X Congreso de Ingeniería de Organización
Valencia, 7 y 8 de septiembre de 2006

Un algoritmo exacto para la secuenciación de piezas en una máquina con tiempos de preparación dependientes de la secuencia

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

[1] Dpto. de Organización de Empresas. Universidad Politécnica de Cataluña. Avda. Diagonal, 647, 08028 Barcelona. Imma.ribas@upc.edu, ramon.companys@upc.edu, manel.mateo@upc.edu

Resumen

Se considera el problema de secuenciar trabajos en una única máquina con tiempos de preparación dependientes de la secuencia. Las piezas se agrupan en familias y los tiempos de preparación están asociados a las familias. El objetivo es minimizar el retraso total. Se propone un algoritmo exacto del tipo Bounded Dynamic Programming (BDP) para su resolución que, en general, requiere la exploración de un número menor de vértices que un procedimiento del tipo branch and bound (o de programación dinámica no acotada). Se define un procedimiento de acotación análogo al utilizado para el problema equivalente sin tiempos de preparación. El procedimiento implementado se ha probado sobre un conjunto de ejemplares con diferente número de trabajos y familias. Los resultados obtenidos muestran que el algoritmo es muy competitivo.

Palabras clave: Secuenciación, Retraso total, Bounded Dynamic Programming.

1. Introducción

Se propone un algoritmo exacto del tipo *Bounded Dynamic Programming* (BDP) para el problema de programación de la producción en un entorno productivo con una única máquina y tiempos de preparación que dependen del tipo de trabajo (o familia), tanto de la que se desea fabricar como la de la anterior. El objetivo buscado es la minimización del retraso medio, o, en forma equivalente, la minimización de la suma de los retrasos de las piezas. Este problema se conoce en la literatura, siguiendo la nomenclatura propuesta por Lawler et al (1993) como $1|s_{j,k}|\sum T$. Si se adopta la clasificación propuesta por Allahverdi et al (1999) para los problemas con tiempos de preparación, esta comunicación considera el problema de secuenciación en una máquina con *sequence-dependent batch setup times*.

2. Definición e hipótesis del problema tratado

En el instante inicial, la única máquina así como los *n* trabajos que deben ser procesados están disponibles. Los trabajos o piezas pueden clasificarse en *q* tipos o familias diferentes, siendo $q \leq n$. Cada trabajo debe recibir una operación sin interrupción y la máquina sólo puede procesar un trabajo a la vez. Cada trabajo i ($i=1,..,n$) requiere de un tiempo de proceso p_i, tiene una fecha de vencimiento d_i, y pertenece a una familia *g(i)*, tal que $g(i) \in \{1,2,...,q\}$. Denominamos φ a la familia del último trabajo procesado por la máquina antes del instante inicial (inicialmente la máquina está preparada para la familia φ).

Sea $s_{h,g(i)}$ el tiempo de preparación necesario cuando se debe procesar en la máquina un trabajo de la familia *g(i)* después de uno de la familia *h*. La consideración de tiempos de preparación por familias no implica una pérdida de generalidad ya que si fuera necesario cada trabajo podría definir su propia familia. Dada una secuencia de trabajos, para un trabajo *i* se puede calcular el instante en que termina su proceso c_i y su retraso $T_i = \max\{0, c_i - d_i\}$. El objetivo es encontrar una secuencia que minimice el retraso total (1).

$$[MIN] Z = \sum_{i=1}^{n} T_i \qquad (1)$$

3. Acotación

Se ha desarrollado un procedimiento de acotación, derivado del procedimiento utilizado en el problema $1||\Sigma T$.

La cota resulta de la relajación de algunas de las condiciones que se dan en la realidad, en particular de los valores concretos, p_i, d_i y *g(i)*, de una pieza. El cálculo de la cota desagrega la asociación real y considera unas piezas ficticias, globalmente con los mismos valores de p_i, d_i y *g(i)* pero asociados en ellas de la forma más favorable para evitar retrasos. En una primera cota la agregación ficticia es totalmente libre, y conduce a suponer que se realizan primero piezas de una familia, lo que no exige tiempo de preparación y que, adicionalmente, es la que contiene más piezas. Se supone, además, que las piezas tienen los valores de p_i y d_i situados en orden creciente. Los tiempos de preparación se tienen en cuenta en la forma también más favorable, sin considerar los necesariamente nulos. A esta cota se ~~la~~ llamará Cota Genérica $b^{(1)}$.

Una cota más ajustada puede obtenerse, en general, analizando para cada familia la cota considerando que a dicha familia pertenece la primera pieza realizada, y adoptando como cota global la menor.

Se calcula la cota para cada una de las *q* familias. Para cada cota se considera que se empieza programando un número de piezas ficticias igual al de la familia, precedido del tiempo de preparación correspondiente si la familia es distinta del estado inicial de la máquina. Para el resto de piezas se sigue el mismo procedimiento que en la Cota Genérica. Se obtiene así *q* valores, b_h, y se elegirá la menor entre ellas.

$$b^{(2)} = \min_{h=1,\ldots,q} \{b_h\} \qquad (2)$$

Reteniendo como cota definitiva (3)

$$b = \max\{b^{(1)}, b^{(2)}\} \qquad (3)$$

4. Grafo asociado al problema $1|s_{i,j}|\Sigma T$

La decisión de secuenciar piezas en la máquina se puede representar mediante un grafo **G** multicapa. Un vértice, *v*, en el grafo **G** queda definido por la pareja (**J**, h), donde **J** es un subconjunto de *n* trabajos y *h* es la familia del último trabajo en **J**. Los vértices representan

todas las posibles combinaciones (secuencias) que se pueden construir con los trabajos en **J** de forma que el último trabajo de la secuencia pertenezca a la familia *h*.

Un arco desde el vértice u = (**I** , b) al vértice v = (**J** , h) existe si **J** = **I** ∪{i} con $i \in$ **N** – **I,** siendo *h* la familia del trabajo *i*, tal y como se muestra en la Figura 1.

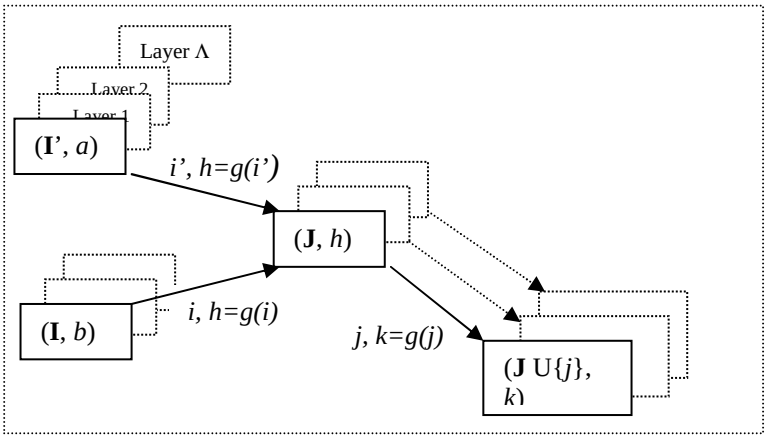

Figura 1. Detalle del grafo multi-capa **G** asociado al problema.

Los vértices en el grafo se pueden organizar en *n+1* niveles. Un vértice v = (**J**, h) pertenece al nivel $|\mathbf{J}|$ que corresponde con el número de elementos en **J.** En el nivel 0, hay un único vértice $\alpha=\{\varnothing, \varphi\}$; en el nivel *n*, hay, potencialmente, *q* vértices $\omega_1, \ldots, \omega_h, \ldots, \omega_q$, donde ω_h = (**N** , h). Un arco va desde un vértice del nivel *j* a un vértice del nivel *j+1*. Un camino desde α a ω_h equivale a una secuencia de *n* trabajos, donde el último trabajo de la secuencia pertenece a la familia *h*. El objetivo es encontrar un camino desde α al vértice ω_h en el nivel *n*, tal que el retraso total de la secuencia asociada sea mínimo.

Asociado a cada vértice *v*, hay una o más de una evaluación bidimensional $EV[v]=\{f(v),\tau(v)\}$, siendo f(v) el retraso total de la secuencia de trabajos y $\tau(v)$ el instante de finalización del último trabajo de la secuencia. Cada camino desde α a *v* puede llevar a evaluaciones diferentes EV[*v*].

Llamaremos *capa* a cada una de las evaluaciones retenidas en un vértice y Λ al número de capas de un vértice. Por lo tanto, las capas de un vértice *v* son $v^{(1)}, v^{(2)}, \ldots, v^{(\Lambda)}$.

Sea v = (**J** , h) el vértice siguiente inmediato del vértice u = (**I** , k). Las evaluaciones de las capas en *u* son $u^{(\lambda)} = \{f^{(\lambda)}(u), \tau^{(\lambda)}(u)\}$ para $\lambda = 1, 2, \ldots, \Lambda$, y cada capa λ del vértice *u* genera una capa en el vértice *v* (Figura 1):

$$\tau^{(\mu)}(v) = \tau^{(\lambda)}(u) + s_{k,h} + p_i \qquad \text{con } g(i)=h \qquad (4)$$

$$f^{(\mu)}(v) = f^{(\lambda)}(u) + \max\{\tau^{(\mu)}(v) - d_i , 0\} \qquad (5)$$

Algunas capas del vértice *v* pueden ser dominadas y, en consecuencia, eliminadas. Dadas dos capas diferentes de *v*, por ejemplo $\{f(v) , \tau(v)\}$ y $\{f'(v) , \tau'(v)\}$, que se corresponden con dos caminos diferentes, la primera domina a la segunda si $f(v) \le f'(v)$ y $\tau(v) \le \tau'(v)$. Las capas dominadas no se consideran y únicamente las capas Pareto-eficientes permanecen en el vértice (Figura 2).

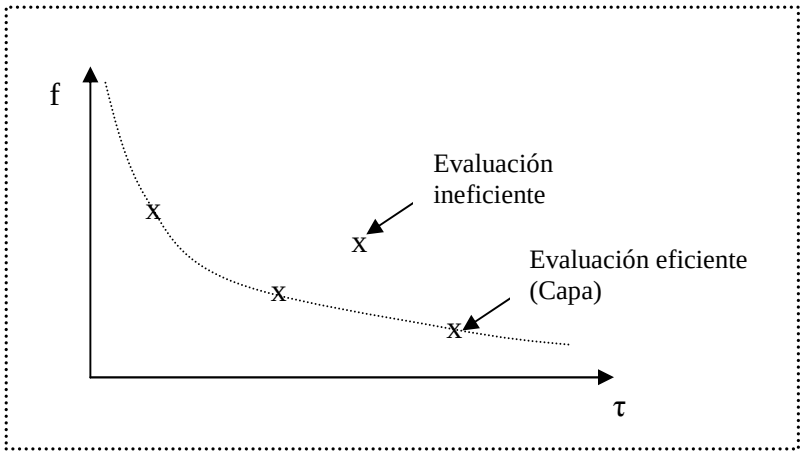

Figura 2. Evaluación bidimensional del vértice *v*

Cada capa $v^{(\lambda)}$ tiene, además de los dos términos utilizados en la evaluación, los siguientes atributos: la cota del retraso de la prolongación $b(v^{(\lambda)})$; la capa $P(v^{(\lambda)})$, desde la que es generada (padre); y la cota del retraso total del camino completo que va desde α hasta dicha capa $f(v^{(\lambda)}) + b(v^{(\lambda)})$.

En este caso, la determinación del camino mínimo se puede calcular a través de un algoritmo similar al de Held and Karp (1962) para el caso en que no hay tiempos de preparación. Puede considerarse que este algoritmo utiliza la ecuación recurrente prototipo de la programación dinámica (6):

$$f(\mathbf{J}) = \min \{ \max\{\tau(v) - d_i , 0\} + f(\mathbf{J} - \{i\}) \mid \forall i \in \mathbf{J}\} \qquad (6)$$

donde $\tau(v) = \sum_{i \in \mathbf{J}} p_i$ (ya que a *v* y **J** están unívocamente asociados).

Las diferencias principales respecto a la situación afrontada por Held and Karp (1962) son dos:

1. Un vértice queda definido por la pareja (**J** , *h*) y no únicamente por **J**.
2. La evaluación no es unidimensional, $f(v) = f(\mathbf{J})$, sino bidimensional $\left\{f^{(\lambda)}(u), \tau^{(\lambda)}(u)\right\}$, ya que τ queda definido de forma ambigua dado **J**, y es necesario evaluar los vértices siguientes inmediatos. Para el objetivo tenido en cuenta, interesa el vértice que tenga la mejor evaluación EV(*v*), por ejemplo el que tenga el mínimo retraso generado por los trabajos programados *f(v)* y el instante de disponibilidad menor para iniciar el resto de trabajos *τ(v)*. Desafortunadamente, es posible que un vértice con el mínimo valor de *f(v)* no coincida con el mínimo valor de *τ(v)*, y, en consecuencia, se deben considerar el conjunto de evaluaciones Pareto-eficientes denominadas capa (Figura 2).

Los algoritmos para encontrar el camino óptimo no requieren la construcción del grafo completo. Los vértices y sus capas asociadas se construyen nivel a nivel. Las dominancias se pueden aplicar en dos sentidos: para reducir el número de capas de un vértice o para reducir el

número de capas entre diferentes vértices. Para este último caso se puede tener en cuenta el Teorema 1.

Teorema 1: Una capa $v^{(\lambda)} = \{f^{(\lambda)}(v), \tau^{(\lambda)}(v)\}$ asociada a un vértice $v = (\mathbf{J}, h)$ domina una capa $w^{(\mu)} = \{f^{(\mu)}(w), \tau^{(\mu)}(w)\}$ asociada a un vértice $w = (\mathbf{J}, k)$ con $h \neq k$ si $f^{(\lambda)}(v) \leq f^{(\mu)}(w)$ y $\tau^{(\lambda)}(v) + s_{h,k} \leq \tau^{(\mu)}(w)$. Una capa dominada, según este teorema, se puede eliminar.

La demostración es obvia y similar a la de Emmons (1969). No se puede alcanzar una solución mejor que la obtenida desde $v^{(\lambda)}$ desde una capa dominada $w^{(\mu)}$.

Corolario: Un vértice con todas las capas eliminadas se puede eliminar.

Si un vértice es eliminado no se considera para extender caminos en el grafo **G.**

Desafortunadamente, el número de vértices y capas activas puede ser demasiado grande para construir eficientemente el camino óptimo. Se puede formular un concepto más fuerte de dominancias utilizando las cotas, que es en lo que se basa el procedimiento de la BDP.

5. Algoritmo BDP progresivo

Inicialmente se fijan las tres características que requiere el algoritmo BDP: *Solución Inicial*, *Cota Inferior* y *Ancho de Ventana*. Las dos primeras características pueden evitar la inspección de un número considerable de vértices que no llevan a buenas soluciones. En cambio el *Ancho de Ventana* limita el número de capas activas a explorar. Si se consideran todas las capas activas a lo largo de la exploración, un algoritmo BDP lleva a la solución óptima, pero un algoritmo de este tipo tiene una complejidad exponencial. Dado que el número de capas de un nivel puede ser impracticable para el procedimiento general de la BDP el algoritmo implementado permite explorar un número máximo de capas por nivel. A esta limitación se la llama *Ancho de ventana*. Si este tamaño de ventana no permite explorar todas las capas activas, el procedimiento actúa de forma heurística.

El procedimiento de acotación explicado en el apartado 4 permite construir una cota inferior $b(v^{(\lambda)})$ del retraso de las prolongaciones desde la capa $v^{(\lambda)}$. Una prolongación es la subsecuencia de los $n - |\mathbf{J}|$ trabajos del subconjunto $\mathbf{N} - \mathbf{J}$ tomando $\tau^{(\lambda)}(v)$ como instante inicial. Previamente se puede establecer que si z_0 es el retraso total de la solución obtenida mediante una heurística, una capa tal que $f^{(\lambda)}(v) + b(v^{(\lambda)}) \geq z_0$ puede se eliminada.

5.1. Solución inicial para la BDP

Dado que la eficiencia del algoritmo está estrechamente relacionada con el número de vértices, una buena solución inicial puede evitar inspeccionar un número considerable de vértices. Un procedimiento heurístico bueno y rápido puede construir un camino desde α hasta ω, siendo z_0 su retraso.

El procedimiento heurístico utilizado en nuestra aplicación ha sido del tipo GRASP.

5.2. Cota inferior de una capa

Para calcular la cota inferior de una capa se calculan las cotas por familia considerando únicamente el conjunto de trabajos no programados $\overline{\mathbf{J}} = \mathbf{N} - \mathbf{J}$ y $\overline{\mathbf{G}}$ el subgrafo de G cuyos vértices corresponden a las familias de los trabajos en $\overline{\mathbf{J}}$. La familia del último trabajo en la secuencia de trabajos de la capa $v^{(\lambda)}$ del vértice v = (**J**,h) es *h*. Los valores $\overline{P}_t$ y $\overline{P}'_t$ se calculan cogiendo $\tau^{(\lambda)}(v)$ como instante inicial.

5.3. Tamaño de ventana por nivel

Sea W el tamaño de ventana. Si el número de capas excede el valor W, sólo se guardan los mejores W candidatos. Este procedimiento es usado de forma dinámica. Existe una lista con W posiciones que guardan las capas activas. Una capa candidata únicamente entrará en la lista, si ésta está llena, si puede sustituir a la mejor capa guardada; "peor" y "mejor" se refiere, normalmente, a la cota inferior global de la capa, pero esto no es obligatorio.

Guardar la mejor cota inferior global de las capas descartadas permite conocer la optimalidad de la solución, a pesar de la simplificación. En caso contrario, permite conocer una cota inferior de la solución óptima.

5.4. Estructura General de la BDP

A continuación se describe el procedimiento de la BDP progresiva para el problema en estudio.

Algoritmo BDP

Paso 0. Generar relación de precedencias entre trabajos siguiendo las reglas de dominancia.

Calcular una solución heurística z_0 y calcular la cota *b*.

nivel=0

Crea un vértice abierto $\alpha=\{\varnothing, \varphi\}$

Create una capa $\alpha^{(1)} = \{ f = 0, \tau = 0, b, P = \_ \}$.

Contador_capas=1

Paso 1. Coger un vértice abierto. Si no hay ninguno ir al Paso 9.

Paso 2. Coger un capa inexplorada. Si no hay ninguna, ir al Paso 8.

Paso 3. Mientras $\exists\, i \in \mathbf{J} - \mathbf{N}$, generar un nuevo candidato (a vértice y/o a capa) añadiendo $i \in \mathbf{J} - \mathbf{N}$ al subconjunto de trabajos **J** basándose en las reglas de dominancia. Si no es posible, ir al Paso 7.

Paso 4. Si no existe un vértice v con $\left(J + \{i\}, g(i)\right)$, crear el vértice abierto *v*.

Paso 5. Crear un candidato a capa $v^{(\mu)}$ y calcular $\tau^{(\mu)}$, $f^{(\mu)}$, $b^{(\mu)}(v)$.

si $\exists v^{(\lambda)}$ tal que $f^{(\lambda)}(v) = f^{(\mu)}(v)$ y $\tau^{(\lambda)} > \tau^{(\mu)}$,

o $f^{(\lambda)}(v) > f^{(\mu)}(v)$,

o $Contador\_capas > W \;\; \forall v \in nivel \;\; \max\{f^{(\lambda)}(v)\} > f^{(\mu)}(v)$,

entonces μ replaza λ.

Si todas las capas de un vértice han sido eliminadas, cerrar el vértice;

Sino crear la capa $v^{(\mu)}$; *Contador_capas*= *Contador_capas*+1

Ir al Paso 3.

Paso 6. Si $f^{(\mu)}(v) + b^{(\mu)} \geq z_0$, o $\exists v^{(\lambda)}$ tal que $f^{(\lambda)}(v) < f^{(\mu)}(v)$ y $\tau^{(\lambda)}(v) < \tau^{(\mu)}(v)$,

o $Contador\_capas > W \;\; \forall v \in nivel \;\; \max\{f^{(\lambda)}(v)\} < f^{(\mu)}(v)$,

o $v^{(\mu)}$ es dominada por $w^{(\rho)}$ según el Teorema 1,

entonces descartar $v^{(\mu)}$ e ir al Paso 3.

Paso 7. Si existe otra capa inexplorada, ir al Paso 2.

Paso 8. Si existe otro vértice abierto, ir al Paso 1.

Paso 9. nivel=*nivel +1. Contador_capas=0*

Paso 10. Si *nivel* < *n,* ir al Paso 1.

Paso 11. Si *nivel* = *n* y no hay ningún vértice abierto, *Solucion* = $z_{0;}$

sino *Solucion* = $\min\{f^{(\lambda)}(v)\} \;\; \forall \lambda, \;\; \forall v \in nivel$.

6. Experiencia Computacional

Se ha construido una colección de ejemplares dividida en 9 grupos de 20 ejemplares cada uno. Cada grupo es una combinación de 15, 20 y 25 trabajos que pertenecen a 4, 5 y 6 familias diferentes.

El generador de ejemplares se basa en tres parámetros $\lambda_1, \lambda_2, \lambda_3$ usados para calcular S_{max}, d_{max} y d_{min}, respectivamente. El tiempo de proceso de los trabajos sigue una distribución uniforme entre [1,P_{max}], con P_{max}=25; el tiempo de preparación está uniformemente distribuido entre [1,S_{max}], donde $S_{max} = \lambda_1 P_{max}$; las fechas de vencimiento siguen una distribución uniforme entre [d_{min}, d_{max}], con $d_{max} = \lambda_2 fv$, $d_{min} = \lambda_3 fv$ y $fv = \sum_{i=1}^{n} p_i + \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} s_{ij}}{q}$ para cada ejemplar. Una vez fijados *n* y *q*, se elige al azar, entre 1 y q, la familia para la que está preparada la máquina y lo mismo se hace para fijar la familia de las diferentes trabajos. En este conjunto de ejemplares se han fijado los siguientes valores: $\lambda_1 = 0{,}5$, $\lambda_2 = 1$ y $\lambda_3 = 0{,}25$. Una vez creado un ejemplar se aplica una heurística sencilla que permita descartarlo si su retraso es cero.

Todos los test realizados se han hecho en un Pentium IV con 512 MB RAM y un procesador de 2.8 GHz. Las Tablas 1, 2 y 3 comparan, sobre un conjunto de ejemplares formados por 15, 20 y 25 trabajos respectivamente, la mejora conseguida por el algoritmo BDP con una ventana de 100 capas (W100) y con una ventana de 50 capas (W50) por nivel. Las columnas

de estas tablas indican: en la columna No se encuentra el número de ejemplar; Min indica el retraso mínimo encontrado indicando con un * que la solución es óptima; GRASP, W100 y W50, indican el porcentaje de desviación respecto a la mejor solución obtenida, calculada como $x = \frac{Alg - Min}{Min} * 100$, donde *Alg* es la solución conseguida por los algoritmos y Min es el mínimo retraso obtenido.

Los resultados indican que la BDP con W100 obtiene, en general, mejores resultados, pero cabe remarcar que en algunos ejemplares el mejor valor es obtenido por W50. Esto se puede deber a que al limitar el número de capas por nivel se puede descartar caminos que llevan a buenas soluciones.

Tabla 1. Resultados para los ejemplares con n=15.

	Familias=4				Familias=5				Familias=6			
No	Min	GRASP (%)	W100 (%)	W50 (%)	Min	GRASP (%)	W100 (%)	W50 (%)	Min	GRASP (%)	W100 (%)	W50 (%)
1	15	0,0	0,0	0,0	46	8,7	0,0	0,0	152	26,3	0,0	0,0
2	45	0,0	0,0	0,0	60	6,7	0,0	6,7	90	0,00	0,0	0,0
3	16	0,0	0,0	0,0	225	13,8	0,0	5,3	4	125,0	0,0	125,0
4	44	0,0	0,0	0,0	276	0,4	0,0	0,0	160	2,50	0,0	1,9
5	172	0,0	0,0	0,0	123	0,0	0,0	0,0	77	0,00	0,0	0,0
6	36	0,0	0,0	0,0	55	0,0	0,0	0,0	139	19,4	0,0	15,1
7	115	6,1	0,0	0,0	11	27,3	0,0	27,3	229	0,0	0,0	0,0
8	45	0,0	0,0	0,0	28	0,0	0,0	0,0	64	0,0	0,0	0,0
9	257	3,1	0,0	0,0	11	27,3	0,0	27,3	148	10,1	0,0	10,1
10	198	6,1	0,0	0,0	118	0,0	0,0	0,0	83	0,0	0,0	0,0
11	87	0,0	0,0	0,0	32	0,0	0,0	0,0	51	3,9	3,9	0,0
12	18	0,0	0,0	0,0	29	0,0	0,0	0,0	58	0,0	0,0	0,0
13	46	0,0	0,0	0,0	117	0,0	0,0	0,0	111	12,6	0,0	0,0
14	61	0,0	0,0	0,0	64	0,0	0,0	0,0	61	0,0	0,0	0,0
15	39	0,0	0,0	0,0	12	0,0	0,0	0,0	28	0,0	0,0	0,0
16	112	7,1	0,0	0,0	9	0,0	0,0	0,0	19	0,0	0,0	0,0
17	41	0,0	0,0	0,0	*12	0,0	0,0	0,0	20	0,0	0,0	0,0
18	193	0,0	0,0	0,0	35	2,9	0,0	2,9	53	0,0	0,0	0,0
19	187	0,0	0,0	0,0	134	13,4	0,0	1,5	36	0,0	0,0	0,0
20	37	0,0	0,0	0,0	45	0,0	0,0	0,0	5	20,0	0,0	0,0

Tabla 2. Resultados para los ejemplares con n=20.

	Familias=4				Familias=5				Familias=6			
No	Min	GRASP (%)	W100 (%)	W50 (%)	Min	GRASP (%)	W100 (%)	W50 (%)	Min	GRASP (%)	W100 (%)	W50 (%)
1	37	8,1	0,0	8,1	34	29,4	0,0	29,4	235	6,9	0,0	0,0
2	43	0,0	0,0	0,0	220	0,0	0,0	0,0	320	0,0	0,0	0,0
3	110	0,0	0,0	0,0	26	0,0	0,0	0,0	1	300,0	0,0	300,0
4	44	0,0	0,0	0,0	89	0,0	0,0	0,0	184	0,0	0,0	0,0
5	177	0,0	0,0	0,0	136	0,0	0,0	0,0	40	50,0	0,0	50,0
6	87	0,0	0,0	0,0	26	0,0	0,0	0,0	77	0,0	0,0	0,0
7	85	0,0	0,0	0,0	18	88,9	0,0	0,0	57	21,1	0,0	1,8
8	182	0,0	0,0	0,0	274	1,8	0,0	1,1	21	0,0	0,0	0,0
9	95	0,0	0,0	0,0	82	0,0	0,0	0,0	75	1,3	1,3	0,0
10	56	0,0	0,0	0,0	50	0,0	0,0	0,0	76	0,0	0,0	0,0
11	24	0,0	0,0	0,0	206	0,0	0,0	0,0	188	4,7	0,0	4,3
12	224	0,0	0,0	0,0	178	0,0	0,0	0,0	200	0,0	0,0	0,0
13	42	19,1	0,0	19,0	263	4,6	1,1	0,0	24	0,0	0,0	0,0
14	20	0,0	0,0	0,0	130	0,0	0,0	0,0	99	22,2	0,0	22,2
15	18	38,9	0,0	38,9	80	0,0	0,0	0,0	397	4,0	0,0	4,0
16	158	0,0	0,0	0,0	150	0,0	0,0	0,0	215	10,2	0,0	4,7

17	49	0,0	0,0	0,0	77	0,0	0,0	0,0	156	0,0	0,0	0,0
18	23	0,0	0,0	0,0	51	27,4	0,0	7,8	11	100,0	0,0	100,0
19	8	50,0	0,0	0,0	200	0,0	0,0	0,0	55	0,0	0,0	0,0
20	42	0,0	0,0	0,0	42	2,9	0,0	0,0	184	0,0	0,0	0,0

Tabla 3. Resultados para los ejemplares con n=25.

	Familias=4				Familias=5				Familias=6			
No	Min	GRASP (%)	W100 (%)	W50 (%)	Min	GRASP (%)	W100 (%)	W50 (%)	Min	GRASP (%)	W100 (%)	W50 (%)
1	32	0,0	0,0	0,0	61	0,0	0,0	0,0	62	8,1	0,0	0,0
2	137	0,0	0,0	0,0	463	0,0	0,0	0,0	232	14,7	0,0	0,0
3	63	0,0	0,0	0,0	222	9,0	0,5	0,0	174	0,0	0,0	0,0
4	289	26,3	0,0	26,3	18	0,0	0,0	0,0	85	0,0	0,0	0,0
5	69	0,0	0,0	0,0	135	28,1	0,0	4,4	125	0,0	0,0	0,0
6	9	0,0	0,0	0,0	40	0,0	0,0	0,0	139	0,7	0,0	0,7
7	209	4,3	0,0	3,8	40	0,0	0,0	0,0	184	0,0	0,0	0,0
8	151	0,0	0,0	0,0	237	0,0	0,0	0,0	313	0,0	0,0	0,0
9	186	0,0	0,0	0,0	112	0,0	0,0	0,0	172	0,0	0,0	0,0
10	76	0,0	0,0	0,0	317	7,5	0,0	7,6	73	0,0	0,0	0,0
11	173	0,0	0,0	0,0	129	8,5	7,8	0,0	214	0,0	0,0	0,0
12	38	0,0	0,0	0,0	130	18,5	0,0	6,9	521	1,9	0,0	0,0
13	42	23,8	0,0	0,0	262	0,0	0,0	0,0	239	0,0	0,0	0,0
14	49	0,0	0,0	0,0	109	63,3	0,0	30,3	72	0,0	0,0	0,0
15	56	0,0	0,0	0,0	219	0,0	0,0	0,0	289	0,0	0,0	0,0
16	23	0,0	0,0	0,0	24	79,2	0,0	79,2	245	0,0	0,0	0,0
17	73	0,0	0,0	0,0	202	0,0	0,0	0,0	21	0,0	0,0	0,0
18	131	0,0	0,0	0,0	117	0,0	0,0	0,0	108	25,0	0,0	25,0
19	189	0,0	0,0	0,0	81	0,0	0,0	0,0	81	0,0	0,0	0,0
20	605	0,0	0,0	0,0	355	0,0	0,0	0,0	82	0,0	0,0	0,0

Analizando separadamente los resultados para cada conjunto de ejemplares, se puede apreciar que la mejora que se consigue con W100 crece a medida que crece el número de familias. La Tabla 4 muestra el porcentaje medio de desviación relativa para cada grupo de ejemplares. Cabe señalar que en el grupo con 25 trabajos, los ejemplares con 6 familias rompen la tendencia, probablemente debido al azar.

Tabla 4. Porcentaje de Mejora media respecto al retraso mínimo.

	Familias =4			Familias =5			Familias =6		
Trabajos	GRASP (%)	W100 (%)	W50 (%)	GRASP (%)	W100 (%)	W50 (%)	GRASP (%)	W100 (%)	W50 (%)
15	1.12	0.00	0.00	5.02	0.00	3.55	11.00	0.20	7.61
20	5.80	0.00	3.30	7.73	0.06	1.92	25.98	0.07	24.35
25	2.72	0.00	1.51	10.71	0.41	6.42	2.52	0.00	1.29

Referencias

Allahverdi, A.; Gupta, J.N.D.; Aldowaisan, T. (1999).A review of scheduling research involving setup considerations. *Omega, International Journal of Management Science*, Vol. 27, pp. 219-239.

Bautista, J.; Companys, R.; Corominas, A. (1996). Heuristic and exact algorithms for solving the Monden problem. *European Journal of Operational Research.* Vol. 88, pp. 101-113

Emmons H. (1969). One-machine sequencing to minimize certain functions of job tardiness. *Operations Research*. Vol. 17, pp. 701-715.

Held M., Karp RM. (1962). A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, Vol. 10, pp. 196-210.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., Shmoys, D. B. (1993). Sequencing and scheduling: algorithms and complexity, in Graves, C. G., Rinnooy Kan, A. H. G., Zipkin, P. (eds) Handbooks in Operations Research and Management Science, 4: Logistics of Production and Inventory, North-Holland, Amsterdam, pp. 445-522.