VIII Congreso de Ingeniería de Organización
Leganés, 9 y 10 de septiembre de 2004

Procedimientos para la secuencia de fabricación en una máquina con tiempos de preparación variables

Ramón Companys Pascual, Manuel Mateo Doll

Departamento de Organización de Empresas, Escuela Técnica Superior de Ingeniería Industrial Barcelona, Universidad Politécnica de Cataluña, Av. Diagonal, 647, 7º. 08028 Barcelona. ramon.companys@upc.es, manel.mateo@upc.es

Resumen

El problema de programación de operaciones con tiempos de preparación dependientes de la secuencia se produce en variados contextos industriales. En estos casos, la máquina, o también la línea, que debe realizar la operación necesita una preparación o ajuste antes de iniciar el proceso. Esta preparación incluye operaciones como la colocación del material, la limpieza, colocación o intercambio de moldes, ajustes de herramientas o inspección del material. Se presenta un estudio que compara los resultados obtenidos mediante dos procedimientos de resolución: una heurística ANED y un Algoritmo Genético. La programación de la producción se aplica a un sistema productivo con una única máquina, con tiempos de preparación dependientes de la secuencia, es decir del tipo de pieza tratado anteriormente y del siguiente tipo de pieza a tratar. El objetivo planteado es la minimización de la suma del retraso de los pedidos, cada uno de los cuales se debe servir en una determinada fecha de entrega.

Palabras clave: secuenciación, una máquina, retraso medio.

1. Introducción

En esta comunicación se estudia el problema de programación de la producción en una única máquina, en la que existen tiempos de preparación dependientes de la secuencia de operaciones y en que se deben servir los pedidos en unas fechas determinadas. El objetivo, en este caso, será programar las operaciones con tal de minimizar el retraso medio, o sea la suma promediada de retrasos debido a la finalización posterior a la fecha de entrega de algún lote de piezas de un cierto pedido. El estudio compara los resultados obtenidos mediante una heurística de optimización local, del tipo Algoritmo No Exhaustivo de Descenso (ANED), con aquellos resultados fruto de la aplicación de un Algoritmo Genético.[*]

El problema de programación tratado, con tiempos de preparación dependientes de la secuencia, se puede encontrar en diferentes ambientes industriales tales como la fabricación de productos químicos, farmacéuticos, metales, semiconductores, … En estos casos, la máquina que debe realizar la operación necesita una preparación o ajuste antes de iniciar el proceso. Esta preparación incluye operaciones como la ubicación del material, la limpieza, la colocación o intercambio de moldes, los ajustes de herramientas o la inspección del material.

[*] Este trabajo se deriva de la participación de sus autores en un proyecto de investigación financiado por CICYT con referencia DPI2001-2169, titulado "Secuenciación de unidades con restricciones múltiples".

De acuerdo con Allahverdi et al. (1999), los problemas de programación relacionados con la preparación se agrupan según la presencia o ausencia de familias, pudiendo ser la preparación dependiente o independiente de la secuencia. La preparación es dependiente de la secuencia si su duración depende tanto de la operación actual como de la inmediatamente precedente y es independiente de la secuencia si únicamente depende de la operación a procesar.

El problema de tiempos de preparación según familia tiene lugar cuando los diferentes tipos de piezas se agrupan en familias, de forma que existen tiempos de preparación cuando la pieza a procesar es de una familia diferente a la de la pieza en proceso. Entre otros, Liaee y Emmons (1997) consideran los tiempos de preparación tanto en una máquina como en máquina paralelas. Uno de los muchos modelos aplicados a situaciones industriales puede ser el caso presentado por Parthasarathy y Rajendran (1997).

2. Definición e hipótesis del problema tratado

2.1. El problema de la secuenciación de operaciones en una máquina

Se dispone de un catálogo de *n* tipos diferentes de piezas, o familias de piezas, que se han de fabricar en una sola máquina. El tiempo de preparación de la máquina fabricar el tipo de pieza *i* (i=1,...,n) depende la pieza anterior *h* (h=1,...,n) en la secuencia de operaciones en la máquina (ST_{ih}). La matriz de tiempos de preparación no necesita ser simétrica. Asimismo, se dispone del tiempo de proceso unitario p_i. Como dato adicional, debe conocerse el último tipo de pieza que pasó por la máquina (estado inicial de la máquina).

En esta situación se dispone de una cartera de pedidos a satisfacer, cada uno de los cuales se compone de una serie de lotes de piezas. Para cada pedido *k* se dispone de la fecha de entrega d_k, de los n_k tipos de pieza *i* solicitados (i=1,...,n_k) y sus respectivas cantidades q_i (i=1,...,n_k). Cada línea de pedido indica un tipo distinto de piezas a producir, cuya fabricación no se puede fraccionar. En función de la secuencia escogida, se tendrá el instante en que el pedido *k* está finalizado F_k (máximo de los n_k valores de finalización de los tipos de pieza solicitados), y se le asigna el posible retraso mediante la expresión (1):

$$T_k = \max\ \{0; F_k - d_k\} \tag{1}$$

La medida de eficiencia para evaluar una secuencia es el retraso total en la entrega de los *p* pedidos. Se deberá programar las operaciones en la máquina con el fin de minimizar la suma del retraso de todos los pedidos teniendo en cuenta que existen tiempos de preparación dependientes de la secuencia. El objetivo de este problema consiste en minimizar dicho valor:

$$[MIN]Z = \sum_{k=1}^{p} T_k \tag{2}$$

El problema en cuestión se trata del caso n/1/ /ΣT, en que *n* piezas deben procesarse en una máquina.

2.2. Hipótesis de partida

Este problema ya fue planteado en la bibliografía hace tiempo (Blazewicz, Dror y Weglarz, 1991). Todas las piezas se encuentran disponibles en el instante inicial, por lo que se trata de un planteo estático. Otras hipótesis aplicadas a este caso son:

- La máquina está disponible desde el instante 0 hasta un instante T muy grande.
- No se permiten interrupciones en una operación.
- La máquina sólo puede tratar una operación a la vez.
- La única restricción es la capacidad de la máquina.

2.3. Procedimientos de resolución

Para resolver el problema planteado, se propone un procedimiento heurístico inspirado en (De Castro, Companys y Mateo, 2003) y un Algoritmo Genético (Fernández-Baños, 2003). El primero se divide en dos partes: un procedimiento para obtener una solución inicial, con sus diversas variantes, y la mejora mediante un procedimiento de optimización local. Para el segundo deben definirse cada uno de los parámetros propios de un algoritmo de esta metaheurística (codificación, tipos de cruce y mutación utilizados, etc.).

3. Programación de la producción mediante una Heurística ANED

Para establecer la programación en la máquina de los *n* posibles tipos de piezas pertenecientes a los *k* pedidos, el procedimiento heurístico propuesto parte de una solución inicial, construida mediante la evaluación de un índice crítico de cada lote *j* de piezas CR_j. La obtención de una secuencia mejor según el índice de eficiencia se efectuará a través de la exploración de su vecindario mediante un algoritmo no exhaustivo de descenso.

3.1. Procedimiento Heurístico

Sea un ejemplar que contiene:

- una matriz cuadrada (de dimensión *n*) de tiempos de preparación *ST*,
- un vector (con *n* componentes) de tiempos de proceso unitario para cada tipo de pieza P,
- el tipo de pieza del último lote procesado en la máquina,
- y un conjunto de pedidos, cada uno con una serie de lotes de piezas. Para cada pedido *k* se conoce la fecha de entrega d_k, los n_k tipos de lotes *i* solicitados (i=1,...,n_k) y sus respectivas cantidades q_i (i=1,...,n_k).

Por tanto, el número de lotes totales a fabricar *L* se calculará como la suma de los n_k tipos diferentes de piezas para cada uno de los diferentes pedidos: $L = \sum_{k=1}^{p} n_k$

Conocido dicho valor *L*, los pedidos se desglosarán según los diferentes tipos de piezas dando lugar a j=1,...,L lotes de piezas diferentes. Cada uno de estos lotes de piezas mantendrá la fecha de entrega d_k del pedido al que pertenece. Para hallar una solución inicial, se construirá una secuencia de fabricación ordenando las operaciones según orden no decreciente de un índice crítico. Se define el índice crítico para cada lote *j* de piezas mediante la expresión (3), que se aplica de forma iterativa:

$$CR_j(t) = \alpha \cdot d_k + (1-\alpha) \cdot (ST_{ih} + p_i) \tag{3}$$

donde α toma valores entre 0 y 1; d_k es la fecha de entrega del pedido *k* al que pertenece; ST_{ih} es el tiempo de preparación para realizar la familia *i* del pedido *k*, que depende de la familia del lote anterior *h*); y p_i es el tiempo de operación de la pieza *i* del pedido *k*.

Si α=0, el índice equivale a aplicar la regla SPT; en cambio, si α=1 supone la regla EDD.

Posteriormente, se aplica un procedimiento de mejora sobre la secuencia hallada **S**, que conduce a **S'**; si el retraso total $T_{tot}(\mathbf{S'})$ es menor que el retraso $T_{tot}*$, de la mejor solución obtenida hasta entonces **S***, se guarda la nueva solución.

Procedimiento Heurístico
    Leer datos
    $\alpha = 0$
    $T_{tot}* =$ M (valor muy grande)
    Mientras $\alpha \leq 1$
        Solución inicial (secuencia **S**)
        Procedimiento de mejora (secuencia **S'**)
        Si $T_{tot}(\mathbf{S'}) < T_{tot}*$
            $T_{tot}* = T_{tot}(\mathbf{S'})$; **S* = S'**
        Fin si
        $\alpha = \alpha+0,1$
    Fin mientras
    Guardar mejor solución (**S***; $T_{tot}*$)
Fin procedimiento

Como se observa, α se incrementa progresivamente de 0,1 en 0,1 desde 0 hasta 1.

3.2. Determinación de una solución inicial

Para disponer de una solución inicial, se construye una secuencia de fabricación ordenando las operaciones sobre los lotes según un orden no decreciente de su índice crítico. En la implementación de la solución inicial se utilizan dos vectores:

- **A**=$(a_1,...,a_L)$: lotes no programados, en un cierto instante de aplicación del algoritmo;
- **B**=$(b_1,...,b_L)$: lotes programados, en un cierto instante de aplicación del algoritmo.

Solución Inicial (dado un valor α)
    Desde j = 1 hasta L
        Calcular CR_j e incluir el lote *j* en el vector **A**
        Ordenar el vector **A**=$(a_1,...,a_L)$ según orden no decreciente de CR_j
    Fin desde
    Mientras **A** $\neq\{0\}$ (no se haya programado todos los lotes)
        $b_1 = a_1$
        $a_j = a_{j+1}$ j = 1,...,L-1
        Para todos los componentes del vector **A**
            Recalcular CR_j
            Ordenar el vector **A** según orden no decreciente de CR_j
        Fin para
    Fin mientras

Así pues, inicialmente los *L* lotes a fabricar se encuentran en el vector **A** mientras que al final de esta fase de la heurística, todos los lotes forman parte del vector **B**.

3.3. Procedimiento de mejora

El procedimiento de mejora implementado explora el vecindario de la solución inicial. La exploración se lleva a cabo permutando, dentro de la secuencia disponible, dos operaciones consecutivas y valorando la suma del retraso de los pedidos. Si la permutación entre ambas consigue reducir el retraso, se mantiene el cambio y se evalúa el vecindario de la nueva solución. En cambio, si la nueva permutación no mejora el retraso anterior, se mantiene la secuencia original y, si es oportuno, se sigue explorando el vecindario de la solución inicial. Cuando no se produce ninguna mejora en la evaluación de un cierto vecindario, se finaliza. Los empates se resuelven mediante un procedimiento aleatorio que simula el lanzamiento de una moneda: si sale cara, se queda con la nueva permutación y es cruz, se mantiene la antigua.

<u>Mejora (ANED)</u>
    $T_{med}^* = T_{med}(\mathbf{B})$
    <u>Mientras</u> $\gamma = 1$
        $\gamma = 0$
        <u>Mientras</u> $j \leq L$
            <u>Mientras</u> $i \leq n$
                Intercambiar dos posiciones consecutivas en el vector **B**: **B’**
                Evaluar la nueva permutación: $T_{med}(\mathbf{B'})$
                <u>Si</u> $T_{med}(\mathbf{B'}) < T_{med}^*$
                    $\mathbf{B}^* = \mathbf{B'}$; $\gamma = 1$
                <u>Si</u> $[T_{med}(\mathbf{B'}) = T_{med}^*]$ y [Aleatorio( ) < 0,5]
                    $\mathbf{B}^* = \mathbf{B'}$; $\gamma = 1$
                $i = i+1$
            <u>Fin mientras</u>
            $i = 1$; $j = j+1$
        <u>Fin mientras</u>
    <u>Fin mientras</u>

4. Algoritmo genético

Los algoritmos genéticos, desde su introducción por Holland (1975), proponen codificar el problema en una serie de cadenas (cromosomas). Estos cromosomas forman una población, mediante intercambio de información sobre los cuales (cruce) y/o variación aleatoria de esta información (mutación) se evalúa la solución definida por las cadenas mediante un valor de *fitness* (de adaptación al entorno). El algoritmo implementado sigue el siguiente esquema:

<u>Algoritmo GA</u>
    Generación población inicial (de tamaño *m*)
    Evaluación de cada individuo (mediante *fitness*)
    <u>Mientras</u> no final (número de generaciones establecido)
        <u>Mientras</u> no final (número de cruces *m/2*)
            Selección de pareja según probabilidades
            Cruce (según probabilidad de cruce)
            Mutación (según probabilidad de mutación)
            Evaluación de descendientes (mediante *fitness*)
            Selección de individuos y regeneración de la población
        <u>fin mientras</u>
    <u>fin mientras</u>

Para ello, a continuación se tratan los operadores específicos del algoritmo implementado:

- La representación cromosómica a modo de código.
- La población inicial.
- La medida de la evaluación.
- El criterio de selección de cromosomas.
- Las operaciones de recombinación o cruce.
- Las operaciones de mutación.
- La posible regeneración de la población.
- La condición de fin del algoritmo.

4.1. Codificación

Cada lote *j*, de los respectivos pedidos a realizar, se simbolizará mediante un gen, que debe contener dos datos: el pedido *k* al cual pertenece y el tipo de pieza *i* a fabricar. Por ejemplo, si en el pedido 1 se solicitan las piezas 1 y 2; en el pedido 2, las piezas 1, 2 y 3; y en el pedido 3, las piezas 2 y 4, un cromosoma podría ser:

[01.01] [01.02] [02.03] [02.02] [02.01] [03.02] [03.04]

4.2. Población inicial

El tamaño *m* de la población inicial se ha pretendido relacionar con los parámetros del problema. Se ha optado por tres posibilidades: el triple del número de piezas (3·n); la suma de los lotes de todos los pedidos del ejemplar (L); o un tamaño cualquiera.

Para garantizar una población heterogénea, que contenga individuos con un *fitness* adecuado, se divide la población en tres partes iguales:

- Un tercio se forma de modo que todos los lotes de un pedido se encuentren agrupados, si bien el orden entre pedidos diferentes es aleatorio.

- Un segundo tercio se crea ordenando los pedidos en orden ascendiente de fecha de vencimiento y se divide el cromosoma en cuatro partes equitativas, en cada una de las cuales los genes se ordenan aleatoriamente.

- La tercera parte es más elaborada: se calculan las fechas acumuladas de finalización de pedidos si éstos se ejecutaran en orden por fecha de vencimiento, sin considerar para nada los tiempos de preparación. Se marca un límite cuando una fecha acumulada supera a su respectiva fecha de vencimiento. Por debajo del límite, las piezas se agrupan por familias, siguiendo entre ellas un orden aleatorio, y a partir del límite, se agrupan por pedidos.

4.3. *Fitness*

Dado que el objetivo principal es minimizar los retrasos en las entregas de los pedidos, en el *fitness* se considera el retraso medio de los pedidos, T_{med}. Como medida secundaria de evaluación, se establece el tiempo de permanencia máximo en el taller, F_{max}, mediante un parámetro β=[0;1]. En este caso, la medida de evaluación del *fitness* para cada individuo será:

$$\text{fitness} = T_{med} + \beta \cdot F_{max} \qquad (4)$$

4.4. Proceso de Selección

A todos los elementos de la población se les asigna una probabilidad en función de su nivel de adaptación o *fitness*. El 80% de la probabilidad de selección se reparte de manera inversamente proporcional al *fitness* de cada uno de los individuos. Para que los individuos con mejores cualidades tengan más posibilidades de ser seleccionados, el 20% restante se reparte a partes iguales entre la cuarta parte de la población con mejor aptitud.

Se constituyen tantas parejas como la mitad entera del tamaño de la población inicial. Este proceso genera la posibilidad que un individuo sea cruzado en más de una ocasión con individuos diferentes, o sea la poligamia.

Para los experimentos se ha utilizado como probabilidades de cruce o mutación, P_c y P_m respectivamente, valores tales que $P_c \geq 0{,}8$, y $P_m \leq 0{,}3$.

4.5. Proceso de Cruce

Cada pareja de progenitores cruzada concibe una pareja de hijos. Las modalidades utilizadas son: 1 punto de cruce, PMX y OX (estas dos últimas con 2 puntos de cruce). Los puntos de cruce se escogen al azar y se repiten en ambos descendientes. Los tres tipos de cruce se pueden simultanear a lo largo del algoritmo, lo que permite la posibilidad de aplicar uno, dos o tres tipos de cruce a la vez, así se aprovecha la conjunción de cualidades de cada uno de ellos.

4.6. Proceso de Mutación

Se ha considerado tres tipos diferentes de mutación con objeto de introducir variaciones elementales en la población: las clásicas mutaciones, blandas y de intercambio, y una tercera más específica del problema, que ha sido llamada “pareja”. Los tres tipos de mutación se pueden simultanear como en el caso del cruce.

El objetivo de la mutación pareja es agrupar lotes del mismo tipo de pieza, disminuyendo la necesidad de tiempos de preparación. Se elige al azar el gen que va a ser mutado. Dicho gen se desplaza a lo largo de la cadena hacia el inicio de la secuencia y situarse después de un gen con idéntico tipo de pieza. En caso que el gen elegido llegue a inicio de la cadena sin éxito, se verifica si coincide con el estado inicial de la máquina (en ese caso, el gen ocuparía el primer lugar); y en caso contrario, la cadena resta igual sin producirse mutación alguna.

Por ejemplo, sea el cromosoma:

[01.01] [01.02] [02.03] [02.02] [02.01] [03.02] [03.04]

Si el gen mutado es el que ocupa la quinta posición, se colocará justo detrás del primer gen, por corresponder ambos a la pieza 1. Así, el cromosoma resultante de la mutación será:

[01.01] [02.01] [01.02] [02.03] [02.02] [03.02] [03.04]

4.7. Regeneración de la población y condición de fin

Tres quintas partes de la población generada se forma con los individuos que posean el mejor *fitness*, sean de la población paternal o de la filial. Así, los individuos más competitivos permanecen en la siguiente generación (elitismo). En condiciones normales, la mayor parte de estos individuos pertenecen a la población filial. Para completar la producción generada, se selecciona la misma proporción de individuos (1/5) de ambas poblaciones. Así, se promueve una cierta heterogeneidad en la población.

El final de la evolución en el algoritmo se establece mediante el número de generaciones, aportando una solución satisfactoria sin un coste computacional excesivo.

4.8. Parámetros seleccionados de los operadores

Para alcanzar las soluciones más competitivas, se debe determinar las mejores combinaciones de operadores. Tomando un tiempo medio de ejecución por ejemplar de un minuto, los parámetros han quedado establecidos en:

- Tamaño de la población inicial: L (suma del número de lotes de todos los pedidos).
- Número de generaciones: 3·L (triple de la número de lotes de todos los pedidos).
- Probabilidad de cruce: 100%.
- Probabilidad de mutación: 30%.
- Tipos de cruce: 1 punto y PMX simultáneamente.
- Tipos de mutación: pareja.

5. Experiencia computacional

La experiencia computacional se ha realizado sobre 3 colecciones, cada una de las cuales formada por 100 ejemplares, y con capacidad para procesar hasta 10, 12 y 15 tipos de piezas diferentes, respectivamente. El número de pedidos por ejemplar oscila entre 5 y 10, y el número de lotes diferentes solicitados en cada pedido puede ser entre 4 y 7. Como datos comunes al conjunto de ejemplares por colección, se dispone de una matriz con los tiempos de preparación en función del tipo de pieza acabada de fabricar y el tipo que va a fabricarse.

La comparación entre ambos procedimientos se ha llevado a cabo en un ordenador Pentium IV, a 2800 Mhz y 512 Mb de Ram. Se ha comparado los mejores resultados obtenidos, en cuanto a retraso total, tanto desde el punto de vista del tiempo de ejecución como de la calidad de la solución obtenida.

5.1. Tiempos de ejecución

En la Tabla 1 se muestra los tiempos medios, en segundos, para resolver un ejemplar usando cada uno de los procedimientos.

Tabla 1. Tiempos medios de ejecución, en segundos, por ejemplar.

Procedimiento	Colección 1 (10 piezas)	Colección 2 (12 piezas)	Colección 3 (15 piezas)
1. Heurística ANED	2,4	2,59	2,66
2. Algoritmo Genético	18	16,2	17,1

Como se puede comprobar, el tiempo para resolver un ejemplar mediante el algoritmo genético es, en promedio, 6,7 veces superior al tiempo de ejecución de la heurística ANED.

5.2. Calidad de la solución

A continuación, se compara las mejores soluciones de las tres colecciones de ejemplares obtenidas al aplicar ambos métodos. La comparación se lleva a cabo calculando el siguiente índice:

$$I = (Retraso\_procedimiento – Retraso\_mínimo) / Retraso\_mínimo \quad (5)$$

Ante todo, cabe comentar que durante el desarrollo del procedimiento heurístico se detectó una gran variabilidad en las soluciones obtenidas en diferentes ejecuciones. Esta variabilidad se debe a la resolución de empates mediante el azar, hecho que provoca que no siempre la permutación elegida entre el vecindario sea la que conduzca a una solución mejor. Por ello, se ha comparado los mejores resultados obtenidos con el Algoritmo Genético con dos grupos de resultados: los obtenidos al aplicar el procedimiento heurístico ANED después de 15 minutos de ejecución, y los obtenidos con el procedimiento después de 1 hora de ejecución.

En la Tabla 2 se muestra el análisis de los resultados, basados en el cálculo del índice I (5) para cada método, comparado con el mismo índice obtenido para los otros métodos. Si el índice I del algoritmo genético es 0 para un ejemplar, indica que es el procedimiento con un retraso menor. Todos los datos se expresan en porcentaje de ejemplares sobre el total (300).

Tabla 2. Análisis de los resultados comparando el índice I para cada procedimiento (en %).

	Algoritmo Genético	**Heurística (15 min.)**	**Heurística (1 hora)**
$I = 0$	55,33	14,00	36,33
$I = 0$ múltiple	1,66	0,33	1,66
$0 < I \leq 0,05$	35,00	51,33	45,00
$I > 0,05$	9,66	34,66	18,66

En la Tabla 2, la fila “I=0” indica el porcentaje de veces que un procedimiento obtiene el mejor resultado, y por tanto, su índice I es nulo; “I=0 múltiple” refleja el subconjunto de ejemplares de la anterior fila cuya mejor solución se alcanza en más de un procedimiento; la tercera fila indica el porcentaje de ejemplares con un valor I hasta 0,05 y que no sea 0; la cuarta incluye el resto de ejemplares, con un índice I superior a 0,05.

De la tabla anterior, se desprende que el Algoritmo Genético ofrece mejores resultados que la heurística ANED, y que ejecutando la heurística durante 1 hora se mejoran substancialmente los resultados obtenidos con sólo 15 minutos de ejecución (y respecto AG). Esto se debe a que el tiempo medio de proceso de los 100 ejemplares pertenecientes a una colección es de unos 4 minutos aproximadamente y si ésta se evalúa durante una hora, se puede aplicar el procedimiento sobre un ejemplar unas 15 veces. Esto da la oportunidad que, gracias al azar, se visite algunos vecindarios de permutaciones anteriormente descartadas que conducían a mejores soluciones.

6. Conclusiones

Este trabajo estudia la resolución del problema n/1/ /ΣT con tiempos de preparación dependientes de la secuencia mediante la aplicación de una heurística ANED y un algoritmo genético. El estudio compara los resultados obtenidos mediante ambos procedimientos, tanto desde el punto de vista de la calidad de la solución como del tiempo de resolución requerido.

Se puede concluir que el algoritmo genético demuestra su más o menos acertada elección de los valores de los operadores, ya que siempre obtiene mejores soluciones que la heurística ANED aunque, a cambio, su tiempo de resolución es más de 6 veces superior. Por otro lado, se comprueba que a medida que se incrementa el tiempo concedido a la heurística para encontrar una solución, la calidad de ésta aumenta considerablemente. Esto se debe al papel del azar en la resolución de empates, provocando que no siempre se escojan las permutaciones cuyo vecindario podría conducir a largo plazo a soluciones de mejor calidad.

Referencias

Allahverdi, A.; Gupta, J.; Aldowaisan, T. (1999) "A review of scheduling research involving setup considerations", *Omega, International Journal of Management science 27 (1999)*, pág. 219-239

Blazewicz, J.; Dror, M.; Weglarz, J. (1991). "Mathematical programming formulations for machine scheduling: A survey", *European Journal of Operational Research 51 (1991)*, pág. 283-300.

De Castro, R.; Companys, R.; Mateo, M. (2003). "Programación de las órdenes de fabricación en una sol máquina con tiempos de preparación", V Congreso de Ingeniería de Organización, Valladolid (septiembre 2003).

Fernández-Baños, I. (2003). "Programación de la secuencia de fabricación en una máquina, con tiempos de preparación variables, mediante la aplicación de Algoritmos Genéticos". Proyecto Fin de Carrera. E.T.S. Ingeniería Industrial Barcelona. UPC.

Holland, J.H. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

Liaee, M.; Emmons, H. (1997). "Scheduling families of jobs with setup times". International Journal of Production Economics 1997; 51, pp. 165-176.

Parthasarathy, S.; Rajendran, C. (1997). "An experimental evaluation of heuristics for scheduling in a real life flowshop with sequence-dependent setup times of jobs". International Journal of Production Economics 1997; 49, pp. 255-263.