Universidade Federal de Santa Catarina | www.periodicos.ufsc.br ISSNe 2175-8018
IJIE
IBEROAMERICAN JOURNAL OF INDUSTRIAL ENGINEERING
REVISTA IBEROAMERICANA DE ENGENHARIA INDUSTRIAL
REVISTA IBEROAMERICANA DE INGENIARÍA INDUSTRIAL

APPLYING MILP/HEURISTIC ALGORITHMS TO AUTOMATED JOB-SHOP SCHEDULING PROBLEMS IN AIRCRAFT-PART MANUFACTURING

Adrián Marcelo Aguirre[1]

Carlos Alberto Méndez[2]

Álvaro García-Sánchez[3]

Miguel Ortega-Mier[4]

ABSTRACT: This work presents efficient algorithms based on Mixed-Integer Linear Programming (MILP) and heuristic strategies for complex job-shop scheduling problems raised in Automated Manufacturing Systems. The aim of this work is to find alternative a solution approach of production and transportation operations in a multi-product multi-stage production system that can be used to solve industrial-scale problems with a reasonable computational effort. The MILP model developed must take into account; heterogeneous recipes, single unit per stage, possible recycle flows, sequence-dependent free transferring times and load transfer movements in a single automated material-handling device. In addition, heuristic-based strategies are proposed to iteratively find and improve the solutions generated over time. These approaches were tested in different real-world problems arising in the surface-treatment process of metal components in the aircraft manufacturing industry.

Keywords: MILP-based algorithm. Automated Manufacturing Systems. Job-shop Scheduling problems. Real-world applications in aircraft-part fabrication process.

[1] Dr., Universidad Nacional del Litoral, Consejo Nacional de Investigaciones Científicas y Técnicas (UNL-CONICET), CAPSE-INTEC. E-mail: aaguirre@intec.unl.edu.ar .

[2] Dr., Universidad Nacional del Litoral, Consejo Nacional de Investigaciones Científicas y Técnicas (UNL-CONICET), CAPSE-INTEC: E-mail: cmendez@intec.unl.edu.ar.

[3] Dr., Escuela Técnica Superior de Ingenieros Industriales, Universidad Politécnica de Madrid (ETSII-UPM), Ingeniería de Organización. E-mail: alvaro.garcia@upm.es .

[4] Dr., Escuela Técnica Superior de Ingenieros Industriales, Universidad Politécnica de Madrid, (ETSII-UPM), Ingeniería de Organización. E-mail: miguel.ortega.mier@upm.es .

1 INTRODUCTION

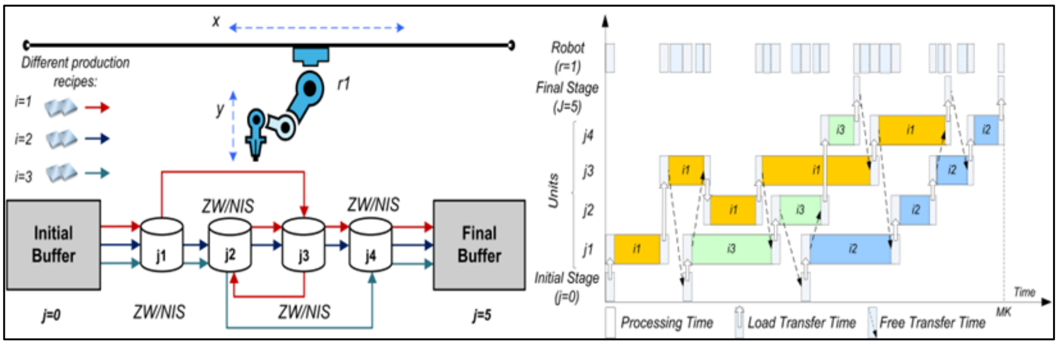
The solution of real-world scheduling problems has greatly attracted the attention of the research and industrial community for many years. In particular, flow-shop scheduling is one of the most treated problems in literature, in which a set of jobs $i=1,2,3,\ldots,N$ has to be transferred through several stages $s=0,1,2,\ldots,M+1$, by using an automated job's transfer device r. In this kind of problems, each job is processed in a sequence of units $j=1,2,3,\ldots,M$, during a flexible processing time, where every machine j can only perform one job at a time, e.g. it is a unary resource where job preemptions are not allowed. Flow-shop problems are usually focused on finding the best processing job sequence that minimizes the completion time of the last job in the system, which is widely known as the MaKespan *(MK)* criterion.

This type of automated manufacturing systems is commonly found in the manufacturing of printed circuit boards (PCBs) in electroplating plants and also in the automated wet-etch station (AWS) in semiconductor manufacturing systems. Moreover, many of those methods and tools developed for these problems, such as heuristic and meta-heuristics procedures (GEIGER; KEMPF; UZSOY, 1997; SHAPIRO; NUTTLE, 1988; BHUSHAN; KARIMI, 2003), full-space MILP models (PHILLIPS; UNGER, 1976; BHUSHAN; KARIMI, 2004; AGUIRRE; MÉNDEZ; CASTRO, 2011; CASTRO;ZABALLOS; MÉNDEZ, 2012), constraint programming approaches (ZEBALLOS; CASTRO; MÉNDEZ, 2011; NOVAS; HENNING, 2012) and hybrid MILP-based formulations (CASTRO et al., 2011; AGUIRRE et al., 2012), can be easily adapted, of their original versions, in order to incorporate the major complexities appeared in real-world industrial problems.

This work is focused on the critical surface-treatment process of large metal components in the aircraft manufacturing industry (PAUL; BIERWIRTH; KOPFER, 2007). Surface-treatment operations of heavy aircraft-parts are characterized by a higher complexity than typical flow-shop scheduling problems. This particular process involves a series of chemical stages $s=0,1,2,\ldots,Li$, disposed in a single production line, in which an automated material-handling tool is in charge of all transfer movements of the aircraft-parts between different stages, including from/to the input and output buffers disposed at front and at the end of the line.

The major assumptions of this problem are; a) unique production sequence for each part, b) re-entrant and possible recycle flows to the same unit, c) flexible processing times and d) load transferring times, e) sequence-dependent times for free travelling operations, f) no intermediate storage between stages, g) single production unit per stage, h) a single automated material-handling device with finite storage capacity on a simple rail, i) stringent storage policies "Zero Wait" (ZW) and "Non-Intermediate Storage" (NIS) for each production stage. Moreover, it is important to remark that, transferring times are directly related to the initial and the final position of the device in the production line. A simple example *(MxN=4x3)* which represents the main features of this problem is shown in Figure 1.

Figure 1 – Automated job-shop system with heterogeneous recipes and free transfer times



These features force that the material-handling tool, as a Robot, must travel large distances, from one unit to another, moving big and heavy aircraft-parts throughout the whole production line, wasting time and decrementing the performance of the line.

According to all of this, is easy to see that the daily operation of the material-handling tool in the surface-treatment process represents a complex issue for the decision-maker. In the past, simple heuristic procedures were used to provide a primary solution, far to the optimal one, for this kind of problems, when full-space methods had become untreatable for solving industrial examples, due to the high number of decisions involved in the model. In the other hand, simple heuristic methods, like two-stage approaches (BHUSHAN; KARIMI, 2004), are difficult to implement when sequencing decisions of both stages are strongly linked. Thus, any changed in one stage's decisions could turn the problem infeasible if other decisions are not carefully revised. Due to this, sequential approaches, based on mathematical programming and/or heuristics-based procedures, that combine robustness and flexibility, seem to be much appropriated to provide integrated solutions in moderate computational time.

The problem addressed in this work considers the scheduling of processing operations and transportation activities in the system by using a single automated job's transfer device (Robot). Thus, hybrid MILP/Heuristic-based approaches are developed to obtain good-quality results of the entire problem in an iterative manner. The principal aim of these mathematical approaches is to provide good-quality results to complex industrial-scale automated job-shop scheduling problems in a computationally efficient way.

2 GENERAL MILP MODEL

The MILP model developed for this work corresponds to an extended version of the previous full-space MILP model presented in Aguirre et al. (2012). This new general approach considers empty transferring times of the Robot between two consecutive load transfers. The performance of this approach will be demonstrated solving an industrial scale example of surface-treatments processes in real-world aircraft industry.

2.1 Nomenclature

2.1.1 Parameters

I, S, J Set of jobs ($i=i_1,...,N$), stages ($s=s_1,...,L_i$), units ($j=j_0,...,M+1$),

I^{ins}, I^{rel} Set of inserted jobs and released jobs in the system,

S_i , L_i Stages belonging to job i and the last stage in the sequence of job i,

$j_{i,s}$ Production unit that performs job i in production stage s,

$Seq_{(i)}$, $p_{(i)}$ Production recipe of job i and position of i in the processing sequence,

$t^{min}{}_{(i,s)}$, $t^{max}{}_{(i,s)}$ Minimum and Maximum processing time of job i in stage s,

$\pi^{min}{}_{(i,s)}$, $\pi^{max}{}_{(i,s)}$ Minimum and Maximum loaded transfer time of job i in stage s,

$\pi^{seq\text{-}dep}{}_{(i,i',s,s')}$ Free transfer times from loaded transfer i',s' to loaded transfer i,s,

M_T Large number (Big-M parameter).

2.1.2 Continuous Variables

$Ts_{(i,s)}$, $Tf_{(i,s)}$ Start time and Final time of job i in stage s,

$\pi^{load}{}_{(i,s)}$, $\pi^{free}{}_{(i,s)}$ Load and Free Transferring time of task i,s,

$t_{(i,s)}$ Processing time of task i,s,

$Pos_{(i,s)}$ Position of task i,s in the transfer sequence of a single Robot,

$K_{(i,s,i',s')}$ Immediate-precedence variable for transfer sequencing decisions,

MK Makespan.

2.1.3 Binary Variables

$X_{(i,s,i',s')}$ General-precedence variable for job's sequencing decisions,

$Y_{(i,s,i',s')}$ General-precedence variable for transfer's sequencing decisions.

2.2 Constraints

This MILP formulation takes into account flexible processing times under ZW/NIS policies by Equations (1-2), flexible load transfer times by Equations 3-5 and sequence-dependent free transferring times. Equations 6-8 and 9-11 are proposed to handle sequencing decisions in the same unit and the transfer's sequencing decisions in different units by binary variables $X_{(i,i',s,s')}$ and $Y_{(i,i',s,s')}$. Then, Equations 12-14 are given to determine the position of every transfer in the transfer sequence provided by $Pos_{(i,s)}$ parameter.

The immediate-precedence variables $K_{(i,i',s,s')}$ for transfer's sequencing decisions in a single resource is determined by Equations 15-18. The calculation of sequence-dependent free transferring times $\pi^{free}_{(i,s)}$ are described by Equation 18. Equation 19-20 is proposed for the partial reduction of the problem size when different jobs have the same production recipe ($Seq_{(i)}$). According to this, jobs with the same recipe must be processed following their lexicographic order. Finally, the objective function *(MK)* is presented in Equation 21.

Flexible timing constraints. Flexible processing times between a minimum a maximum time are considered by Equations 1-2 under stringent ZW policy in each production stage.

$$Tf_{(i,s)} = Ts_{(i,s)} + t_{(i,s)} \qquad \forall i \in I^{ins}, s \in S_i : (s \neq L_i), ZW \tag{1}$$

$$t^{\min}{}_{(i,s)} \leq t_{(i,s)} \leq t^{\max}{}_{(i,s)} \qquad \forall i \in I^{ins}, s \in S_i \tag{2}$$

Flexible transfer constraints. Non-Intermediate Storages (NIS) policy is followed in the system by the robot as stated in Equations 3-5. According to this, once the processing time of an immersion process is reached, the production lot must be removed by the robot to this bath and immediately transferred to the next unit in its production sequence.

$$X_{(i,i',s,s')} = \begin{cases} 1 & \text{if task } i,s \text{ is processed after task } i',s' \text{ in the same unit } j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$Ts_{(i,s)} \geq Tf_{(i',s')} + \pi^{load}{}_{(i,s)} + \pi^{load}{}_{(i',s'+1)} + \pi^{free}{}_{(i,s)} - M_T(1 - X_{(i,i',s,s')})$$
$$\forall i,i' \in I^{ins} : (i > i'), s \in S_i, s' \in S_{i'}, j_{i,s} = j_{i',s'} \tag{4}$$

$$Ts_{(i',s')} \geq Tf_{(i,s)} + \pi^{load}{}_{(i',s')} + \pi^{load}{}_{(i,s+1)} + \pi^{free}{}_{(i',s')} - M_T(X_{(i,i',s,s')})$$
$$\forall i,i' \in I^{ins} : (i > i'), s \in S_i, s' \in S_{i'}, j_{i,s} = j_{i',s'} \tag{5}$$

Transfer's sequencing decisions. Sequencing variables for transfer decisions *(i,s)* and *(i',s')* between different units are modeled by binary variable $Y_{(i,i',s,s')}$ in Equations 9-11.

$$Y_{(i,i',s,s')} = \begin{cases} 1 & if\ transfer\ i,s\ is\ processed\ after\ transfer\ i',s'\ in\ different\ units \\ 0 & otherwise \end{cases} \quad (6)$$

$$Ts_{(i,s)} \geq Ts_{(i',s')} + \pi^{load}{}_{(i,s)} + \pi^{free}{}_{(i,s)} - M_T(1 - Y_{(i,i',s,s')})$$

$$\forall i,i' \in I^{ins} : (i > i'), s \in S_i, s' \in S_{i'} \quad (7)$$

$$Ts_{(i',s')} \geq Ts_{(i,s)} + \pi^{load}{}_{(i',s')} + \pi^{free}{}_{(i',s')} - M_T(Y_{(i,i',s,s')})$$

$$\forall i,i' \in I^{ins} : (i > i'), s \in S_i, s' \in S_{i'} \quad (8)$$

Estimate the position of transfers in robot's sequence. The absolute position ($Pos_{(i,s)}$) of transfer task *i,s* in the robot sequence is defined by Equations 12-14. This variable is derived by the information of global precedence decisions $Y_{(i,i',s,s')}$. Thus, when $Y_{(i,i',s,s')}=1$, $Pos_{(i,s)} > Pos_{(i',s')}$ by Equation 12 and $Pos_{(i,s)} < Pos_{(i',s')}$ if $Y_{(i,i',s,s')}=0$ as is stated in Equation 13. Position $Pos_{(i,s)}$ variable is positive and could be integer or continuous. In order to reduce model complexity $Pos_{(i,s)}$ is defined as continuous variable using Equation 14.

$$Pos_{(i,s)} \geq Pos_{(i',s')} + 1 - M_T(1 - Y_{(i,i',s,s')}) \quad \forall i,i' \in I^{ins} : (i \geq i'), s \in S_i, s' \in S_{i'} : (i,s) \neq (i',s') \quad (9)$$

$$Pos_{(i',s')} \geq Pos_{(i,s)} + 1 - M_T(Y_{(i,i',s,s')}) \quad \forall i,i' \in I^{ins} : (i \geq i'), s \in S_i, s' \in S_{i'} : (i,s) \neq (i',s') \quad (10)$$

$$1 \leq Pos_{(i,s)} \leq \sum_{i' \in I^{ins}}^{N} \sum_{s' \in S_{i'}}^{L_{i'}} 1 \quad \forall i \in I^{ins}, s \in S_i \quad (11)$$

Immediate-precedence constraints. Using the absolute position information ($Pos_{(i,s)}$) a new variable $K_{(i,i',s,s')}$ is proposed in Equations 15-17 to determine the immediate-precedence of transfer *i,s* in the robot sequence. This new variable $K_{(i,i',s,s')}$ is then used to estimate the sequence-depending free transferring times in Equation 18. $K_{(i,i',s,s',r)}$ is a free variable but with some changes in Equations 15-18 can be also redefined as a positive or even integer domain.

$$K_{(i,i',s,s')} = \begin{cases} = 0 & if\ transfer\ i,s\ just\ after\ transfer\ i',s'\ in\ the\ robot \\ \neq 0 & otherwise \end{cases} \quad (12)$$

$$K_{(i,i',s,s')} = Pos_{(i,s)} - Pos_{(i',s')} - 1 + M_T(1 - Y_{(i,i',s,s')})$$

$$\forall i,i' \in I^{ins} : (i \geq i'), s \in S_i, s' \in S_{i'} : (i,s) \neq (i',s') \quad (13)$$

$$K_{(i',i,s',s)} = Pos_{(i',s')} - Pos_{(i,s)} - 1 + M_T(Y_{(i,i',s,s')})$$

$$\forall i,i' \in I^{ins} : (i \geq i'), s \in S_i, s' \in S_{i'} : (i,s) \neq (i',s') \quad (14)$$

$$\pi^{free}{}_{(i,s)} \geq \pi^{seq-dep}{}_{(i,i',s,s')} - M_T(K_{(i,i',s,s')}) \quad \forall i,i' \in I^{ins}, s \in S_i, s' \in S_{i'} : (i,s) \neq (i',s') \quad (15)$$

Predefined transfer decisions. Eqs 19-20 are proposed to reduce the search space of the entire problem without losing optimal results. Thus, sequencing decisions of two production lots *i* and *i'* that have the same recipe $Seq_{(i)} = Seq_{(i')}$ could be defined beforehand by Equation 19 ensuring that job *i* is produced after job *i'* in unit *j*. In addition, certain transfer sequencing decisions could be predefined by Equation 20 when two transfer tasks *i,s* and *i',s'* are following the same production recipe.

$$X_{(i,i',s,s')} = 1 \quad \forall i,i' \in I^{ins} : (i > i'), s \in S_i, s' \in S_{i'}, j_{i,s} = j_{i',s'}, Seq(i) = Seq(i') \tag{16}$$

$$Y_{(i,i',s,s')} = 1 \quad \forall i,i' \in I^{ins} : (i \geq i'), s \in S_i, s' \in S_{i'} : (s \geq s'), Seq(i) = Seq(i') \tag{17}$$

Objetive Function: Makespan Minimization. The principal aim is to optimize the throughput of aircraft-parts in the production line by using the makespan criterion *MK* as the measure of performance. Then, *MK* is estimated as the completion time of all tasks in the last production stage, as is stated in Equation 21.

$$MK \geq Ts_{(i,s)} \quad \forall i \in I^{ins}, s \in S_i : (s = L_i) \tag{18}$$

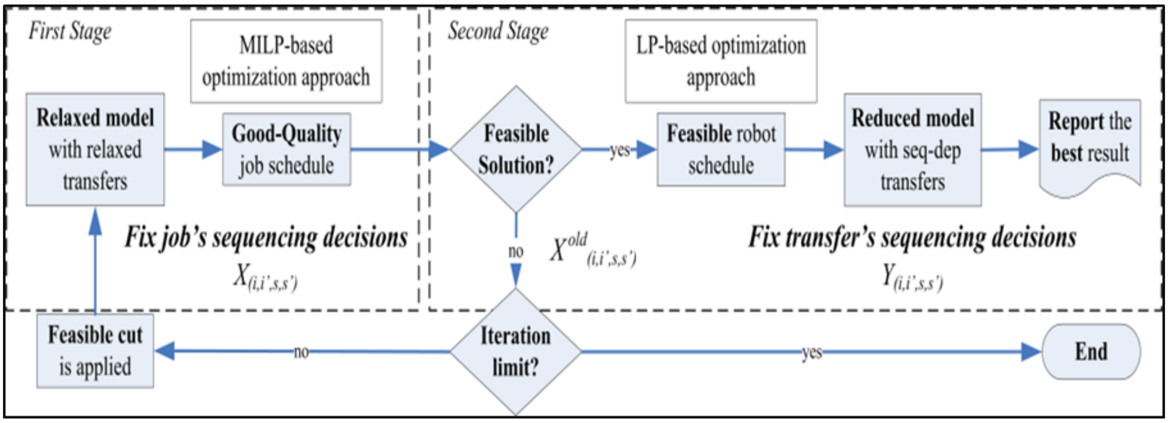
3 MILP/Heuristic-based algorithms

3.1 Sequential MILP-based algorithm

An MILP-based iterative solution method is presented here for dealing with this complex optimization problem in a sequential manner (see Figure 2). Thus, an adapted version of bi-level approach, developed by Bhushan and Karimi (2003) and later used by Aguirre, Méndez and De Prada (2012) for job-shop scheduling problems, is proposed in Figure 2. The solution algorithm allows solving the whole problem in two stages. In the first stage, a relaxed model is solved considering relaxed transfers and a feasible solution of this problem is provided in each iteration.

Then, in the second stage, sequence-dependent transfers are taken into account and then a reduced model is solved by fixing the job's sequencing and transfer sequencing decisions provided before. Here, it is worth to remark that the job's sequence of the first stage may not always provide feasible results. According to this, additional integer cuts, defined by Equation 22, are applied to provide alternative job's sequences that allow finding good-quality solutions. In addition, variable $Ts_{(i,s)}$ and $Tf_{(i,s)}$, are copied in each sub-model to accelerate the convergence. The algorithm ends when an iteration limit is reached.

Figure 1 – Pseudo-code scheme of sequential solution approach



The first stage algorithm could report a feasible result of the entire problem considering relaxed transfers by using Equations 1-11, 19-21. In order to provide a feasible result, a reduced model is solved considering overestimated transferring times. Then, job and transfer's sequencing decisions are fixed and a LP model, defined by Equations 1-21, is solved by considering sequence-dependent transfers. In order to do that, estimated transferring times are calculated as the maximum transfer value. Then integer cuts are applied by Equation 22 and the previous job's sequence is removed from the feasible region in following iterations. Then a new job's sequence will be found by the model in the first stage algorithm in order to improve the solution found.

Additional integer cuts to general alternative results.

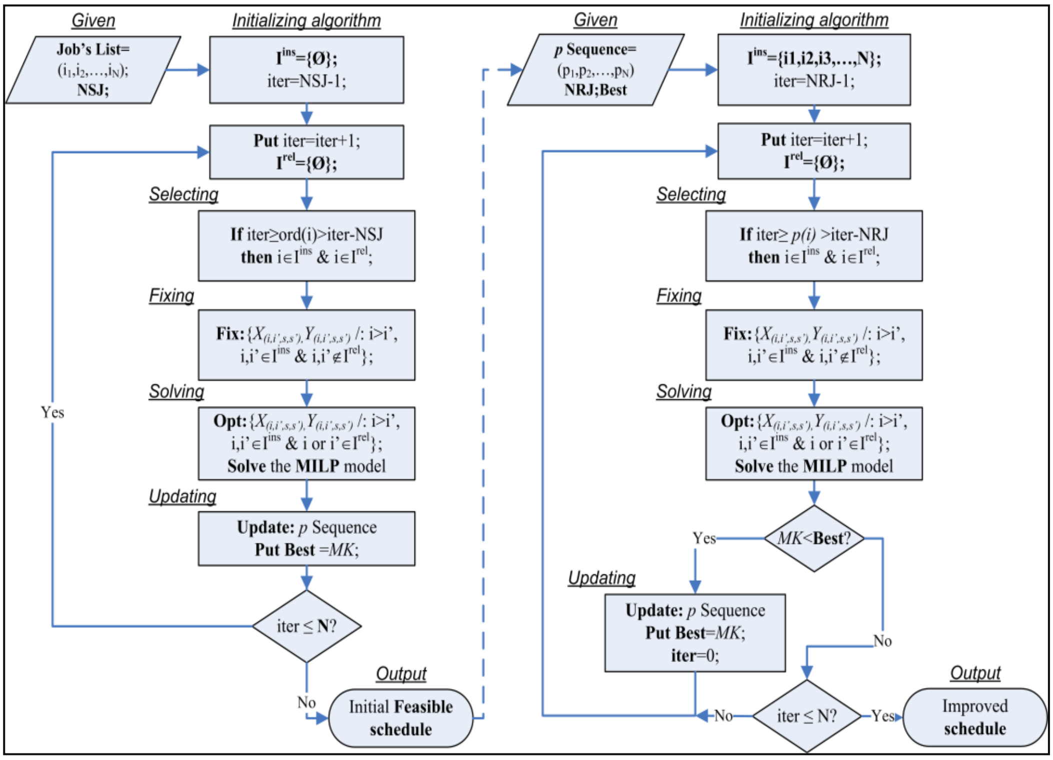
$$\left(\sum_{i,i'>i}^{N}\sum_{s\in Si}^{Li}\sum_{\substack{s'\in Si'\\ X^{old}_{(i,i',s,s')}=1}}^{Li'} X_{(i,i',s,s')}\right) \le \left(\sum_{i,i'>i}^{N}\sum_{s\in Si}^{Li}\sum_{\substack{s'\in Si'\\ X^{old}_{(i,i',s,s')}=1}}^{Li'} 1\right) + \left(\sum_{i,i'>i}^{N}\sum_{s\in Si}^{Li}\sum_{\substack{s'\in Si'\\ X^{old}_{(i,i',s,s')}=0}}^{Li'} X_{(i,i',s,s')}\right) - 1$$

$$\forall i,i'\in I^{ins}:(i>i'), s\in S_i, s'\in S_{i'}, j_{i,s}=j_{i',s'} \qquad (19)$$

3.2 Hybrid Constructive-Improvement algorithm

The constructive-improvement algorithm developed in this work is explained as follow in Figure 3. This iterative solution method allows decompose the problem in small sub-problems that can be solved separately, in a sequential way, consuming short computational time (AGUIRRE et al., 2012).

Figure 3 – Pseudo-code of Constructive-Improvement algorithm



Each step algorithm consists in 5 phases: initialization, selection procedure, setting binary variables, model resolution and updating parameters. In each iteration (iter) of the constructive step, NSJ jobs are selected, from the Job's List $(i_1,i_2,...,i_N)$, to be inserted in the system I^{ins} by following the NEH ordering rule (Nawas;). Thus, jobs with the maximum total production time are selected first to be included into set I^{rel} in order to be scheduled by optimizing variables $X_{(i,i',s,s')}$ and $Y_{(i,i',s,s')}$.

Before solving the MILP model, binary variables $X_{(i,i',s,s')}$ and $Y_{(i,i',s,s')}$ of already inserted but non-selected jobs can be fixed. Then, a reduced MILP model is solved obtaining a new sequence *p* and *MK*. When all jobs are already inserted, this step finishes reporting an initial feasible schedule $p=(p_1,p_2,...,p_N)$ and the Best makespan result. Starting from this solution, the second step algorithm determinates the jobs to be realized I^{rel} per iteration by chosen the NRJ consecutive jobs in the *p* sequence. Job's released in the selecting phase are re-scheduled in the system by optimizing $X_{(i,i',s,s')}$ and $Y_{(i,i',s,s')}$ while binary variables of non-released jobs remain fixed. Releasing consecutive jobs allows synchronizing transfer operations efficiently. After solving, the *MK* result of the MILP model is compared with the Best solution obtained until this iteration. Better solutions are reported and their sequence p is updated. The improvement step finish when no more released jobs can enhance the Best solution found.

4 COMPUTATIONAL ANALYSIS

4.1 Motivating Case Study

The following is a small case study proposed by Aguirre et al. in where sequence-dependent transferring times are taken into account in job-shop system. In it, jobs i_1-i_6 must be schedule in j_1-j_{36} units by following specific sequences or recipes $Seq_{(i)}$ which their information is presented in Table 1.

Table 1 – Flexible processing times of task *(i,s)* in unit *j* [min.]

Seq	**jobs**	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8
1	i_1-i_5	j_3:10-15 π:1-6	j_5:5-15 π:1-6	j_4:1 π:1-6	j_5:10-15 π:1-6	j_7:10 π:1-6	j_{25}:1 π:3-6	j_{35}:30-60 π:2-6	j_{36}: 0 π:1-6
2	i_2-i_3-i_4	j_3:10-15 π:1-6	j_5:5-10 π:1-6	j_7:1-5 π:1-6	j_9:5-10 π:1-6	j_{16}:51 π:2-6	j_{35}:30-60 π:3-6	j_{36}: 0 π:1-6	
3	i_6	j_3:10-15 π:1-6	j_5:5-10 π:1-6	j_{35}:30-60 π:3-6	j_{36}: 0 π:1-6	-	-	-	

In this work, free transferring movements are also considered. Free transfer time of tasks *(i,s)* is derived by the information of the current position of the robot along the line which is closely related to the last transfer movement *(i',s')* in robot sequence. According to this, the sequence-dependent transferring time to move job i from unit $j_{i,s}$ to unit $j_{i,s-1}$ is estimated by $\pi^{seq\text{-}dep}_{(i,i,s,s)} = 0.05[min.]^{*}abs(j_{i,s} - j_{i,s-1})$.

Table 2 shows the results obtained by the monolithic and the sequential approach presented above. The optimal solution of 259.5 min. is reached by the MILP model in < 500 sec. while sequential algorithm could provide only a feasible result of 301.6 min., far from the optimal one, after 40 CPUs.

Table 2 – Statistics and Results of the small example analyzed

Units x Jobs	**Statistics**	**Monolithic MILP-based model**	**MILP model with relaxed transfers**	**LP model with seq-dep transfers**
36x6	Binary Var.	1075	950	-
	Cont. Var.	1887	206	1887
	Equations	9149	2048	9149
	MK *(Gap%)*	**259.5 (0.0%)**	**304**	**301.6**
	CPUtime(s)*	320	3.8	0.6
	Total time*(s)*	415	**40	

*Using Gurobi 5.0 in a PC Intel Core 2 Quad 2,5 GHz with parallel processing in 4 threads.
**Maximum number of iterations by the algorithm = 10. Time limit per iteration = 120 sec.

Different values of NSJ/NRJ algorithm parameter are tested. The results reported in Table 3 show that the decomposition algorithm could find an optimal solution 259.5 min. in less than 60 sec. using certain configurations, e.g. NSJ/NRJ=3/1 or 2/2.

Table 3 – Statistics and Results of test problem analyzed

Jobs	**Constructive Step algorithm**		**Improvement Step Algorithm NRJ=1**		**NRJ=2**		**NRJ=3**	
(2,3,1)	**MK**	**CPU**/iter	**MK**(ip)	**CPU**/iter	**MK**(ip)	**CPU**/iter	**MK**(ip)	**CPU**/iter
NSJ =1	290.55	4.8/6	290.55	21/6	259.5(11)	58/21	259.5(11)	127/7
NSJ =2	268.55	18/5	268.55	24/5	259.5(3.4)	38/15	259.5(3.4)	48/2
NSJ =3	268.55	49/4	259.5(3.4)	9/6	259.5(3.4)	13/5	259.5(3.4)	26/4

Using Gurobi 5.0 in a PC Intel Core 2 Quad 2,5 GHz with parallel processing in 4 threads. ip = Improvement percent from the initial solution. iter = Iterations to reach the solution. Time limit per iteration = 120 sec. *Computational time limit = 3600 sec.

4.2 Industrial application example

An industrial application example of real-life operations in the aircraft industry is presented in this work. This information was obtained from a previous work of Aguirre et al.[15]. In this example, ten jobs i_1-i_{10} have to be schedule in different units, from j_0-j_{36}, where j_0 and j_{36} represent the input and the output buffer. The information of processing times $t_{(i,s)}$, transferring times $\pi_{(i,s)}$ of every task (i,s) and the processing sequences $Seq_{(i)}$ of each job i is presented in Table 1. Free transfer times between two consecutive load transfer are estimated as sequence-dependent variable $\pi^{seq\text{-}dep}_{(i,i,s,s)}=0.05[min.]^{*}abs(j_{i,s} - j_{i,s-1})$, according to the absolute distance of departure and arrival units $j_{i,s}$ and $j_{i,s-1}$.

Table 4 – Flexible processing times and transfer times of task *(i,s)* in unit *j* [min.]

Seq	**Jobs**	**s₁**	**s₂**	**s₃**	**s₄**	**s₅**	**s₆**	**s₇**	**s₈**	**s₉**	**s₁₀**
1	i_1-i_5-i_6-i_9-i_{10}	j_3:12-15 π:1-6	j_5:5-15 π:1-6	j_7:1 π:1-6	j_9:10-15 π:1-6	j_{35}:30-60 π:3-6	j_{36}:0 π:1-6	-	-	-	-
2	i_2-i_3	j_3:12-15 π:1-6	j_5:5-15 π:1-6	j_4:6-10 π:1-6	j_5:5-10 π:1-6	j_7:1 π:1-6	j_9:10-15 π:1-6	j_{35}:30-60 π:3-6	j_{36}:0 π:1	-	-
3	i_4-i_7-i_8	j_3:12-15 π:1-6	j_5:5-15 π:1-6	j_7:8-10 π:1-6	j_8:5-10 π:1-6	j_9:5-10 π:1-6	j_{16}:56 π:2-6	j_{18}:5-10 π:1-6	j_{30}:5-15 π:2-6	j_{35}:30-60 π:2-6	j_{36}:0 π:1-6

Table 4 shows the main statistics and results of the industrial problem analyzed. Here, monolithic model cannot reach optimal the result after 1 hour, providing only a good initial solution with 4% relative gap in 250 CPUs. However, sequential approach can reach a better result (383.75 min.) in very short total CPU time of 35 seconds.

Table 5 – Statistic and Results for the industrial problem analyzed

Units x Jobs	**Statistics**	**Monolithic MILP-based model**	**MILP model with relaxed transfers**	**LP model with seq-dep transfers**
36x10	Binary Var.	3494	2926	-

Units x Jobs	Statistics	Monolithic MILP-based model	MILP model with relaxed transfers	LP model with seq-dep transfers
	Cont. Var.	6157	607	6157
	Equations	30868	6928	30868
	MK *(Gap%)*	**384.15 (4.0%)**	**392.4**	**383.75**
	CPUtime(s)*	250	6.2	0.8
	Total time*(s)*	3600	**35	

*Using Gurobi 5.0 in a PC Intel Core 2 Quad 2,5 GHz with parallel processing in 4 threads.
**Maximum number of iterations by the algorithm = 10. Time limit per iteration = 120 sec.

The solution obtained by the decomposition algorithm (378 min.), testing different NSJ/NRJ combinations, improves the one reported by sequential approach.

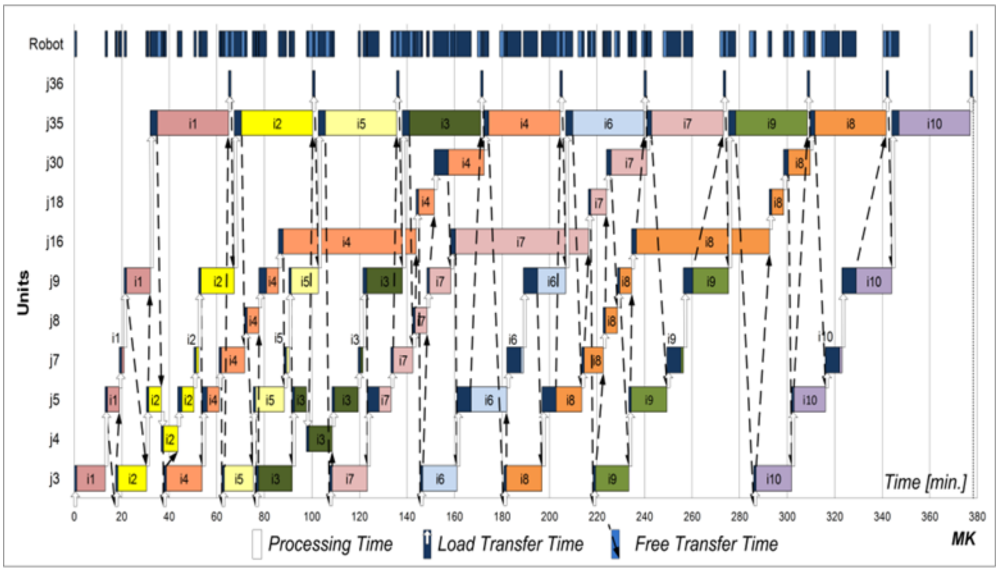
Table 6 – Statistics and Results of test problem analyzed

Jobs	Constructive Step algorithm		Improvement Step Algorithm NRJ=1		NRJ=2		NRJ=3	
(5,2,3)	**MK**	**CPU**/iter	**MK**(ip)	**CPU**/iter	**MK**(ip)	**CPU**/iter	**MK**(ip)	**CPU**/iter
NSJ =1	395.75	101.3/10	395.75(0.0)	74.1/5	378(4.5)	1039/15	378(4.5)	2014/14
NSJ =2	391.2	167.6/9	379.25(3.0)	130.2/14	378(3.3)	965.8/13	378(3.3)	1331/7
NSJ =3	379.4	476.6/8	378(0.4)	91.1/7	378(0.4)	952.7/14	378(0.4)	1106/2

Using Gurobi 5.0 in a PC Intel Core 2 Quad 2,5 GHz with parallel processing in 4 threads. ip = Improvement percent from the initial solution. iter = Iterations to reach the solution. Time limit per iteration = 120 sec. *Computational time limit = 3600 sec.

Table 6 shows that the best solution found by the algorithm is obtained in less than 10 minutes using NSJ/NRJ=3/1. In general, larger values of NSJ/NRJ can provide better results but with more CPU time. Thus, the reported solution starts from a good-quality result using higher NSJ, provided in < 500 sec., and then it is improved until achieving the best result after a few minutes. The detailed schedule is shown in Figure 4.

Figure 4 – Solution Schedule of the industrial example



4.3 Testing a daily scheduling problem

This problem, provided by Paul et al.[12], represents a real industrial example at the surface treatment process of aircraft-parts used at the body and wings of airplanes. Here, 12 jobs have to be scheduled following one of the production recipes $Seq_{(i)}$ where the initial and final units are j_0 and j_{20}. Also, information of flexible processing and load transferring times are shown in Table 7.

Table 7 – Flexible processing times and transfer times of task *(i,s)* in unit *j* [min.]

Seq	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{11}
1	j_1:23-26 π:1-6	j_2:3-4 π:1-6	j_3:3-4 π:1-6	j_4:15-∞ π:1-6	j_6:3-∞ π:1-6	j_5:15-∞ π:1-6	j_{20}:0 π:2-6	-	-	-	-
2	j_7:2-3 π:2-6	j_8:3-4 π:1-6	j_9:3-4 π:1-6	j_6:3-∞ π:1-6	j_5:15-∞ π:1-6	j_{20}:0 π:2-6	-	-	-	-	-
3	j_{13}:70-73 π:2-6	j_{14}:2-3 π:1-6	j_{15}:2-∞ π:1-6	j_{18}:15-20 π:1-6	j_{17}:3-∞ π:1-6	j_{16}:40-45 π:1-6	j_{11}:2-3 π:1-6	j_{10}:2-∞ π:1-6	j_6:3-∞ π:1-6	j_5:15-∞ π:1-6	j_{20}:0 π:2-6
4	j_{13}:70-73 π:2-6	j_{14}:2-3 π:1-6	j_{15}:2-∞ π:1-6	j_{12}:40-45 π:1-6	j_{11}:2-3 π:1-6	j_{10}:2-∞ π:1-6	j_6:3-∞ π:1-6	j_5:15-∞ π:1-6	j_{20}:0 π:2-6	-	-

Load and free transfer times were changed to this original version in order to much more emphasize robot activities. Thus, pick-up and drop-down a part into a bath are estimated in 30 seconds while the travelling time is approximately to 3 sec./meter. The distance between adjacent baths is 1 meter. Thus, the free travelling time from j_1 to j_2 takes 3 sec. while load travel time is rounded in 1 min. According to this, for small distances, less than 15 meters, the time to travel of charged robot, considering pick-up and drop-down movements, is estimated in 1 min. while for medium distances (>15 meters) is 2 minutes. The current product mix of the original problem is (8,2,1,1).

Table 8 and Table 9 show the main results of the monolithic model, the sequential approach and the decomposition algorithm for this particular problem. As observed in the reported statistics, this problem seems to be very challenging due to the number of variables and equations defined in the MILP formulation.

Table 8 – Statistics and Results of test problem analyzed

Units x Jobs	Statistics	Monolithic MILP-based model	MILP model with relaxed transfers	LP model with seq-dep transfers
20x12	Binary Var.	4523	4234	-
	Cont. Var.	8185	353	8185
	Equations	41318	9950	41318
	MK *(Gap%)*	**268.3 (4.2%)**	**279**	**270.55**
	CPUtime(s)*	1660	240	0.7
	Total Time*(s)*	3600	**1155	

*Using Gurobi 5.0 in a PC Intel Core 2 Quad 2,5 GHz with parallel processing in 4 threads.
**Maximum number of iterations by the algorithm = 10. Time limit per iteration = 120 sec.

Table 9 – Statistics and Results of test problem analyzed

Jobs	Constructive Step algorithm		Improvement Step Algorithm NRJ=1		NRJ=2		NRJ=3	
(8,2,1,1)	**MK**	**CPU**/iter	**MK**(ip)	**CPU**/iter	**MK**(ip)	**CPU**/iter	**MK**(ip)	**CPU**/iter
NSJ =1	308.1	150/12	288.9(6)	3200/24	272.4(11)	3100/13	270.0(12)	3300/11
NSJ =2	289.45	250/11	271.25(6)	1394/19	271.25(6)	3600/34	270.0(7)	3600/32
NSJ =3	285.75	440/10	285.45	2173/47	271.25(5)	3600/20	270.0(5)	3600/16

Using Gurobi 5.0 in a PC Intel Core 2 Quad 2,5 GHz with parallel processing in 4 threads. ip = Improvement percent from the initial solution. iter = Iterations to reach the solution. Time limit per iteration = 300 sec. *Computational time limit = 3600 sec.

Despite of this, results indicate that monolithic approach can be solved up to 4.2% of relative gap in 1660 sec. while sequential procedure provides similar solution in 240 seconds. Decomposition approach only could provide good-quality results after 1500 sec. using NSJ/NRJ=2/1 configuration.

5 CONCLUSIONS

An MILP-based model and sequential heuristic approaches were developed for the scheduling of multiple aircraft-parts in the surface-treatment process in the aircraft industry. Results demonstrate that MILP-based model could obtain good-quality results in less than 1 hour of CPU time. While, heuristic-based algorithms, were able to decompose the problem in reduced sub-problems that were solved in moderate CPU time. Thus, a primary solution of these complex scheduling problem have been easily found while extra computational time has been used to improve the solutions obtained over time. Finally, different algorithm parameters were tested in order to find the best configuration, in terms of *MK* and CPU effort, for these particular problem's instances.

Acknowledgments

Financial support received from CONICET under Grant PIP 112 20110100940, from UNL under Grant CAI+D 2011-UNL-PI-81-481 and from Erasmus Mundus Scholarship Program are fully appreciated.

REFERENCES

AGUIRRE, A.M.; MÉNDEZ, C.; DE PRADA, C. An iterative MILP-based approach to automated multi-product multi-stage manufacturing systems. **Computer Aided Chemical Engineering**, v. 31, p. 1085-1089.

AGUIRRE, A.M.; MÉNDEZ, C.A.; CASTRO, P.M. A novel optimization method to automated wet-etch station scheduling in semiconductor manufacturing systems. **Computers & Chemical Engineering**, v. 35, n. 12, p. 2960-2972, 2011.

AGUIRRE, A.M.; MÉNDEZ, C.A.; CASTRO, P.M.; DE PRADA, C. MILP-based Approach for the Scheduling of Automated Manufacturing System with Sequence-Dependent transferring times. **Computer Aided Chemical Engineering**, v. 30, p. 477-481, 2012.

AGUIRRE, A.M.; MÉNDEZ, C.A.; GUTIERREZ, G.; DE PRADA, C. An improvement-based MILP optimization approach to complex AWS scheduling. **Computers & Chemical Engineering**, v. 47, p. 217-226, 2012.

BHUSHAN, S.; KARIMI, I.A. An MILP approach to automated wet-etch station scheduling. **Industrial & Engineering Chemistry Research**, v. 42, n. 7, p. 1391-1399, 2003.

BHUSHAN, S.; KARIMI, I.A., Heuristic algorithms for scheduling an automated wet-etch station. **Computers & Chemical Engineering**, v. 28, n. 3, p. 363-379, 2004.

CASTRO, P.M.; AGUIRRE, A.M.; ZEBALLOS, L.J.; MÉNDEZ, C.A., Hybrid mathematical programming discrete-event simulation approach for large-scale scheduling problems. **Industrial & Engineering Chemistry Research**, v. 50, n. 18, p. 10665-10680, 2011.

CASTRO, P.M.; ZEBALLOS, L.J.; MÉNDEZ, C.A. Hybrid time slots sequencing model for a class of scheduling problems. **AIChE Journal**, v. 58, n. 3, p. 789-800, 2012.

GEIGER, C.E.; KEMPF, K.G.; UZSOY, R. A Tabu search approach to scheduling an automated wet etch station. **Journal of Manufacturing Systems**, v.16, n. 2, p. 102-116, 1997.

MÉNDEZ C.A; CERDÁ, J. An MILP Continuous-time framework for short-term scheduling of multipurpose batch processes under different operation strategies. **Optimization and Engineering**, v. 4, n. 1-2, p. 7-22, 2003.

NAWAZ, M.; ENSCORE Jr., E.E.; HAM, I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. **Omega**, v. 11, p. 91-95, 1983.

NOVAS, J.M.; HENNING, G.P. A comprehensive constraint programming approach for the rolling horizon-based scheduling of automated wet-etch stations. **Computers & Chemical Engineering**, v. 42, p. 189-205, 2012.

PAUL, H.J.; BIERWIRTH, C.; KOPFER, H. A heuristic scheduling procedure for multi-item hoist production lines. **International Journal Production Economics v.** 105, p. 54-69, 2007.

PHILLIPS, L.W.; UNGER, P.S. Mathematical programming solution of a hoist scheduling programm. **AIIE Transactions**, v. 28, n. 2, p. 219-225, 1976.

SHAPIRO, G.W.; NUTTLE, H.L. Hoist scheduling for a PCB electroplating. **IIE Transactions**, v. 20, n. 2, p. 157-1, 1988.

ZEBALLOS, L.J.; CASTRO, P.M.; MÉNDEZ, C.A. Integrated constraint programming scheduling approach for automated wet-etch stations in semiconductor manufacturing. **Ind Eng Chem Res**. v. 50, p.1705-1715, 2011.

Originais recebidos em: 26/10/2013

Aceito para publicação em: 15/04/2014