Algoritmo genético para reducir el makespan en un flow shop híbrido flexible con máquinas paralelas no relacionadas y tiempos de alistamiento dependientes de la secuencia*

Juan Camilo López-Vargas
Candidato a Doctor en Ingeniería Industrial y Organizaciones de la Universidad Nacional de Colombia Sede Manizales, Colombia.
jclopezva@unal.edu.co

Jaime Antero Arango-Marín
Candidato a Doctor en Ingeniería Industrial y Organizaciones de la Universidad Nacional de Colombia Sede Manizales, Colombia. Profesor Asociado Facultad de Ingeniería y Arquitectura Universidad Nacional de Colombia Sede Manizales, Colombia.
jaarangom@unal.edu.co

Resumen

El artículo propone el algoritmo genético simple o estándar (AGS) como enfoque de solución al problema de programación de producción para un ambiente tipo flow shop híbrido flexible minimizando el makespan. La codificación del algoritmo propuesto permite obtener resultados con tiempos de cómputo bastante razonables y con un nivel de convergencia del makespan cercano al 2%, con mejores soluciones que un algoritmo alternativo diseñado para el mismo caso de programación de producción. A partir de los resultados obtenidos en el proceso de experimentación y del posterior análisis comparativo, se concluye que a partir del modelamiento más completo de las condiciones reales de producción, el algoritmo genético ejecuta la programación de producción reduciendo el tiempo máximo de procesamiento, o makespan. En futuros trabajos, el enfoque de investigación será la búsqueda de más escenarios alternativos de producción, con el fin de incrementar la aplicación de este tipo de herramientas y generar impacto en los entornos empresariales reales.

Palabras clave

Algoritmo genético, flow shop híbrido flexible, makespan, máquinas paralelas no relacionadas, tiempos de alistamiento dependientes de la secuencia.

Genetic algorithm for reducing the makespan in a flexible hybrid flow shop with unrelated parallel machines, and sequence-dependent setup times

Abstract

This article proposes a simple or standard (AGS) genetic algorithm as a focus of solution to the problem of production scheduling for a flexible hybrid flow shop environment, minimizing makespan. The coding of the proposed algorithm makes it possible to obtain results with rather reasonable computation times and with a level of convergence of the makespan close to 2%, with better solutions than those of an alternative algorithm designed for the same case of production scheduling. Based on the results obtained from the testing process and on the subsequent comparative analysis, it can be concluded that, based on the most complete modeling of the actual production conditions, the genetic algorithm executes production scheduling, reducing the maximum processing time or makespan. In future work, the focus of research will be the search for other alternative production scenarios in order to increase the application of this type of tool and generate an impact on the actual business environment.

Keywords

Genetic algorithm, flexible hybrid flow shop, makespan, unrelated parallel machines, sequence-dependent setup times.

Recibido: 10/10/2014 **Aceptado:** 09/12/2014

* Artículo producto del proyecto de investigación: "Mejora de tiempos de entrega en un flow shop híbrido flexible usando técnicas inteligentes. Aplicación a la industria de tejidos técnicos", financiado por la Universidad Nacional de Colombia, convocatoria de apoyo a tesis de postgrado DIMA 2012, Proyecto Hermes 15917.

http://dx.doi.org/10.18041/entramado.2015v11n1.21103

Cómo citar este artículo: LÓPEZ-VARGAS, Juan Camilo; ARANGO-MARÍN, Jaime Antero. Algoritmo genético para reducir el makespan en un flow shop híbrido flexible con máquinas paralelas no relacionadas y tiempos de alistamiento dependientes de la secuencia. En: Entramado. Enero - Junio, 2015 vol. 11, no. 1, p. 250-262, http://dx.doi.org/10.18041/entramado.2015v11n1.21103

Algoritmo genético para reduzir o makespan em um flow shop híbrido flexível com máquinas paralelas não relacionadas e tempos de preparação dependentes da sequência

Resumo

O artigo propõe um algoritmo genético simples ou padrão (AGS) como um enfoque de resolução do problema de programação de produção para um ambiente de tipo flow shop híbrido flexível minimizando o makespan. A codificação do algoritmo proposto permite obter resultados com tempos de computação bastante razoáveis e com um nível de convergência do makespan próximo a 2%, com melhores soluções que um algoritmo alternativo concebido para o mesmo caso de programação de produção. A partir dos resultados obtidos no processo de experimentação e da análise comparativa subsequente, concluímos que a partir da modelagem mais completa das condições reais de produção, o algoritmo genético executa a programação de produção reduzindo o tempo máximo de processamento, o makespan. Nos trabalhos futuros o foco da pesquisa será a busca de mais cenários alternativos de produção, a fim de aumentar a aplicação desse tipo de ferramentas e gerar impacto sobre os ambientes empresariais reais.

Palabras-chave

Algoritmo genético, flow shop híbrido flexível, makespan, máquinas paralelas não relacionadas, tempos de alistamento dependentes da sequência.

Introducción

La gestión de la producción implica todo el proceso de planeación desde el largo plazo (planeación estratégica), pasando por la programación táctica o de mediano plazo, y llegando a la programación operativa o de corto plazo, en lo que se conoce como el enfoque jerárquico de la producción (Domínguez-Machuca, García, Domínguez-Machuca, Ruiz y Álvarez, 1995). Los procesos de planificación de producción actúan en el mediano y largo plazo de la empresa y las decisiones tomadas en este nivel de planificación afectan directamente los procesos de la programación de producción (Pinedo, 2009). El proceso de programación de la producción tiene como objetivo asignar los trabajos a las máquinas en las etapas correspondientes y definir la secuencia de procesamiento en cada máquina, con el fin de minimizar el tiempo máximo de terminación (Dai, Tang, Giret, Salido y Li, 2013).

Asimismo, la distribución de las máquinas en la planta productiva o layout, define tanto la secuencia de los procesos productivos como la metodología para realizar la programación de la producción (Phanden, Jain y Verma, 2012).

El flow shop resulta un caso especial del conocido *job shop*, donde los trabajos i siguen una misma secuencia de procesos y además es lineal a través de las etapas k presentes en la fábrica (Akhshabi, Haddadnia y Akhshabi, 2012; Shabtay, 2012). Además, el flow shop híbrido resulta ser una extensión del flow shop, en el que existen dos o más máquinas j en una o más etapas k en el proceso (Pinedo, 2009; Wang y Liu, 2013). Cuando existen dos o más máquinas en una etapa, éstas se conocen como máquinas paralelas, de acuerdo con sus características técnicas y mecánicas, se pueden clasificar de la siguiente manera (Cevikcan, Durmusoglu y Baskak, 2011): máquinas paralelas idénticas, si los tiempos de procesamiento son los mismos para cada máquina; máquinas paralelas uniformes, si las máquinas tienen una relación paramétrica en términos del tiempo de procesamiento; y máquinas paralelas no relacionadas, si las diferencias entre los tiempos de procesamiento en las máquinas no pueden ser expresadas en una relación paramétrica.

Finalmente, el flow shop híbrido flexible resulta ser una extensión del flow shop híbrido, donde los trabajos i siguen presentando una secuencia lineal a través de las etapas k, pero uno o más trabajos puedan saltar una o más etapas durante su procesamiento, es decir, se hace referencia a trabajos que no necesitan ser procesados en todas las etapas del proceso (Zandieh y Karimi, 2011).

El problema de la programación de un flow shop híbrido flexible, teniendo más de dos etapas en el proceso ($m>2$), además de la cantidad de máquinas presentes en la planta, y del número y variedad de trabajos a procesar, produce una cantidad enorme de alternativas de solución al problema, es decir, las combinaciones posibles de las asignaciones de todos los trabajos a las máquinas en todas las etapas son demasiadas y llevan a que el problema de programación de producción en estos ambientes sea de un nivel de complejidad NP-Hard (Tavares Neto y Godinho Filho, 2013; Zhang y van de Velde, 2012). Esto es, los tiempos computacionales requeridos para hallar una solución óptima al problema, son poco eficientes y de poca utilidad e interés en las situaciones reales de producción. Por tanto, es necesario aplicar innovaciones tecnológicas para facilitar procesos de mejor calidad y mejorar la competitividad y la sostenibilidad de la organización (Rada, Chaverra, Morante y Mosquera, 2011).

Como acercamiento a este tipo de problemas, distintas herramientas de optimización combinatoria han surgido en las últimas décadas, conocidas en la literatura como meta-heurísticas, entre ellas son conocidas los sistemas expertos, los agentes inteligentes, los algoritmos aleatorios y los algoritmos genéticos, entre otros; éstos aportan soluciones factibles de muy buena calidad frente a problemas complejos que implican un número significativo de variables (Castrillón, Giraldo y Sarache, 2010). Los algoritmos genéticos se destacan por su carácter iterativo y evolutivo, por la búsqueda diversa en el universo de soluciones posibles, por su buen desempeño computacional y por la calidad de las soluciones que ofrece (Gallego, Escobar y Romero, 2006).

Muchos resultados de investigación muestran la gran aplicación que han tenido los algoritmos genéticos como enfoque a la solución del problema de programación del flow shop y el flow shop híbrido, y que además definen como función objetivo el makespan (Xiao, Hao, Zhang y Xu, 2000; Su y Li, 2009). Para el problema de programación en una configuración tipo flow shop, un algoritmo genético híbrido fue construido demostrando mayor eficiencia en los resultados con respecto al algoritmo genético convencional (Tang, Zhang, Lin y Zhang, 2010).Vallada y Ruiz proponen un algoritmo genético para la programación de dos máquinas paralelas no relacionadas con tiempos de alistamiento dependientes de la secuencia (Vallada y Ruiz, 2011). Otra publicación muestra la manera en que se aplicó un algoritmo genético para programar un flow shop híbrido en el sector de la industria cerámica (Gómez-Gasquet, 2007).

Debido al amplio uso de los algoritmos genéticos en los trabajos académicos de los años recientes, es clara también la necesidad de incluir en el diseño de estos algoritmos, los aspectos y las condiciones que afectan la dinámica de la producción real y más aún, aplicar y llevar la implementación de estas herramientas en los entornos productivos reales. Este trabajo muestra el desarrollo de un algoritmo genético simple (AGS) donde se han considerado aspectos comunes en las industrias que presentan configuraciones tipo flow shop híbrido flexible, como las máquinas paralelas no relacionadas y los tiempos de alistamiento dependientes de la secuencia, junto con factores y valores de tiempo ajustados a la realidad, para así incrementar su oportunidad de implementación y aportar soluciones aplicables a los ámbitos reales de la producción en el contexto de la industria local.

1. Descripción del problema

En los contextos reales de la producción, se manejan una diversidad de factores o restricciones que definen las características y capacidades de la producción. Entre estos están, además de la presencia de máquinas paralelas, los factores de eficiencia que afectan los procesos causados por las máquinas y las personas, las prioridades en los pedidos, y los tiempos de alistamiento dependientes de la secuencia (López, 2013; Toledo, de Oliveira y Morelato França, 2013). Asimismo, los empresarios del contexto local latinoamericano reconocen la necesidad de ejecutar procesos de calidad que les permitan, además de aumentar sus niveles de eficiencia, mayor presencia en los mercados y mayor reconocimiento por parte de la industria (Merchán y Gómez, 2011).

Para la formulación del problema se establece un sistema productivo tipo flow shop híbrido flexible. La elección de esta configuración obedece a que éste está presente en diversos sectores productivos, entre otros: la industria cerámica, la industria textil, la industria de procesamiento de alimentos, el procesamiento de la madera y la fabricación de muebles, la industria de formas metálicas, la industria del papel, la industria química, la industria de envases de vidrio, la industria del azulejo y la industria de los semiconductores (Linn y Zhang, 1999; Ruiz, Şerifoğlu y Urlings, 2008).

Para este trabajo se establece un proceso que consta de cinco etapas secuenciales, se describe el número de máquinas por etapa como sigue: tres máquinas en la primera etapa, dos máquinas en la segunda, tres máquinas en la tercera, cuatro máquinas en la cuarta, y una sola máquina en la quinta etapa. Un modelo gráfico del proceso para esta configuración se representa en la Figura 1, donde se muestra, además, ejemplos de los procesos que siguen algunos trabajos.

Otra consideración importante es la presencia de máquinas paralelas en algunas de las etapas. En este sentido, se ha decidido para la construcción del modelo, trabajar con máquinas paralelas no relacionadas, dado que no todas las máquinas están en condiciones de procesar todos los trabajos y además el tiempo de procesamiento dependerá de la referencia, de la máquina asignada y de la cantidad del pedido. Lo anterior con el propósito de ser más ajustado a los entornos de producción reales, donde no en todos los casos se tienen máquinas idénticas dentro de las plantas productivas.

Otros elementos importantes a tener en cuenta para el modelado del problema y que responden a la dinámica de producción propuesta, son los siguientes:

Los tiempos de alistamiento dependientes de la secuencia, en todas las etapas. Esto es, si el trabajo que entra en una máquina tiene distinta referencia al trabajo que sale, se tomará un tiempo en arrancar el proceso dado el requerimiento de cambios, ajustes y alistamiento de la máquina para arrancar el procesamiento del nuevo pedido.

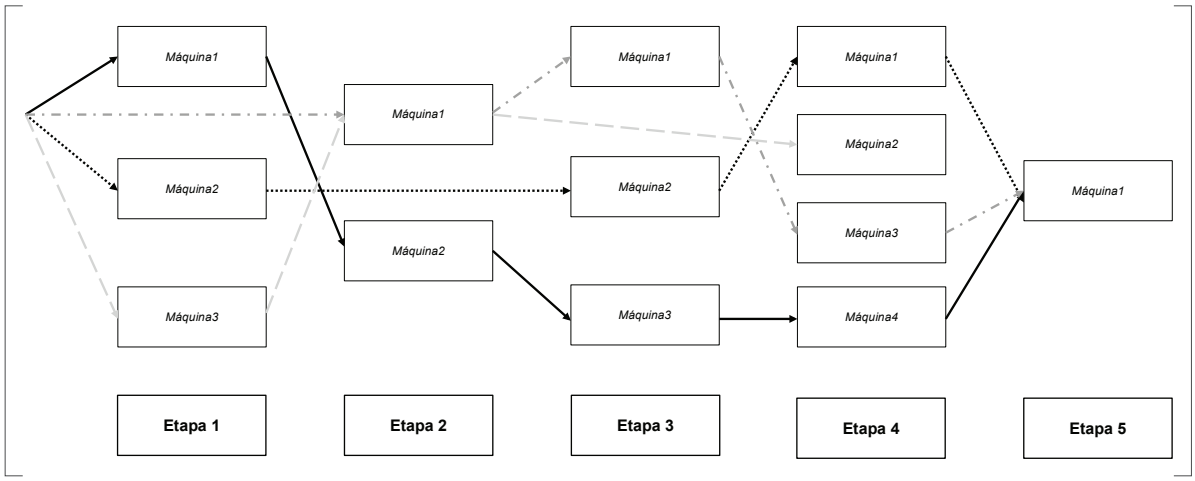

Figura 1. Representación gráfica del sistema productivo y secuencia de procesos para algunos trabajos.
Fuente: Elaboración propia.

En vista de que no todos los trabajos pasan por todas las etapas, se modela un tiempo de procesamiento de cero ($t=0$), en caso de que un trabajo no requiera pasar por una etapa.

Se podrá disponer de las máquinas antes de arrancar la asignación y producción de los pedidos, es decir, todas las máquinas estarán disponibles en el tiempo $t=0$.

En cuanto a la capacidad de las máquinas, se considera que una máquina puede procesar un trabajo a la vez como máximo.

Frente a los trabajos, éstos se consideran indivisibles, es decir, un trabajo en una etapa sólo se podrá asignar y procesar en una sola máquina.

Se consideraron en total diez referencias distintas, para distinguir los trabajos a producir y que se asignarán a las máquinas a través de las etapas del proceso.

La función objetivo está enfocada a la reducción del makespan. El makespan viene definido como el máximo tiempo de procesamiento, o el tiempo comprendido entre el inicio del primer trabajo en la primera etapa y el tiempo de terminación del último trabajo en la última etapa del proceso (Hmida, Haouari, Huguet y López, 2011). Se ha definido el makespan como función objetivo a partir de los aportes que éste genera en la productividad. Tales como, maximizar la utilización de las máquinas, ya que se busca disminuir los tiempos de alistamiento y los tiempos de ocio de las máquinas, mejorar el nivel de utilización de los recursos (Hekmatfar, Fatemi-Ghomi y Karimi, 2011) y reducir los tiempos de terminación es un método efectivo para eliminar demoras y tardanzas en los trabajos (Tavakkoli-Moghaddam, Taheri, Bazzazi, Izadi y Sassani, 2009).

Dado que el makespan (C_{max}) es el máximo tiempo de terminación de todos los trabajos, la función objetivo ($Min(C_{max})$) debe estar relacionada con los tiempos de terminación de todos los trabajos en la última etapa del proceso. De manera que el problema de programación de producción en el flow shop híbrido flexible debe considerar la asignación de los trabajos a cualquiera de las máquinas en todas las etapas del sistema productivo. Se definen entonces los siguientes conjuntos:

Para los trabajos $i, i = \{1,2, ..., m\}$
Para las etapas $k, k = \{1,2, ..., n\}$
Para las máquinas $j, j = \{1,2, ..., p_k\}$

El índice p_k indica que el conjunto máquinas es un subconjunto del conjunto etapas, donde la etapa 1 contiene p_1 máquinas, la etapa 2 posee p_2 máquinas, y así hasta tener en p_k máquinas en la etapa k. Los parámetros definidos en el modelo incluyen aspectos relacionados con la ruta de procesamiento de los trabajos i, los tiempos de procesamiento, tiempos de alistamiento, el factor de eficiencia de la máquina y la capacidad de procesamiento de las máquinas.

RP_i, ruta de procesamiento del trabajo i.

TP_{ijk}, tiempo de procesamiento del trabajo i en la máquina j de la etapa k.

TA_{ijk}, tiempo de alistamiento cuando el trabajo i ingresa a la máquina j de la etapa k.

E_j, porcentaje de eficiencia de la máquina j.

U_j, porcentaje de utilización de la máquina j afectada por aspectos humanos, de calidad y de condiciones de la máquina.

M_j, factor de mantenimiento de la máquina j que afecta el porcentaje de eficiencia de la máquina j.

$E_j = U_j - M_j, \ \forall j$ (1)

$U_j > M_j, \ \forall j$ (2)

$U_j, M_j \in [0,1]$

CP_j, capacidad de procesamiento de la máquina j.

Como la función objetivo se calcula por los tiempos de inicio y terminación de cada trabajo en cada máquina, éstos se consideran variables intermedias que dependen de las asignaciones y de los tiempos de proceso de cada trabajo en cada máquina. Así mismo, un aspecto importante en el desarrollo del problema de programación de producción es la asignación de los trabajos a las máquinas. Las variables de decisión, por lo tanto, deben estar relacionadas con la asignación de los trabajos. Entonces, se tiene:

TI_{ijk}, tiempo de inicio del procesamiento del trabajo i en la máquina j de la etapa k.

TF_{ijk}, tiempo de finalización del procesamiento del trabajo en la máquina de la etapa .

X_{ijk}, la variable de decisión binaria representa con el valor de 1 si se asigna el trabajo i a la máquina j de la etapa k, y un valor de 0 en otro caso.

La Tabla 1 resume la información anterior en el modelo matemático, definiendo la función objetivo y las restricciones del problema planteado.

2. Metología propuesta

Los algoritmos genéticos (AG), son una herramienta de optimización combinatoria desarrollados en la década de los 70 (Holland, 1992; Gómez-Gasquet, Andrés y Lario, 2012). Estos algoritmos siguen los principios y conceptos de la evolución natural mediante el mejoramiento de las características de los individuos a través de las generaciones (Feng, Lu y Li, 2009).

En relación con la herramienta de optimización, se ha decidido diseñar un algoritmo genético simple o estándar (AGS) como metodología de programación de producción para el sistema productivo modelado. El procedimiento general del algoritmo propuesto se describe a continuación.

Generación de la población inicial

La representación de un individuo o cromosoma de la población, se hace mediante una cadena de valores con una cantidad de posiciones igual al producto del número de trabajos a procesar por el número de máquinas en la planta, es decir, el cromosoma tendrá un tamaño de $i * j$ posiciones. Se define tal cantidad de posiciones ya que cada celda debe representar la asignación o no asignación de los trabajos a cada una de las máquinas. Para una comprensión gráfica de esta idea se elabora la Figura 2, que muestra un ejemplo de la estructura básica de un cromosoma. Las posiciones a lo largo del cromosoma tendrán valores binarios de 0 y 1: con un 0 se comprende que un trabajo no pasará por la máquina que represente la posición de esa celda; mientras que un 1

Tabla 1.
Modelo matemático del problema a resolver.

Función objetivo:

$Min(C_{max}) = Min\ (Max(TF_{ijn}))\ i = 1,2, ..., p_k\ n =$ *última etapa.* (3)

Sujeto a:

$$\sum_{i=1}^{m} X_{ijk} = 1, \quad \forall j,k \tag{4}$$

$$\sum_{j=1}^{p_k} X_{ijk} = 1, \quad \forall i,k \tag{5}$$

$$TF_{ijk} = TI_{ijk} + TA_{ijk} + (TP_{ijk}/E_j), \ \forall i,j,k \tag{6}$$

$$TI_{ijk} \geq Max(TF_{i_o jk}, TF_{i jk_o}), \quad \forall i,j,k \tag{7}$$

$$\sum_{i=1}^{m} TP_{ijk} \leq CP_j, \quad \forall i,k \tag{8}$$

$$X_{ijk} \in \{0,1\} \tag{9}$$

$$k, k_o \in RP_i \tag{10}$$

$$E_j \in [0,1] \tag{11}$$

Fuente: Elaboración propia.

Trabajo	1													2												
Etapa	1			2		3			4				5	1			2		3			4				5
Máquina	1	2	3	1	2	1	2	3	1	2	3	4	1	1	2	3	1	2	1	2	3	1	2	3	4	1
Posición	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Asignación	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**1**

Trabajo	...													*i*												
Etapa	1			2		3			4				5	1			2		3			4				5
Máquina	1	2	3	1	2	1	2	3	1	2	3	4	1	1	2	3	1	2	1	2	3	1	2	3	4	1
Posición	27	28	29	30	31	32	33	34	35	36	37	38	39	*ij* -12	*ij* -11	*ij* -10	*ij* -9	*ij* -8	*ij* -7	*ij* -6	*ij* -5	*ij* -4	*ij* -3	*ij* -2	*ij* -1	*ij*
Asignación	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**1**

Figura 2. Representación gráfica de un cromosoma.
Fuente: Elaboración propia.

significará la asignación del trabajo a la máquina que represente esa posición.

La creación de un cromosoma parte de la información contenida en la matriz de factibilidad (Tabla 2). Esta matriz distingue la capacidad de cada máquina para procesar los distintos trabajos de acuerdo con la referencia del producto requerido. Así con un 1, la máquina podrá producir el trabajo y con un 0 se entiende que el trabajo no podrá ser procesado por esta máquina.

La codificación diseñada por el algoritmo toma los valores de la matriz de factibilidad para copiar los valores de ésta en la creación del cromosoma, de acuerdo con la referencia de cada trabajo. Dado que es factible que, en una etapa, más de una máquina pueda estar en condiciones de procesar un trabajo, y que un trabajo en una etapa será asignado a una sola máquina para su procesamiento, es necesario realizar un procedimiento de ajuste para el proceso de asignación de los trabajos a las máquinas, y que se acomode a las condiciones reales de manufactura modeladas y descritas anteriormente. A través de todo el cromosoma, desde las celdas que representan el primer trabajo en la primera etapa, hasta las que representan el último trabajo en la última etapa, se toman las celdas que representan cada etapa y se elige una de ellas de forma aleatoria y se cambia su valor por un 0. Este procedimiento se repite hasta que la suma de los valores de las celdas que representan las máquinas de una etapa, sea 1. Así, se garantiza que todos los trabajos sean asignados una única vez a una sola máquina de cada una de las etapas de todo el proceso productivo.

El procedimiento para crear un cromosoma se repite tantas veces como se defina el tamaño de la población para crear los demás individuos. Esto crea una matriz donde cada fila representa un individuo, y por tanto el número de filas será igual al tamaño de la población del algoritmo genético.

Operación de selección de individuos

Luego de tener definida la población, se calcula el valor del makespan para cada uno de los individuos. Tomando las asignaciones hechas a cada máquina se procede con la secuenciación de los trabajos asignados, mediante la regla de prioridad SPT (tiempo más corto de procesamiento), teniendo en cuenta la relación directa que tiene con la función obje-

Tabla 2.
Matriz de factibilidad.

Etapa	**1**			**2**		**3**			**4**				**5**
Máquina	**1**	**2**	**3**	**1**	**2**	**1**	**2**	**3**	**1**	**2**	**3**	**4**	**1**
Referencia 1	1	0	1	0	1	0	0	0	0	1	1	0	1
Referencia 2	0	1	1	1	0	0	1	1	1	0	1	1	0
Referencia 3	1	1	1	1	1	1	1	1	1	1	1	1	1
Referencia 4	1	1	1	1	0	1	0	1	1	1	1	1	0
Referencia 5	1	1	0	1	1	1	1	1	0	0	0	0	1
Referencia 6	0	1	1	1	0	1	0	0	0	1	1	0	0
Referencia 7	1	1	1	1	1	1	1	1	1	1	1	1	1
Referencia 8	0	0	0	1	0	0	1	1	1	1	1	1	0
Referencia 9	1	1	1	1	1	0	0	0	1	0	1	1	1
Referencia 10	1	0	1	1	0	1	1	1	0	0	0	0	0

Fuente: Elaboración propia.

tivo del makespan. Después de la secuenciación, se procede a calcular los tiempos de inicio y fin de cada trabajo en cada máquina y en todas las etapas, teniendo en cuenta los tiempos de procesamiento, que están definidos para cada máquina (recordando que se tienen máquinas paralelas no relacionadas), los tiempos de alistamiento dependientes de la secuencia en todas las etapas, y el tamaño de los pedidos o trabajos.

Teniendo para todos los individuos de la población los tiempos de finalización máximos de procesamiento o makespan, es decir, el tiempo en que el último trabajo finaliza en la última etapa, se procede a calcular una función de fitness para cada individuo, que ayude a diferenciar la calidad de la alternativa de solución frente a las demás de la población. La función fitness se calcula a partir de la siguiente expresión:

$$f = 1/C_{max} \qquad (12)$$

Donde C_{max} es la función objetivo del modelo o el makespan calculado para cada individuo de la población. Esta función favorece a los individuos con mejores valores del makespan, cuando los tiempos máximos de terminación de los trabajos son bajos y tendrán consecuentemente mayores valores de función fitness. Para el proceso de selección se ha elegido el método de la ruleta proporcional, que otorga mejores posibilidades a los individuos con mejor función de fitness para realizar las operaciones genéticas y generar la descendencia. Se generan aleatoriamente dos números entre 0 y 1 y se comparan con la ruleta proporcional para seleccionar dos individuos de la población.

Operaciones genéticas de cruce y mutación

A partir de la selección de los dos individuos padres se crean dos cromosomas descendientes. Estos últimos contendrán la información genética de los padres de manera permutada y en función de los puntos de cruce definidos para el algoritmo genético. La Figura 3 muestra, a manera de ejemplo, la idea de la operación de cruce diseñada para dos cromosomas y con tres puntos de cruce.

Al concluir el cruce de los individuos y habiéndose generado los dos descendientes, se compara la tasa de mutación pm definida para el algoritmo y un número generado aleatoriamente, si el aleatorio es menor a pm, se ejecuta el operador de mutación. De forma aleatoria, se elige una posición del cromosoma y según el trabajo, la máquina y la etapa que represente dicha celda, se reasignará el trabajo a otra máquina de la etapa correspondiente, respetando las condiciones de producción descritas en la matriz de factibilidad de cada referencia en todas las máquinas. El procedimiento se repite para el segundo descendiente.

Actualización de la población

Teniendo los dos cromosomas listos, se calculan los valores de makespan y se tendrá como candidato el descendiente con menor valor de makespan. De la población se elige al peor de los individuos, es decir, aquel con el valor más alto de makespan. Si al comparar los valores de makespan del peor individuo de la población y del descendiente candidato, se encuentra que el candidato resulta con una solución de mejor calidad, se descarta el peor individuo de la población y entra en su reemplazo el candidato descendiente a actualizar la población.

Cierre

Se ha diseñado como criterio de parada el número de iteraciones que deberá ejecutar el algoritmo genético en sus operaciones para llegar a una población final. Luego de que el número de iteraciones se cumpla, se compila la información de salida del algoritmo. Se elige el individuo con mejor valor de la función fitness, o dicho de otra manera, el individuo con menor makespan. De este individuo se extrae la información de la secuencia de los trabajos asignados a las máquinas y el valor del tiempo máximo de procesamiento.

Posición	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Cromosoma 1	0	0	1	0	1	0	1	0	0	0	0	1	1	0	1	0	1	0	0	1	0	0	0	0	1	0
Cromosoma 2	0	1	0	1	0	1	0	0	1	0	0	0	1	1	0	0	0	1	0	0	1	0	1	0	0	0

Posición	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Descendiente 1	0	0	1	0	1	1	0	0	1	0	0	0	1	0	1	0	1	0	0	1	0	0	1	0	0	0
Descendiente 2	0	1	0	1	0	0	1	0	0	0	0	1	1	1	0	0	0	1	0	0	1	0	0	0	1	0

Figura 3. Representación de la operación de cruce para dos individuos y 3 puntos de cruce.
Fuente: Elaboración propia.

3. Experimentación y resultados

El diseño y la codificación del algoritmo genético permite ingresar cinco variables de entrada para su ejecución, a saber: tamaño de la población, número de iteraciones, tasa de mutación, puntos de cruce, y número de trabajos a producir. Para el proceso de experimentación del algoritmo se trabajaron los valores que muestra la Tabla 3 para todas las variables de entrada.

Tabla 3.
Valores elegidos de las variables de entrada del algoritmo genético para su experimentación.

Parámetro	Valores
Número de trabajos	40, 50
Tamaño de la población	100, 200
Número de iteraciones	3000, 5000
Tasa de mutación	0.1, 0.2
Puntos de cruce	20, 30

Fuente: Elaboración propia.

Teniendo dos valores distintos para cada variable, el diseño experimental implica el desarrollo de un problema exponencial del tipo 2^n. Además, teniendo en cuenta dos valores distintos del número de trabajos, se configuraron dos situaciones diferentes para la producción: una con 40 trabajos a producir, y otra distinta con 50 trabajos requeridos para la producción.

La ejecución del algoritmo se realizó a través del software MATLAB, ejecutándolo 30 veces por cada combinación posible de los valores de las variables de entrada. Los resultados de la ejecución cuando se tienen 40 trabajos (i=40), se muestran en la Tabla 4; mientras que en la Tabla 5 (ver pág. 258), se resumen los principales resultados cuando i=50 . Las Tablas 4 y 5 contienen por cada fila, las distintas combinaciones posibles para las variables de entrada, se muestran los cálculos respectivos para el makespan de las mejores soluciones encontradas y la media de las soluciones de las 30 ejecuciones hechas. También la desviación estándar de las 30 corridas y el porcentaje del coeficiente de variación.

Con el fin de resumir gráficamente la información presentada, se han construido las Figuras 4 y 5 (ver pág. 258). La Figura 4 muestra las medias del makespan calculado por el algoritmo genético cuando i=40 ubicándolas de acuerdo con cada combinación de las variables de entrada. La Figura 5 resume gráficamente el panorama de la producción cuando se tienen programados 50 trabajos (i=50), mostrando las medias de los valores del makespan calculados de acuerdo con cada combinación posible de las variables de entrada.

Los gráficos muestran la forma en que los valores del makespan calculados por el algoritmo genético propuesto varían cuando se tratan distintas combinaciones de las variables de entrada. Claramente puede notarse que con 40 y con 50 trabajos a producir, el rendimiento del algoritmo mejora cuando se tiene un número de iteraciones de 5000.

Tabla 4.
Resumen de las ejecuciones del algoritmo genético realizadas con i=40.

					Makespan		
Combinación	Población	Iteraciones	T. mutación	P. cruce	Media	Desv. Est.	Coef. Var. %
1	100	3000	0,1	20	207,1	3,31	1,60%
2	100	3000	0,1	30	207,5	3,14	1,51%
3	100	3000	0,2	20	206,6	3,29	1,59%
4	100	3000	0,2	30	207,8	2,80	1,35%
5	100	5000	0,1	20	201,5	2,89	1,44%
6	100	5000	0,1	30	201,0	3,51	1,75%
7	100	5000	0,2	20	201,5	2,59	1,29%
8	100	5000	0,2	30	201,7	4,49	2,23%
9	200	3000	0,1	20	214,0	2,68	1,25%
10	200	3000	0,1	30	214,1	2,22	1,04%
11	200	3000	0,2	20	214,1	2,50	1,17%
12	200	3000	0,2	30	214,8	2,42	1,13%
13	200	5000	0,1	20	208,3	2,67	1,28%
14	200	5000	0,1	30	209,2	2,69	1,28%
15	200	5000	0,2	20	208,0	2,08	1,00%
16	200	5000	0,2	30	209,1	2,57	1,23%

Fuente: Elaboración propia.

Tabla 5.

Resumen de las ejecuciones del algoritmo genético realizadas con i=50.

					Makespan		
Combinación	**Población**	**Iteraciones**	**T. mutación**	**P. cruce**	**Media**	**Desv. Est.**	**Coef. Var. %**
1	100	3000	0,1	20	227,7	4,03	1,77%
2	100	3000	0,1	30	229,3	3,28	1,43%
3	100	3000	0,2	20	227,9	3,61	1,59%
4	100	3000	0,2	30	227,6	3,51	1,54%
5	100	5000	0,1	20	221,3	3,54	1,60%
6	100	5000	0,1	30	221,4	3,50	1,58%
7	100	5000	0,2	20	222,0	3,07	1,38%
8	100	5000	0,2	30	220,6	4,12	1,87%
9	200	3000	0,1	20	235,2	2,64	1,12%
10	200	3000	0,1	30	235,8	2,29	0,97%
11	200	3000	0,2	20	235,5	3,03	1,29%
12	200	3000	0,2	30	236,0	2,32	0,98%
13	200	5000	0,1	20	229,9	2,41	1,05%
14	200	5000	0,1	30	230,7	2,65	1,15%
15	200	5000	0,2	20	230,6	3,14	1,36%
16	200	5000	0,2	30	229,7	2,69	1,17%

Fuente: Elaboración propia.

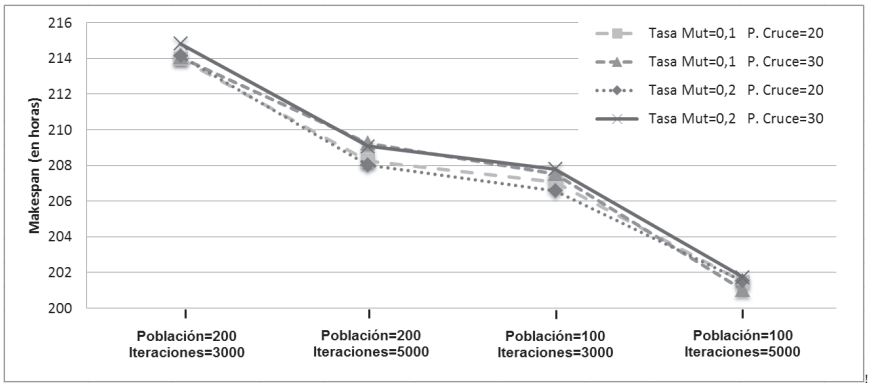

Figura 4. Media de los valores del makespan calculados cuando i=40.
Fuente: Elaboración propia.

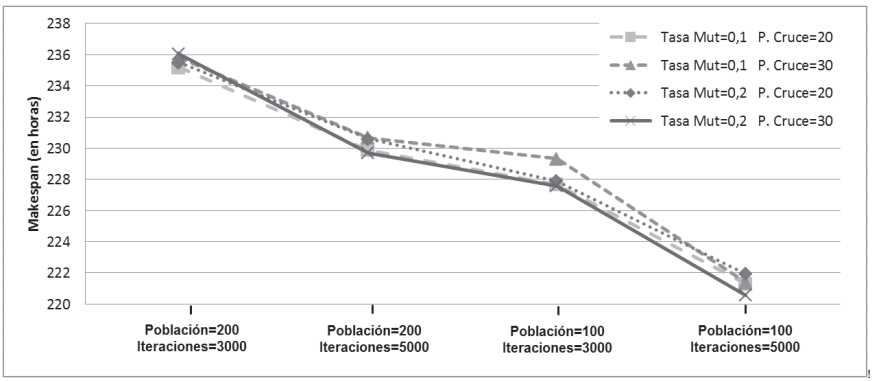

Figura 4. Media de los valores del makespan calculados cuando i=50.
Fuente: Elaboración propia.

Esto es, el algoritmo diseñado otorga mejores soluciones a medida que aumentan las generaciones a través de las iteraciones del algoritmo. Caso contrario ocurre con el tamaño de la población que tuvo mejores resultados cuando se tuvieron 100 individuos en lugar de una población con 200 individuos. La variedad y diversidad que presenta una población de mayor tamaño implica mayores esfuerzos para mejorar la calidad de la población; en cambio una población de tamaño medio tiende a facilitar mejoras en la calidad de sus individuos.

Por otra parte, las variables de tasa de mutación y puntos de cruce, parecen no tener mucha afectación en cuanto a los resultados obtenidos por el algoritmo genético. Gráficamente se observa que los resultados comparados por estas dos variables no presentan cambios sustanciales y están ubicados muy cercanos entre sí. Con el objetivo de soportar estadísticamente estas afirmaciones se han elaborado las Tablas 6 y 7 con los análisis de varianza o ANOVA para las dos situaciones de producción planteadas, con i=40 e i=50 respectivamente.

Las Tablas 6 y 7 fueron construidas tomando un nivel de confianza del 95% (α=0,05). Puede considerarse estadísticamente entonces, y con base en los resultados con i=40 e i=50, que la tasa de mutación y los puntos de cruce no tienen significancia en el diseño del algoritmo genético propuesto. Mientras que las variables tamaño de la población y número de iteraciones tienen un efecto altamente significativo para el hallazgo de soluciones de mejor calidad por parte del algoritmo. Es decir, y soportado en el análisis ANOVA, se pueden obtener resultados mucho más competitivos con un número de iteraciones de 5000, que cuando se definen 3000 iteraciones en el algoritmo genético.

Es necesario resaltar la poca variabilidad de los resultados del makespan obtenidos a través de las ejecuciones realizadas. La Figura 6 (ver pág. 260) describe el porcentaje de coeficiente de variación de las 30 ejecuciones hechas por cada una de las combinaciones cuando i=40. Asimismo, la Figura 7 (ver pág. 260) presenta los coeficientes de variación porcentuales de las ejecuciones hechas con 50 trabajos frente a las combinaciones obtenidas por las variables de entrada del algoritmo.

Los gráficos relacionados como Figuras 6 y 7, muestran claramente que los porcentajes de variación para todo el experimento están por debajo del nivel del 2%, solamente en una combinación de las 32 posibles, el coeficiente de variación superó por poco este valor. La gran mayoría de las

Tabla 6.
Tabla ANOVA para las ejecuciones realizadas cuando i=40.

Análisis de Varianza					
Variable	**Suma cuadrados**	**GL**	**Cuadrados medios**	**F**	**Valor-p**
Población	6064,83	1	6064,83	719,02098	3,956E-97
Iteraciones	3912,49	1	3912,49	463,84875	2,747E-72
Mutación	2,16	1	2,16	0,2560905	0,6130536
P. cruce	34,03	1	34,03	4,0340697	0,0451558
Error	4006,55	475	8,43		
Totales	14020,06	479			

Fuente: Elaboración propia.

Tabla 7.
Tabla ANOVA para las ejecuciones realizadas cuando i=50

Análisis de Varianza					
Variable	**Suma cuadrados**	**GL**	**Cuadrados medios**	**F**	**Valor-p**
Población	8075,36	1	8075,36	795,27306	1,598E-103
Iteraciones	4487,19	1	4487,19	441,90456	7,650E-70
Mutación	3,64	1	3,64	0,3584806	0,5496367
P. cruce	2,11	1	2,11	0,2074757	0,6489613
Error	4823,24	475	10,15		
Totales	17391,54	479			

Fuente: Elaboración propia

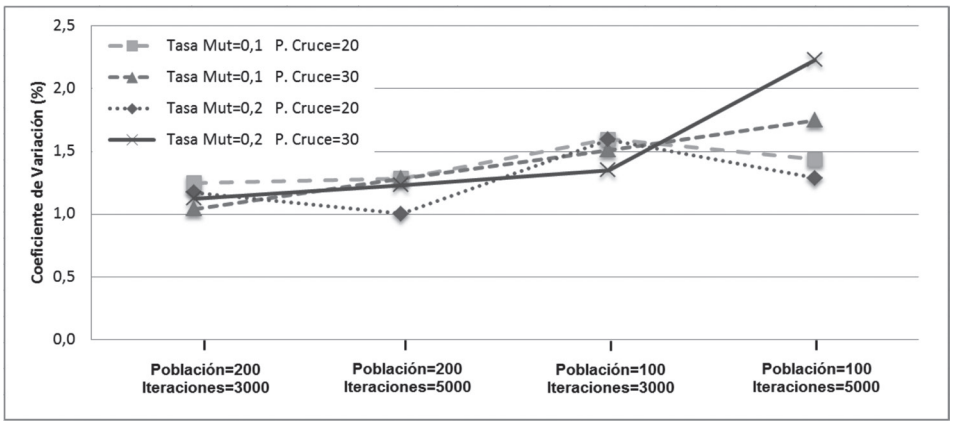

Figura 6. Coeficientes de variación de las corridas para cada combinación con i=40.
Fuente: Elaboración propia.

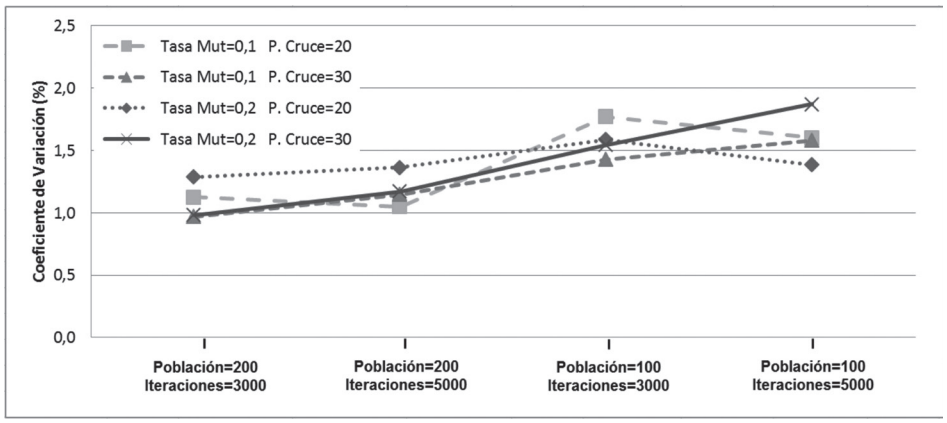

Figura 7. Coeficientes de variación de las corridas para cada combinación con i=50.
Fuente: Elaboración propia.

combinaciones apenas supera el 1% de variación. Teniendo en cuenta el número de veces que se ejecutó el algoritmo genético diseñado, se encuentran niveles de convergencia bastante importantes. La codificación hecha para el algoritmo propuesto permite obtener resultados muy estables, dada la poca variabilidad de los valores alcanzados en el experimento mostrado en este documento. Esto se logra gracias a un buen número de iteraciones y simultáneamente con tasas de mutación del orden $p_m = 0{,}2$, aspectos que ofrecen al algoritmo no quedar estancado en óptimos locales.

Con el propósito de llevar a cabo una comparación de los resultados y del rendimiento del algoritmo genético propuesto, se diseñó un algoritmo aleatorio para la programación de producción del mismo sistema tipo flow shop híbrido flexible, con el mismo número de máquinas y etapas en el proceso, con las mismas restricciones de procesamiento y condiciones de producción, y los mismos tiempos de producción. Este algoritmo aleatorio hace una asignación de los trabajos a las máquinas bajo la regla de prioridad SPT y calcula los tiempos de inicio y terminación de cada trabajo en cada máquina obteniendo al final, el valor del makespan para la solución generada por el algoritmo.

La Tabla 8 (ver pág. 261) muestra los resultados obtenidos a partir de la ejecución del algoritmo aleatorio 30 veces para los casos con i=40 e i=50 trabajos. Claramente se evidencia tanto para 40 como para 50 trabajos, que las soluciones arrojadas por el algoritmo aleatorio presentan variaciones bastante considerables, frente a los obtenidos mediante el algoritmo genético simple (AGS) con coeficiente de variación cercanos al 10%.

Los valores del makespan obtenidos a través de la programación de producción hecha por el algoritmo genético, están por encima de los valores del makespan logrados por el AGS. En promedio, los resultados para i=40 trabajos superan las 280 horas, mientras que para i=50 trabajos, el promedio del makespan ronda las 360 horas de producción. De hecho, las mejores alternativas alcanzadas por el algoritmo aleatorio siguen estando por encima de las peores solucio-

Tabla 8.
Resultados de la ejecución del algoritmo aleatorio.

	Makespan (Tiempo en horas)			
Trabajos	Mejor	Media	Desv. Est.	Coef. variación %
40	250,4	286,78	26,50	9,2%
50	298,2	359,14	39,84	11,1%

Fuente: Elaboración propia.

nes encontradas por el algoritmo genético, tanto cuando se tienen 40 como cuando se tienen 50 trabajos a programar.

A partir de este paralelo, se evidencia una mejoría en el rendimiento del sistema productivo y una reducción en los tiempos totales de procesamiento, cuando el algoritmo genético simple (AGS) propuesto logra llevar a cabo el proceso de programación de producción frente a una asignación de los trabajos a las máquinas realizada de forma aleatoria, con secuenciación a partir de la regla de prioridad SPT.

4. Conclusiones y trabajo futuro

El algoritmo genético estándar se diseñó con el propósito de plantear una alternativa de solución al problema de programación de un flow shop híbrido flexible. Teniendo en cuenta, en el modelamiento del problema aspectos de la producción real, como las máquinas paralelas no relacionadas y los tiempos de alistamiento dependientes de la secuencia. Estas restricciones de fabricación resultan comunes en diversos sectores industriales en los contextos local y regional.

El algoritmo propuesto ofrece soluciones con niveles de convergencia muy estables, que rondan el 2% de coeficiente de variación, el análisis ANOVA indica la diferencia altamente significativa de emplear el algoritmo con valores distintos en las variables número de iteraciones y tamaño de la población. Además, considerando los factores de producción incluidos en su modelamiento, el algoritmo propuesto logra encontrar soluciones de mejor calidad frente al algoritmo alternativo con el cual se comparó, con tiempos computacionales bastante razonables que no superan el minuto y medio de ejecución. Aspecto de bastante interés por parte del sector empresarial para mejorar sus procesos en la gestión de la producción.

Teniendo en cuenta que se tomaron datos reales de la industria para la definición de los tiempos de procesamiento, el principal aporte de este trabajo radica en la validación de la metodología del algoritmo genético simple (AGS) en los problemas reales de la producción. Esto es, obtener a partir de un programa que se ejecuta con tiempos razonables para un empresario, soluciones prácticas y eficientes al problema de la programación de la producción en sistemas tipo flow shop híbrido flexible. Las soluciones ofrecidas por el algoritmo cuentan con la ventaja de ser aplicables en los entornos reales y locales de la producción.

En futuros trabajos, el tema de investigación puede enfocarse en la búsqueda de escenarios alternativos de producción y la consideración de otros aspectos que se presentan en los ambientes reales empresariales, en específico: la maleabilidad del tamaño de los trabajos a producir, la variabilidad en las etapas de producción y la entrada dinámica de pedidos. Ello con el propósito de incrementar el campo de aplicación de este tipo de herramientas de optimización, para generar impacto en los ambientes empresariales reales. El objetivo es ampliar el número de trabajos de este tipo aplicados a la industria y propender al desarrollo y la productividad local.

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 y la tesis de maestría del coautor, Juan Camilo López Vargas.

Conflicto de intereses

Los autores declaran no tener ningún conflicto de intereses.

Referencias bibliográficas

1. AKHSHABI, Mostafá; HADDADNIA, Javad y AKHSHABI, Mohammad. Solving flow shop scheduling problem using a parallel genetic algorithm. In: Procedia Technology. 2012, vol. 1, p. 351-355.
2. CASTRILLÓN GÓMEZ, Omar Danilo; GIRALDO GARCIA, Jaime Alberto y SARACHE CASTRO, Willian Ariel. Técnicas Inteligentes y

Estocásticas en Scheduling: un Enfoque en la Producción y las Operaciones. Manizales: Universidad Nacional de Colombia, 2010.

3. CEVIKCAN, Emre; DURMUSOGLU, M. Bulent y BASKAK, Murat. Integrating parts design characteristics and scheduling on parallel machines. In: Expert Systems with Applications. September 2011, vol. 38, no. 3, p. 13232-13253.

4. DAI, Min; TANG, Dunbing; GIRET, Adriana; SALIDO, Miguel Angel y LI, Wei Dong. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. In: Robotics and Computer-Integrated Manufacturing. October 2013, vol. 29, no. 5, p. 418-429.

5. DOMÍNGUEZ-MACHUCA, Jose Antonio; GARCÍA GONZÁLEZ, Santiago; DOMÍNGUEZ-MACHUCA, Miguel Angel; RUIZ JIMÉNEZ, Antonio y ÁLVAREZ, Gil María José. Dirección de Operaciones. Aspectos estratégicos en la Producción y los servicios. Madrid: Editorial McGraw-Hill, 1995.

6. FENG, Haodi; LU, Shenpeng y LI, Xiuqian. Genetic algorithm for hybrid flow-shop scheduling with parrel batch processors. En: Actas del WASE International Conference on Information Engineering, 2009. ICIE '09. 2009, vol. 2, p. 9-13.

7. GALLEGO RENDON, Ramón A.; ESCOBAR Z., Antonio H. y ROMERO LÁZARO, Rubén A. Técnicas de Optimización Combinatorial. Pereira: Universidad Tecnológica de Pereira, 2006.

8. GÓMEZ-GASQUET, Pedro. Un nuevo algoritmo genético basado en un sistema multiagente para la programación de la producción en un taller de flujo híbrido. En: Actas del International Conference on Industrial Engineering & Industrial Management - CIO 2007. September 2007, p. 1675-1685.

9. GÓMEZ-GASQUET, Pedro; ANDRÉS, Carlos y LARIO, Francisco Cruz. An agent-based genetic algorithm for hybrid flowshops with sequence dependent setup times to minimise makespan. In: Expert Systems with Applications. July 2012, vol. 39, no. 9, p. 8095-8107.

10. HEKMATFAR, Masood; FATEMI-GHOMI, Seyyed Mohammad Taghi y KARIMI, Behrooz. Two stage reentrant hybrid flow shop with setup times and the criterion of minimizing makespan. In: Applied Soft Computing. December 2011, vol. 11, no. 8, p. 4530-4539.

11. HMIDA, Abir Ben; HAOUARI, Mohamed; HUGUET, Marie-José y LOPEZ, Pierre. Solving two-stage hybrid flow shop using climbing depth-bounded discrepancy search. In: Computers & Industrial Engineering. March 2011, vol. 60, no. 2, p. 320-327.

12. HOLLAND, John H. Adaptation in Natural and Artificial Systems. Michigan: The MIT Press, 1992.

13. LINN, Richard y ZHANG, Wei. Hybrid flow shop scheduling: A survey. En: Actas del 24° International Conference on Computers and Industrial Engineering. October 1999, vol. 37, no. 1-2, p. 57-61.

14. LÓPEZ VARGAS, Juan Camilo. Metodología de Programación de Producción en un Flow Shop Híbrido Flexible con el Uso de Algoritmos Genéticos para Reducir el Makespan. Aplicación en la Industria Textil. Tesis de Maestría. Manizales: Universidad Nacional de Colombia. Facultad de Ingeniería y Arquitectura. Departamento de Ingeniería Industrial, 2013.

15. MERCHÁN PAREDES, Luis y GÓMEZ, Diego Armando. Gestión de configuración. Validación de un modelo liviano para pequeñas empresas de desarrollo de software. En: Entramado. Enero-junio, 2011, vol. 7, no. 1, p. 190-201.

16. PHANDEN, Rakesh Kumar; JAIN, Ajai y VERMA, Rajiv. A genetic algorithm-based approach for job shop scheduling. In: Journal of Manufacturing Technology Management. 2012, vol. 23, no. 7, p. 937-946.

17. PINEDO, Michael L. Planning and Scheduling in Manufacturing and Services. New York: Springer Science + Business Media, LLC, 2009.

18. RADA, Omar; CHAVERRA, Yurlady; MORANTE, Diego Fernando y MOSQUERA, Omaira. La gestión tecnológica: Una herramienta para el desarrollo de la cadena productiva del ají en el Valle del Cauca. En: Entramado. Enero-junio, 2011, vol. 7, no. 1, p. 12-30.

19. RUIZ, Rubén, ŞERIFOĞLU, Funda Sivirikaya y URLINGS, Thijs. Modeling realistic hybrid flexible flowshop scheduling problems. In: Computers & Operations Research. April 2008, vol. 35, no. 4, p. 1151-1175.

20. SHABTAY, Dvir. The just-in-time scheduling problem in a flow-shop scheduling system. In: European Journal of Operational Research. February 2012, vol. 216, no. 3, p. 521-532.

21. SU, Zhixiong y LI, Tieke. Genetic algorithm for minimizing the makespan in hybrid flow shop scheduling problem. En: Actas del International Conference on Management and Service Science, 2009. MASS '09. September 2009, p. 1-4.

22. TANG, Jianchao; ZHANG, Guoji; LIN, Binbin y ZHANG, Bixi. Hybrid genetic algorithm for flow shop scheduling problem. En: Actas del 2010 International Conference on Intelligent Computation Technology and Automation (ICICTA). May 2010, vol. 2, p. 449-452.

23. TAVAKKOLI-MOGHADDAM, Reza; TAHERI, Farid; BAZZAZI, Mohammad; IZADI, Mohammad Javad y SASSANI, Farrokh. Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints. In: Computers & Operations Research. December 2009, vol. 36, no. 12, p. 3224-3230.

24. TAVARES NETO, Moacir y TAVARES NETO, Roberto Fernandes. Literature review regarding ant colony optimization applied to scheduling problems: Guidelines for implementation and directions for future research. In: Engineering Applications of Artificial Intelligence. January 2013, vol. 26, no. 1, p. 150-161.

25. TOLEDO, Claudio Fabiano Motta; DE OLIVEIRA, Renato Rosende Ribeiro; y MORELATO FRANÇA, Paulo. A hybrid multi-population genetic algorithm applied to solve the multi-level capacitated lot sizing problem with backlogging. In: Computers & Operations Research. April 2013, vol. 40, no. 4, p. 910-919.

26. VALLADA, Eva y RUIZ, Rubén. A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. In: European Journal of Operational Research. June 2011, vol. 211, no. 3, p. 612-622.

27. WANG, Shijin y LIU, Ming. A heuristic method for two-stage hybrid flow shop with dedicated machines. In: Computers & Operations Research. January 2013, vol. 40, no. 1, p. 438-450.

28. XIAO, Wendong; HAO, Peifeng; ZHANG, Sen y XU, Xinhe, Hybrid flow shop scheduling using genetic algorithms. En: Actas del 3° World Congress on Intelligent Control and Automation. June 2000, vol. 1, p. 537-541.

29. ZANDIEH, Mostafa y KARIMI, Nada. An adaptive multi-population genetic algorithm to solve the multi-objective group scheduling problem in hybrid flexible flowshop with sequence-dependent setup times. In: Journal Intelligent Manufacturing. December 2011, vol. 22, no. 6, p. 979-989.

30. ZHANG, Xiandong y VAN DE VELDE, Steff. Approximation algorithms for the parallel flow shop problem. In: European Journal of Operational Research. February 2012, vol. 216, no. 3, p. 544-552.