Programación de Máquinas Paralelas no Relacionadas con Tiempos de Montaje dependientes de la Secuencia y Entrada Dinámica usando Algoritmos Genéticos

Jaime A. Arango, Jaime A. Giraldo y Omar D. Castrillón

Universidad Nacional de Colombia, Campus la Nubia, Manizales-Colombia

(e-mail: jaarangom@unal.edu.co, jaiagiraldog@unal.edu.co, odcastrillong@unal.edu.co)

Recibido Nov. 22, 2012; Aceptado Ene. 16, 2013; Versión final recibida Feb. 07, 2013

Resumen

Este trabajo describe una propuesta de solución al problema de procesar n trabajos en m máquinas paralelas no relacionadas. Es un problema de asignación generalizado de orden lineal y tipo combinatorial que considera tiempos de montaje dependientes de la secuencia y entrada dinámica de trabajos. Se propone un algoritmo genético con codificación entera y conformación de la población, selección de padres, cruzamiento y mutación aleatorios. Hay dos descendientes por generación que compiten contra el peor elemento existente para entrar a la población. Se itera un número de generaciones proporcional al producto de $n \times m$ para obtener la solución. En cada máquina se secuencian los trabajos por fecha de entrega y los tiempos computacionales son aceptables. Se concluye que el problema pudo ser solucionado mediante el algoritmo genético propuesto de forma eficaz y eficiente y que las soluciones se enfocan en reducir el tiempo de procesamiento y cumplimiento de fechas de entrega.

Palabras clave: meta-heurísticas, máquinas paralelas, optimización, programación de actividades

Scheduling of Non-Related Parallel Machines with Sequence Dependent Setup Times and Dynamic Entry using Genetic Algorithms

Abstract

This paper describes a solution method to the problem of processing n jobs on m non-related parallel machines. It is a linear and combinatorial generalized allocation problem that considered a sequence-dependent setup time and dynamic job entry. A genetic algorithm with integer coding and random generation of population, parent selection, crossover and mutation is proposed. There are two descendants per generation that are compared against the worst existing element to enter to population. After a number of generations that is proportional to the product of $n \times m$ the solution is generated. The jobs are sequenced on each machine by due date and computational times are acceptable. It is concluded that the proposed genetic algorithm is an effective and efficient solution that focuses on reducing processing time and on meeting deadlines.

Keywords: meta-heuristics, parallel machines, optimization, activity scheduling.

INTRODUCCIÓN

En muchos procesos de producción industrial, se pueden tener en una o más de las etapas de producción, máquinas paralelas que permiten agilizar los trabajos, repartir un trabajo entre varios recursos, o tener la flexibilidad suficiente para diferentes tipos de productos. Existe una relación muy estrecha entre la gestión de la capacidad y la secuenciación de trabajos; es la priorización de las órdenes de producción, la que define la forma más eficiente de asignar la capacidad según los requerimientos. (Escobar et al, (2012)).

Aunque es frecuente en la literatura el tratamiento de este tipo de problema partiendo del supuesto de que las máquinas paralelas son idénticas como puede verse en trabajos recientes como los de Salazar-Hornig & Medina (2013), Gicquel et al (2012), Zhang & van de Velde (2012), Carpov et al (2012) y Kis & Kovacs (2012) entre otros; en la realidad de las empresas es más común encontrar máquinas paralelas no relacionadas, por razones de evolución tecnológica en el tiempo o por decisión de la compañía para tener equipos adecuados a las necesidades de producción de diferentes líneas de productos.

Se pueden encontrar referencias muy antiguas sobre esta temática. El paralelismo es frecuente en la industria textil, y autores como Morton (1958), Hurst & McNamara (1967) y Serafini (1996) resolvieron casos relacionados con ese tipo de industria teniendo en cuenta esa circunstancia. Los trabajos más recientes, en su mayoría, toman como objetivo minimizar el tiempo de tardanza y resuelven el problema por metaheurísticas. Salazar-Hornig & Medina (2013) utilizaron un algoritmo genético para minimizar el makespan suponiendo máquinas paralelas idénticas. Zhang & Van de Velde (2012) propusieron un algoritmo de aproximación, Driessel & Monch (2011) y James & Almada-Lobo (2011) aplicaron búsqueda en vecindario variable, Lin et al (2011) resolvieron un problema de paralelismo con un algoritmo voraz; mientras Chang & Chen (2011) adaptaron algoritmos genéticos. Ribas-Vila et al. (2009) estudiaron un conjunto de heurísticas teniendo a la vez el criterio de minimizar el tiempo medio de fabricación y la tardanza. Tavakkoli-Moghaddam et al. (2009) y Balasubramanian et al. (2009) propusieron algoritmos genéticos para minimizar tanto los trabajos que incumplen las fechas de entrega, como el tiempo total de trabajo. Liu & Lu (2009) probaron el algoritmo DSPT en la programación de máquinas idénticas. Hu et al. (2009) utilizaron una heurística para optimizar un proceso de construcción de edificios. Baesler et al. (2008) propusieron un algoritmo de recocido simulado para una optimización multiobjetivo. Chen (2006) analizó la influencia de la programación del mantenimiento de las máquinas en la programación óptima. Chang et al. (2008) aplicaron un algoritmo de colonia de hormigas para hacer optimización multiobjetivo cuantitativa y cualitativa.

Este artículo se ocupa de formular y presentar una alternativa de solución al problema de asignación de trabajos en un proceso de producción con máquinas paralelas no relacionadas. La programación de máquinas paralelas no relacionadas con función objetivo basada en tiempo total de terminación y restricción de plazo ha sido modelado en la literatura como $R_m | d_i | \sum C_i$ (Ver Castrillón et al (2010)). Los trabajos a programar son lotes que pueden estar compuestos de una o más unidades de producto. La función objetivo es minimizar el tiempo total de procesamiento, es decir, la suma de los tiempos de proceso de todos los trabajos en todas las máquinas. Las restricciones tienen que ver con las fechas de entrega de los diferentes trabajos y con las condiciones técnicas que limitan la capacidad de producción y restringen el dominio de máquinas en la que puede programarse cada trabajo.

Las máquinas pueden agruparse de acuerdo a sus características técnicas relacionadas con los trabajos que pueden procesar. Aunque todas las máquinas realizan, en esencia, el mismo trabajo, difieren en ciertas características técnicas relacionadas con especificaciones particulares de los productos que pueden procesar. Esas diferencias delimitan los grupos de máquinas elegibles para cada tipo de producto y repercuten en la eficiencia asociada a cada tipo de producto. Los grupos de máquinas se conforman de acuerdo a características técnicas y no son excluyentes entre sí. Una misma máquina puede pertenecer a varios grupos. Un grupo puede estar compuesto por una o más máquinas. Incluso, si existen ciertos tipos de trabajos que pueden programarse en cualquier máquina, un grupo estará conformado por todas las máquinas disponibles.

El tiempo requerido para preparar una máquina entre dos trabajos consecutivos depende de la similitud que haya entre esos dos trabajos. Trabajos similares permitirán un tiempo de montaje hasta 4 veces menor que el que se tarda un montaje entre dos trabajos con amplias diferencias en sus especificaciones. Esta circunstancia de tiempos de montaje dependientes de la secuencia ha sido modelada en varios trabajos recientes, como el de Montoya-Torres et al. (2010) y el de D'Armas (2010). En el proceso de solución del problema deben privilegiarse las soluciones que programen trabajos similares consecutivos, reduciendo así los tiempos dedicados a los cambios de producto.

Es importante también tener en cuenta que al hacer la programación, las máquinas tienen trabajos programados con anticipación de los que se conoce su tasa de producción y su eficiencia estimada, de tal

forma que se pueda anticipar con algún margen de error, el momento en el que la máquina quedará disponible para programar un nuevo trabajo.

En la literatura alrededor del tema abundan las generalizaciones y simplificaciones que dificultan la aplicación práctica de los resultados. Es común encontrar propuestas que desprecian, ignoran o consideran proporcionales al tiempo de procesamiento los tiempos de montaje, parten del supuesto de que todos los recursos están disponibles permanentemente y consideran que todas las máquinas paralelas son idénticas, y por lo tanto todos los trabajos se pueden programar indistintamente en cualquiera de las máquinas con igual expectativa de rendimiento.

El tratamiento clásico del problema, lo asimilaba a un problema del transporte, donde las máquinas eran las fuentes y su capacidad era la oferta, mientras que los trabajos eran los destinos y su tamaño era la demanda. Los tiempos de procesamiento de cada trabajo en cada máquina eran los costos unitarios. Se resolvía el problema del transporte y luego en cada máquina se trabajaba un problema tipo mochila para aprovechar mejor su capacidad. Esa óptica puede apreciarse en trabajos como los de Lawler & Labetoulle (1978), Bruno et al. (1974) y Horn (1973), citados por Leung (2004). No obstante, cuando se consideran complejidades como la condición de que las máquinas paralelas no están relacionadas, la entrada dinámica de los trabajos, y los tiempos de montaje dependientes de la secuencia; esa óptica se hace inaplicable por la misma naturaleza del algoritmo del transporte y su limitación a $n + m - 1$ elementos en la solución.

Algunos trabajos recientes, como el de França & Chaves (2012), han modelado algunas complejidades, buscando una formulación más realista. En el presente trabajo se presenta un modelo que tiene en cuenta simultáneamente los tiempos de montaje dependientes de la secuencia, la entrada dinámica de pedidos y máquinas paralelas no relacionadas. La función objetivo general es minimizar el tiempo total de procesamiento de todos los trabajos y se hace una secuenciación de trabajos en cada máquina con el objetivo de minimizar la tardanza.

DESARROLLO METODOLÓGICO

Como en este trabajo se pretende minimizar el tiempo total de procesamiento de todos los trabajos en un conjunto de máquinas paralelas no relacionadas, la función objetivo es una expresión matemática que suma los tiempos de procesamiento de todos los trabajos a programar. Considerando que esos tiempos pueden variar de acuerdo a la programación obtenida, la expresión debe considerar la variabilidad tanto en relación al tipo de trabajo como a las características técnicas propias de cada máquina, involucrándose en esta función una sumatoria triple respecto a las máquinas, a los trabajos y a la secuencia en cada máquina.

A cada término de la función objetivo se le ha agregado, además, el tiempo de cambio de artículo que está relacionado con la similitud entre dos trabajos consecutivos en la misma máquina, que modela la sensibilidad del tiempo de procesamiento a la secuencia de programación. La variable de decisión x_{ijk} es de carácter binario y tiene el valor de 1 si el artículo i se programa en la máquina j en la secuencia k , y el valor de 0 en el caso contrario.

La función objetivo sería de la siguiente forma:

$$\text{Minimizar Tiempo total de procesamiento } z = \sum_{i=1}^n \sum_{j=1}^m \sum_{k=1}^r \left(\frac{t_{ij}}{e_i} + p_{ijk} \right) x_{ijk} \quad (1)$$

Fuente: Elaboración propia de los autores, basada en Arango (2009) y Arango et al (2010)

Donde:

t_{ij} = Parámetro para el tiempo estimado de procesamiento del producto i en la máquina j .

e_i = Parámetro para la eficiencia asociada al procesamiento del producto i .

p_{ijk} = Parámetro para el tiempo de preparación de la máquina j para procesar el trabajo i en la secuencia k . Si el trabajo i es similar al trabajo que procesó la máquina j en la secuencia $k-1$, $p_{ijk} = 2$ horas, de lo contrario, $p_{ijk} = 8$ horas.

x_{ijk} = Variable de decisión: Asignación del trabajo i a la máquina j en la secuencia k . $x_{ijk} = 1$, si el trabajo i se procesará en la máquina j en la secuencia k , $x_{ijk} = 0$ en caso contrario.

n = Número total de trabajos a programar.

m = Número total de máquinas disponibles.

r = Número total de trabajos programados en la máquina j .

Hay un conjunto de restricciones que tienen que ver con el cumplimiento de las fechas de entrega de los diferentes trabajos. Se debe calcular la fecha prevista de terminación de cada trabajo y su límite teórico sería la fecha de entrega prometida.

$$f_{ijk} + \frac{t_{ij}}{e_i} + p_{ijk} \quad x_{ijk} \leq d_i \quad i = 1, 2, \dots, n; j = 1, 2, \dots, m; k = 1, 2, \dots, r \quad (2)$$

Fuente: Elaboración propia de los autores, basada en Arango (2009) y Arango et al (2010)

Donde:

f_{ijk} = Parámetro para la fecha prevista para el inicio del procesamiento del trabajo i en la máquina j en la secuencia k . Si $k = 1$, será igual a la fecha en que estará disponible la máquina teniendo en cuenta programación hecha con anterioridad, de lo contrario será igual a la terminación del trabajo programado en la secuencia $k-1$. En esta formulación se está considerando la entrada dinámica de trabajos y los tiempos de montaje dependientes de la secuencia.

d_i = Parámetro para la fecha límite de entrega del trabajo i , pactada con el cliente respectivo.

Este conjunto de restricciones, al procesar el problema en un método exacto, podría hacer en algunos casos que el problema fuera infactible, si al menos uno de los trabajos no puede entregarse antes de la fecha límite (d_i). Al procesar el modelo por una metaheurística como el algoritmo genético que se propone en este trabajo se penaliza la función objetivo cuando hay incumplimiento, pero se llega a una solución viable desde el punto de vista práctico de la programación de los trabajos en una planta de producción.

Se debe considerar un conjunto de restricciones de consistencia del modelo para garantizar que cada trabajo solo se puede programar en una sola máquina:

$$\sum_{j=1}^m \sum_{k=1}^r x_{ijk} \leq 1 \quad i = 1, 2, \dots, n \quad (3)$$

Fuente: Elaboración propia de los autores, basada en Arango (2009) y Arango et al (2010)

Así mismo hay que establecer que cada trabajo solo puede procesarse en un grupo de máquinas que comparten las mismas características técnicas, sabiendo que las máquinas son paralelas no relacionadas:

$$j \in G_h \quad \text{Para todo } i, \text{ con } j = 1, 2, \dots, m; h = 1, 2, \dots, q \quad (4)$$

Fuente: Elaboración propia de los autores basada en Arango (2009) y Arango et al (2010)

Donde:

G_h = Grupo de máquinas h .

h = Subíndice para los grupos de máquinas.

q = Número de grupos de máquinas.

Otro grupo de restricciones considera la limitación técnica referente a que cada máquina solo puede procesar un producto a la vez:

$$\sum_{i=1}^n x_{ijk} \leq 1 \quad j = 1, 2, \dots, m; k = 1, 2, \dots, r \quad (5)$$

Fuente: Elaboración propia de los autores

El modelo es binario. Las variables de decisión x_{ijk} toman el valor de 1 cuando el trabajo i se programa en la máquina j en la secuencia k y el valor de 0 en otro caso:

$$x_{ijk} = 0, 1 \quad i = 1, 2, \dots, n; j = 1, 2, \dots, m; k = 1, 2, \dots, r \quad (6)$$

Fuente: Elaboración propia de los autores basada en Arango (2009) y Arango et al (2010)

ALGORITMO GENÉTICO

El modelo planteado es un problema de programación lineal entera con codificación binaria. Los modelos de optimización binaria son NP-Hard, y los métodos exactos de optimización han encontrado severas dificultades de rendimiento en esas aplicaciones en particular (Hvattum et al, 2012). Esta circunstancia los hace poco apropiados para el uso habitual en la programación de la producción en entornos reales, donde se requieren respuestas rápidas.

En el presente trabajo, se propone una solución por un algoritmo genético que llegue a una solución de muy buena calidad a un costo computacional razonable. Los algoritmos genéticos por su carácter aleatorio no garantizan el óptimo ni pueden generar soluciones iguales en ejecuciones independientes (Montoya et al., 2006), pero son eficientes, como otras técnicas inteligentes, para generar buenas soluciones en corto tiempo para casos de alta complejidad en la programación (Ruíz et al., 2012).

El algoritmo propuesto funciona de la siguiente forma:

a. Población Inicial: La primera población se integrará por un número de vectores solución igual a la parte entera de la raíz cuadrada del número de trabajos a programar. Los elementos de la población son vectores de n elementos (n = número de trabajos a asignar). El valor de cada elemento corresponde a la máquina en la que se programaría el trabajo respectivo. La primera población se genera de manera aleatoria teniendo en cuenta las limitaciones técnicas que determinan el grupo de máquinas apto para cada trabajo. Es decir, para cada posición del cromosoma, se generan exclusivamente números aleatorios que pertenezcan al conjunto de máquinas adecuadas a ese tipo de producto. Por lo tanto, cada elemento en las soluciones integrantes de la población se compondrá de n identificadores de máquinas, donde el primer elemento será la máquina en la que se programaría el primer trabajo, el segundo sería la máquina en la que se programaría el segundo trabajo, el tercero sería la máquina para programar el tercer trabajo, y así sucesivamente. Al final, el último elemento entonces será la máquina donde se programaría el último trabajo. Y, generalizando, el elemento i es la máquina donde se programará el i -ésimo trabajo.

Tabla 1: Esquema del cromosoma (vector solución) Fuente: Elaboración propia de los autores.

Trabajo	1	2	...	i	$i+1$	...	N
Máquina	j_1	j_2	...	j_i	j_{i+1}	...	j_n

b. Selección de padres: Se evalúa la función objetivo presentada en (1), en cada uno de los vectores solución componentes de la población. Se evalúa también el cumplimiento de las restricciones. Si se incumple la restricción de fecha de entrega, se penaliza el valor de la función objetivo sumándole el valor en días del incumplimiento multiplicado por el cociente entre el número de preparaciones largas (cuando los dos trabajos consecutivos son diferentes) y el número de preparaciones cortas (cuando los dos trabajos consecutivos son similares) que estén programadas en la misma máquina. De esta manera se castiga también una solución con alto número de preparaciones largas o con bajo número de preparaciones cortas. Para cada elemento de la población se calcula un indicador igual a 1 sobre el valor de la función objetivo evaluada en el vector solución respectivo. Con los valores de los indicadores se genera la "ruleta" para seleccionar los padres. La ruleta es proporcional a la participación de cada indicador en la sumatoria de todos los indicadores. Se generan dos números aleatorios entre 0 y 1, para seleccionar dos padres entre la población actual. Se compara el valor de cada número aleatorio con la suma de los indicadores hasta que el aleatorio supere al acumulado. En ese punto se selecciona como padre el vector solución correspondiente.

c. Cruzamiento: El número de puntos de cruce (pc) será igual a la parte entera de la raíz cuadrada del total de máquinas a programar. Se generarán pc números aleatorios entre 2 y $n - 1$ para determinar los pc puntos donde se ubicarán los cruces dentro de los descendientes. De acuerdo a la ubicación de los cruces se intercalarán los valores de ambos padres en los dos descendientes generados, comenzando con los valores del primer padre en el primer descendiente y con los valores del segundo padre en el segundo descendiente hasta el primer punto de cruce donde se seguirá con los valores del segundo padre en el

primer descendiente y con los valores del primer padre en el segundo descendiente. Se sigue alternando hasta completar los pc puntos de cruce.

d. Mutación: La tasa de mutación, entre 0,01 y 0,30, se puede modelar haciendo diferentes simulaciones para determinar el valor con el que se obtienen mejores resultados. Se evaluará en ambos descendientes de cada iteración. Para determinar si en un descendiente de una iteración hay mutación, se genera un número aleatorio entre 0 y 1. Si el valor está por debajo de la tasa de mutación, se elige en forma aleatoria una posición dentro del descendiente que se está evaluando. Para esa posición se vuelve a generar un número aleatorio que le asigne al azar una máquina entre las que tienen la capacidad técnica de procesar ese trabajo. Se evalúa de nuevo la función objetivo y la penalización correspondiente a las restricciones en el descendiente afectado por la mutación.

e. Actualización de la población: El elemento de la población con peor valor de la función objetivo se compara con cada descendiente. El mejor de los dos elementos comparados quedará en la población y el otro será desechado. Se regresa a 2. Selección de padres.

f. Criterio de Parada: El proceso se repite hasta que pase un número de generaciones igual al producto entre el total de máquinas a programar multiplicado por el tamaño de la población. Cuando se complete esa cantidad de iteraciones, se tendrá la asignación de cada uno de los trabajos a cada una de las máquinas. Durante las iteraciones del algoritmo se estima el tiempo total de procesamiento en cada máquina acumulando los tiempos parciales de cada uno de los trabajos.

g. Secuenciación: Después de hecha la asignación, los artículos asignados a una misma máquina se secuencian por fecha prometida de entrega teniendo en cuenta que el tiempo de montaje depende de la secuencia. El criterio de secuenciación, por lo tanto, primero verifica si hay similitud entre los artículos a montar para programarlos consecutivamente y luego aplica el criterio de fecha de entrega prometida más próxima así como se muestra en la Fig. 1.

Fig. 1: Diagrama de Flujo de secuenciación de trabajos asignados a una máquina. Fuente: Elaboración propia de los autores.

RESULTADOS Y DISCUSIÓN

El procedimiento se aplicó a la programación real de una planta textil con 20 máquinas paralelas en su proceso de tejeduría. En las pruebas se tomó una instancia con un total de 79 trabajos a programar, con un horizonte de fechas de entrega de un mes. La tabla 2 muestra la agrupación de las máquinas según sus características técnicas. La tabla 3 contiene los datos de los trabajos (TRABAJOS = Lista según consecutivo, ENTREGA = Fecha de entrega pactada en formato día/mes, GRUPO = Grupo de máquinas donde pueden programarse según la Tabla 2, y EFICIENCIA = Rendimiento estimado en unidades producidas realmente / unidades que se producirían si la máquina trabajara el 100% del tiempo) y la Tabla 4 contiene los resultados (MÁQUINA = Número de Máquina asignada, DISPONIBLE DESDE = Fecha en que se puede iniciar la programación, SECUENCIA = Trabajos programados ordenados para minimizar tiempos de montaje y cumplir fechas de entrega, TERMINA = Fecha estimada de terminación de la secuencia de trabajos. TARDANZA MAX = Máxima tardanza de los trabajos asignados y secuenciados).

Tabla 2: Grupos de Máquinas Fuente: Elaboración propia de los autores.

GRUPO	MÁQUINAS
A	1, 8, 9, 11, 12
B	10, 13, 14, 15, 16, 17, 18, 19, 20
C	12, 13, 14, 15, 16, 17, 18, 19, 20
D	13, 14, 15, 16, 17, 18, 19, 20
E	16, 17, 18, 19, 20
F	17, 18, 19
G	17, 19
H	2, 3, 5, 6, 7, 8, 9
I	2, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 20
J	2, 5, 6, 8, 10, 12, 16, 20
K	4, 6
L	5, 6, 11
M	7, 8, 9, 10, 11, 12
N	7, 8, 9, 10, 11, 12, 13, 14, 15
O	7, 8, 9, 10, 11, 12, 13, 14, 15, 16
P	7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 20
Q	8, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20
R	8, 10, 13, 14, 15, 16, 17, 18, 19, 20

Tabla 3: Especificaciones de trabajos a programar. Fuente: Elaboración propia de los autores

TRABAJOS	ENTREGA	GRUPO MÁQUINAS	EFICIENCIA
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11	04-jun	K	0,6116
12	08-jun	I	0,7109
13	17-jun	B	0,7533
14, 15	08-jun	Q	0,7559
16, 17	08-jun	C	0,7559
18	21-jun	D	0,7187
19, 20, 21, 22	08-jun	D	0,7187
23, 24, 25	15-jun	D	0,7187
26	08-jun	D	0,6702
27	15-jun	D	0,6702
28	08-jun	E	0,6702
29, 30, 31, 32	15-jun	E	0,6702
33	11-jun	R	0,6702

Tabla 3: (continuación).

TRABAJOS	ENTREGA	GRUPO MÁQUINAS	EFICIENCIA
34, 35	15-jun	F	0,6702
36, 37, 38	08-jun	F	0,7213
39, 40	15-jun	F	0,7213
41	15-jun	G	0,7241
42	22-jun	A	0,6631
43	30-jun	D	0,7104
44	05-jun	H	0,8765
45	20-jun	H	0,8765
46, 47, 48	28-jun	H	0,8765
49	05-jun	P	0,7466
50	13-jun	O	0,7466
51	30-jun	P	0,7135
52	05-jun	M	0,7135
53, 54, 55, 56, 57, 58, 59	11-jun	Q	0,6827
60, 61, 62	25-jun	N	0,8509
63, 64, 65, 66, 67	10-jun	N	0,7817
68, 69, 70, 71	30-jun	N	0,7817
72, 73, 74, 75, 76, 77	30-jun	N	0,8325
78	27-jun	J	0,7662
79	04-jun	L	0,6827

Tabla 4: Resultados Asignación y secuenciación de trabajos. Fuente: Elaboración propia de los autores.

MÁQUINA	DISPONIBLE DESDE	SECUENCIA TRABAJOS	TERMINA	TARDANZA MAX
1	23-may	42	29-may	0
2	23-may	46	10-jun	0
3	25-may	45	02-jun	0
4	23-may	1-2-4-9-10-11	04-jun	0,7536
5	23-may	44-47	10-jun	0
6	24-may	3-5-6-7-8-48	19-jun	0
7	26-may	49-12-62-72	05-jun	0
8	23-may	52-71	28-may	0
9	23-may	64-50-76-77	05-jun	0
10	23-may	66-33-55-59-13	01-jun	0
11	24-may	79-70-75	01-jun	0
12	23-may	63-65-67-74	05-jun	0
13	25-may	22-25-26-57-51-73	04-jun	0
14	26-may	58-18-61-68	05-jun	0
15	24-may	15-21-56-60-69	05-jun	0
16	24-may	16-54-23-29	02-jun	0
17	27-may	14-20-37-41	04-jun	0
18	25-may	36-27-30-31-32-34-35	02-jun	0
19	23-may	38-39-40-24-43	01-jun	0
20	24-may	17-19-28-53-78	12-jun	0

En la Tabla 5, aparece un ejemplo de los resultados que genera el modelo propuesto. Las columnas son los días y las filas las máquinas, cada celda tiene tres números: el consecutivo del pedido, el código del producto y el tiempo estimado de procesamiento (incluido el montaje) en días. Para probar el algoritmo, se utilizó un programa de computador desarrollado por los autores, integrado a la base de datos de programación de la producción de la empresa de donde provinieron los datos de entrada. El tiempo medio de procesamiento estuvo alrededor de 20 segundos con 1580 generaciones, 4 puntos de cruce y un factor de mutación de 0.1. En la planta donde se implementó el sistema, el algoritmo reemplazó un procedimiento empírico, basado en estimaciones subjetivas que se registraban en una hoja de cálculo que consumía entre 4 y 8 horas de trabajo de una persona cada semana. No es posible hacer una comparación entre los resultados del algoritmo y el procedimiento empírico dada la naturaleza subjetiva de las estimaciones utilizadas por este último y la ausencia de datos sobre la incidencia de esa programación en los tiempos de procesamiento y las entregas oportunas a los clientes.

Tabla 5: Ejemplo de programación generada por el software que implementa este algoritmo. Fuente: Aplicativo de software elaborado por los autores.

Máq	jul21	jul22	jul23	jul24	jul25	jul26	jul27	jul28
1	10773 2101761 7.40 días	10773 2101761 7.40 días	10773 2101761 7.40 días	10773 2101761 7.40 días	10773 2101761 7.40 días	10773 2101761 7.40 días	10773 2101761 7.40 días	10773 2101761 7.40 días
2	11045 2101823 4.53 días	11045 2101823 4.53 días	11045 2101823 4.53 días	11045 2101823 4.53 días	10921 2101774 2.15 días	10921 2101774 2.15 días	11039 2101829 16.25 días	11039 2101829 16.25 días
3	10751 2101752 23.89 días	10751 2101752 23.89 días	10751 2101752 23.89 días	10751 2101752 23.89 días	10751 2101752 23.89 días	10751 2101752 23.89 días	10751 2101752 23.89 días	10751 2101752 23.89 días
4	11044 2101823 1.45 días	11063 2101855 1.45 días	90663 2090068 2.10 días	90663 2090068 2.10 días	90663 2090068 2.10 días	10851 2101749 7.24 días	10851 2101749 7.24 días	10851 2101749 7.24 días
5	10903 2101793 12.74 días	10903 2101793 12.74 días	10903 2101793 12.74 días	10903 2101793 12.74 días	10903 2101793 12.74 días	10903 2101793 12.74 días	10903 2101793 12.74 días	10903 2101793 12.74 días
6	10850 2101763 7.24 días	10850 2101763 7.24 días	10850 2101763 7.24 días	10850 2101763 7.24 días	10850 2101763 7.24 días	10850 2101763 7.24 días	10850 2101763 7.24 días	10850 2101763 7.24 días

En los resultados se aprecia la importancia que da el modelo al tiempo de montaje, dado que productos similares se programan preferiblemente en las mismas máquinas en forma consecutiva. De igual forma se nota una búsqueda del equilibrio entre los diferentes tipos de trabajo, programando en unas máquinas pocos pedidos grandes que tardan mucho más tiempo y acumulando en otras, muchos pedidos pequeños que se terminan rápidamente.

Tabla 6: Desempeño del algoritmo en 30 corridas para un caso de 65 trabajos en 20 máquinas. Fuente: Elaboración propia de los autores.

	Valor Función Objetivo (segundos totales)	Iteraciones para mejor valor	Tiempo de procesamiento (segundos)
MÍNIMO	2.508.425,48	992	6,00
PROMEDIO	2.515.465,53	1.442	9,57
MÁXIMO	2.534.342,28	1.560	14,00
DESVIACIÓN ESTANDAR	11.423,47	126	1,57

Aunque, dadas las complejidades que se manejaron simultáneamente no es posible la comparación uno a uno con otros trabajos, se puede apreciar que el algoritmo es eficiente, pues logra una muy buena solución en sólo 9,57 segundos (en promedio) a un problema de 65 trabajos en 20 máquinas (ver Tabla 6). El trabajo de França & Chaves (2012), que también tiene en cuenta los tiempos de montaje dependientes de la secuencia en máquinas paralelas, trabajando con recocido simulado, reporta 840,91 segundos para un problema con 60 trabajos en 5 máquinas.

CONCLUSIONES

De los resultados mostrados, de su análisis y discusión se pueden obtener las siguientes conclusiones:

El problema estudiado es posible representarlo mediante un modelo matemático de orden lineal entero, con codificación binaria y solucionarlo mediante un algoritmo de programación de n trabajos en m máquinas paralelas no relacionadas, cuya función objetivo es minimizar el tiempo total de procesamiento con restricciones relativas al cumplimiento de fechas de entrega, a las diferencias técnicas entre las máquinas y a que en forma unívoca cada trabajo solo puede estar en una máquina y cada máquina solo puede hacer un trabajo a la vez.

Las diferencias de tiempos de procesamiento de los productos y de tamaño de los lotes inciden directamente en los resultados del proceso, debido a que los lotes grandes de productos que tienen procesamiento lento copan la programación de una sola máquina en el horizonte contemplado, haciendo que se acumulen muchos trabajos rápidos de lotes pequeños en otras máquinas, afectando su eficiencia total. En concreto se ven pedidos con tiempos estimados de 18,47 días frente a otros de sólo 0,33 días (igual al tiempo de montaje cuando los dos pedidos no son similares).

Es necesario considerar las restricciones de tipo de máquina de cada pedido versus la versatilidad de las máquinas, dado que si un producto que puede ser programado en cualquier máquina se direcciona a una máquina que tiene capacidades especiales, afecta la posibilidad de cumplir pedidos de productos que sólo pueden procesarse en esa máquina en particular.

Es recomendable complementar la aplicación del algoritmo con programas de cambio rápido de producto, buscando reducir los tiempos de montaje, dada su incidencia directa en la productividad en este tipo de problemas.

La aplicación real del modelo en un entorno de producción ha arrojado resultados satisfactorios en tiempos computacionales apropiados a la programación periódica de una planta de producción. Futuros trabajos podrían comparar los resultados del algoritmo propuesto con otros métodos y avanzar en la generalización del modelo considerando todas las etapas del proceso y no sólo la que tiene paralelismo.

AGRADECIMIENTOS

Los autores desean dar reconocimiento a la Universidad Nacional de Colombia por su apoyo al desarrollo de esta investigación (Convocatoria de Apoyo a Tesis de Posgrado-DIMA 2012. Proyecto: "Mejora de Tiempos de Entrega en un Flow Shop Híbrido Flexible Usando Técnicas Inteligentes. Aplicación en la Industria de Tejidos Técnicos", código Hermes 15917). Este trabajo hace parte de la tesis doctoral del coautor Jaime Antero Arango Marín.

REFERENCIAS

Arango, J.A., Programación Óptima de Múltiples Trabajos en un Conjunto de Telares Utilizando Algoritmos Genéticos. En: VIII Congreso Chileno de Investigación Operativa, OPTIMA 2009. Universidad del Bío-Bío, Concepción (Chile), (2009).

Arango, J.A.; Castrillón G., O.D.; Giraldo G., J.A., Optimal scheduling for a set of looms using evolutionary algorithms. En: II Congreso Peruano de Investigación de Operaciones y Sistemas COPIOS 2010. Universidad Nacional de San Agustín, Arequipa (Perú), (2010).

Baesler, F.; Moraga, R.; Cornejo, O., Introducción de Elementos de Memoria en el Método Simulated Annealing para resolver Problemas de Programación Multiobjetivo de Máquinas Paralelas. Ingeniare. Revista Chilena de Ingeniería, Vol. 16 N° 3. 428-437. (2008).

Balasubramanian, H.; Fowler, J.; Keha, A.; Pfund, M., Scheduling interfering job sets on parallel machines. European Journal of Operational Research 199. 55-67. (2009).

Bruno, J.; Coffman, E.G. Jr.; Sethi, R., Scheduling Independent Tasks to Reduce Mean Finishing Time. En: Communications of the ACM, July 1974. Volume 17. Number 7 Pp. 382-387. (1974).

Carpov, S.; Carlier, J.; Nace, D.; Sirdey, R., Two-Stage Hybrid Flow shop With Precedence Constraints And Parallel Machines At Second Stage. Computers & Operations Research. No. 39. Pp 736-745. (2012).

Castrillón, O.D.; Sarache, W.A.; Giraldo García, Jaime Alberto. Técnicas inteligentes y estocásticas en scheduling. Un enfoque en la producción y las operaciones. En: Colombia 2010. Editorial Universidad Nacional De Colombia ISBN: 978-958-719-485-2 v. 1 pags. 114. (2010).

Chang, Pei-Chann; Chen, Shih-Hsin. Integrating Dominance Properties With Genetic Algorithms For Parallel Machine Scheduling Problems With Setup Times. *Applied Soft Computing*, 11. Pp 1263-1274. (2011).

Chang, P. -T.; Lin, K. -P.; Pai, P. -F.; Zhong, C. -Z.; Lin, C. -H.; Hung, L. -T. Ant colony optimization system for a multi-quantitative and qualitative objective job-shop parallel-machine-scheduling problem. *International Journal of Production Research*, 46:20, 5719 — 5759. (2008).

Chen, Jen-Shiang. Optimization models for the machine scheduling problem with a single flexible maintenance activity, *Engineering Optimization*, 38:1, 53 — 71. (2006).

D'Armas R., Mayra. Algoritmos evolutivos para la secuenciación de operaciones con tiempos de preparación. *Revista Ingeniería Industrial. Actualidad y Nuevas Tendencias*, Universidad de Carabobo (Venezuela), núm. 4, enero-junio, 2010, pp. 7-26. (2010).

Driessel, R.; Mönch, L., Variable Neighborhood Search Approaches For Scheduling Jobs On Parallel Machines With Sequence-Dependent Setup Times, Precedence Constraints, And Ready Times. En: *Computers & Industrial Engineering*, 61. Pp 336-345. (2011).

Escobar, P.; Giraldo, J.A.; Cárdenas, D.M., Programación de Sistemas de Producción Híbridos, Para inventario/Bajo pedido, mediante un Proceso Analítico Jerárquico de Ordenación Grupal (GAHPO). *Información Tecnológica* [online]. ISSN 0718-0764. 2012, vol.23, n.5, pp. 33-46. Disponible en: <http://www.scielo.cl/pdf/infotec/v23n5/art05.pdf> (Consultado en Noviembre 14 de 2012). (2012).

França, E.L.; Chaves, A.A., Sequenciamento de Tarefas com Restrições de Compatibilidade em Máquinas Paralelas com Tempos de Preparação Dependentes da Sequência. En: XVI Congresso Latino-Iberoamericano de Investigación Operativa – XLIV Simpósio Brasileiro de Pesquisa Operacional (CLAIO-SBPO). Sobrapo, Rio de Janeiro (Brasil). Disponible en: <http://www2.claiosbpo2012.iltc.br/pdf/102049.pdf> (Consultado en Noviembre 14 de 2012). (2012).

Gicquel, C.; Hege, L.; Minoux, M; Van Canneyt, W. A Discrete Time Exact Solution Approach For A Complex Hybrid Flow shop Scheduling Problem With Limited-Wait Constraints. En: *Computers & Operations Research*, 39. Pp 629-636. (2012).

Horn, W. A., Minimizing Average Flow Time with Parallel Machines. En: *Operations Research* May/June 1973 Vol. 21 No. 3 Pp. 846-847. (1973)

Hu, X.; Bao, J.; Jin, Y., Minimizing makespan on parallel machines with precedence constraints and machine eligibility restrictions. *International Journal of Production Research*, 1-13. (2009).

Hurst, G.; McNamara, A., Heuristic Scheduling in a Woolen Mill. *Management Science*, Vol. 14, No. 4, Application Series. B182-B203. (1967).

Hvattum, L.; Løkketangen, A.; Glover, F. Comparisons of Commercial MIP Solvers and an Adaptive Memory (Tabu Search) Procedure for a Class of 0-1 Integer Programming Problems. *Algorithmic Operations Research*, 7, Pp. 13-20. Disponible en: <http://journals.hil.unb.ca/index.php/AOR/article/view/18595>. Consultado en Enero 23 de 2013. (2012)

James, J.W.; Almada-Lobo, B., Single And Parallel Machine Capacitated Lotsizing And Scheduling: New Iterative Mip-Based Neighborhood Search Heuristics. En: *Computers & Operations Research*, 38. Pp 1816-1825. (2011).

Kis, T.; Kovacs, A., A Cutting Plane Approach For Integrated Planning And Scheduling. En: *Computers & Operations Research*, 39. Pp 320-327. (2012).

Lawler, E. L.; Labetoulle, J., On Preemptive Scheduling of Unrelated Parallel Processors by Linear Programming. En: *Journal of the Association for computing Machinery*. Vol 25. No 4 October 1978, pp 612-619. (1978).

- Leung, J. , Handbook of scheduling : algorithms, models, and performance analysis / edited by Josep Y-T. Leung. Chapman & Hall/CRC, Boca Ratón, Florida, USA. (2004).
- Lin, Shih-Wei; Lee, Zne-Jung; Ying, Kuo-Ching; Lu, Chung-Cheng. Minimization Of Maximum Lateness On Parallel Machines With Sequence-Dependent Setup Times And Job Release Dates. En: Computers & Operations Research. , 38. Pp 809-815. (2011).
- Liu, P.; Lu, X., On-line scheduling of parallel machines to minimize total completion times. Computers & Operations Research 36. 2647—2652. (2009).
- Montoya, F.G.; Espin, A.; Gil, C. y Banos, R., Optimización de Tensión en Redes de Distribución utilizando Técnicas de Optimización Evolutiva. Inf. tecnol. [online]. 2006, vol.17, n.4 [citado 2012-09-11], pp. 81-88. Disponible en: <http://www.scielo.cl/scielo.php?script=sci_arttext&pid=S071807642006000400015&lng=es&nrm=iso>. ISSN 0718-0764. doi: 10.4067/S0718-07642006000400015. (2006).
- Montoya-Torres, J.R.; Soto-Ferrari, M., González-Solano, F., Programación de la Producción con Tiempos de Preparación Dependientes de la Secuencia y Fechas de Llegada de Trabajos. Dyna, Año 77, Nro. 163, pp. 260-269. (2010).
- Morton, G., Linear Programming -- An Application in an Indian Textile Mill. OR, Vol. 9, No. 3. 198-206. (1958).
- Ribas-Vila, I.; Companys-Pascual, R.; Mateo-Doll, M. Programación bicriterio para máquinas en paralelo. Dyna. Vol. 84 No. 5, Pp. 429-440. (2009).
- Ruiz, S.; Castrillón, O.D.; Sarache, W.A., Una Metodología Multiobjetivo para Optimizar un Ambiente Job Shop. Información Tecnológica. [online]. ISSN 0718-0764. 2012, vol.23, n.1, pp. 35-46. Disponible en: <http://www.scielo.cl/pdf/infotec/v23n1/art05.pdf> (Consultado en Noviembre 14 de 2012). (2012).
- Salazar-Hornig E., Medina S. J. C., Minimización del *makespan* en máquinas paralelas idénticas con tiempos de preparación dependientes de la secuencia utilizando un algoritmo genético. *Ingeniería Investigación y Tecnología*, volumen XIV (número 1), enero-marzo 2013: 43-51.(2013).
- Serafini, P., Scheduling Jobs on Several Machines with the Job Splitting Property. Operations Research, Vol. 44, No. 4. 617-628. (1996).
- Tavakkoli-Moghaddama, R.; Taheri, F.; Bazzazi, M.; Izadi, M.; Sassani, F. Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints. Computers & Operations Research 36 3224—3230. (2009).
- Zhang, X.; Van De Velde, S., Approximation Algorithms For The Parallel Flow shop Problem. En: European Journal Of Operational Research. , 216. Pp 544-552. (2012).