Scheduling of flexible multipurpose back-up chemical/assembly process systems

E. Capón, [a] A. Bonfill, [a] A.Espuña, [a] L. Puigjaner [a]

[a]*Department of Chemical Engineering, Universitat Politécnica de Cataluña, 08028 Barcelona, Spain*

Summary

This paper focuses on the scheduling of flexible multipurpose back-up chemical/assembly process systems in a fully robotized environment. The problem is modeled using both mathematical programming and heuristic rules, which are compared in terms of optimality and computation time. Stochastic data is also considered to take into account production environmental dynamics and identify robust schedules. The results obtained suggest that the two strategies should be combined to achieve well balanced results in terms of optimality and time.

Keywords: flexible systems, multipurpose scheduling

1. Introduction

In a continuously changing and competitive environment, increasing productivity and quality is crucial for process and manufacturing industries. Significant improvements in technology have been achieved in recent years. One of the frequent solutions to low productivity and poor quality is automation (Kaighobadi et al., 1994). Higher degree of automation implies higher system’s flexibility, better operation control, reduced manpower, and many other benefits. In that sense, the effective configuration and enhanced coordination of operations in automated lines and workstations play an important role not only in manufacturing but also in pharmaceutical and food industries.

This paper deals with the scheduling of flexible multipurpose back-up chemical/assembly processes in a robotized environment aiming to allocate resources (robotized units, intermediate storages and other resources) to assembly activities and determine the sequencing and timing of operations so as to optimize makespan and resource use.

The short-term scheduling problem of multipurpose batch plants has been extensively studied; recent reviews of models and optimization methods can be found in Shaik et al. (2006) and Méndez et al. (2006). On the other hand, a large bibliography dealing with flexible assembly systems is already available. Most of these contributions rely on the use of dispatching rules to circumvent the problem complexity. For example, Reeja (2000) develops new dispatching rules based on the “operation synchronization date”. Sawik (2004) also presents a mixed integer programming approach for loading and scheduling of a general flexible assembly system made up of a network of assembly stages with limited in-process buffers and with no revisiting of stations.

However, most of the situations studied so far do not consider the coexistence of information/constraints typical from the process industry (flow management, recycles, capacity and time limitations, dependency on operating conditions, etc.) and from the manufacturing or assembly industry (robotized tools, geometrical problems,…), and in many cases their interaction, which happens frequently in these industries. In this approach, alternative production sequences are considered along with constraints on resources availability and simultaneity requirements.

2. Problem statement

The considered automated process consists of a set of equipment units (robots, identification and verification units), intermediate storages and other resources such as manpower, steam, tools, etc. The process to obtain the final product comprises the assembling, and related operations, namely identification, verification and transport, of different components and subassemblies, and is constrained by resource availability, simultaneity of assembling operations and coordination of robotised units. Therefore, the aim of this work is to allocate the available limited resources to a set of defined activities and determine the detailed sequencing and timing of operations so that production makespan is minimized and resource use maximised, taking into account the aforementioned constraints. In addition, the process environment is dynamic, so it is important to consider the randomness in input data in order to achieve robust and efficient solutions.

On the whole, the problem stems from a set of components and subassemblies to be processed throughout a set of operations with their associated times and resource consumption, and seeks the values of initial and final processing times of the different tasks to be carried out in diverse equipment that best fit the needs of the processing environment.

3. Modelling and solution approach

The problem has been modelled by defining a set of components and subassemblies for each final product (Figure 1). Each component and subassembly comprises a set of process stages. Each process stage consists of a set of operations to be performed sequentially in one of the available equipments units provided with the specific functionality to fulfil that task. Each operation is characterized by its initial and end times, operation time, maximum waiting time as well as resources consumption levels. Resource consumption, simultaneity and sequence restrictions between operations of different stages are contemplated.

The general stage-operation scheme devised is presented in Figure 2. It shows how components are processed through stages e1-e4, whereas subassemblies through stages e5 and e6. Stages of components are sequential whereas subassembly stages must be parallel since subassembly operations O7 and O8 must be performed simultaneously. In addition, Figure 1 also indicates use of resources at each stage.

The described model structure is flexible and general enough to represent different types of problems in this area.

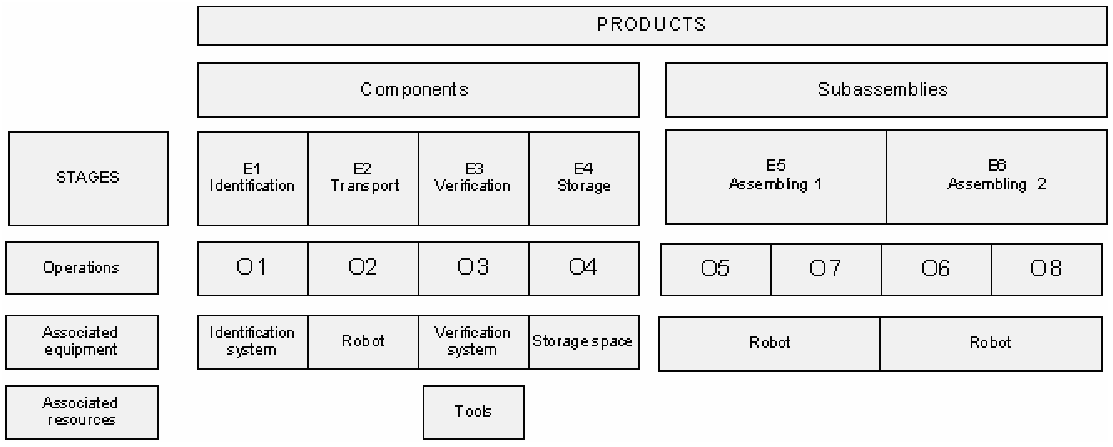

Figure 1. Model definition scheme

Different methodologies, either equation- or heuristic-based can be considered to address the underlying problem. In this work, the use of a rigorous MILP formulation is assessed and compared with a rule-based heuristic approach.

3.1. Rigorous approach

A rigorous MILP mathematical model has been developed based on a continuous time representation.

The objective function is to minimize makespan, which is defined as the maximum final time of a given operation.

$$MIN\ z_{obj} = MIN\ M_k = MIN\left(\max_{(o,p)|p\in(p_1\ldots p_P),o\in o(o_e\cap p_e)}\left(T_{fn}(o,p)\right)\right) \qquad Eq.\ 1$$

Precedence conditions are defined for each component and subassembly. Therefore, sequencing conditions must be defined for each subassembly. Equation 2 imposes that the operations of a given stage must be done consecutively, whereas equation 3 imposes this relationship between operations of different stages, initially defined in oseq. Equation 4 imposes the relation of sequence for components and subassemblies. Moreover, equations 5 and 6 impose simultaneity between operations of different stages, formerly defined in osim.

$$T_{in}\left(o,p\right) = T_{fn}\left(o\text{-}1,p\right) \quad \forall p,\forall e\in P_e,\ \forall o\left|\left\{o\in O_e \wedge not\left(ose\right)\right\}\right. \qquad Eq.\ 2$$

$$T_{fn}\left(o,p\right) \ \le \ T_{in}\left(o',p\right) \quad \forall p, \forall(e,e') \in P_e, \ \forall\left(o \in O_e, o' \in O_{e'}\right)\Big| oseq \qquad Eq.\ 3$$

$$T_{fn}\left(o,p\right) \ \le \ T_{in}\left(o',p'\right) \quad \forall(p,p') \in pseq, \forall(e,e') \in P_e, \forall\left(o \in O_e, o' \in O_{e'}\right) \qquad Eq.\ 4$$

$$T_{in}\left(o,p\right) \ = T_{in}\left(o',p\right) \quad \forall p, \forall(e,e') \in P_e, \ \forall\left(o \in O_e, o' \in O_{e'}\right)\Big| osim \qquad Eq.\ 5$$

$$T_{fn}\left(o,p\right) \ = T_{fn}\left(o',p\right) \quad \forall p, \forall(e,e') \in P_e, \ \forall\left(o \in O_e, o' \in O_{e'}\right)\Big| osim \qquad Eq.\ 6$$

Timing conditions define operation initial and final times. Eq.7 defines the operation time for a given product that follows a specific sequence as the difference between final time, and initial and waiting times. Eq.8 sets the maximum limit of waiting time for a given operation.

$$T_{fn}\left(o,p\right) - T_{in}\left(o,p\right) - T_{wt}\left(o,p\right) \ = minot(o,p) \quad \forall p, \forall e \in P_e, \forall o \in O_e \qquad Eq.\ 7$$

$$T_{wt}\left(o,p\right) \ \le \ maxot(o) \quad \forall p, \forall e \in P_e, \forall o \in O_e \qquad Eq.\ 8$$

Allocation constraint assures through eq.9 that a single equipment unit is assigned to each stage. Moreover, eq.10 states that two simultaneous operations cannot be done in the same equipment. Additionally, eq.10 reduces the search space of solutions of the problem.

$$\sum_{r \in re(r,e)} Y\left(e,p,r\right) \ = \ 1 \quad \forall p, \forall e \qquad Eq.\ 9$$

$$Y\left(e,p,r\right) + Y\left(e',p,r\right) \ = 1 \quad \forall p, \forall(e,e') \in P_e, \forall r \in \left(R_e \cap R_{e'}\right), \quad \forall\left(o \in O_e, o' \in O_{e'}\right)\Big| osim \qquad Eq.\ 10$$

Sequencing constraints permit timing without overlapping between operations in the different resources (equipments).

$$T_{fn}\left(o,p\right) \le T_{in}\left(o',p'\right) + M \cdot \left(1\text{-}X_o\left(e,e',p,p'\right)\right) + M \cdot \left(2\text{-}Y\left(e,p,r\right) - Y\left(e',p',r\right)\right)$$
$$\forall p,p', \ \left(\left(\forall e \in P_e, \forall e' \in P_{e'}\right)\Big|\left(p<p' \vee e<e'\right)\right), \forall r \in \left(R_e \cap R_{e'}\right), \forall o \in \left(O_e \cap ose\right), \forall o' \in \left(O_{e'} \cap ofe\right)$$

Eq. 11

$$T_{fn}\left(o',p'\right) \le T_{in}\left(o,p\right) + M \cdot X_o\left(e,e',p,p'\right) + M \cdot \left(2\text{-}Y\left(e,p,r\right) - Y\left(e',p',r\right)\right)$$
$$\forall p,p', \ \left(\left(\forall e \in P_e, \forall e' \in P_{e'}\right)\Big|\left(p<p' \vee e<e'\right)\right), \forall r \in \left(R_e \cap R_{e'}\right), \forall o \in \left(O_e \cap ose\right), \forall o' \in \left(O_{e'} \cap ofe\right)$$

Eq. 12

Resource limitations is represented by equations 13 to 15, and assure that resource consumption does not exceed availability. Namely, eq.13 imposes that the consumption of a given operation *o*, must be covered through the availability of that resource. Equations 14 and 15 permit timing without overlapping between operations in the different resources.

$$\sum_{z \in kz(k,z)} V\left(o,p,z\right) = 1 \quad \forall p, \forall e \in P_e, \forall o \in O_e, \forall k \in \left(K_o\right) \qquad Eq.\ 13$$

$$T_{fn}\left(o,p\right) \le T_{in}\left(o',p'\right) + M \cdot \left(1\text{-}X_o\left(e,e',p,p'\right)\right) + M \cdot \left(2\text{-}V\left(o,p,z\right) - V\left(o',p',z\right)\right)$$
$$\forall p,p', \ \left(\left(\forall e \in P_e, \forall e' \in P_{e'}\right)\Big|\left(p<p' \vee e<e'\right)\right), \forall o \in O_e, \forall o' \in O_{e'}, \forall k \in \left(K_o \cap K_{o'}\right), \forall z \in Z_k$$

Eq. 14

$$T_{fn}\left(o',p'\right) \leq T_{in}\left(o,p\right) + M \cdot X_o\left(e,e',p,p'\right) + M \cdot \left(2-V\left(o,p,z\right) - V\left(o',p',z\right)\right)$$

$$\forall p,p',\ \left(\left(\forall e \in P_e,\ \forall e' \in P_{e'}\right) \middle| \left(p<p' \vee e<e'\right)\right), \forall o \in O_e, \forall o' \in O_{e'}, \forall k \in \left(K_o \cap K_{o'}\right), \forall z \in Z_k$$

Eq. 15

3.2. Heuristic approach

The integrated support system for planning and scheduling developed by Cantón (2003) is also adopted to solve the scheduling problem. The system is based on heuristic procedures commonly used in commercial packages, and it has been designed in a modular way allowing the implementation of alternative algorithms, as well as additional functionalities to solve and further optimize the problem as needed.

Rule-based algorithms are available to establish the number of batches to be processed, the sequence and the assignment of stages to specific resources. These algorithms are applied in combination with the Event Operation Network (EON) timing model, also proposed by Cantón (2003), to perform the timing of the operations. This timing model takes into account storage and resource constraints, and it is based on a graph representation involving a network of events (time instants at which some change occurs) and operations (time intervals to be observed between start and end events).

The main priority rules implemented in the module are reported in Table 1. Based on the characteristics of the problem and the objective function, different combinations of these rules can be selected.

An initial feasible schedule is first identified, which can be further improved according to some objective function. Meta-heuristic algorithms such as simulated annealing and genetic algorithms are implemented for this purpose within the module.

Assignment	AUA: already used assignment
	FU: first unit
	HPU: highest priority unit
	LUU: less used unit
	MAU: most available unit
	SPTU: shortest processing time unit
Sequencing	EDD: earliest due date
	HSL: highest storage level
	LCT: longest cycle time
	LPT: longest processing time
	LSL: lowest storage level
	SCT: shortest cycle time
	SPT: shortest processing time

Table 1. Priority rules implemented for assignment and sequencing decisions.

3.2.1. Uncertainty

The scheduling problem under uncertainty entails taking decisions before the real value of the parameters is known. Therefore, the behaviour of the system is only defined after uncertain parameters are unveiled. In these cases, deterministic models are not suitable for stochastic problems. The best way to confront uncertainty is to caracterize it and to introduce it in the decision-making process.

Variables such as processing and transportation times can be considered in a two stage probabilistic approach, with the simulation of different scenarios to eventually identify a robust schedule. In this case, assembling and identification operation times have been considered random variables.

Stochastic information is introduced by means of a set of scenarios. There are as many scenarios as combinations of realizations of random variables. Therefore, in the model there must be as many second-stage variable sets as scenarios, and constraints that contain second-stage variables must be duplicated for each scenario.

Uncertainty is managed using a two-stage stochastic approach, so new considerations must be undertaken in the previously presented rigorous model to introduce stochastic information. As for decision variables, those related to allocation and sequencing of operations are considered first-stage variables, whereas initial, waiting and final times belong to the second-stage since they can only be decided once uncertainty is unveiled.

For the proposed random operations, three possible times with an associated discrete probability are considered. From these values, a combinatorial amount of scenarios can be derived. In order to reduce the number of scenarios without loosing the essence of the problem, a representative number of scenarios must be sampled and simulated so that a robust schedule is achieved.

4. Case study

For illustration purposes consider the case study based on an assembly process with 7 components (p1-p7) and 6 subassemblies (p8-p13), 40 stages and 52 operations. The precedence diagram is represented in Figure 2. From this diagram, component and subassembly sequence restrictions can be deduced. For instance, it can be seen that component p1 and p7 must be processed before subassembly p9, and p2 and p7 before p10; however, there is no precedence condition between p9 and p10.

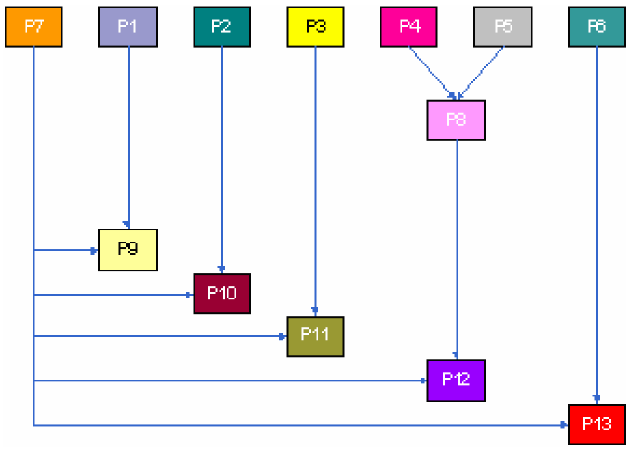

Figure 2. Tree-representation of the sequence of the case study

Other initial data are operation times (table 2) and resources availability (table 3). Moreover, waiting time for all operations is 100 s, which means in practice an unlimited intermediate storage policy (UIS).

	p1	p2	p3	p4	p5	p6	p7
o1	5	5	4	4	3	6	4
o2	3	3	3	3	3	3	3
o3	3	3	6	6	2	9	6

	p8	p9	p10	p11	p12	p13
o5	3	3	3	3	3	3
o6	4	3	3	4	4	3
o7	3	3	3	3	3	3
o8	4	3	3	4	4	3

Table 2. Case study operation times (in s).

Resource	Availability (units)
Identification system	1
Verification system	1
Robot	2
Storage area	Unlimited

Table 3. Resources availability.

The mathematical model has been implemented in GAMS and solved using the MILP solver of CPLEX (9.0) on AMD Athlon 3000 computer. The best schedule obtained in 250 s CPU time has a makespan of 67 TU (Figure 3a). Otherwise, the heuristic rules have been implemented in C++ as an integrated module of the scheduling system developed by Cantón (2003). The schedule obtained has about 38% increase in makespan but is reached in 0.3 s CPU time (Figure 3b). In addition, metaheuristic optimisation algorithms, such as simulated annealing or genetic algorithm, allow reducing makespan to about 1.5% in 5 s CPU time (Figure 3c). Despite the fact that the resources usage is not optimum, the computational requirements are significantly lower. This entails concluding that processes which require long calculation times for solving the exact approach can have a good first estimate through the use of heuristic rules and metaheuristic optimisation methods.

It has been observed that introducing the solution of the heuristic procedure as an upper bound to the exact approach increases the rate of convergence to the solution and reduces the optimality gap for a given calculation time. Namely, in a smaller case study where optimality was reached in 186 s CPU with less than 1% optimality gap, when introducing as an upper bound the metaheuristic solution, which was about 13% worse than the optimum, the computation time reduces to 58 s CPU to get the same 1% final optimality gap. The overall time savings is about 56% calculation time. Therefore, combining both approaches offers the best solution compromise.

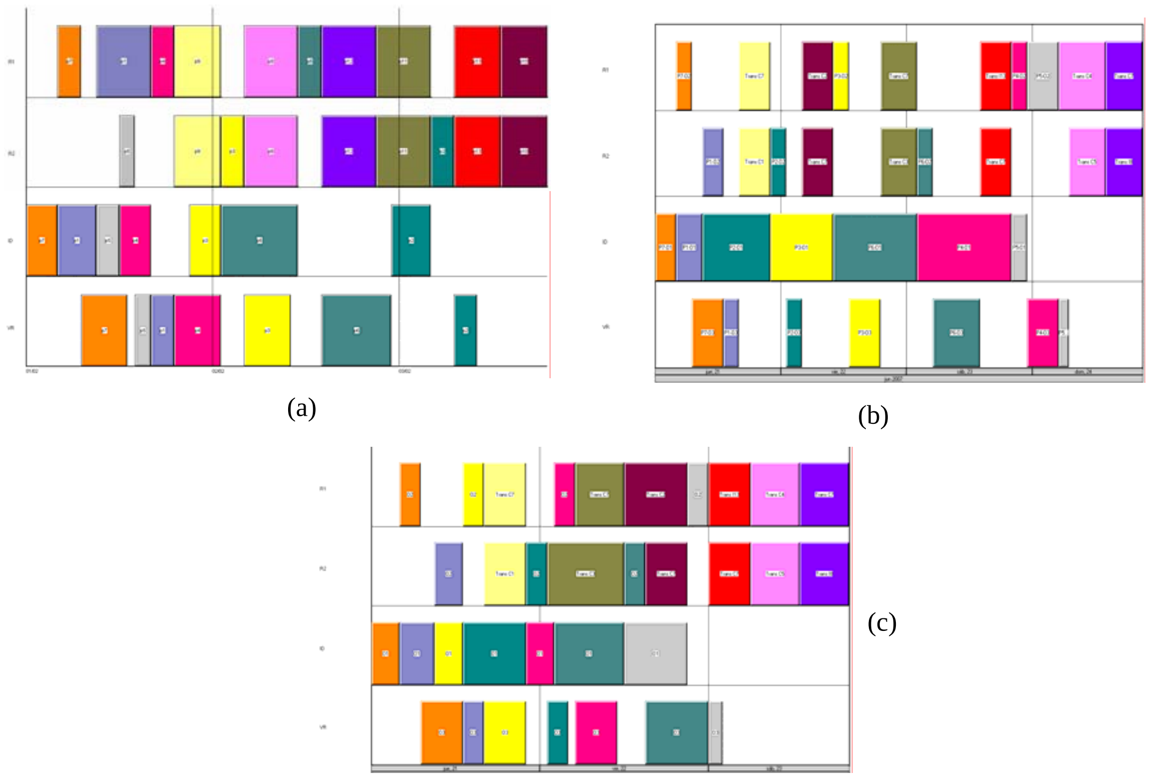
(a) (b) (c)

Figure 3. Gantt chart representation for case study: (a) Mathematical approach (b) Heuristic approach (c) Metaheuristic optimisation approach

4.1. Uncertainty

The main data have been kept the same as those in the case study. Tables 4 and 5 present stochastic data and probability related to the random operations. Twenty random scenarios have been sampled by Monte Carlo sampling to be studied.

The mathematical model has been implemented in GAMS and solved using the MILP solver of CPLEX (9.0) on AMD Athlon 3000 computer. The final used schedule was obtained after 36000 s CPU calculation time. Results show that the difference between applying the nominal case study and applying the stochastic solution to the randomly selected scenarios, that is to say, the value of the stochastic solution (Birge, 1997) is about 2 seconds (Figure 4), which represents a 2,8% of the total makespan. This means that every hour, on average, an additional unit could be produced if the stochastic plan was implemented instead of the nominal case.

		s1	s2	s3
o1	p1	3	5	8
	p2	3	5	8
	p3	2	4	6
	p4	2	4	6
	p5	2	3	5
	p6	4	6	8
	p7	2	4	6
o7-o8	p8	3	4	6
	p9	2	3	4
	p10	2	3	4
	p11	3	4	6
	p12	3	4	6
	p13	2	3	4

Table 4. Case study stochastic operation times (in s).

	s1	s2	s3
O1	0.1	0.3	0.6
O7-o8	0.6	0.3	0.1

Table 5. Probability of each representation of random operation times.

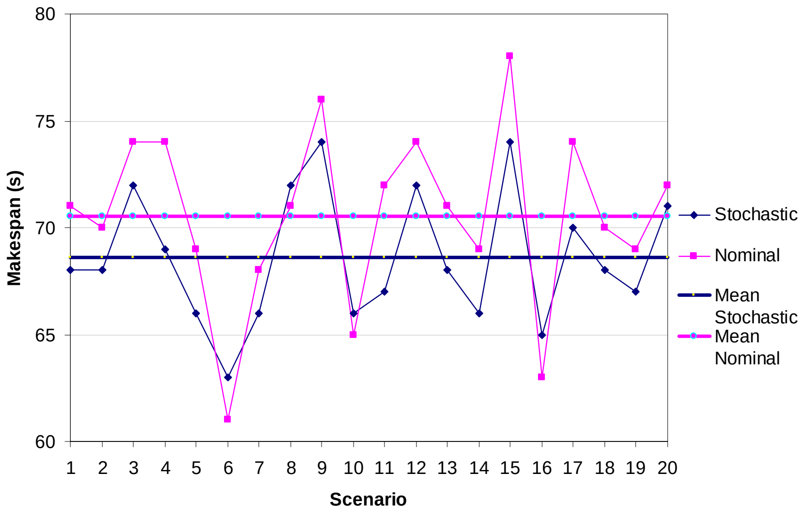

Figure 4. Makespan comparison for the application of the nominal and stochastic solutions to random scenarios

5. Conclusions

Flexible multipurpose back-up chemical/assembly process systems are very attractive for the efficient management of resources and to replace labour in short series production and production peaks. However, their optimum production scheduling implies large computational effort due to the combinatorial nature of the problem. The use of a hybrid approach that exploits the capabilities of rigorous and heuristic methods offers the optimum strategy.

Moreover, the stochastic approach offers significant benefits to the average scenario realization for back-up chemical/assembly process systems. Although the application of the stochastic model entails more complexity to the problem and higher computational resources consumption, it allows to reduce overall makespan and to improve system production robustness and efficiency.

Acknowledgements

Financial support received from "Generalitat de Catalunya" (project no. I0898), Univ. Pol. de Catalunya (UPC research grant) and European Community (project PRISM-MRTN-CT-2004-512233) is fully appreciated. Partial support from Spanish Ministery of Education and Science, Project DPI2006-05673 is also acknowledged with thanks.

References

1. Birge, J.R.; Louveaux, F. *Introduction to stochastic programming*, Springer, New York (1997)
2. Cantón, J. (2003). PhD. Thesis, Universitat Politécnica de Catalunya.
3. Kaighobadi, M.; Venkatesh, K., (1994) International Journal of Operations & Production Management, 4, 26-49.
4. Méndez, C.A.; Cerdá, J.; Grossmann, I.E.; Harjunkoski, I.; Fahl, M., (2006).Comp. Chem.Eng. 30, 13-916.
5. Reeja, M.K.; Rajendran, C. (2000) International Journal of Production Research, 38 (9), 2051-2066.
6. Sawik, T. (2004). European Journal of Operational Research, 154, 1-19.
7. Shaik, M.A.; Janak, S.L.; Floudas, C.A. (2006) Industrial Engineering and Chemical Research, 45, 6190-6209.

Nomenclature

<u>Indices</u>

p	components and subassemblies of the product to be produced
e	process stages
o	operations
r	resources (robots, identification and verification equipment, storages, ...)
k	resources related to operations
z	units of resource type k

<u>Sets</u>

C_p	components of product p
P_e	stages related to each component or subassembly of p
O_e	operations included in stage e
R_e	resources that can perform stage e

Z_k	types of resource k
K_o	available resources of type k for operation o
ose	starting operations of the different stages
ofe	final operations of the different stages
osim	simultaneous operations
oseq	consecutive operations of different stages
pseq	components and subassemblies to be done sequentially

<u>Initial data</u>

minot(o)	time of operation o
maxot(o)	maximum wait time of operation o

<u>Variables</u>

z_{obj}	objective function
Mk	makespan value
Tin (o,p)	initial processing time of o of component or subassembly p
Tfn (o,p)	final processing time of o of component or subassembly p
Twt (o,p)	wait time in operation o of component or subassembly p
Y(e,p,r)	binary variable denoting that stage e of piece p is assigned to robot r
X_O(e,e’,p,p’)	binary variable denoting that stage e of component or subassembly p is done before stage e’ of component or subassembly p’
V(o,p,z)	binary variable denoting that operation o of component or subassembly p consumes resource z