Representation and Decoding of Solutions

As an approximation, it can be considered that the time-way chart is divided into *time units.* One time unit is the average time to make a transfer plus an idle move.

The genome is an array of N integers (encoded as characters) (t_1, t_2, ..., t_n). The number of genes in the array is equal to the number of steps in the sequence. Thus, the length of the genome is fixed for a given problem.

Each cell in the array corresponds to the tank from which a transfer begins, and the value of t_i is interpreted as the immersion time, in time units, in that tank. A representation of a time-way chart can be obtained from the genome, whereby the chart is represented as an array of integers, each corresponding to the corresponding number of the step of the sequence. If a time unit corresponds to a non-zero integer, then that integer gives the transfer that is supposed to happen at that time.

Due to conflictual situations (see below), it is usually not possible to convert directly the genome into a time-way chart; the genome must be modified in order to obtain a feasible chart. This means that different genomes can give the same time-way chart.

The decoding algorithm proceeds as follow:

1.- In the beginning, the array is filled with zeros, corresponding to waiting times. In order to increase the chances of finding an acceptable proposition, analysis is started by adding to the number of steps some time units (representing 10 % to 15 % of the number of steps, coming from practical experience), corresponding to waiting times for the hoist.

2. The order in which the transfers will be executed is created according to the genome. The first transfer (1) is put in the first cell of the array. The second transfer (2) is put t_2 cells after the cell just filled, t_2 being the number of time units, given from the genome, corresponding to the immersion time in the first tank.

The reader should notice that t_1 is the number of time units of the last treatment.

The array is considered as circular, the first cell following the last one. It is usually not possible to respect all times defined in the genome, as some transfers would have to be simultaneous. In this case, the transfer is pushed (or pulled) to the closest free cell, without caring whether the corresponding immersion time would respect the tolerances.

The addition of some idle waiting times does not create a real problem since, during fitness calculation, the algorithm tries to reduce all waiting times as much as possible. Fitness calculation penalises any violations of the immersion time tolerances, as well as of the cycle time.

Example

Genome: (12, 11, 7, 6, 11, 11, 12, 3, ...)

$t_1^{min} = 10$	$t_1^{max} = 19$ time units	$t_1 = 12$ units.
$t_2^{min} = 10$	$t_2^{max} = 13$ time units	$t_2 = 11$ units.
$t_3^{min} = 5$	$t_3^{max} = 10$ time units	$t_3 = 7$ units.
	Etc.	

1 means a very short immersion (e.g. 1 sec)
Number of steps in the sequence: 20

Empty time-way chart representation (including 3 additional time units) with the first transfer in the beginning:
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Second transfer after 11 time units:
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Third transfer after 7 time units:
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0)
Up to this point all transfers are possible:
(1, 0, 0, 4, 6, 0, 0, 0, 0, 0, 0, 2, 0, 0, 5, 0, 7, 0, 0, 3, 0, 0)
The next transfer would come at the point marked with *:
(1, 0, 0, 4, 6, 0, 0, 0, 0, 0, 0, 2, 0, 0, 5, 0, 7, 0, 0, 3*, 0, 0)
It has to be shifted the cell immediately after:
(1, 0, 0, 4, 6, 0, 0, 0, 0, 0, 0, 2, 0, 0, 5, 0, 7, 0, 0, 3, 8, 0)
The final representation of the time-way chart is:
(1, 12, 18, 4, 6, 20, 0, 0, 0, 16, 19, 17, 2, 11, 14, 5, 12, 7, 9, 15, 3, 8, 10)

4.2 Initialisation

The easiest way to initialise the algorithm would be to start with a purely random organisation of the transfers (initial population). The experience has shown that it is very difficult to obtain a good individual from a chaotic population.

As it is not difficult to restrict the parameters of the initial population to eliminate most of the unfeasible propositions, the initial population was selected in order to have a certain coherence with the desired results.

The number of possibilities in a coherent population is greatly reduced compared to a fully randomly selected one (from about 10^{18} to 10^{15} for a typical 20 steps sequence)

Different ways were used to make this first selection; the one that has given up to now the best results works on the following principle:

- The immersion times are translated into a minimum and a maximum number of time units, according to the corresponding tolerances.
- In order to avoid the risk of elimination of possible propositions, the number of minimum units is calculated with a "long" time unit, and the number of maximum units with a "short" time unit. (Long time units correspond to long tranfer distances and/or long waiting times for drainage. Short time units correspond to short transfer distances without any drainage).

Of course, as the sequence of transfers is not an exact translation of the genome, some different genomes may create identical transfer chains (duplication).

As the initial population follows certain rules, it can be considered that the population evolves in a subspace of the whole feasible space.

4.3 Crossover

We considered two ways of selecting parents individuals for crossover:

- 1.- Random selection of the parents inside the population
- 2.- Probability of selection proportional to the individual fitness.

Method 2 has lead to a quicker improvement of the best individual, but with a higher risk of being locked on a local minimum. For this reason, method 1 was preferred.

As for the crossover operation itself, we employed standard one-point crossover by randomly choosing a crossover point in both individuals respecting the boundaries between genes (i.e. the integers representing immersion times cannot be split). In many cases the offspring, when decoded into a time-way chart do not give feasible phenotypes. The algorithm employ a “repair” procedure to ensure that valid diagrams are obtained. Every individual participates in the birth of two children that compete with their parents to enter the new population (see “selection” below).

4.4 Mutation

Mutation is done on a fixed percentage of children, by changing the number of time units in one position randomly selected, before decoding.

4.5 Selection

Fitness proportionate selection with elitism is employed, in the sense that a few (namely 5) of the best individuals are always passed on unchanged to the next generation.

4.6 Fitness Calculation

To determine the fitness, two parameters are important:

- Respect of the immersion times.
- Cycle time as short as possible.

The cycle time constraint allows the selection of the best individuals inside acceptable ones, but the non respect of immersion times must be much more penalised than a poor cycle time.

A time-way chart is first designed from the direct representation; then waiting times are added or reduced in optimised places in order to improve immersion times.

The immersion time fitness is the sum of the square of the time errors (in seconds).

Then the cycle time penalises the fitness as follow:

- Calculation of the real cycle time.

- Calculation of a theoretical optimum of the cycle time, by adding the time of all active movements, plus 20% for idle moves and waiting times.

- Cycle time fitness = square of the difference between real and theoretical cycle time.

- A variable weight factor is given to the cycle time fitness, in order to give to the user the possibility to give more importance to a strict respect of immersion times or to the highest possible level of production

The global fitness is the sum of the immersion time fitness and the cycle time fitness.

5 Experiments and results

A number of experiments have been carried out, and the results obtained were positively judged by experts for the examples studied (sequences of 20 steps). The average quality of the results is slightly better than what has been obtained with previous methodologies for that class of problems. In addition, those results were arrived at in a negligible amount of time while it would have taken many hours to attain the same quality with older methods. The main difficulty is due to a very high number of local minima, which increases the risk of concentrating the whole population around such values. Depending on the time constraints, these minima could be broad enough to eliminate from the population all individuals which are not part of the group. It has been shown that, as soon as a group inside a population has a certain importance, its character takes over better individuals (externally introduced in the population), which are usually destroyed.

The best way to reduce this risk is to slow down the improvement of the population.

5.1 Validation

Since for most real problems the optimal solution is not known, it is difficult or impossible, in general, to assess whether the solution found by the algorithm is optimal or it lies within a given distance from the optimum.

In order to provide a validation of the algorithm in this sense, the following test was performed, whereby a problem is constructed from its optimal solution, allowing therefore a direct assessment of the results obtained :

1.- Draw an optimal time-way chart of 20 steps, without time constraints.

2.- Introduce in the system immersion times coming from this chart, with enough tolerances to have a representative number of possibilities (approx. 10^{12}).

3.- The question is asked whether the application is able to regenerate the original time-way chart.

In 30% of the cases, we did obtain the original time-way chart, under the following conditions:

- Population: 250 individuals
- Number of generations: limited to 150 (due to convergence phenomena, no significant improvement of the best individual were shown after 150 generations).

The original time-way chart was obtained after 100 to 150 generations

As expected, we never obtained a better time-way chart then the original one.

In the other cases, the system was concentrating on different local minima.

Usually the proposed solutions, even though not optimal, were anyway acceptable for a practical application.

6 Discussion and Future Work

Some practical considerations were also introduced in the algorithm, such as

- Use of multiple positions
- Multiple use of one position

The advantage of using a soft computing method to generate propositions is the fact that practical limitations, such as the ones mentioned above, can be treated just by changing the fitness calculation algorithm. If a single tank is used more than one time in the sequence, the crossover could propose a solution with the tank occupied with two loads (not possible). It is important not to eliminate this proposition, which could drive to a good and possible one just by crossing two transfers. It is sufficient to penalise it, but not too much in order to avoid its systematic destruction.

An interesting practical case is the possibility to adapt the fitness parameters to the type of industry using plating machines; giving more importance to the cycle time than to the strict respect of immersion times drives the system to find a different time-way chart, more productive, but slightly less performing regarding immersion time constraint.

In order to have a tool really usable in this industry, the extension to multiple hoists must be done. Problems associated with multiple hoists machines are:

- selection of the best hoist to do every job.
- collision constraints analysis.

Another problem is linked to non cyclical time-way charts. If the loads must follow different sequences, more or less randomly, it is not possible to use a cyclical time-way chart. Hoists movements must be calculated in real time, as well as the correct selection of the load allowing the best productivity and its possible introduction time in the machine.

Actually different systems are used in this case, but every system has some weaknesses:

- Either no insurance that all immersion times will be respected
- or no insurance that the best productivity is achieved.

A soft computing method could also be an interesting approach to solve such problems.

Bibliography

1 - Yih, Y., Yin, N.: An algorithm for crane scheduling in flexible circuit board electroplating lines, in Proceedings of the Symposium on Manufacturing Aspects in Electronic Packaging, 1992 ASME Winter annual meeting, Anaheim, pp 19-25, 1992.

2 - Baptiste P., Legeard B., Manier M.A and Varnier C. : Optimization with constraint logic programming: the hoist scheduling problem solved with various solvers, in Proceedings of Applications of Artificial Intelligence Engineering (AIENG93), Toulouse, Vol. 2 , Elsevier, pp 599-614, 1993.

3 - Yih, Y. , Chiu,C.: Incremental learning for hoist scheduling problem in circuit board electroplating lines, in Proceedings of the Symposium on Manufacturing Aspects in

Electronic Packaging, 1993 ASME Winter annual meeting,New Orleans, pp119-124, 1993.

4 - Baptiste P., Legeard B., Manier M.A. , Varnier C. : A scheduling problem optimization solved with Constraint Logic Programming, in Proceedings of the Second International Conference on the practical application of Prolog (ICPAP), London, pp 47-66, 1994.

5 - Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, 1989.

6 - Manier M.A., Varnier C., Baptiste P.: A Multi-Hoists Scheduling Problem Approach, in Proceedings of the Fourth International Workshop on Project Management and Scheduling, Leuven, 1994.

7 - Bachelu A., Baptiste P., Varnier C., Boucher E., Legeard B.: Multi-criteria comparison between algorithmic constraint logic and specific constraint programming on a real scheduling problem, in Proc. Of the 3rd Int. Conf. on the Practical Application of Constraint Technology, London ,pp 47-53, 1997.

8.- Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, 1992.