II International Conference on Industrial Engineering and Industrial Management
XII Congreso de Ingeniería de Organización
September 3-5, 2008, Burgos, Spain

Programación bicriterio para máquinas en paralelo y tiempos de preparación dependientes de la secuencia*

Imma Ribas[1], Ramon Companys[1], Manel Mateo[1]

[1] Dpto. de Organización de Empresas. Escuela Técnica Superior de Ingenieros Industriales de Barcelona. Universitat Politècnica de Catalunya. Avda. Diagonal, 647 08028. Barcelona. imma.ribas@upc.edu, ramon.companys@upc.edu, manel.mateo@upc.edu

Resumen

Se ha estudiado el problema de programación de la producción bicriterio en un entorno de fabricación con máquinas idénticas en paralelo y tiempos de preparación dependientes de la secuencia. Se han implementado seis heurísticas resultantes de la combinación de 3 procedimientos para obtener una solución inicial y dos procedimientos de mejora, uno que actúa sobre el vecindario de una permutación de piezas y otro que actúa sobre el vecindario del problema en cuestión, con el fin de analizar la eficiencia tanto del vecindario como del procedimiento inicial. A partir de los resultados obtenidos, se ha estudiado la eficiencia de los procedimientos para obtener una solución inicial al variar el factor de ponderación de los criterios.

Palabras clave: secuenciación, tiempos de preparación, bicriterio

1. Introducción

El problema de secuenciación en máquinas en paralelo, considera un conjunto de *n* piezas que deben sufrir una operación en una de *m* máquinas idénticas. Cada máquina puede procesar un único trabajo a la vez y cada trabajo se procesa sin interrupción. Las piezas y las máquinas están disponibles en el instante inicial y cada pieza tiene un tiempo de proceso determinado. Se desea establecer la secuencia en cada una de las máquinas que optimice un cierto criterio de eficiencia. En la versión clásica del problema no se consideraba el tiempo necesario para preparar la máquina al cambiar de producto, pero en muchos entornos industriales este tiempo puede ser significativo y debe tenerse en cuenta si se quieren determinar secuencias eficientes.

En la mayoría de los entornos reales, la programación de la producción persigue un objetivo complejo, que considera, implícita o explícitamente, más de un criterio simple de eficiencia. Con ello se pretende dar respuesta a las diferentes preocupaciones, a veces contrapuestas, que un planificador tiene: el nivel de stock y el nivel de servicio ofrecido a los clientes. El primer criterio se puede medir a través del plazo medio de fabricación, $\overline{F}$ y el segundo, a través del retraso medio, $\overline{T}$, los cuales se desea minimizar. Tener en cuenta ambos criterios puede ser

* Este trabajo se enmarca en un proyecto de investigación financiado por el Ministerio de Educación y Ciencia con referencia DPI2007-61371, titulado "Programación de operaciones multicriterio con máquinas paralelas, en varias etapas, sin interrupciones ni almacenajes"

satisfactorio en un entorno industrial; por ello utilizamos la combinación ponderada de dichas medidas, como función objetivo, en la obtención de programas eficientes en un sistema con máquinas idénticas en paralelo y tiempos de preparación dependientes de la secuencia.

El problema de programación bicriterio en máquinas en paralelo considerando los tiempos de preparación ha sido escasamente tratado en la literatura (Webster y Azizoğlu (2001) y Azizoğlu y Webster (2003)) a los que hay que añadir Lin y Jeng (2004) y Cao *et al* (2005) que son los únicos que consideran, además, la programación bicriterio.

2. Definición del problema

El problema considerado se denota, siguiendo la nomenclatura propuesta por T'Kindt y Billaut (2002), como $P\left|s_{h,i}, r_i\right|F_l(\overline{C},\overline{T})$. Se consideran *n* piezas, con una única operación, que pueden procesarse en *m* máquinas idénticas. Las máquinas no pueden procesar más de una pieza a la vez y una vez iniciada la operación no se interrumpe hasta que haya finalizado. Cada pieza *i* tiene asociado un tiempo de proceso p_i, una fecha de vencimiento d_i, y está disponible en el instante r_i para iniciar su operación (la preparación de la misma, si la hubiera, puede haber empezado antes). $s_{h,i}$ es el tiempo de preparación necesario para pasar de la fabricación de la pieza *h* a la pieza *i*; Se supone que la matriz de tiempos de cambio cumple la desigualdad triangular. Se conoce además, el instante, τ_j, en que cada máquina queda libre del trabajo anterior para realizar las operaciones sobre las *n* piezas y, por lo tanto, se debe considerar el estado inicial de cada máquina, φ_j, que corresponde a la última pieza procesada por ésta. Siendo c_i el instante en que la pieza *i* termina su proceso, el retraso viene dado por $T_i = \max\{0, c_i - d_i\}$. El objetivo es encontrar una secuencia de las piezas que minimice la suma ponderada del tiempo de proceso medio y el retraso medio de las piezas.

3. Modelo matemático

El problema considerado es NP-hard dado que el caso $1\left|\right|\sum T$ ya lo es (Du y Leung, 1990). En consecuencia el uso de procedimientos de exploración exactos queda limitado a ejemplares de dimensión reducida. Como base de comparación, hemos construido un modelo de Programación Lineal Mixta (PLM), que se inspira en el propuesto por Guinet y Solomon (1996) para el problema de flow shop híbrido. El modelo implementado utiliza una única variable binaria, x_{ijk}, que toma valor 1 si el trabajo *j* es el precedente del trabajo *i* en la máquina *k*.

$$MIN[z] = \alpha \cdot \sum_{i=1}^{n} c_i + (1-\alpha) \cdot \sum_{i=1}^{n} T_i \tag{1}$$

$$\sum_{i=0}^{n}\sum_{k=1}^{m} x_{ijk} = 1 \qquad \mathrm{j} = 1...\mathrm{n}+1 \tag{2}$$

$$\sum_{\mathrm{j}=1}^{\mathrm{n}+1} \mathrm{x}_{0\mathrm{jk}} = 1 \qquad \mathrm{k} = 1...\mathrm{m} \tag{3}$$

$$\sum_{\mathrm{i}=0}^{\mathrm{n}} \mathrm{x}_{\mathrm{in}+1\mathrm{k}} = 1 \qquad \mathrm{k} = 1...\mathrm{m} \tag{4}$$

$$\sum_{i=0}^{n} x_{ihk} - \sum_{j=1}^{n+1} x_{hjk} = 0 \qquad \begin{matrix} h = 1...n \\ k = 1...m \end{matrix} \tag{5}$$

$$c_i \geq c_i + p_i + \sum_{k=1}^{m} s_{ij} \cdot x_{ijk} + \left(\sum_{k=1}^{m} x_{ijk} - 1 \right) \cdot M \qquad \begin{matrix} 1 = 1...n \\ i \neq j \end{matrix} \tag{6}$$

$$T_i \geq c_i - d_i \qquad \forall i \tag{7}$$

$$T_i \geq 0 \qquad \forall i \tag{8}$$

A través de la función objetivo (1) se quiere minimizar la suma ponderada del retraso medio de las piezas y el tiempo de fabricación medio. El planificador, a través del parámetro α, puede adaptar la programación a las necesidades del momento, penalizando más o menos los retrasos respecto al plazo medio de fabricación.

La ecuación (2) obliga a asignar cada pieza a una máquina. Las ecuaciones (3) y (4) son ecuaciones de contorno. La (3) fija la primera pieza, *j*, asignado a la máquina *k*. La (4) fija la última pieza, *i*, asignada a la máquina *k*. La ecuación (5) secuencia las piezas en las máquinas. La (6) determina el instante de finalización de las piezas teniendo en cuenta el tiempo de preparación necesario para pasar de una pieza a otra. En esta implementación se ha considerado que las máquinas, inicialmente, requieren un tiempo de preparación que depende de la primera pieza que se va a procesar. La ecuación (7) y (8) permiten calcular el retraso asociado a las piezas.

Mediante este modelo hemos podido resolver, únicamente, ejemplares de dimensión muy reducida (6 piezas y dos máquinas). y, por tanto, no se ha podido utilizar para contrastar los resultados obtenidos por los procedimientos heurísticos implementados.

4. Procedimientos heurísticos

El problema de programación multicriterio puede abordarse de diferentes formas tal y como se explica en T'kindt y Billaut (2002). Aquí se ha optado, siguiendo la clasificación de Hwang y Masud (1979), por un método a priori en el que el planificador proporciona el peso que quiere dar a cada uno de los criterios. El procedimiento de programación desarrollado es una adaptación, al caso bicriterio, del procedimiento propuesto en Ribas (2007) para el caso monocriterio. Este procedimiento permite, dada una secuencia inicial, explorar su vecindario mediante el procedimiento de mejora Soft Simulated Annealing (SSA). Por otro lado, y a efectos de poder contrastar los resultados, se ha adaptado también el procedimiento de mejora que explora el vecindario de las máquinas a través de diferentes movimientos.

Se han implementado seis procedimientos heurísticos que son combinación de tres procedimientos para obtener una solución inicial y dos procedimientos de mejora. Los procedimientos propuestos para hallar una solución inicial se basaban en las reglas de prioridad EDD, SPT y en una combinación ponderada de ambas. Los primeros resultados obtenidos en la experiencia computacional desaconsejaron el uso de la fecha de vencimiento como criterio de prioridad ya que, en las colecciones generadas, el retraso medio obtenido era de un orden muy distinto al del tiempo medio de fabricación. Con el fin de analizar el efecto que tiene la solución inicial sobre el resultado final cuando se utiliza un vecindario u otro, se ha optado por implementar un procedimiento basado en la regla *SPT*, implementar un segundo procedimiento, que llamamos *pseudo-SPT*, basado en el anterior, pero introduciendo

una aleatoriedad ponderada y un último procedimiento, que llamamos *Random*, con el que se obtiene una solución inicial dando a todas las piezas la misma probabilidad.

4.1. Procedimiento *SPT*

Se calcula de forma dinámica, para cada combinación de pieza y máquina, un índice que pretende valorar el instante de finalización de cada pieza en cada máquina. Este índice, es una adaptación de la regla SPT al problema con tiempos de preparación dependientes de la secuencia. El índice se utiliza como criterio de asignación de las piezas a las máquinas. Se asigna la pieza a la máquina que obtenga el índice de menor valor, es decir, se asigna la pieza a la máquina que la termine antes.

Para cada pieza y máquina se calcula $CR_{j,i.}$ según la expresión (9).

$$CR_{j,i}(t) = \left(\max\left\{\tau_j + s_{hi}, r_i\right\} + p_i\right) \tag{9}$$

donde d_i es la fecha de entrega de la pieza i, p_i es el tiempo de proceso de la pieza i, s_{hi} es el tiempo de preparación necesario para realizar la pieza i cuando anteriormente se ha procesado la pieza h, τ_j es el instante en que la máquina j queda libre y r_i es el instante de disponibilidad de la pieza.

$$\overline{CR_i} = \min\left\{CR_{j,i}\right\} \tag{10}$$

Para cada pieza se retiene el valor de $\overline{CR_i}$ calculado según (10) y se asigna la pieza que tiene el menor valor a la máquina que proporciona el mismo, actualizando su instante de disponibilidad.

Se procede de esta forma hasta haber programado todas las piezas calculando su temporización y el retraso asociado.

4.2. Procedimiento *pseudo-SPT*

Se calcula para cada pieza su $\overline{CR_i}$ asociado según (10). A continuación se calcula el valor $\overline{CR}_{\min} = \min \overline{CR_i}$ y se toma como valor umbral el resultado de (11).

$$Umbral = (1+\beta)\cdot \overline{CR}_{\min} \tag{11}$$

En la implementación realizada β=0,2. Posteriormente se elige al azar una de las piezas cuyo valor $\overline{CR_i} \le umbral$ y se asigna a la máquina que ha proporcionado dicho valor. El procedimiento continúa hasta que se han programado todas las piezas.

4.3. Procedimiento *Random*

Se genera una permutación de las *n* piezas al azar y se asigna a las máquinas mediante el algoritmo *grinder* propuesto en Ribas (2007).

4.4. Procedimiento de mejora *SSA*

Este procedimiento es una variante del algoritmo no exhaustivo de descenso (ANED). Esta variante incorpora dos herramientas que permiten hacer frente a dos problemas existentes en muchos procedimientos de exploración: La existencia en el espacio de las soluciones de mesetas, zonas en las que el valor de la función objetivo es la misma y en las cuales la exploración se encalla, y el orden en que se explora el vecindario que suele ser prefijado. Para el primer inconveniente se aceptan empates con cierta probabilidad y para el segundo, se incorpora un vector de posiciones (que llamamos técnica revolver) que permite recorrer el vecindario de forma aleatoria.

4.5. Procedimiento de mejora *vecindario máquina*

Este procedimiento explora el vecindario de las máquinas mediante tres movimientos. Inicialmente se aplica el procedimiento SSA, sobre cada una de las secuencias asignadas a las máquinas con el fin de mejorar dicha secuencia. A continuación se aplica un procedimiento de intercambio entre piezas de diferentes máquinas y, si éste produce alguna mejoría, se aplica de nuevo la mejora SSA sobre cada máquina. Finalmente se aplica un procedimiento que estudia la extracción de una pieza de la secuencia de una máquina y la inserción en otra máquina y, de nuevo, si hay mejoría, se aplica la mejora SSA sobre cada máquina.

5. Experiencia computacional

Se ha construido un generador de ejemplares que sigue la filosofía propuesta en Potts y Van Wassenhove (1982) para el problema de secuenciar piezas en una máquina con tiempos de preparación dependientes de la secuencia. Está basado en cuatro parámetros: *P*max, λ, *T* y *R* (además de *n* y *m*). Para cada ejemplar los tiempos de proceso se distribuyen uniformemente entre 1 y *P*max, los tiempos de preparación entre 1 y *S*max, siendo *S*max = $\lambda \cdot P$max , efectuando las correcciones necesarias para garantizar la desigualdad triangular. Los tiempos de preparación iniciales, en las máquinas, se distribuyen uniformemente entre 1 y *S*max/2. Las fechas de vencimiento se distribuyen uniformemente entre $Cmax \cdot (1 - T - R/2)$ y $Cmax \cdot (1 - T + R/2)$ donde $C\max = \frac{n}{m} \cdot \frac{2 + P\max + S\max}{2} - \frac{1 + S\max}{4}$ es el makespan teórico.

Las colecciones generadas están formadas por 10 ejemplares cada una y son combinación de tres valores de λ (0.5, 0.8, 1.2), 4 valores de *T* (0.2, 0.4, 0.6, 0.8) y 4 valores de *R* (0.2, 0.4, 0.6, 0.8). Un total de 48 colecciones para cada combinación de número de piezas y máquinas. Se han generado ejemplares de 15 piezas con 2 máquinas, 20 piezas con 2 y 3 máquinas y 50 piezas con 3 y 4 máquinas. Las colecciones se han codificado con 4 dígitos. El primero indica el número de máquinas, el segundo el nivel de λ, representado por (5, 8 y 1) respectivamente y el tercer y cuarto dígitos corresponden al nivel de *T* y *R* codificado como 2, 4, 6,y 8.

Se han realizado dos test en un Pentium IV con 512 MB RAM y un procesador de 2.8 GHz. El primero tiene como objetivo analizar la influencia de la solución inicial sobre el resultado final, en función del vecindario implementado. En el segundo test se analiza el comportamiento de los algoritmos cuando se varía el peso asignado a cada criterio.

Para comparar los procedimientos se utiliza el índice I_j definido según (12).

$$I_j = \frac{\text{a lg}_j - \min}{\max - \min} \tag{12}$$

Donde alg_j indica el valor obtenido por el procedimiento j, min es el valor mínimo obtenido por cualquiera de los procedimientos y max es el mayor valor obtenido por cualquiera de los procedimientos.

Para analizar los resultados del primer test realizado se ha representado, gráficamente, el valor promedio de I_j por colección. Con el fin de estudiar la influencia que el número de piezas y máquinas puede tener sobre la eficiencia de los algoritmos, se ha representado los resultados obtenidos separando las colecciones según el número de piezas y máquinas consideradas.

La Figura 1 permite comparar los resultados obtenidos, en las colecciones de 15 piezas, con cada uno de los vecindarios implementados. Se observa que la adaptación de la regla SPT no resulta eficiente para ninguno de los dos vecindarios. También se observa que para el *vecindario máquina* la solución inicial que lleva a mejores resultados es la obtenida de forma aleatoria y que esta combinación es la que constituye el procedimiento más eficiente para estas colecciones.

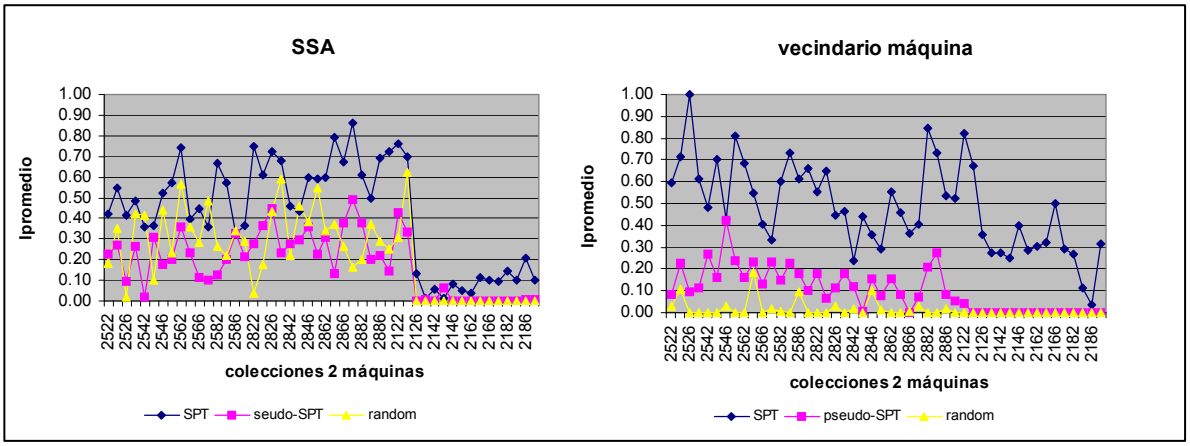

Figura 1. Valor de promedio de Ij para las colecciones de 15 piezas según vecindario utilizado

La Figura 2 representa los valores obtenidos, en las colecciones de 20 piezas, con los procedimientos que utilizan SSA, separando las colecciones según el número de máquinas consideradas, y la Figura 3 los valores obtenidos mediante los procedimientos que utilizan el *vecindario máquina*.

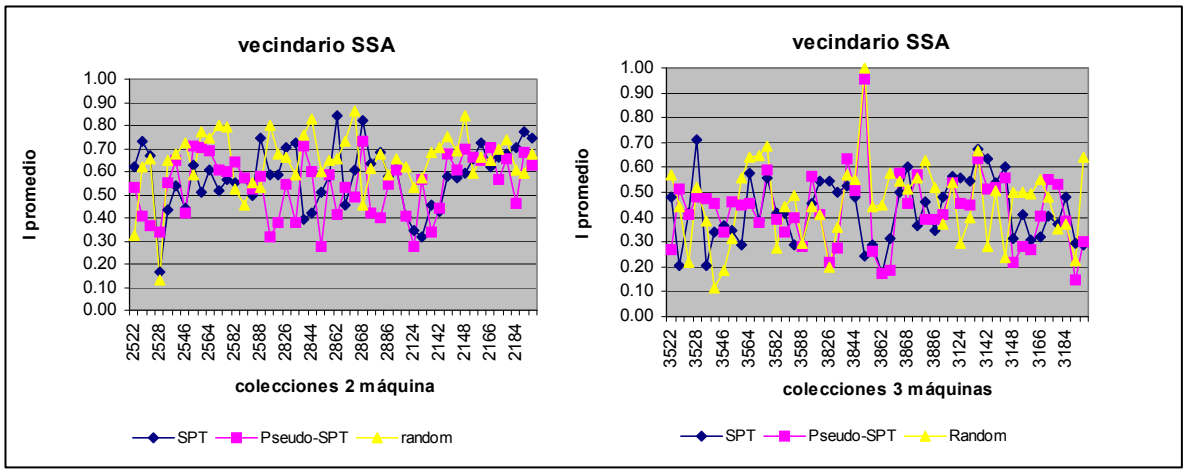

Figura 2. Valor de promedio de Ij para las colecciones de 20 piezas, con 2 y 3 máquinas, para el vecindario SSA

Observando los gráficos de la Figura 2 no se aprecia ninguna diferencia significativa que permita concluir que alguno de los procedimientos iniciales se comporta mejor que otro. En cambio, en los gráficos de la Figura 3 se observan diferencias significativas entre el comportamiento de los algoritmos en las colecciones de 2 y 3 máquinas. Mientras que para las colecciones con 2 máquinas el procedimiento *pseudo*_SPT parece aconsejable, en las colecciones con 3 máquinas se aconsejaría el uso del procedimiento *Random*, tal y como sucedía en las colecciones de 15 piezas. En este caso el procedimiento de mejora *vecindario máquina* también es el más aconsejable.

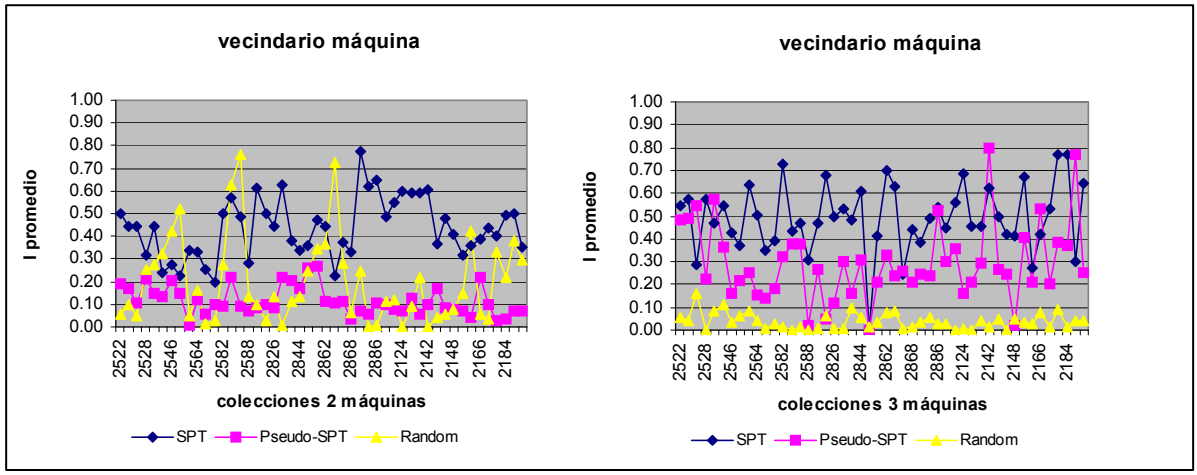

Figura 3. Valor promedio de Ij, en colecciones de 20 piezas, con 2 y 3 máquinas, para el vecindario Máquina

En la Figura 4 se muestran los valores obtenidos en las colecciones de 50 piezas, con 3 y 4 máquinas, con los algoritmos que utilizan el procedimiento SSA. Comparando los dos gráficos entre sí no se aprecia ningún comportamiento diferente, en los algoritmos, atribuible al número de máquinas consideradas. Se llega a la misma conclusión comparando los gráficos de la Figura 5 que representan los valores obtenidos con los algoritmos que utilizan el procedimiento *vecindario máquina*. En cambio, en la Figura 4 se observa que la solución inicial aleatoria, combinada con SSA no lleva a buenos resultados, mientras que para el *vecindario máquina*, Figura 5, no se puede concluir que un procedimiento inicial sea mejor que otro.

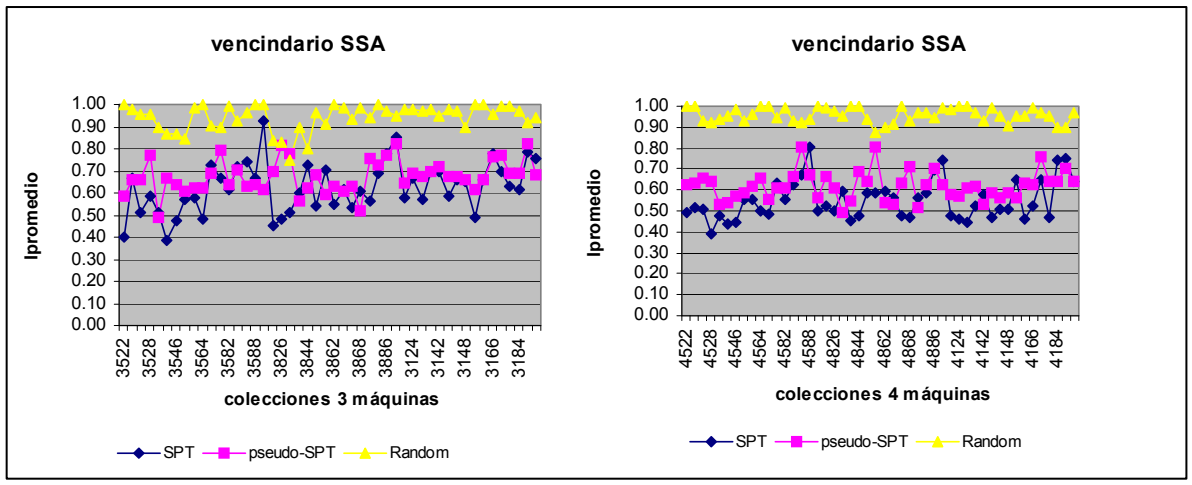

Figura 4. Valor promedio de Ij, en las colecciones de 50 piezas, con 3 y 4 máquinas, para el vecindario SSA

Comparando la Figura 4 y 5 se concluye que el procedimiento de mejora más eficiente es, también, el *vecindario máquina*.

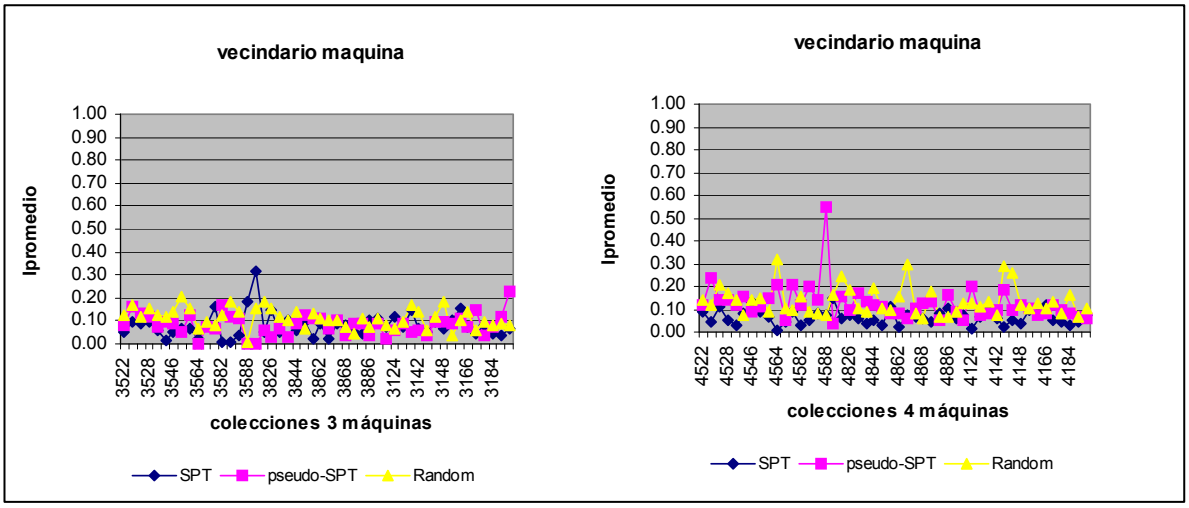

Figura 5. Valor promedio de Ij, en las colecciones de 50 piezas, con 3 y 4 máquinas, para el vecindario Máquina

El segundo test se ha realizado con el fin de analizar si la variación del peso asignado a cada criterio afecta al comportamiento de los algoritmos implementados. En este caso se han elegido 3 colecciones de 50 piezas y 3 máquinas que tuvieran un retraso considerable ya que, como se ha comentado anteriormente, hay muchos ejemplares en que el retraso es nulo o de orden muy inferior al tiempo medio de finalización. Las 3 colecciones elegidas tienen un retraso del orden del 10% del tiempo medio de fabricación. Por esta razón, el test realizado analiza los resultados cuando el peso asignado al retraso es del 90, 95 y 97 %.

En este caso, se ha modificado el cálculo del índice CR_{ji}, utilizado en el procedimiento SPT, para que también considere la fecha de vencimiento asociada a la pieza, la ecuación que define ahora al índice CR_{ji} es la descria en (13).

$$CR_{j,i}(t) = \alpha \cdot d_i + (1-\alpha) \cdot \left(\max\left\{\tau_j + s_{h,g(i)}, r_i\right\} + p_i\right) \qquad (13)$$

A la solución inicial que utiliza este índice le llamaremos, de ahora en adelante, CR. Al haber modificado el índice también queda modificada la solución inicial que hasta ahora llamábamos *pseudo*-SPT que, aunque mantiene la misma estructura, trabaja sobre el nuevo índice y que, siendo coherentes con la nomenclatura, pasaremos a llamar *pseudo*-CR. De esta forma los 6 algoritmos ahora implementados son combinación de los procedimientos de obtención de una solución inicial CR, *pseudo*-CR y *Random* y los procedimientos de mejora SSA y *vecindario máquina*.

En la Tabla 1 se muestran los resultados obtenidos con cada procedimiento, para cada uno de los valores de α considerados. Se observa que, de nuevo, el procedimiento de mejora *vecindario máquina* es con el que se obtiene mejores resultados. Sin embargo, no resulta evidente sacar ninguna conclusión respecto a la eficiencia en el uso de un procedimiento u otro para obtener una solución inicial.

Con el fin de poder comparar el comportamiento de los algoritmos al variar el peso otorgado a cada criterio se ha representado, gráficamente, Figuras 6, 7 y 8, el valor de I_j de los 30 ejemplares que forman las 3 colecciones seleccionadas: 3588, 3888 y 3188.

Tabla 1. Valor promedio de Ij por colección y procedimiento.

	CR SSA	CR vec. Máq	Pseudo-CR SSA	Pseudo-CR vec. Máq	Random SSA	Random vec.máq
α = 0.9	0.77	0.13	0.74	0.25	0.73	0.22
	0.65	0.15	0.67	0.11	0.85	0.11
	0.64	0.13	0.70	0.09	0.70	0.13
α = 0.95	0.77	0.05	0.70	0.12	0.85	0.17
	0.68	0.18	0.77	0.06	0.72	0.19
	0.73	0.24	0.66	0.12	0.91	0.14
α = 0.97	0.69	0.16	0.80	0.09	0.83	0.23
	0.74	0.06	0.80	0.13	0.73	0.26
	0.75	0.17	0.63	0.09	0.86	0.22

Observando las Figura 6 vemos que, para un valor de α de 0.9, el comportamiento de los 3 procedimientos es similar y que no se puede afirmar que uno sea mejor que el otro.

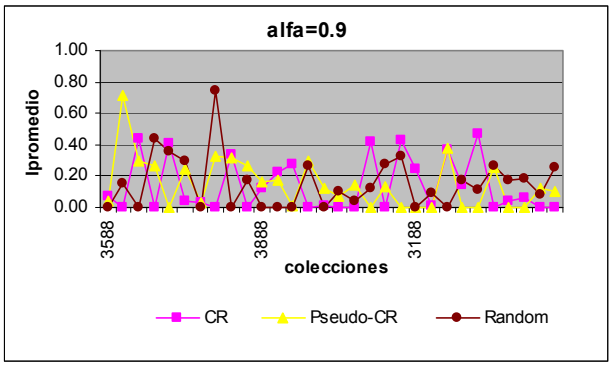

Figura 6. Valor de Ij para los ejemplares de las 3 colecciones de 50 piezas y 3 máquinas con α=0.9

Sin embargo, comparando los gráficos de la Figura 7, observamos que a medida que los criterios están más equilibrados, la utilización de una solución inicial más dirigida a los objetivos perseguidos permite obtener mejores soluciones.

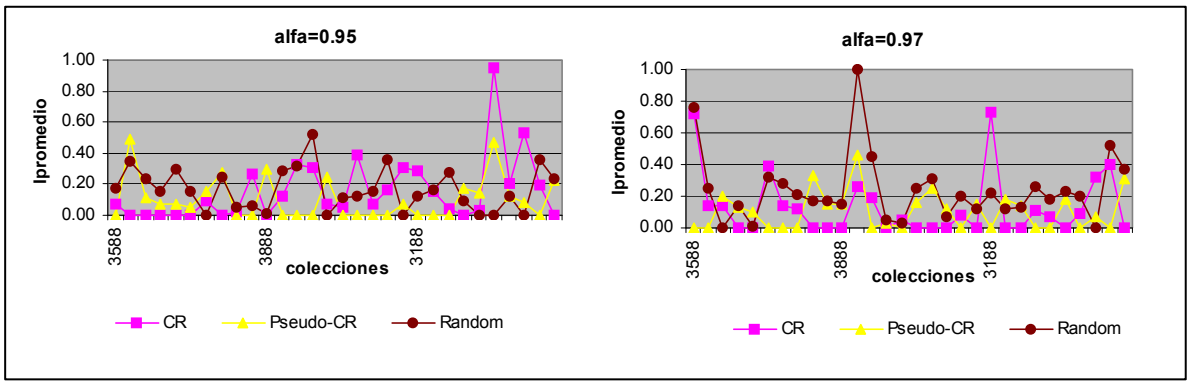

Figura 7. Valor de Ij para los ejemplares de las 3 colecciones de 50 piezas y 3 máquinas con α=0.95 y α=0.97

6. Conclusiones

Se ha analizado el problema de programación bicriterio en un entorno productivo con máquinas idénticas en paralelo y tiempos de preparación dependientes de la secuencia. Los criterios de eficiencia considerados son el retraso medio de las piezas y el tiempo medio de finalización. Se ha utilizado la suma ponderada de ambos criterios como función objetivo. El

estudio realizado ha analizado la eficiencia de 6 procedimientos que combinan dos tipos de vecindarios, el utilizado normalmente para el caso de una máquina y un vecindario específico del problema en cuestión, con 3 procedimientos para obtener una solución inicial. Los resultados obtenidos en el primer test permiten concluir que con el vecindario específico del problema se obtienen mejores resultados. El segundo test ha analizado el comportamiento de los algoritmos cuando se varía el factor de ponderación de los criterios concluyendo que, a medida que éstos están más equilibrados, las soluciones iniciales obtenidas teniendo en cuenta dichos criterios tienen más impacto en la calidad de la solución.

Referencias

Azizoğlu, M. Webster,S. (2003). "Scheduling Parallel Machines to Minimize Weighted Flowtime with Family Setup Times". *International Journal of Production Research*, 41:1199-1215.

Cao. D., Chen, M., Wan, G. (2005). "Parallel machine selection and job scheduling to minimize machine cost and job tardiness". *Computers and Operations Research*, 32:1995-2012.

Du, J.; Leung, J.Y.T. (1990). "Minimizing total tardiness on one machine is NP-hard". *Mathematics of Operations Research*, 15:483-495.

Guinet, A.G.P.; Solomon, M.M. (1996). "Scheduling hybrid flowshop to minimize maximum tardiness or maximum completion time". *International Journal of Production Research*, 34(6):1643-1654.

Hwang, C.L.; Massud, A.S.M. (1979). *Multiple objective decision making-methods and applications*. Springer-Verlag,, Heidelberg.

Lin, B.M.T, Jeng, A.A.K (2004). "Parallel-machine batch scheduling to minimize the maximum lateness and the number of tardy jobs". *International Journal of production economics*, 91:121-134.

Potts, C.N., Van Wassenhove, L.N. (1982). "A decomposition algorithm for the single machine tardiness problem". *Operations Research Letters*, 1:177–181.

Ribas, I. (2007). *Programación Multicriterio de un Sistema Productivo con Flujo Regular sin Esperas y Estaciones en Paralelo. Aplicación a una Fábrica de Helados*. Tesis Doctoral. UPV.

T'kindt, V.; Billaut, J.C. (2002). *Multicriteria scheduling: theory, models and algorithms*. Springer, Heidelberg.

Webster, S.; Azizoğlu, M. (2001). "Dynamic Programming Algorithms for Identical Parallel Machines with Family Setup Times". *Computers and Operations Research*, 28:127-137.