IX Congreso de Ingeniería de Organización
Gijón, 8 y 9 de septiembre de 2005

Programación de un proceso sin esperas con varias etapas en serie dotadas de máquinas en paralelo

Manuel Mateo Doll[1], Imma Ribas Vila[1], Ramon Companys Pascual[1]

[1] Dpto. de Organización de Empresas. Escuela Técnica Superior de Ingeniería Industrial de Barcelona, 08028 Barcelona. manel.mateo@upc.edu, imma.ribas@upc.edu , ramon.companys@upc.edu

Resumen

Una de las configuraciones que se observa actualmente en un buen número de sistemas productivos es la compuesta por varias etapas en serie cada una de las cuales puede constar de más de una máquina o estación en paralelo. En la literatura este tipo de sistemas se conoce por el nombre de "flow shop híbrido" o "sistemas de flujo híbrido". En este tipo de sistemas productivos, es importante establecer cuál de los recursos en paralelo debe realizar cada una de las operaciones a programar, dado que repercutirá significativamente en la eficiencia del programa. Esto adquiere una mayor importancia cuando los trabajos, debido a los diversos tratamientos que reciben, no deben esperar entre el final de una operación y el inicio de la siguiente. Esta situación particular se conoce como "no-wait", y conlleva que sean claves tanto la asignación como la secuenciación. Se presenta un procedimiento heurístico desarrollado para programar la producción en una empresa dedicada a la fabricación de placas de metacrilato con el objetivo de minimizar los tiempos muertos en los recursos así como un modelo que permite obtener, para ejemplares reducido, la solución óptima mediante programación lineal mixta.

Palabras clave: programación, no-wait, flow-shop híbrido

1. Introducción

En este trabajo se estudia la programación de la producción en una empresa dedicada a la fabricación de placas de metacrilato cuyo proceso productivo (Figura 1) consiste inicialmente en amoldar lotes de 20 placas del mismo espesor en una línea de producción y colocarlas en una torre, que a partir de este momento constituye la unidad productiva.

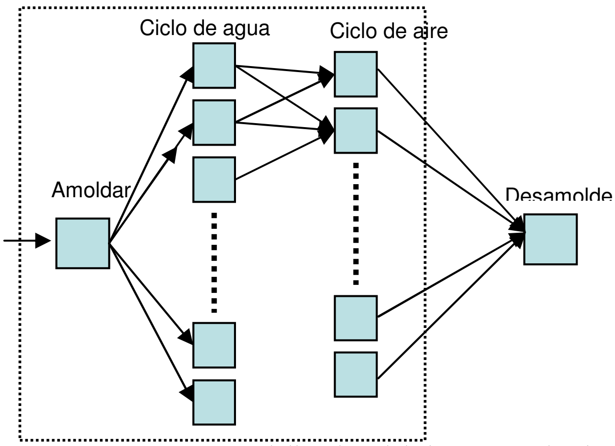


Figura 1. Modelización del sistema productivo.

A continuación las torres se someten a un ciclo de polimerización en agua (en uno de los baños en paralelo) e, posteriormente, a un ciclo de secado (en una de las estufas en paralelo) para, finalmente, proceder al desamolde de las mismas.

Las torres tienen que pasar inmediatamente de la línea inicial de producción a un baño y de un baño a una estufa, sin sufrir demora alguna. Las placas pueden presentar diferentes espesores, con lo cual los tiempos de operación también son diversos (existen variaciones en los tiempos de proceso mayores al 650% para una misma etapa (Tabla 1)). La limitación de los recursos disponibles en cada etapa y la gran diferencia en los tiempos de proceso de los diferentes artículos dotan al proceso de programación de una mayor complejidad.

Tabla 1. Tiempos de Ciclo.

Espesor (mm)	Amolde (min.)	Ciclo Agua (min.)	Ciclo Aire (min.)
2	60	210	240
3	60	240	240
4	60	240	240
5	60	240	240
6	60	240	240
7	60	300	300
8	70	330	480
10	75	435	480
12	75	540	540
13	75	660	540
15	85	840	600
18	100	1800	660
20	120	1800	660
25	120	1440	660

El problema de programación de la producción en un sistema productivo compuesto por diferentes etapas en serie cada una de las cuales puede constar de más de un recurso en paralelo, es conocido en la literatura como *flow-shop* híbrido o según Brah y Hunsucker (1991), como *flow-shop* con procesadores múltiples (FSMP). En este tipo de sistemas, es tan importante la secuenciación de piezas en cada máquina como la asignación de éstas a una de los diferentes recursos de la etapa (carga). La asignación además es más crítica cuando las operaciones deben realizarse sin demoras intermedias, situación que recibe el nombre de "*no-wait*", Pinedo (1995).

La programación de la producción en la fábrica se realiza semanalmente intentando programar toda la producción adjudicada a la semana. El programa de producción debe detallar, para cada torre o producto, tanto la asignación concreta a uno de los recursos alternativos como los intervalos de realización de las operaciones.

En este estudio se ha desarrollado un procedimiento heurístico y un procedimiento exacto que permite obtener una solución óptima en ejemplares de medida reducida.

2. Definición e hipótesis del problema tratado

Se dispone de un conjunto I de lotes a producir durante una semana, que deben procesarse en k etapas diferentes. Cada etapa está formada por M_k máquinas idénticas en paralelo. Se dispone, además, de los tiempos de proceso p_{ik} requeridos por cada lote en cada etapa. La fábrica trabaja *d* días a la semana con *t* turnos cada día.

En esta situación, se deberán programar las operaciones, en cada etapa, con el fin de maximizar el número de lotes a producir durante una semana. Este objetivo se puede evaluar a través de la minimización del C_{max} o asignando un coeficiente de utilidad a cada lote e intentando maximizar la suma de la utilidad de los lotes a producir. En la primera alternativa el programa resultante puede programar lotes con fechas posteriores al límite de programación establecido, ya que si se limita el horizonte de programación se puede caer en programar imposibles de resolver. Los lotes con inicio o fin posterior al límite establecido deberán descartarse del programa final. La segunda alternativa implica asignar a cada lote un coeficiente de utilidad acorde con el objetivo perseguido. En este caso se puede limitar el horizonte del programa y será éste el que propondrá únicamente aquellos lotes capaces de producirse en las fechas acordadas y que contribuyen a maximizar la suma de utilidad.

En este estudio se ha optado por la segunda alternativa que será la utilizada en la formulación del modelo.

2.1. Hipótesis de partida

- Las máquinas y los lotes, en forma de torres, están disponibles en el instante 0.
- Los baños y las estufas no pueden procesar más de un lote a la vez.
- Los lotes deben procesarse de inicio a fin sin interrupciones.
- Todos los baños y todas las estufas son equivalentes.
- Una vez se inicia un ciclo, ya sea de agua o de aire, debe terminar. Es decir, no se permiten interrupciones.

2.2 Formulación del modelo

Se detallan los datos de partida a utilizar en el modelo y se formula el modelo utilizado.

2.2.1. Datos del modelo

Se definirán los índices y los parámetros, para luego definir las variables del modelo.

2.2.1.1 Índices

i,i': índice de los lotes a procesar
j: índice de las máquinas
k: índice de las etapas

2.2.1.2 Parámetros

I: número de lotes a procesar
K: número de etapas
M: número de máquinas
U: número arbitrariamente grande
t: número de turnos por día
d: número de días a programar
$\min_t$: minutos por turno

M_k : máquinas de la etapa k
p_{ik} : tiempo de proceso del lote *i* en la etapa *k*
u_i : utilidad del lote *i*

2.2.2. Modelo

Primero se definen las variables, a continuación la función objetivo, y finalmente las restricciones.

2.2.2.1. Variables

Se definen las siguientes variables:

$$v_i = \begin{cases} 1 & \text{si se fabrica el lote } i \\ 0 & \text{en caso contrario} \end{cases}$$

$$y_{mi} = \begin{cases} 1 & \text{si el lote } i \text{ se procesa en la máquina } m \\ 0 & \text{en caso contrario} \end{cases}$$

$$x_{i'im} = \begin{cases} 1 & \text{si el lote } i' \text{ es el precedente inmediato del lote } i \text{ en la máquina } m \\ 0 & \text{en caso contrario} \end{cases}$$

e_{ik} : instante de inicio del lote *i* en la etapa *k*
c_{ik} : instante de fin del lote *i* en la etapa *k*

2.2.2.2. Función Objetivo

La función objetivo pretende maximizar la utilidad de los lotes a fabricar:

$$Maximizar[Z] = \sum_{i=1}^{I} u_i . v_i \qquad (1)$$

2.2.2.3. Restricciones

$$\sum_{i=1}^{I} y_{mi} \cdot p_{ik} \leq d \cdot \theta \cdot t \qquad \begin{array}{l} k = 1...K \\ \forall m \in M_k \end{array} \qquad (2)$$

$$\sum_{m \in M_k} y_{mi} - v_i = 0 \qquad \begin{array}{l} i = 1...I \\ k = 1...K \end{array} \qquad (3)$$

$$\sum_{\substack{i'=0 \\ i' \neq i}}^{I} x_{i'im} - y_{mi} = 0 \qquad \begin{array}{l} i = 1...I \\ k = 1...K \\ \forall m \in M_k \end{array} \qquad (4)$$

$$\sum_{\substack{i'=1 \\ i' \neq i}}^{I+1} \boldsymbol{x}_{\boldsymbol{ii'm}} - \boldsymbol{y}_{\boldsymbol{mi}} = 0 \qquad \begin{array}{l} i = 1...I \\ k = 1...K \\ \forall m \in M_k \end{array} \qquad (5)$$

$$\sum_{i=1}^{I+1} x_{0im} = 1 \qquad \begin{array}{l} k = 1...K \\ \forall m \in M_k \end{array} \qquad (6)$$

$$\sum_{i=0}^{I} x_{iI+1m} = 1 \qquad \begin{array}{l} k = 1...K \\ \forall m \in M_k \end{array} \qquad (7)$$

$$\boldsymbol{e}_{\boldsymbol{ik}} = \boldsymbol{e}_{\boldsymbol{ik}-1} + \boldsymbol{p}_{\boldsymbol{ik}-1} . \boldsymbol{v}_{\boldsymbol{i}} \qquad \begin{array}{l} i = 1..I \\ k = 2...K \end{array} \qquad (8)$$

$$e_{ik} \geq c_{i'k} + U.(x_{i'im} - 1) \qquad \begin{array}{l} i = 1...I \\ i' = 1...I \\ k = 1...K \\ \forall m \in M_k \end{array} \qquad (9)$$

$$c_{ik} = e_{ik} + p_{ik} . v_i \qquad \begin{array}{l} i = i...I \\ k = 1...K \end{array} \qquad (10)$$

$$c_{iK} \leq \theta \cdot t \cdot d \qquad i = 1...I \qquad (11)$$

Las restricciones de tipo (2) limitan la carga asignada a cada máquina a la capacidad existente en la fábrica durante el horizonte de programación. Las restricciones (3) a (7) están relacionadas con la asignación. Las de tipo (3) obligan, si el lote se fabrica, a asignarle una máquina en cada etapa. Las de tipo (4) implican que, una vez asignado un lote a una máquina éste tenga un predecesor que tanto puede ser otro lote como el estado inicial de la máquina. Las de tipo (5) implican que si un lote se asigna a una máquina éste debe tener un sucesor, ya sea otro lote o el estado final de la máquina. Las restricciones tipo (6) inicializan la secuencia en cualquier máquina empezando por el lote 0; las (7), en cambio, sirven para finalizar la secuencia en cualquier máquina, colocando en último lugar el lote I+1. Las restricciones (8) a (11) son responsables de la secuenciación de los lotes en las máquinas. En la restricción (8) se indica que el proceso de un lote en una etapa no puede empezar si no ha terminado su proceso en la etapa anterior, así como la condición *"no-wait"*, lo que implica que el lote deba procesarse de forma continua de inicio a fin. En la restricción (9) se condiciona el inicio de un lote en una etapa al fin de proceso del lote predecesor en dicha etapa. En la restricción (10) se indica que un lote debe procesarse sin interrupciones. Finalmente, en la de tipo (11) se limita el tiempo de fin de proceso de los lotes al horizonte de programación establecido.

2.3. Procedimientos de resolución

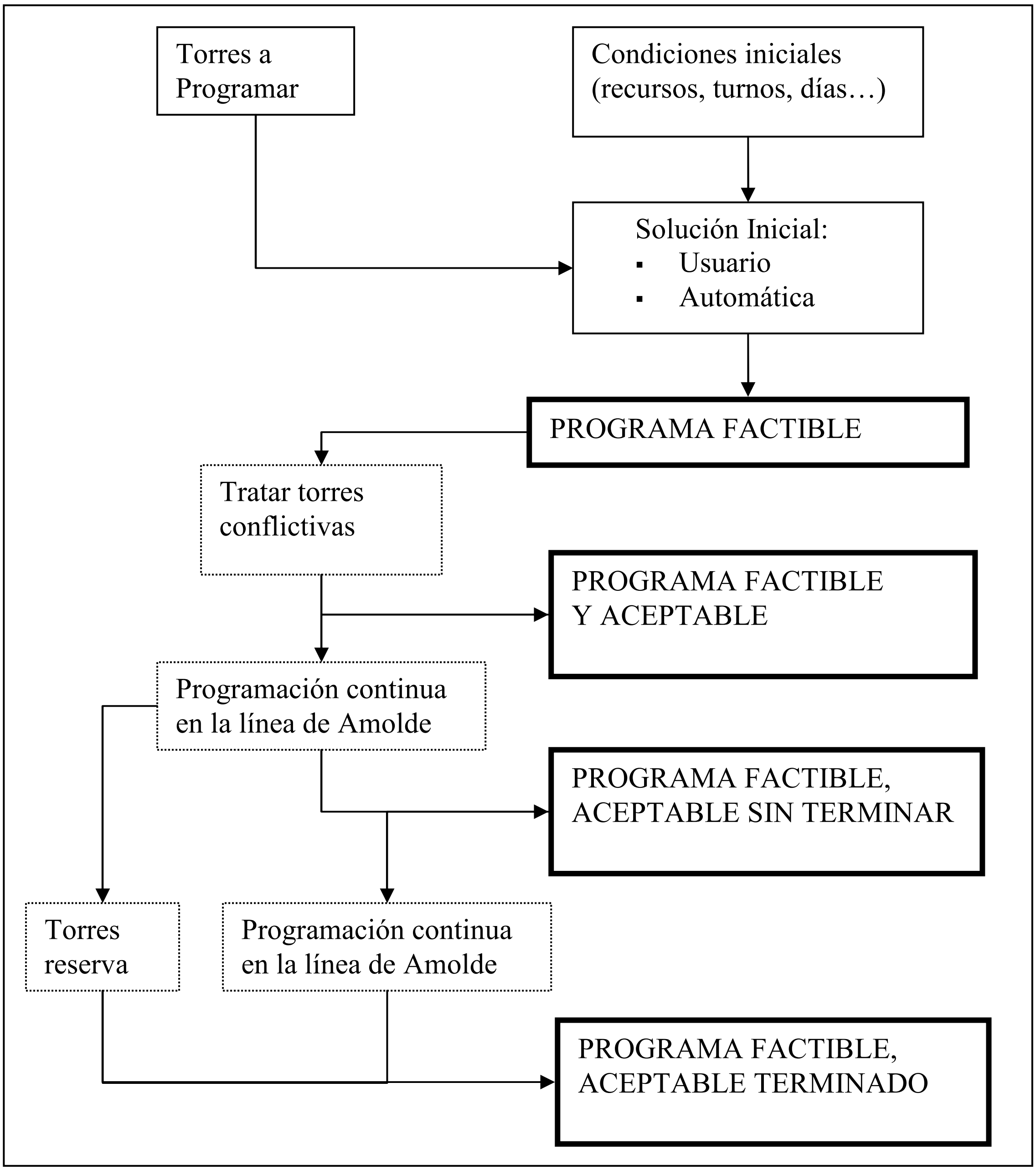
Para resolver el problema planteado se propone un procedimiento heurístico Roglà (1999) que consta de dos partes: un procedimiento para obtener una solución inicial y una mejora mediante un procedimiento de optimización local y un procedimiento exacto que permite solucionar de forma óptima ejemplares de medida reducida.

3. Procedimiento heurístico

Según Companys (2003), diversas subfunciones se asocian a la programación de operaciones, entre las que cabe señalar las de carga, secuenciación y temporización. En este caso la carga corresponde al proceso de asignar un baño y una estufa a cada torre. La secuenciación consiste en ordenar el conjunto de torres a producir. Ambas acciones se resuelven separadamente. El proceso básico de programación de las torres consta de 3 etapas (Figura 2):

- Identificación de las condiciones iniciales
- Obtención de un primer programa factible mediante un algoritmo que es una adaptación de los algoritmos de *dispatching* considerando características particulares del problema.
- Mejora del programa mediante diferentes opciones, que se escogerán en función de las necesidades del usuario, cada opción corresponde a una variante de optimización local.

Figura 2. Esquema de la relación entre las diferentes fases de la programación.



A modo de síntesis, la Figura 2 esquematiza las relaciones existentes entre las diferentes fases del proceso de programación. La terminología utilizada es la siguiente:

- Programa factible: programa en el que todas las torres están asignadas a un baño y a una estufa.
- Programa aceptable: programa factible y en el que el proceso es continuo de principio a fin, es decir no hay paros por saturación de los recursos.
- Torre conflictiva: aquella torre a la que precede una pausa en la línea de amolde y que no tiene asignada ni un baño ni una estufa.
- Torres reserva: conjunto de torres problemáticas que se han separado de un programa y que pueden ser tratadas posteriormente.

El sistema implementado incorpora todas las opciones necesarias para llegar a soluciones, que puedan ser válidas no sólo en el marco teórico sino también en la práctica. De esta forma, durante el proceso de programación el usuario puede optar por aceptar el primer programa, que sujeto a las condiciones de trabajo, amolda las torres introducidas por el usuario tratando de minimizar el tiempo de parada de la línea de amolde (solución factible). Si este primer programa contiene paros en la línea de amolde, el usuario puede optar por extender su análisis y obtener, en una segunda fase, un programa factible y aceptable, sin paradas. Si esta solución aún no satisface al usuario, ya sea por que no se encuentra en estado estacionario o por que no contiene todas las torres introducidas inicialmente, el usuario puede obtener un programa sin pausa, factible y en estado estacionario a través de una tercera fase en la que el sistema substituirá alguna de las torres introducidas, si éstas no pueden dar lugar a programas factibles, aceptables y en estado estacionario, por torres que estén en cartera para ser programadas en semanas posteriores. El tiempo de ejecución del proceso de programación que se obtiene con el sistema desarrollado es del orden de 0,6-0,8 segundos por torre.

Además de la funcionalidad básica, que resuelve el problema de programación, el sistema desarrollado incluye funcionalidad adicional, entre la que destaca la generación de estadísticas, gráficos y la posibilidad de archivado y recuperación de programas. Esta funcionalidad permite agilizar el proceso de análisis para la toma de decisiones.

4. Procedimiento Exacto

El modelo descrito en el apartado 2.2 proporciona soluciones óptimas del problema pero según las primeras experiencias computacionales su ámbito de aplicación eficiente está limitado a un horizonte de planificación reducido. Por ello hemos iniciado otro procedimiento exacto, éste bietápico, que se inspira en el trabajo de Harjunkoski y Grossmann (2002). Estos autores proponen dividir el problema general en un problema de asignación y un problema de secuenciación como se muestra en la Figura 3. Mediante la resolución del problema de asignación, en un primer nivel, se determina de forma óptima los lotes a producir y la asignación de estos lotes a una máquina concreta en cada una de las etapas. En un segundo nivel, mediante el modelo de secuenciación, se secuencian los lotes en las máquinas. Dado que una asignación obtenida en la primera fase puede ser imposible de secuenciar en la segunda, si ello ocurre se añaden cortes al problema de asignación para provocar nuevas asignaciones hasta llegar a una asignación que sea secuenciable.

Tanto el modelo inicial como los subproblemas de asignación y de secuenciación se han modelizado mediante programas lineales mixtos y se han resuelto utilizando el paquete comercial OPL Studio 3.7.

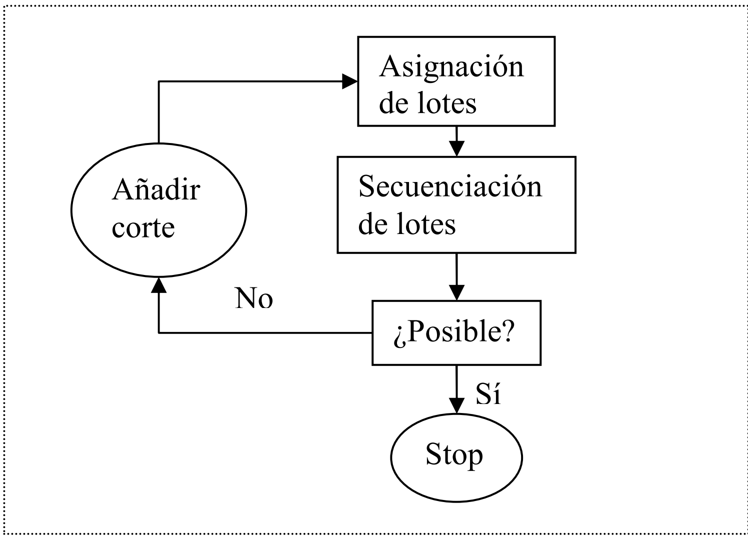


Figura 3. Estrategia de descomposición.

Para la formulación de ambos subproblemas se ha conservado la nomenclatura utilizada en el apartado 2.2.

4.1. Formulación para el problema de asignación

La resolución de este problema indica qué lotes se deben fabricar y en qué máquinas se deben procesar.

4.1.1. Función objetivo

La función objetivo, como en el caso anterior, pretende maximizar la utilidad de los lotes a fabricar:

$$Maximizar[Z] = \sum_{i=1}^{I} u_i . v_i \qquad (12)$$

4.1.2. Restricciones

En este caso, sólo son necesarios dos grupos de restricciones.

$$\sum_{i=1}^{I} y_{mi} \cdot p_{ik} \leq d \cdot \theta \cdot t \qquad \begin{array}{l} k = 1...K \\ \forall m \in M_k \end{array} \qquad (13)$$

$$\sum_{m \in M_k} y_{mi} - v_i = 0 \qquad \begin{array}{l} i = 1...I \\ k = 1...K \end{array} \qquad (14)$$

Las restricciones (13) comprueban la capacidad global para procesar los lotes y las restricciones (14) imponen que si un lote se fabrica se le debe asignar una máquina por etapa.

4.2. Formulación para el problema de secuenciación

Los datos de entrada para la resolución de este problema, además de los parámetros descritos, son los lotes a fabricar y las máquinas en los que se deben procesar. El resultado que se obtiene resolviendo este subproblema es tanto la secuenciación de los lotes en cada máquina como la temporización de los mismos.

4.2.1. Función Objetivo

En este caso la función objetivo elegida consiste en minimizar el tiempo de proceso máximo, $C_{\max}$, que equivale, aproximadamente, a maximizar la productividad del sistema.

$$Minimizar[Z] = C_{\max} \tag{15}$$

4.2.2. Restricciones

En este caso se utilizan las restricciones (4) a (11) más unas restricciones adicionales (15) que indican que $C_{\max}$ es el mayor de los tiempos de finalización de los lotes.

$$c_{ik} \leq C_{\max} \qquad \begin{array}{l} i = 1...I \\ k = 2...K \end{array} \tag{16}$$

4.2.3. Cortes

Se ha estudiado diferentes posibilidades, todas ellas conducentes a ajustar las capacidades productivas en una o varias máquinas, lo que provoca una reasignación de los lotes.

5. Conclusiones

Los sistemas de *flow-shop* híbrido representan con gran fidelidad la mayoría de los sistemas productivos industriales de las PYMES, y generalmente esta estructura está acompañada de restricciones específicas (*no stop*, *no wait*, *blocking*, tiempos de preparación dependientes e independientes de la secuencia, etc.). Sorprendentemente es raro encontrar en la literatura procedimientos prácticos basados en casos reales para proceder a su programación. Por ello todo intento de modelización y de obtención de soluciones es importante, sobre todo si se basa en un caso industrial.

En el este trabajo hemos presentado una situación real, su modelización, su tratamiento a través de procedimientos exactos y un procedimiento heurístico eficiente que está siendo utilizado actualmente. Dada la existencia de otras situaciones parecidas iniciamos hace ya cierto tiempo una línea de investigación al respecto que, a pesar de no haber alcanzado los resultados finales, está aportando sus frutos en diferentes campos.

Las experiencias computacionales parecen indicar la superioridad de los procedimientos heurísticos sobre los exactos en esta temática.

Agradecimientos

Esta investigación ha sido financiada por el proyecto DPI2001-2169 (MCYT).

Referencias

Brah, S.A.; Hunsecker, J.L. (1991). Branch and Bound algorithm for the flow shop with multiple processors, *European Journal of Operational Research*, Vol. 51, pp. 88-99.

Companys, R. (2003), *Secuenciación.* 1ª ed. Publicaciones d'Abast S.L.L.

Guinet, A.G.P.; Solomon, M.M.(1996). Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion time, *International Journal of Production Research*, Vol. 34, No. 6, pp. 1643-1654.

Gupta, J.N.D. (1989), Two-stage hybrid flowshop scheduling problem, *Operational Research Society*, Vol. 39, No. 4, p.p. 359-364

Harjunkoski, I.; Grossmann, I.E. (2002). Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming models, *Computers and Chemical Engineering* Vol. 26, pp. 1533-1552.

Maravelias, C.T.; Grossmann I.E. (2004). A hybrid MILP/CP decomposition approach for the continuous time scheduling of multipurpose batch plants, *Computers and Chemical Engineering* Vol. 28, pp. 1921-1949

Pinedo, M. (1995). Scheduling: *Theory, Algorithms and Systems*. Prince Hall.

Roglà, J., (1999). *Anàlisi de la viabilitat de l'ampliació de la planta de producció d'una PIME mitjançant l'estudi de capacitat de producció en diferents escenaris*, PFC, ETSEIB-UPC.