Ingeniería, investigación y tecnología
ISSN: 1405-7743
Universidad Nacional Autónoma de México, Facultad de Ingeniería

Espitia-Mendez, Julieth Andrea; Mendoza-Rojas, Germán Leonardo

Metodología basada en un algoritmo genético para programar la producción de una empresa del sector textil

Ingeniería, investigación y tecnología, vol. XXII, núm. 4, e1831, 2021, Octubre-Diciembre
Universidad Nacional Autónoma de México, Facultad de Ingeniería

DOI: https://doi.org/10.14482/INDES.30.1.303.661

Disponible en: https://www.redalyc.org/articulo.oa?id=40471804004

Ingeniería Investigación y Tecnología
volumen XXII (número 4), octubre-diciembre 2021 1-16
ISSN 2594-0732 FI-UNAM artículo arbitrado
Información del artículo: Recibido: 10 de noviembre de 2020, reevaluado: 21 de abril de 2021, aceptado: 7 de septiembre de 2021

https://doi.org/10.22201/fi.25940732e.2021.22.4.032

Ingeniería
Investigación y
Tecnología

Metodología basada en un algoritmo genético para programar la producción de una empresa del sector textil

Methodology based on genetic algorithm for a textil industry company production scheduling

Espitia-Mendez Julieth Andrea
Escuela Colombiana de Ingeniería Julio Garavito
Bogotá, Colombia
Correo: julieth.espitia0224@gmail.com
https://orcid.org/0000-0001-8445-0225

Mendoza-Rojas Germán Leonardo
Escuela Colombiana de Ingeniería Julio Garavito
Bogotá, Colombia
Correo: Leonardo.mendoza_09@hotmail.com
https://orcid.org/0000-0003-4924-1916

Resumen

En este trabajo se presenta la aplicación de un sistema basado en algoritmos genéticos, dirigida a la minimización del *makespan* (C_max) y cantidad de trabajos tardíos (U(γ)) en un ambiente *flow shop* híbrido flexible, perteneciente a la industria textil. La metodología fue evaluada en escenarios de producción reales de una empresa. El resultado del modelo desarrollado refleja una disminución de C_max y U(γ) sobre las muestras reales de la empresa, presentando una disminución de 73 % de C_max y una reducción de 11 a 0 trabajos tardíos respecto a la muestra.
Descriptores: *Makespan*, trabajos tardíos, algoritmo genético, *flow shop*, multiobjetivo.

Abstract

This paper presents the application of a system based on genetic algorithms, aimed at the optimization of the makespan and the amount of late works in a flexible hybrid flow shop environment in the textile industry. The methodology was evaluated in real production scenarios of the company. The developed model results reflects a decrease of C_max and U(γ), over the real company´s sample results, presenting a 79 % decrease of C_max and the reduction from 11 to 0 late work compared to the real scenarios.
Keywords: Makespan, late jobs, genetic algorithm, flow shop, multiobjetive.

Introducción

El principal objetivo de este estudio es desarrollar e implementar una metodología de programación de producción basada en el uso de algoritmos genéticos como método de solución para minimizar el *makespan* y los trabajos tardíos de la línea textil de la empresa objeto de estudio.

La empresa objeto de estudio fabrica un textil técnico de cierre de contacto, este producto maneja dos clases, cierre de contacto tipo felpa y tipo gancho. Dado que los dos tipos de artículos omiten al menos una de las etapas del proceso de fabricación y además se cuenta con máquinas paralelas en dos de las fases de producción se determina que esta planta tiene una configuración *flow shop* híbrido flexible.

El estudio se divide en las siguientes secciones: marco teórico, en donde se analizan las características de un ambiente *flow shop* híbrido flexible, metaheurísticas, metodología propuesta, análisis experimental, análisis de resultados y sus correspondientes conclusiones y recomendaciones.

Marco teórico

Flow Shop

El problema de *flow shop* es de programación en el que cierta cantidad de trabajos (n) tienen que ser procesados por m máquinas, este problema se denomina NP-Hard cuando el número de máquinas es mayor a 3 ($m > 3$) (Revetti *et al.*, 2012), cada trabajo j $(j = 1, 2,\ldots,n)$ pasa a través de las m máquinas en un orden definido, requiere un tiempo de procesamiento p (i,j) en cada máquina i $(i = 1, 2, \ldots,m)$ y tiene una fecha de entrega r (j) (Ren *et al.*, 2012). En el tipo de configuración *flow shop* tradicional, los trabajos deben ser procesados en cada una de las etapas, lo que significa que no se permite omitir fases en el proceso productivo. Gomez *et al.* (2012) abordaron un tipo de configuración *flow shop* con tiempos de preparación dependientes de la secuencia con el objetivo de minimizar el *makespan* de un conjunto de datos de problemas de referencia. Para esto, ellos diseñaron un algoritmo genético mejorado al que denominan *Multi-Agent Genetic Scheduling Algorithm* (MASGA) usando el paradigma del agente del *software*, con el que obtienen mejoras significativas en el *makespan* logrando el objetivo planteado. En este artículo se expone la alternativa de mejorar los resultados de Gomez *et al.* (2012) combinando y modificando un algoritmo genético simple con otras metodologías, se realiza un detallado análisis teórico del ambiente de producción *flow shop*.

Dadas las necesidades de la industria para incluir más maquinas a una o varias etapas de producción, se presenta el sistema *Flow Shop Híbrido* (HFS), el cual puede verse como una generalización de dos tipos de problema de programación: el problema de Programación de Máquina Paralela (PMS) y el problema de Programación del *Flow Shop* (FSS). La decisión clave del problema PMS es la asignación de trabajos a las máquinas y la del FSS es la secuencia de trabajos a través del flujo productivo, por lo tanto, una vez que la configuración del HFS ha sido diseñada, la principal decisión en el funcionamiento de esta es programar los trabajos a las máquinas en cada una de las etapas, es decir, determinar el orden en el que se realizarán los trabajos (Ribas *et al.*, 2010). Por otro lado, el *flow shop* híbrido flexible es una extensión del *flow shop* híbrido, donde los trabajos siguen presentando una secuencia lineal a través de las etapas, pero uno o más trabajos pueden omitir varias etapas durante su procesamiento (Zandieh & Karimi, 2011).

El problema de programación en un *flow shop* híbrido flexible donde existen más de dos etapas, además de la cantidad de máquinas que intervienen en el proceso y el número y variedad de trabajos a producir, genera un crecimiento exponencial de la cantidad de alternativas de solución, es decir, las posibles combinaciones de asignaciones de los trabajos a las máquinas en las diferentes etapas son demasiadas y llevan a que el problema obtenga un nivel de complejidad NP-Hard. Para encontrar la solución a este tipo de problemas han surgido distintas herramientas de optimización combinatoria que tienen la capacidad de encontrar soluciones de muy buena calidad en tiempos de cómputo razonables, estas son conocidas en la literatura como metaheurísticas (López & Arango, 2015).

Metaheurísticas

Las metaheurísticas cuentan con las propiedades indicadas a continuación (Chicano, 2007):

a) Son métodos que orientan el proceso de búsqueda.
b) El objetivo es realizar un análisis eficiente del espacio de búsqueda para encontrar resultados semi óptimos.
c) Las metaheurísticas son algoritmos no exactos y generalmente no deterministas.
d) Pueden agregar mecanismos para evitar regiones poco favorables del espacio de búsqueda.
e) El esquema básico de cualquier metaheurística tiene una estructura preestablecida.

f) Utiliza el conocimiento del problema que se trata de resolver en forma de heurísticos que son controlados por una estrategia de más alto nivel.
g) No saben si llegan a la solución óptima, por lo tanto, se les debe indicar una cantidad de iteraciones finita y limitada.
h) Aceptan malos movimientos, es decir, se trata de procesos de búsqueda en los que cada nueva solución no es necesariamente mejor que la anterior. Algunas veces aceptan soluciones no factibles como paso intermedio para acceder a nuevas regiones no exploradas.

Comparadas con las heurísticas, las metaheurísticas encuentran soluciones superiores con mayores esfuerzos computacionales. Estas técnicas de optimización han sido utilizadas en gran variedad de aplicaciones en campos como la ingeniería, economía, teoría de juegos, ciencias computacionales, mercadeo, biología, medicina, entre otras, logrando ajustarse para cada caso y obteniendo buenos resultados (Acuña *et al.,* 2012). Las metaheurísticas más conocidas son algoritmos genéticos, búsqueda tabú, recocido simulado, colonia de hormigas, algoritmo GRASP, entre otras.

Algoritmos genéticos

Los algoritmos genéticos son una meta heurística que imita en su funcionamiento a la evolución natural de las especies y su desarrollo se atribuye a John Holland (Gestal, 2013).

Existen elementos fundamentales en el algoritmo genético: la representación de los individuos (cromosoma), la función objetivo o función de aptitud que mide si el individuo es apto como solución, la población inicial, el reemplazo de la población y el criterio de parada (Golberg, 1989). También existen tres operaciones esenciales que se aplican a los cromosomas: selección, cruce y mutación.

Cromosoma: Esta codificación debe contener información acerca de la solución que representa y se puede construir de varias formas, la elección de la forma de codificar depende del problema a resolver. Las clases de representación de cromosomas según (Arranz & Parra, n.d) son: codificación binaria, numérica, por valor directo o codificación de árbol. Para la programación de producción se han propuesto métodos como codificación basada en los trabajos, codificación basada en las máquinas y codificación basada en la operación. Chen *et al.* (2008) diseñan en su artículo un algoritmo genético híbrido para minimizar el *makespan* en un *Flow Shop Reentrante* (RFS), esta investigación fue la base para definir una codificación de cromosoma basada en los trabajos.

Población inicial: Corresponde al conjunto de posibles soluciones del problema, es sometida a evaluación y se seleccionan los individuos (cromosomas) más aptos según su aptitud para dar origen a nuevos cromosomas y a nuevas poblaciones. Puede ser generada de forma aleatoria, sin incluir ningún conocimiento previo o mediante heurísticas que introducen conocimiento previo de un conjunto probable de buenas soluciones. Salazar & Sarzuri (2015) utilizaron un algoritmo genético mejorado para minimizar la tardanza total en un entorno de producción *flow shop* flexible con tiempos de preparación dependientes de la secuencia. De este artículo se consideró una heurística para generar una población adaptada al problema con el fin de mejorar el rendimiento del algoritmo genético.

Función fitness: La función de evaluación o *fitness* establece una medida numérica de la bondad de una solución, esta medida recibe el nombre de ajuste. En el funcionamiento de los algoritmos genéticos esta medición servirá para controlar la aplicación de los operadores genéticos (Gestal *et al.,* 2010).

Reemplazo de la población: Establece el criterio de los supervivientes en cada iteración generacional. Dependiendo de los métodos de selección y cruce que se utilicen, es posible que al final de cada iteración el tamaño de la población inicial aumente, para evitar esto, se pueden escoger varios individuos y los descendientes que estos han generado. Existen varias formas de generar la población de reemplazo (Corral, 2017): generacional, estado-estacionario, gap generacional y reemplazo catastrofista.

Criterio de parada: El proceso iterativo del algoritmo finaliza en caso de que se cumpla una condición establecida, dentro de la cual se podría encontrar el procesamiento de una cantidad definida de generaciones, la aptitud de una cantidad que supere un valor definido o que el valor de la función objetivo no presente una mejor solución después del número de iteraciones establecido (Solari & Ocampo, 2006).

Selección: Fase responsable de seleccionar a los individuos que serán cruzados o mutados con el fin de generar una nueva población de individuos procesados (Gesta *et al.,* 2010). Esta selección se puede realizar de varias formas: por ruleta, por torneo, por rango, elitista y jerárquica.

Cruce: Consiste en mezclar el material genético de dos cromosomas que serán progenitores y que han sido seleccionados (utilizando cualquier técnica). Por lo general, los cromosomas son aleatoriamente divididos y

mezclados, por lo que ciertos genes de los descendientes provienen de un progenitor, mientras otros provienen del otro, es decir, los descendientes tendrán en su genotipo información de ambos progenitores. El cruce garantiza la conservación de las características que hacen a un individuo fuerte para sobrevivir y la eliminación de las que lo hacen débil (Aguilar, 2016). Las diferentes técnicas de cruce que más se usan son: cruce de un punto, cruce de dos puntos, cruce uniforme, Operador *Partially Mapped Crossover*, entre otras.

Mutación: Tras el cruce, tiene lugar la mutación. Este operador transforma levemente a un individuo, puede evitar la convergencia y cambia la dirección de búsqueda. Se eligen dos posiciones en forma aleatoria entre 1 y n, los genes de cada posición se intercambian. La probabilidad de mutación generalmente es menor a 1 %, sin embargo, se realizan mutaciones para garantizar que ningún punto del espacio de búsqueda no tenga la probabilidad de ser evaluado (Salazar & Sarzuri, 2015).

Algoritmos Genéticos Multiobjetivo: La optimización multiobjetivo involucra múltiples objetivos frecuentemente en conflicto, que deben ser solucionados de forma simultánea (Esquivel *et al.,* 2002).

Estableciendo el problema que cuenta con $k \leq 2$ FOB (de ahora en adelante, funciones objetivo) a ser maximizadas o minimizadas de forma simultánea, la solución de este problema busca establecer la solución x perteneciente al grupo de soluciones factibles X que den solución al problema:

$$Min\ x \in X\ f(x) = \{f1\ (x), f2\ (x), ..., f\,k(k)\}$$

Para la solución de los problemas multiobjetivo Schaffer (1985) propone el primer algoritmo genético multiobjetivo (llamado VEGA-Algoritmos genéticos evaluados por vectores), cuya idea principal se ve plasmada en la división de la población en subpoblaciones (de igual tamaño), para cada subpoblación, se asigna un único objetivo para su respectiva búsqueda de solución; dando así, un procedimiento de selección de mejor hijo independiente por cada uno de los objetivos, así como la ejecución de los cruces a través de límites de subpoblación (Schaffer, 1985).

En los diferentes estudios analizados se encontró que la función objetivo más estudiada ha sido el *makespan*; muy pocos se enfocan en el criterio de trabajos tardíos y la mayoría utiliza como método de solución el algoritmo genético. Autores como Zobolas *et al.* (2009), desarrollaron una propuesta de metaheurística híbrida, diseñando múltiples componentes para el proceso del algoritmo, primero, el desarrollo de un método de heurística constructiva aleatoria para la generación de la población inicial, posteriormente un algoritmo genético (GA) para soluciones generales y una posterior búsqueda de vecindad variable (VNS) para la mejora en las soluciones arrojadas, minimizando el *makespan* en un ambiente *flow shop* de permutación, esta investigación atribuye el concepto de combinación de ciclos para el proceso de búsqueda de la mejor solución. Karmakar & Manhaty (n.d) centraron su investigación en un problema de *Flexible Flow Shop* (FFS), donde buscaron la minimización del *makespan* a través de la aplicación de la teoría de restricciones (TOC) y un algoritmo genético que busca soluciones para el escenario de múltiples productos en máquinas en etapas con tiempos de preparación dependientes de la secuencia y almacenamientos intermedios infinitos, con esta investigación se complementa la construcción de la lógica correspondiente a tiempos de preparación y aplicación de restricciones a la primera fase del modelo propuesto. Zhang (2019) planteo para un modelo de FFSP (*Flexible Flow Shop Shceduling Problem*) una consideración del uso de GA mezclado con el algoritmo heurístico de Palmer para la solución de un objetivo, para esto, se realiza el diseño del simulador de eventos discretos (DES) en el cual se plantean diferentes escenarios de *testing* para corroborar la efectividad del modelo planteado. Oguz & Ercan (2005) estudiaron un ambiente *flow shop* híbrido con tareas multiprocesadas en el que emplean un algoritmo genético con un nuevo modelo *crossover*, un test preliminar de resultados parametrizables, y una generación de mejores soluciones, posteriormente realizan una comparación con la metaheurística búsqueda tabú para encontrar el mejor *makespan*. Este artículo atribuye al modelo construido el planteamiento de tests preliminares y los resultados parametrizables para encontrar mejores soluciones. Toro *et al.* (2006) resuelven el problema de minimización del tiempo total de producción utilizando el algoritmo genético modificado de Chu-Beasley, este artículo atribuye a la investigación la selección del método de cruce y mutación, así como el proceso de modificación de la población N+1. Salazar & Sarzuri (2015) estudiaron una configuración *flow shop* flexible con tiempos de preparación dependientes de la secuencia en el que usan un algoritmo genético mejorado con el objetivo de minimizar la tardanza total del proceso de producción. Otro punto que se pudo observar en la revisión del estado del arte es que son pocos los autores que enfocan la investigación a problemas multiobjetivo, aun cuando; los problemas reales de ingeniería tienen múltiples objetivos. Allahverdi & Al-

Anzi (2008) proponen tres métodos: recocido simulado, colonia de hormigas y evolución diferencial auto adaptativa para minimizar tiempo de finalización medio y *makespan*. Jarboui *et al.* (2011) buscan minimizar el *makespan* y el tiempo de flujo total en un ambiente *flow shop* sin espera para más de dos máquinas, para ello proponen un algoritmo genético híbrido en el que usan la metodología de vecino más próximo como mecanismo de mejora en el último paso del algoritmo. Czajkowsk & Kretowski (2019) construyen una solución basada en *Global Model Tree* (GMT) que atribuye a los algoritmos la capacidad de búsqueda de soluciones multiobjetivo con la generación de soluciones que satisfacen los objetivos en paralelo, y un posterior proceso de Pareto para su decisión de mejor solución. Adicionalmente, se resaltan componentes de funciones de aglomeración y soluciones elitistas, lo cual atribuye al proyecto en puntos de vista para la mejor unificación de soluciones en las dos funciones objetivo planteadas. En la literatura existente son escasos los trabajos que abordan sistemas reales de manufactura, la mayoría de las investigaciones son desarrolladas sobre conjuntos de problemas de pruebas generados aleatoriamente. Es por eso que este trabajo supone un gran aporte al plantear la solución de dos objetivos (*makespan* y trabajos tardíos) e implementarse en un ambiente real de manufactura (textiles técnicos).

Metodología propuesta

El método seleccionado ha sido el algoritmo genético multi objetivo (MOGA), considerando las premisas del ambiente *flow shop* híbrido flexible. A continuación, se enumeran los supuestos manejados en el estudio:

a) Cada máquina tiene la capacidad de procesar como máximo un montaje (conjunto de artículos por tipo y color) a la vez.
b) Cada montaje es procesado solamente en una máquina a la vez.
c) Toda máquina se caracterizará como disponible en el momento T=0 y no se tendrán en cuenta los tiempos de mantenimiento.
d) No se permiten interrupciones durante el procesamiento de los montajes en cada máquina.
e) Los tiempos de procesamiento son determinísticos y previamente conocidos.
f) Los tiempos de montaje y alistamiento de máquina, están inmersos en los tiempos de procesamiento en cada una de las etapas.
g) La flexibilidad del *flow shop* híbrido en el modelo se verá reflejada a partir de la asignación de un tiempo de procesamiento igual a 0 (cero) en la (s) etapas (s) donde el producto no requiera de su procesamiento.
h) No se tendrán en cuenta los tiempos de transporte entre etapas, dado que no son relevantes en el tiempo del proceso.
i) Todos los artículos tienen la misma prioridad para su procesamiento.

Para la metodología se definieron los siguientes conjuntos, parámetros y variables:

a) Conjuntos:

γ = artículo
m = máquina
t = turno
j_y = posición de artículo
$P_{\gamma\eta}$ = agrupación de artículos por tipo de producto (gancho o felpa) y de color η

b) Parámetros:

α = iteraciones
β = subpoblaciones
μ = cromosomas
ш = % de mutación
η = color
λ = tipo de artículo
$R_{\gamma\eta}$ = tiempo de procesamiento del artículo γ
$F_{\gamma\eta}$ = due date del artículo γ de color η
$N_{\gamma\eta}$ = cantidad de metros del artículo γ de color η

c) Variables:

T_{γ} = asignación del artículo γ al turno T
{1 *si el atículo* γ *es asignado al turno t,* 0 *en caso contrario*}
$x_{\gamma tm}$ = asignación del artículo γ al turno T en la máquina m
{1 *si el atículo* γ *es asignado al turno t en la máquina m* 0 *en caso contrario*}
$G_{\gamma mj}$ = asignación del artículo γ en la máquina *m* en la posición j
{1 *si el atículo* γ *es asignado a la máquina m en la posición j,* 0 *en caso contrario*}
Q_{γ} = tiempo de terminación del artículo γ
$U(\gamma)$ = {1, *si el artículo es tardío* ($R_{\gamma\eta} > F_{\gamma\eta}$) 0, *en caso contrario*}
C_{max} = *Makespan*, tiempo máximo de terminación de todos los artículos en la última etapa de procesamiento.

Con base en los conjuntos, parámetros, variables y supuestos definidos en el estudio, el modelo propuesto se

divide en dos fases, la primera corresponde al método de asignación de artículos (γ) al proceso de tintorería (este proceso cuenta con tres máquinas paralelas con diferente capacidad, pero con el mismo tiempo de procesamiento), donde se utiliza la heurística basada en la regla de despacho LPT dando como resultado una matriz que representa la asignación de los artículos (γ) en las tres máquinas (m=3) de tintorería y en los turnos (t) con duración igual a 3 horas, capacidad de tres turnos al día y un tamaño de artículo (γ) adaptado a las restricciones de capacidad de maquinaria. La segunda fase abarca la programación de artículos (γ) desde el proceso de mageba (proceso posterior a la tintorería) hasta el empaque (proceso final) mediante el uso de un algoritmo genético cuyos resultados serán medidos por *Makespan* (C_{max}) y por la sumatoria de trabajos tardíos ($\Sigma U(\gamma)$).

La metodología diseñada se diferencia de la expuesta por Jarboui *et al.* (2011) en el momento de uso de las heurísticas, ya que Jarboui *et al.* (2011) usó la heurística del vecino más próximo en el último paso del algoritmo genético para prevenir que las posibles soluciones quedaran en el espacio de óptimos locales; en esta investigación se utilizó la heurística LPT para la generación de la población inicial del algoritmo teniendo en cuenta las restricciones de capacidad de las máquinas del proceso de tintorería.

La descripción de la metodología se indica a continuación:

Fase 1. Asignación de artículos a proceso de tintorería:

1. Realizar consolidación de la totalidad de artículos (γ) pertenecientes a las órdenes de venta cargadas en SAP para su programación de producción.
2. Consultar la cantidad de metros de cada artículo (γ), y verificar que no supere la capacidad máxima de metros (capacidad máxima: 1920 m, correspondiente a la capacidad instalada de la máquina 1 de tintorería).

   2.1 Si la cantidad de metros de un artículo (γ) es mayor a 1920 m, se divide el número de metros del artículo (γ) en 1920 generando un artículo (γ_k). En la Figura 1 es visible la transformación de la información a procesar cuando los artículos superan la cantidad de 1920 m. En este ejemplo, los artículos con Id 2 y 6 se dividen y se generan los artículos 2.0 y 6_0 con cantidad de metros igual a 1920 y los artículos 2_1 y 6_1 con la cantidad de metros restantes al valor original.

   2.2 Si la cantidad de metros del artículo (γ) es menor o igual a 1920 m se conserva el Id del artículo como único.

Id Artículo	Tamaño
1	1200
2	1950
3	100
4	1200
5	1000
6	1990
7	1200
8	200
9	700
10	940
11	800
12	160
13	120

Id Artículo	Tamaño
1	1200
2_0	1920
3	100
4	1200
5	1000
6_0	1920
7	1200
8	200
9	700
10	940
11	800
6_1	70
2_1	30
12	160
13	120

Figura 1. Transformación de artículos a procesar cuando el tamaño es mayor a 1920 m
Fuente: Elaboración propia

3. Agrupar los artículos (γ) y los artículos (γ_k) por tipo de artículo (λ) (felpa y gancho), creando los conjuntos de artículos (P_λ).

$P_{\lambda\eta}$	Artículos γ	Artículos γ	Artículos γ_k	$P\gamma$	Color (η)
3	3	100		F	Verde 1
3	4	1200		F	Verde 1
3	11	800		F	Verde 1
3	13	120		F	Verde 1
1	1	1200		F	Negro
1	2	1950		F	Negro
1	6_1		70	F	Negro
1	2_1		30	F	Negro
1	8	200		G	Negro
1	9	700		G	Negro
4	6	1990		G	Azul
2	5	1000		G	Amarillo
2	7	1200		G	Amarillo
2	10	940		G	Amarillo
2	12	160		G	Amarillo

Tabla 1. Tabla de agrupación de artículos
Fuente: Elaboración propia

4. Desglosar los conjuntos de tipo de artículo (P_λ), en agrupación por color (η) generando un conjunto ($P_{\lambda\eta}$) (Tabla 1).
5. Realizar la sumatoria de metros correspondientes a los artículos (γ) y artículos (γ_k), que contiene cada conjunto ($P_{\lambda\eta}$).

6. Ordenar los conjuntos ($P_{\lambda\eta}$) de mayor a menor cantidad en metros (Tabla 2).

Tabla 2. Agrupación de conjuntos

$P_{\lambda\eta}$	$\Sigma\gamma+\gamma k$
1	5050
2	3300
3	2220
4	1990
Total	12560

Fuente: Elaboración propia

7. Evaluar la asignación de conjuntos de artículos ($P_{\lambda\eta}$) en orden de mayor a menor cantidad de metros y a la maquinaria en orden de mayor a menor capacidad, teniendo en cuenta las capacidades y la capacidad de turnos a asignar (Tabla 3).

8. Generar una subpoblación (β) (cantidad de subpoblaciones parametrizable) seleccionando de forma aleatoria los artículos (γ) que contiene el conjunto ($P_{\lambda\eta}$) y asignándolos a cada máquina (m) y a cada turno (t) teniendo en cuenta que cada uno de los artículos (γ) a asignar, deberá evaluar la capacidad de cada turno de la máquina, si el tamaño del artículo (γ) es menor o igual a la capacidad del turno (t) en la maquina (m), este deberá ser asignado a la máquina (m) que cumpla con estas restricciones, esto, aprovechando la capacidad máxima de cada turno (t) y máquina (m). A continuación la descripción de esta lógica:
   8.1 Máquina 1: ¿La máquina m en el turno t (t de menor a mayor), tiene capacidad para el artículo (γ)?
   Sí: Asignar el artículo al turno "t".
   No: validar capacidad del turno "t+1" para el artículo (γ).
   8.2 Máquina 2: ¿El artículo (γ) es mayor a la capacidad de la maquina 2?
   Sí: Asignar nuevo turno "t" a la máquina 1.
   No: Asignar a turno "t" de máquina 2 teniendo en cuenta la lógica de capacidad por turno "t".
   Máquina 3: ¿El artículo (γ) es mayor a la capacidad de la maquina 3?
   Sí: Evaluar restricción 1 (¿La cantidad de metros del trabajo (γ) es menor o igual a la capacidad de la máquina 2?).
   Sí: Asignar a turno de máquina 2 teniendo en cuenta la lógica de capacidad por turno "t".
   No: Asignar nuevo turno M_t a la máquina 1 teniendo en cuenta la lógica de capacidad por turno "t".
   No: Asignar a máquina 3 teniendo en cuenta la lógica de capacidad por turno "t". (Figura 2).
9. Una vez la totalidad de artículos (γ) sean distribuidos en la matriz de máquinas / turnos se da por cerrada la generación de la subpoblación (β) y se continuará con la subpoblación (β+1) hasta la cantidad de subpoblaciones (β) parametrizadas.

Fase 2. Algoritmo genético

1. Una vez generada la subpoblación (β) reflejada en la asignación de artículos (γ) a máquinas (m) y turnos (t), se construye un cromosoma compuesto por genes que representan los turnos de las máquinas de tintorería, los cuales se componen de los artículos ordenados en secuencia de procesamiento para las etapas de mageba en adelante.
   La Figura 3 representa una subpoblación y la Figura 4 representa un cromosoma generado a partir de la subpoblación.
   Cada cromosoma se compone de un grupo de genes conformados por los artículos (γ) ordenados de forma aleatoria y asignados a cada turno.

Tabla 3. Tabla asignación de artículos a máquinas y turnos Fuente: Elaboración propia

Capacidades		Turno 1	Turno 2	Turno 3	...	Turno N
1920	Máquina 1					
960	Máquina 2					
160	Máquina 3					

- La máquina 1, tiene una capacidad de 1920 metros por turno
- Cada máquina, tendrá un N de turnos que serán asignados según la cantidad de artículos / conjuntos a producir

Capacidades Max	Máquina	T_1	T_2	T_3	T_4	T_5	T_6
960 - 1920	Autoclave 1	1	2_0	4 ; 3	6_0	5	7
160 - 960	Autoclave 2	8 ; 9	10	11			
0 - 160	Autoclave 3	6_1 ; 2_1	12	13			

Figura 2. Ejemplo de asignación de artículos a máquinas y turnos
Fuente: Elaboración propia

https://doi.org/10.22201/fi.25940732e.2021.22.4.032

ID subpoblación	Capacidades Max	Máquina	T_1	T_2	T_3	T_4	T_5	T_6
1	960 - 1920	Autoclave 1	1	2_0	4 ; 3	6_0	5	7
	160 - 960	Autoclave 2	8 ; 9	10	11			
	0 - 160	Autoclave 3	6_1 ; 2_1	12	13			

Figura 3. Representación de una subpoblación
Fuente: Elaboración propia

Gen 1 Gen 2

Turno (T)	T_1	T_2	T_3	T_4	T_5	T_6
Secuencia (y)	1; 6_1; 8; 9; 2_1	12; 2_0; 10	4; 13; 11; 3	6_0	5	7

Figura 4. Representación de un cromosoma
Fuente: Elaboración propia

ID subpoblación	Capacidades Max	Máquina	T_1	T_2	T_3	T_4	T_5	T_6
1	960 - 1920	Autoclave 1	1	2_0	4 ; 3	6_0	5	7
	160 - 960	Autoclave 2	8 ; 9	10	11			
	0 - 160	Autoclave 3	6_1 ; 2_1	12	13			
2	960 - 1920	Autoclave 1	4	2_0	1	5	6_0	7
	160 - 960	Autoclave 2	9	11	8; 3	10		
	0 - 160	Autoclave 3	13	6_1; 2_1	12			
3	960 - 1920	Autoclave 1	1;6_1	4	6_0	2_0	5	7
	160 - 960	Autoclave 2	10	11; 3	9	8		
	0 - 160	Autoclave 3	2_1	12	13			

ID subpoblac	ID Cromosoma
1	1
1	2
1	3
2	1
2	2
2	3
3	1
3	2
3	3

T_1	T_2	T_3	T_4	T_5	T_6
1; 6_1; 8; 9; 2_1	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
1; 2_1; 8 ; 6_1; 9	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
2_1; 6_1; 8; 9; 1	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
13; 4; 9	2_0; 6_1; 11; 2_1	1; 12; 8; 3	10; 5	7	6_0
9; 13; 4	2_1; 11; 2_0; 6_1	3; 1; 12; 8	10; 5	7	6_0
13; 9; 4	6_1; 2_0; 11; 2_1	12; 8; 3; 1	5; 10	7	6_0
10; 2_1; 1; 6_1	12; 3; 4; 11	13; 6_0; 9	8; 2_0	5	7
6_1; 2_1; 1; 10;	11; 3; 4; 12	9; 6_0; 13	2_0; 8	5	7
2_1; 6_1; 10; 1	3; 12; 4; 11	6_0; 13; 9	2_0; 8	5	7

Figura 5. Ejemplo de generación de subpoblaciones y cromosomas
Fuente: Elaboración propia

2. Generar una cantidad (parametrizable) de cromosomas (μ) por cada subpoblación (β). A través de la Figura 5 se puede ver un ejemplo de generación de tres subpoblaciones con tres cromosomas parametrizados.
3. Medir el *makespan* y trabajos tardíos a cada uno de los cromosomas (μ) generados, las funciones objetivo son tomadas como las funciones *fitness* planteadas para la evaluación de los individuos (Figura 6).
4. Una vez ejecutada la iteración (α) 0, que genera las subpoblaciones (β) y sus correspondientes cromosomas (μ), la selección del mejor cromosoma (μ), deberá basarse en los resultados del *makespan* o en la sumatoria de la cantidad de trabajos tardíos dados por cada subpoblación (β), según la medición seleccionada en cada iteración (α), la cual, deberá ser de forma intercalada:
   - Iteración 0: *Makespan*
   - Iteración 1: Trabajos tardíos
   - Iteración 2: *Makespan*

Identificadores	
ID subpoblac	ID Cromosoma
1	1
1	2
1	3
2	1
2	2
2	3
3	1
3	2
3	3

Cromosomas					
T_1	T_2	T_3	T_4	T_5	T_6
1; 6_1; 8; 9; 2_1	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
1; 2_1; 8 ; 6_1; 9	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
2_1; 6_1; 8; 9; 1	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
13; 4; 9	2_0; 6_1; 11; 2_1	1; 12; 8; 3	10; 5	7	6_0
9; 13; 4	2_1; 11; 2_0; 6_1	3; 1; 12; 8	10; 5	7	6_0
13; 9; 4	6_1; 2_0; 11; 2_1	12; 8; 3; 1	5; 10	7	6_0
10; 2_1; 1; 6_1	12; 3; 4; 11	13; 6_0; 9	8; 2_0	5	7
6_1; 2_1; 1; 10;	11; 3; 4; 12	9; 6_0; 13	2_0; 8	5	7
2_1; 6_1; 10; 1	3; 12; 4; 11	6_0; 13; 9	2_0; 8	5	7

FOB	
Makespan	Q Trabajos T.
2329	4
2325	2
3475	2
1490	4
2066	0
2167	0
2561	3
3295	0
3383	3

Figura 6. Medición de *fitness* a cada individuo Fuente: Elaboración propia

https://doi.org/10.22201/fi.25940732e.2021.22.4.032

Como ejemplo, en la Figura 7 se seleccionan los mejores padres de la subpoblación dos según medición de *makespan* (medición seleccionada para la iteración 0), los cuales se encuentran resaltados en color amarillo.

1. Una vez dada la selección de los mejores cromosomas (μ) según *makespan* de cada subpoblación (β), se debe llevar a cabo el proceso de cruce uniforme, el cual deberá realizarse intercambiando la posición de los genes (Figura 8).
2. Una vez cruzados los cromosomas, se realiza la medición del *fitness* (Figura 9).

7. Al generar los hijos, determinar si se realiza el proceso de mutación de acuerdo con los criterios descritos a continuación:
   - El % de probabilidad de mutación (ш) deberá ser parametrizable
   - El número aleatorio para determinar si se realiza la mutación, será realizado para cada uno de los hijos:

   Hijo 1

   - **% Mutación parametrizado**: 10 % (resultados de 1 a 10 en número aleatorio de 1 a 100)
   - **Número aleatorio:** 86
   - **Resultado:** No mutación

   Hijo 2

   - **% Mutación parametrizado**: 10 % (resultados de 1 a 10 en número aleatorio de 1 a 100)
   - **Número aleatorio:** 5
   - **Resultado:** Cromosoma a mutar
8. En caso de que se cumpla el porcentaje para el proceso de mutación, en este caso, el hijo Núm. 2 del proceso de cruce de cromosomas presentado en el numeral 6, se deberán seleccionar dos genes de forma aleatoria, e intercambiar sus posiciones (Figura 10).
9. Los cromosomas (μ) cruzados y mutados (si aplica), deberán reemplazar a los cromosomas (μ) con mayor *makespan* generados en la iteración 0 para generar la nueva población; este método se denomina reemplazo por estado estacionario, teniendo en cuenta los siguientes puntos:
   - Si los 2 cromosomas (μ) hijos no mutaron, pasan a reemplazar los 2 peores de la iteración 0

Identificadores	
ID subpoblac	ID Cromosoma
1	1
1	2
1	3
2	1
2	2
2	3
3	1
3	2
3	3

Cromosomas					
T_1	T_2	T_3	T_4	T_5	T_6
1; 6_1; 8; 9; 2_1	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
1; 2_1; 8 ; 6_1; 9	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
2_1; 6_1; 8; 9; 1	12; 2_0; 10	4; 13; 11; 3	6_0	5	7
13; 4; 9	2_0; 6_1; 11; 2_1	1; 12; 8; 3	10; 5	7	6_0
9; 13; 4	2_1; 11; 2_0; 6_1	3; 1; 12; 8	10; 5	7	6_0
13; 9; 4	6_1; 2_0; 11; 2_1	12; 8; 3; 1	5; 10	7	6_0
10; 2_1; 1; 6_1	12; 3; 4; 11	13; 6_0; 9	8; 2_0	5	7
6_1; 2_1; 1; 10;	11; 3; 4; 12	9; 6_0; 13	2_0; 8	5	7
2_1; 6_1; 10; 1	3; 12; 4; 11	6_0; 13; 9	2_0; 8	5	7

FOB	
Makespan	Q Trabajos T.
2329	4
2325	2
3475	2
1490	4
2066	0
2167	0
2561	3
3295	0
3383	3

Figura 7. Selección de mejores padres según *makespan* Fuente: Elaboración propia

Padre 1

T_1	T_2	T_3	T_4	T_5	T_6
13; 4; 9	2_0; 6_1; 11; 2_1	1; 12; 8; 3	10; 5	7	6_0

Padre 2

T_1	T_2	T_3	T_4	T_5	T_6
9; 13; 4	2_1; 11; 2_0; 6_1	3; 1; 12; 8	10; 5	7	6_0

Hijo 1

T_1	T_2	T_3	T_4	T_5	T_6
13; 4; 9	2_1; 11; 2_0; 6_1	1; 12; 8; 3	10; 5	7	6_0

Hijo 2

T_1	T_2	T_3	T_4	T_5	T_6
9; 13; 4	2_0; 6_1; 11; 2_1	3; 1; 12; 8	10; 5	7	6_0

Figura 8. Proceso de cruce uniforme
Fuente: Elaboración propia

Hijo 1

T_1	T_2	T_3	T_4	T_5	T_6
13; 4; 9	2_1; 11; 2_0; 6_1	1; 12; 8; 3	10; 5	7	6_0

Makespan	Q Trabajos T.
2025	1

Hijo 2

T_1	T_2	T_3	T_4	T_5	T_6
9; 13; 4	2_0; 6_1; 11; 2_1	3; 1; 12; 8	10; 5	7	6_0

Makespan	Q Trabajos T.
2320	0

Figura 9. Generación de hijos en proceso de cruce
Fuente: Elaboración propia

https://doi.org/10.22201/fi.25940732e.2021.22.4.032

Antes:

T_1	T_2	T_3	T_4	T_5	T_6
9; 13; 4	2_0; 6_1; 11; 2_1	3; 1; 12; 8	10; 5	7	6_0

Makespan	Q Trabajos T.
2320	0

Mutado:

T_1	T_2	T_3	T_4	T_3	T_6
9; 13; 4	2_0; 6_1; 11; 2_1	7	10; 5	3; 1; 12; 8	6_0

Makespan	Q Trabajos T.
2410	1

Figura 10. Proceso de mutación
Fuente: Elaboración propia

- Si de los 2 cromosomas (μ) hijos mutó 1, pasan a reemplazar los 3 peores de la iteración 0
- Si los 2 cromosomas (μ) mutan, se generarán 4 cromosomas (μ) que reemplazarán a los 4 peores
- Este caso se repetirá en cada iteración, y se escogerá dependiendo de la medición intercalada de cada una.

10. El proceso de cruce y mutación deberá realizarse para cada conjunto de cromosomas (μ) pertenecientes a cada una de las subpoblaciones (β) generadas.
11. Una vez integrados los cromosomas (μ) (hijos) en cada subpoblación (β), se deberá llevar a cabo la iteración α+1, en donde la medición del *fitness* será intercalada (diferente a la iteración α).
12. La cantidad de iteraciones (α) deberá ser parametrizable.
13. Para cada cromosoma (μ), realizar el cálculo de los *Key Performance Indicators* (KPIs) correspondientes a *makespan* y trabajos tardíos normalizándolos con la misma escala de medida (%) y promediarlos (Figura 11).

    Indicador *makespan*: ($C_{max\mu}$ (μ) / Min *100
    Indicador trabajos tardíos:
    (Min $\Sigma U(\gamma)_\mu$ /Max $\Sigma U(\gamma)_\mu$)*100
14. Seleccionar el menor valor del KPI total como mejor resultado (Figura 12).

Pseudocódigo de la metodología propuesta
Obsérvense las Figuras 13 y 14.

Análisis experimental

Se realizó la prueba del modelo propuesto utilizando datos históricos de la empresa objeto de estudio con el fin de comparar los resultados reales de *makespan* y la cantidad de trabajos tardíos *vs.* resultados arrojados por el modelo.

El algoritmo diseñado ha sido desarrollado e implementado para el uso de la empresa a través del lenguaje de programación Python versión 3.8.5 y las pruebas fueron ejecutadas en un equipo portátil con procesador Intel Core i5 g8 y capacidad de memoria RAM de 8 GB.

Para el funcionamiento del modelo es necesario el ingreso de los parámetros iniciales indicados en Tablas 4 y 5 requeridos por el algoritmo genético.
El procesamiento de los dos tipos de artículo (Gancho y Felpa) se presenta en la Tabla 5, si existe 1 el producto puede fabricarse en esa máquina, si se visualiza 0 el trabajo no debe procesarse en esa máquina.

Identificadores		Cromosomas						FOB		KPI´S		
ID subpoblac	ID Cromosoma	T_1	T_2	T_3	T_4	T_5	T_6	Makespan	Q Trabajos T.	Makespan	Q Trabajos T.	KPI Norm.
1	1	1; 6_1; 8; 9; 2_1	12; 2_0; 10	4; 13; 11; 3	6_0	7	5	1829	2	104%	151%	128%
1	2	1; 2_1; 8 ; 6_1; 9	12; 2_0; 10	11; 3; 4; 13	6_0	5	7	2138	0	121%	100%	111%
1	3	2_1; 6_1; 8; 9; 1	12; 2_0; 10	13; 4; 3; 11	5	6_0	7	3053	1	173%	126%	150%
2	1	13; 4; 9	2_0; 6_1; 11; 2_1	1; 12; 8; 3	10; 5	7	6_0	1761	3	100%	177%	139%
2	2	9; 13; 4	3; 1; 12; 8	2_1; 11; 2_0; 6	10; 5	6_0	7	2579	1	146%	126%	136%
2	3	13; 9; 4	6_1; 2_0; 11; 2_1	12; 8; 3; 1	5; 10	7	6_0	1830	0	104%	100%	102%
2	4	13; 9; 4	11; 6_1; 2_0;2_1	3; 1; 12; 8	6_0	7	5; 10	1920	2	109%	151%	130%
3	1	10; 2_1; 1; 6_1	12; 3; 4; 11	13; 6_0; 9	8; 2_0	5	7	1868	0	106%	100%	103%
3	2	11; 3; 4; 12	6_1; 2_1; 1; 10;	9; 6_0; 13	2_0; 8	7	5	3120	0	177%	100%	139%
3	3	2_1; 6_1; 10; 1	6_0; 13; 9	3; 12; 4; 11	8; 2_0	7	5	3042	1	173%	126%	149%

Figura 11. Medición de KPI´s Fuente: Elaboración propia

Identificadores		Cromosomas						FOB		KPI´S		
ID subpoblac	ID Cromosoma	T_1	T_2	T_3	T_4	T_5	T_6	Makespan	Q Trabajos T.	Makespan	Q Trabajos T.	KPI Norm.
1	1	1; 6_1; 8; 9; 2_1	12; 2_0; 10	4; 13; 11; 3	6_0	7	5	1829	2	104%	151%	128%
1	2	1; 2_1; 8 ; 6_1; 9	12; 2_0; 10	11; 3; 4; 13	6_0	5	7	2138	0	121%	100%	111%
1	3	2_1; 6_1; 8; 9; 1	12; 2_0; 10	13; 4; 3; 11	5	6_0	7	3053	1	173%	126%	150%
2	1	13; 4; 9	2_0; 6_1; 11; 2_1	1; 12; 8; 3	10; 5	7	6_0	1761	3	100%	177%	139%
2	2	9; 13; 4	3; 1; 12; 8	2_1; 11; 2_0; 6	10; 5	6_0	7	2579	1	146%	126%	136%
2	3	13; 9; 4	6_1; 2_0; 11; 2_1	12; 8; 3; 1	5; 10	7	6_0	1830	0	104%	100%	102%
2	4	13; 9; 4	11; 6_1; 2_0;2_1	3; 1; 12; 8	6_0	7	5; 10	1920	2	109%	151%	130%
3	1	10; 2_1; 1; 6_1	12; 3; 4; 11	13; 6_0; 9	8; 2_0	5	7	1868	0	106%	100%	103%
3	2	11; 3; 4; 12	6_1; 2_1; 1; 10;	9; 6_0; 13	2_0; 8	7	5	3120	0	177%	100%	139%
3	3	2_1; 6_1; 10; 1	6_0; 13; 9	3; 12; 4; 11	8; 2_0	7	5	3042	1	173%	126%	149%

Figura 12. Selección de mejor cromosoma Fuente: Elaboración propia

https://doi.org/10.22201/fi.25940732e.2021.22.4.032


```
Estructura cromosoma: codificación numérica que indica el número de artículo (γ) a programar
Parametrizar α (iteraciones), β (subpoblaciones), μ (cromosomas) y Ш (% de probabilidad de
mutación)
Conjuntos: γ (artículos), t (turnos), m (máquinas), P_γ (agrupación de artículos por tipo de producto y
color)
Variables: T_γ (conjunto de turnos de artículos)
Crear conjuntos de artículo "ρ_γ" según color (η) y tipo (λ) de artículo (γ)
While ∑β > 0
    While ∑ρ_γ > 0
        Ordenar aleatoriamente los (γ) en cada (ρ_γ)
        While γ ϵ ρ_γ > 0
            Evaluar capacidad y prioridad de máquina (m)
            Evaluar capacidad de turno (t)
                IF (γ) < Capacidad de máquina (m) Then
                    IF (γ) < Capacidad del turno (t) Then
                    Asignar artículo (γ) a turno (t)
                    Else Evaluar turno (t + 1)
                Else Evaluar máquina (m+1)
            γ ϵ ρ_γ -1
        ρ_γ − 1
        Agrupar (γ) asignado a cada (t) y conformar el conjunto (T_γ)

    While ∑μ > 0
        Generar (μ) mediante orden aleatorio de (γ) correspondientes a cada conjunto (T_γ)
        μ - 1
    β - 1
```

Figura 13. Pseudocódigo de asignación de máquinas para proceso tinturado
Fuente: Elaboración propia

```
Calcular fitness de cada cromosoma:
Calcular C_max (makespan): Tiempo máximo de terminación de todos los (γ) de cada (μ)
Calcular trabajos tardíos U(γ): Cantidad de artículos (γ) con due date (F_γ) < tiempo de terminación
de los artículos Q_γ de cada (μ)
For α=1 to α
    For β =1 to β
        Seleccionar de los padres con mejor fitness
            If α= 2n+1
                Then: Seleccionar (μ) padres con menor makespan
                Else: Seleccionar (μ) padres con menor cantidad de trabajos
            tardíos
            End If
        Operación de cruce uniforme
        Generar (μ) hijos
        Calcular random (0-100%)
            If random ≤ Ш
                Then: Mutar 2 posiciones aleatorias de (μ) hijo
                Else: Omitir proceso de mutación
            End If
        Calcular fitness de (μ) hijos y (μ) hijos mutados
        Comparar el fitness de los (μ) hijos y (μ) hijos mutados, con los (μ) de las
        subpoblaciones (β)
            If Fitness (μ) hijos y (μ) hijos mutados < fitness (μ) de la subpoblación (β)
                Then: Reemplazar (μ) de menor fitness
                Else: Omitir proceso de reemplazo
            End If
    End for
End for
Calcular KPI's de fitness para cada (μ) en cada (β)
Seleccionar (μ) con mejor KPI
```

Figura 14. Pseudocódigo del algoritmo genético Fuente: Elaboración propia

Tabla 4. Parámetros del sistema Fuente: Elaboración propia

Parámetro	Descripción
Cantidad de subpoblaciones (β)	Número de grupos de cromosomas construidos a partir de la asignación a máquinas de tintorería
Cantidad de cromosomas (μ)	Orden aleatorio de trabajos, basados en la asignación de turnos por máquina en cada subpoblación
% Mutación (ш)	Probabilidad de mutación de los cromosomas, lo cual aplica solamente para cromosomas de la misma subpoblación
Iteraciones AG	Cantidad de iteraciones del algoritmo genético, desde el proceso de cruce y mutación hasta el reemplazo de cromosomas y cálculo de *fitness*
Fecha inicio de producción	Fecha en la cual se inicia el proceso operativo de producción; base para el cálculo de tiempos de proceso

Tabla 5. Paso de cada tipo de artículo por máquinas
Fuente: Elaboración propia a partir de datos de la empresa

	Tipo de artículo	
Máquinas m	Felpa	Gancho
Tintorería-autoclave 1	1	1
Tintorería-autoclave 2	1	1
Tintorería-autoclave 3	1	1
Mageba	1	1
Cepillo breiten	1	0
Termofijadora 1	1	1
Termofijadora 2	0	1
Corte monofilamento	0	1
Corte ultrasonido	1	1
Enrollado	1	1
Empaque	1	1

Pruebas reales

El procedimiento y *software* desarrollado se aplicó a la producción real de la empresa objeto de estudio con 165 trabajos correspondientes a 10 programas de producción pertenecientes a los meses de enero, febrero y junio con fechas de entrega de 14 días.

La Figura 15 muestra los resultados obtenidos de las mejores soluciones, la media del *makespan*, la desviación estándar muestral (calculada a partir de la desviación entre las soluciones generadas en cada una de las iteraciones de los cromosomas, indicando la dispersión de las soluciones comparadas con la media dada como resultado de las funciones objetivo) y la desviación porcentual (calculada a partir de las soluciones de cada una de las iteraciones, captando la propagación de los resultados de las funciones objetivo respecto a la media calculada en cada uno de los escenarios de prueba de una forma porcentual) en las simulaciones ejecutadas con cada una de las combinaciones de parámetros para los γ=165 trabajos. Teniendo en cuenta que una desviación estándar baja indica que la mayoría de los datos de una muestra tienden a estar agrupados cerca de su media, mientras que una desviación estándar alta indica que los datos se extienden sobre un rango de valores más amplio, se analiza que las desviaciones altas presentadas como resultado generan en su mayoría los mejores *fitness*, es decir, que las mejores soluciones se encontraron realizando una exploración más amplia del espacio de búsqueda.

Los tiempos computacionales del algoritmo varían de acuerdo con las configuraciones y la cantidad de trabajos (en un rango de 1 a 50), para aquellas que manejan 10 iteraciones y 2 subpoblaciones el tiempo requerido por el algoritmo para encontrar una solución, oscila entre 19 seg y 2 min; al ejecutar los escenarios de 20 iteraciones y 10 subpoblaciones el tiempo aumenta y varía en un rango de 3 a 36 min dado que la búsqueda de soluciones se vuelve más rigurosa.

En la Figura 16 se visualizan los resultados de la mejor combinación de los parámetros α (iteraciones), β (sub poblaciones), μ (cromosomas) y ш (% de mutación), para cada programa de producción y el tipo de cromosoma que arrojó la mejor solución y se comparan con los datos reales proporcionados por la empresa, referentes a los datos de entrada del sistema que son generados en un archivo desde SAP Business One, que contiene información de las órdenes de venta (número de orden, código de producto, cantidad, tipo de producto, color, fecha de entrega prometida al cliente, fecha contabilización de la orden y un identificador único para cada producto); con base en la ejecución de las pruebas se evidencia una mejora promedio de 911 min (78 %) respecto al C_{max} real que tuvieron los 10 programas de producción y en ningún escenario ejecutado se presentan ($\Sigma U(\gamma)$). En esta tabla es posible evidenciar

https://doi.org/10.22201/fi.25940732e.2021.22.4.032

Fecha de prueba	Cantidad de trabajos	Iteraciones	Subpoblaciones	Cromosomas	% Mutación	Mejor ME pruebas	Media	Desvest	Desv%	TTpruebas
enero 12 2020	5	10	2	2	5	468	4.788	7,7	1,61%	0
		10	2	5	5	474	4.819	5,6	1,16%	0
		10	10	2	5	468	4.788	7,7	1,61%	0
		10	10	5	5	465	4.773	8,7	1,83%	0
		20	2	2	5	466	4.778	8,4	1,75%	0
		20	2	5	5	479	4.844	3,9	0,80%	0
		20	10	2	5	466	4.778	8,4	1,75%	0
		20	10	5	5	466	4.778	8,4	1,75%	0
enero 15 2020	10	10	2	2	5	3791	3.809	13	0,34%	0
		10	2	5	5	3758	3.792	24,6	0,65%	0
		10	10	2	5	3758	3.814	39,7	1,04%	0
		10	10	5	5	3758	4.051	207,6	5,12%	0
		20	2	2	5	3758	3.799	29,2	0,77%	0
		20	2	5	5	3758	3.823	46,6	1,22%	0
		20	10	2	5	3758	4.036	197	4,88%	0
		20	10	5	5	3758	4.071	221,6	5,44%	0
enero 22 2020	23	10	2	2	5	1617	1.814	139,9	7,71%	0
		10	2	5	5	1618	1.73	79,3	4,58%	0
		10	10	2	5	1597	1.804	146,7	8,13%	0
		10	10	5	5	1616	1.81	137,4	7,59%	0
		20	2	2	5	1622	1.803	128,2	7,11%	0
		20	2	5	5	1628	1818	134,6	7,40%	0
		20	10	2	5	1604	1.827	158	8,65%	0
		20	10	5	5	1617	1.826	148,1	8,11%	0
enero 27 2020	46	10	2	2	5	4456	4.46	3,2	0,07%	0
		10	2	5	5	4456	4729	193,5	4,09%	0
		10	10	2	5	4456	4.941	343,3	6,95%	0
		10	10	5	5	4456	5046	417,8	8,28%	0
		20	2	2	5	4456	4.742	202,4	4,27%	0
		20	2	5	5	4466	5.067	425	8,39%	0
		20	10	2	5	4456	5.203	528,4	10,16%	0
		20	10	5	5	4456	5.127	474,8	9,26%	0
febrero 03 2020	28	10	2	2	5	5869	5904	23,78	0,40%	0
		10	2	5	5	5999	6002	22,49	0,37%	1
		10	10	2	5	5835	6013	127,41	2,12%	0
		10	10	5	5	5873	6046	113,58	1,88%	0
		20	2	2	5	6058	6051	127,24	2,10%	1
		20	2	5	5	5856	6019	139,39	2,32%	0
		20	10	2	5	5854	6013	139,84	2,33%	0
		20	10	5	5	5853	6049	142,95	2,36%	0
febrero 07 2020	10	10	2	2	5	1170	1212	52,89	4,36%	0
		10	2	5	5	1165	1191	47,26	3,97%	0
		10	10	2	5	1170	1257	63,64	5,06%	0
		10	10	5	5	1165	1221	60,09	4,92%	0
		20	2	2	5	1165	1207	54,79	4,54%	0
		20	2	5	5	1165	1247	64,79	5,20%	0
		20	10	2	5	1165	1234	65,08	5,27%	0
		20	10	5	5	1165	1208	55,73	4,61%	0
febrero 14 2020	11	10	2	2	5	2651	2679	41,44	1,50%	0
		10	2	5	5	2646	2668	36,59	1,37%	0
		10	10	2	5	2646	2676	33,47	1,25%	0
		10	10	5	5	2646	2672	29,34	1,10%	0
		20	2	2	5	2651	2690	33,73	1,25%	0
		20	2	5	5	2651	2680	33,55	1,25%	0
		20	10	2	5	2646	2671	29,98	1,12%	0
		20	10	5	5	2646	2675	38,01	1,42%	0
febrero 17 2020	8	10	2	2	5	671	681	7,94	1,17%	0
		10	2	5	5	673	678	6,06	0,89%	0
		10	10	2	5	671	683	7,88	1,15%	0
		10	10	5	5	671	680	8,33	1,23%	0
		20	2	2	5	676	681	4,45	0,65%	0
		20	2	5	5	675	681	5,14	0,75%	0
		20	10	2	5	672	683	9,54	1,40%	0
		20	10	5	5	670	682	9,62	1,41%	0
febrero 24 2020	12	10	2	2	5	1928	1.99	44,2	2,22%	0
		10	2	5	5	2005	2.035	21,4	1,05%	0
		10	10	2	5	1948	2.155	146,7	6,81%	0
		10	10	5	5	1901	2.24	240,2	10,72%	0
		20	2	2	5	1969	2.163	137,5	6,35%	0
		20	2	5	5	1938	2060	86,4	4,19%	0
		20	10	2	5	1889	2.207	225,3	10,21%	0
		20	10	5	5	1944	2.157	151,1	7,00%	0
junio 04 2020	12	10	2	2	5	688	690	1,88	0,27%	0
		10	2	5	5	691	702	15,85	2,26%	0
		10	10	2	5	690	706	14,49	2,05%	0
		10	10	5	5	690	712	11,64	1,63%	0
		20	2	2	5	695	703	14,17	2,02%	0
		20	2	5	5	696	708	12,08	1,71%	0
		20	10	2	5	688	706	13,47	1,91%	0
		20	10	5	5	688	699	11,28	1,61%	0

Figura 15. Resultados de las simulaciones
Fuente: Elaboración propia

Figura 16. Comparación mejores soluciones vs. datos reales Fuente: Elaboración propia

Fecha de prueba	Cantidad de trabajos	Iteraciones	Subpoblaciones	Cromosomas	Mejor makespan pruebas	Makespan Real	Trabajos tardíos pruebas	Trabajos tardíos real	Tipo de cromosoma
enero 12 2020	5	10	10	5	465	523	0	0	Hijo
enero 15 2020	10	10	10	5	3758	5940	0	1	Hijo
enero 22 2020	23	10	10	2	1597	1789	0	0	Mutación
enero 27 2020	46	20	2	5	4466	6380	0	6	Hijo
febrero 03 2020	28	10	10	2	5835	5986	0	3	Hijo
febrero 07 2020	10	20	2	5	1165	1534	0	0	Hijo
febrero 14 2020	11	20	10	5	2646	6238	0	1	Hijo
febrero 17 2020	8	20	10	5	670	779	0	0	Hijo
febrero 24 2020	12	20	10	2	1889	2359	0	0	Padre
junio 04 2020	12	20	10	2	688	766	0	0	Hijo

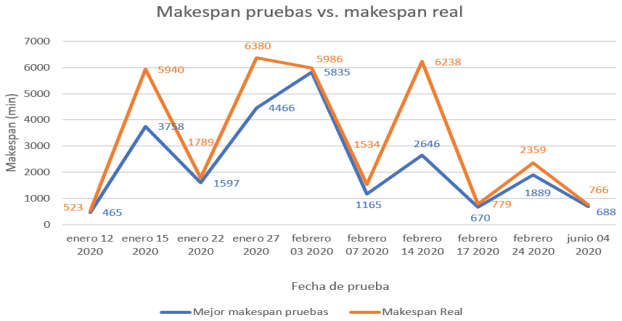

Figura 17. Comparación *makespan* pruebas vs. *makespan* real
Fuente: Elaboración propia

que las mejores soluciones para 80 % de los escenarios se dieron con una configuración de 10 subpoblaciones, lo que quiere decir que este parámetro es el que mayor influencia tiene sobre una buena generación de resultados, esto con base en los análisis de las desviaciones estándar presentadas (mayor exploración de resultados), adicional, los μ (cromosomas) que se seleccionaron por tener un mejor *fitness* fueron en su mayoría de tipo hijo, por el contrario, en 53 resultados de las 80 pruebas ejecutadas (66 %), los cromosomas con menor calificación fueron de tipo mutación.

Conclusiones

1. Los resultados arrojados por el sistema desarrollado demuestran que la nueva metodología de programación logra reducir el *makespan* y la cantidad de trabajos tardíos en la empresa (Figura 17); este aspecto toma gran importancia teniendo en cuenta que el reducir estos dos elementos no solo tiene impacto en la productividad de las empresas sino a nivel organizacional, ya que puede ser un factor importante de competitividad en la industria.
2. Una de las ventajas que ofrece el sistema radica en el tiempo que demora en generar una secuencia de programación comparado con un modelo de programación empírico.
3. Dada la complejidad del sistema productivo elegido objeto de estudio, el modelo que se debió desarrollar no estaba contemplado dentro de los modelos matemáticos estándares encontrados en las investigaciones que abarcan este tipo de problema.
4. En la literatura se encontraron pocos proyectos que abordan sistemas reales de manufactura con configuraciones de producción de tipo *flow shop* híbrido flexible en la industria textil, además, se evidenció que se enfocan en la solución de solo una función objetivo. López (2013) en su investigación aborda un sistema de producción tipo *flow shop* híbrido flexible en una empresa del sector textil y establece una metodología de *scheduling* con el objetivo de reducir el makespan, aunque el trabajo también está enfocado a la industria textil, los procesos de producción estudiados difieren de los procesos que se analizan en el presente artículo, lo que conlleva a que puntos como las definiciones propias del algoritmo genético (estructura del cromosoma, generación de la población inicial, técnicas de cruce y mutación) también sean distintos en las dos investigaciones; es por eso que este trabajo aporta un valor a la investigación de este tipo de problemas a través de la solución de dos objetivos (*makespan* y trabajos tardíos), el uso de una regla de despacho para la generación de la población inicial con el fin de que esta estuviera adaptada al problema para que el algoritmo tuviera un mejor rendimiento, y el desarrollo de un sistema con la capacidad de procesar casos en un ambiente real de manufactura (textiles técnicos).

5. Para este trabajo se eligió como base a los algoritmos genéticos dado que es una metaheurística que no se ve altamente afectada por los óptimos locales, es decir, que explora varios espacios de búsqueda y no se quedan con la primera solución encontrada. Otra ventaja que ofrecen los algoritmos genéticos frente a otras heurísticas o metaheurísticas es que tiene la capacidad de trabajar con varios parámetros de forma simultánea y además trabaja realizando cambios aleatorios en las posibles soluciones validando una mejora o no en la función de aptitud, adicional, es capaz de conseguir soluciones de alta calidad de forma eficiente en problemas de gran tamaño y complejidad.
6. Es posible mejorar los acuerdos de nivel de servicio (ANS) teniendo en cuenta que el *makespan* y la cantidad de trabajos tardíos disminuyeron en los resultados de las pruebas, con esto se pueden lograr tiempos de entrega menores a 14 días dependiendo de la cantidad de trabajos a procesar y dando cumplimiento al tiempo de entrega acordado, lo cual se ve reflejado a través de los resultados presentados en la Figura 16 (comparación de mejores soluciones vs datos reales).
7. Según el análisis de correlación de los datos de prueba suministrados por la empresa y el análisis realizado a los resultados de las pruebas ejecutadas en el sistema desarrollado, los cuales son formulados como *makespan*/cantidad de metros del artículo, se puede concluir que con la metodología propuesta el coeficiente de correlación presenta un resultado de R^2=0.97, al comparar con el resultado real suministrado por la empresa, correspondiente R^2 =0.72, se evidencia una diferencia de 0.24, lo cual indica una menor desviación entre la correlación lineal de las variables cantidad de metros vs. *makespan*, dando como conclusión una mejora en la relación de cantidad de metros procesados y el respectivo *makespan*, aumentando la eficiencia del proceso de producción. Para este caso no se realizó un análisis de correlación de número de trabajos tardíos dado que de los 80 escenarios de prueba solamente dos presentaron cromosomas con un trabajo tardío.

Recomendaciones

1. En futuras investigaciones es posible lograr un mejor rendimiento del algoritmo en cuanto a secuencias de programación si se tienen en cuenta restricciones de la industria textil como tiempos de preparación dependientes de la secuencia, presencia de máquinas paralelas no relacionadas en las diferentes etapas de procesamiento, lotes de transferencia variables y factores que puedan afectar el cumplimiento y resultados de la programación, entre los cuales se pueden presentar: modificaciones de pedidos, disponibilidad de recursos, avería de máquinas, restricciones de precedencia, entre otros.
2. Otro aspecto que se puede trabajar es la utilización de una metaheurística en la fase 1 de asignación de trabajos a las máquinas de tintorería, ya que para el desarrollo de la metodología diseñada se utilizó el fundamento de la heurística LPT, el usar una metaheurística puede generar la exploración de espacios de búsqueda no examinados por el modelo propuesto.

Referencias

Acuña, Domínguez & Toro. (2012). Una comparación entre métodos estadísticos clásicos y técnicas metaheurística en el modelamiento estadístico. *Scientia et Technica*, 17(50), 68-77.

Aguilar, R. (2016). *Algoritmo genético aplicado a sistemas de manufactura flexible.* (Trabajo de grado). Universidad Nacional Autónoma de México, Ciudad de México.

Allahverdi, A. & Al-Anzi, F. (2008). The two-stage assembly flowshop scheduling problem with bicirteria of makespan and mean completion time. *The International Journal of Advanced Manufacturing Technology*, (37), 166-177. https://doi:10.1007/s00170-007-0950-y

Arranz, P. & Parra, T. (n.d). *Algoritmos genéticos.* (Trabajo de grado). Universidad Carlos III, Madrid, España.

Chen, Pan & Lin. (2008). A hybrid genetic algorithm for the reentrant flow-shop scheduling problem. *Expert Systems with Applications*, 34(1), 570-577.

Chicano-García, J. (2007). *Metaheurísticas e ingeniería del software.* (Tesis doctoral). Universidad de Málaga, Málaga, España.

Corral, A. (2017). *Desarrollo y evaluación del algoritmos genéticos multiobjetivo. Aplicación al problema del viajante.* (Trabajo de grado). Escuela Técnica Superior de Ingeniería Informática. Universidad Politécnica de Valencia, España.

Czajkowski, M. & Kretowski, M. (2019). A multi-objective evolutionary approach to Pareto-optimal model trees. *Soft Computing*, (23), 1423-1437.

Esquivel, S., Ferrero, S., Gallard, R., Salto, C., Alfonso, H. & Schuts, M. (2002). Enhanced evolutionary algorithms for single and multiobjective optimization in the job shop scheduling problem. *Knowledge-Based Systems*, 13-25.

Gestal, M. (2013). *Introducción a los algoritmos genéticos*. Universidade da Coruña, Coruña, España.

Gestal, Rivero, Rabuñal, Dorado & Pazos. (2010). *Introducción a los algoritmos genéticos y la programación genética.* Universidade da Coruña, Coruña, España.

Golberg, D. (1989). *Genetic algorithms in search, optimization, and machine learning.* Estados Unidos: Addison-Wesley publishing company. Recuperado de https://www.semanticscholar.org/paper

Gomez, Romano & Cruz. (2012). An agent-based genetic algorithm for hybrid flowshops with sequence dependent setup times to minimise makespan. *Expert Systems with Applications*, 39(9), 8095-8107.

Jarboui, B., Eddaly, M. & Siarry, P. (2011). A hybrid genetic algorithm for solving no-wait flowshopscheduling problems. *The International Journal of Advanced Manufacturing Technology*, 54, 1129-1143. https://doi.org/10.1007/s00170-010-3009-4

Karmakar & Manhaty. (n.d). *Minimizing makespan for a flexible flow shop scheduling problem in a paint company*. Indian Institute of Technology, Department of Industrial Engineering and Management, Kharagpur, India.

López, J. & Arango J. A. (2015). 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. *Entramado*, 2(1), 250-262.

López-Vargas, J. C. (2013). 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. Universidad Nacional de Colombia. Manizales, Colombia.

Oguz, C. & Ercan, M. (2005). A genetic algorithm for hybrid flowshop scheduling with multiprocessor tasks. *Journal of Scheduling*, (8), 323-351.

Ren, Diao & Luo. (2012). Optimal results and numerical simulations for flow shop scheduling problems. *Journal of Applied Mathematics*, 2012, 1-9. https://10.1155/2012/395947

Revetti, M., Riveros, C., Mendes, A., Resende, M. & Pardalos, P. (2012). Parallel hybrid heuristics for the permutation flow shop problem. *Annals of Operations Research*, 1-16.

Ribas, Leisten & Framiñan. (2010). Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers & Operations Research*, 37, 1439-1454.

Salazar & Sarzuri. (2015). Algoritmo genético mejorado para la minimización de la tardanza total en un flowshop flexible con tiempos de preparación dependientes de la secuencia. *Revista Chilena de Ingeniería*, 23(1), 118-127.

Schaffer, J. (1985). Multiple objetive optimization with vector evaluated denetic algroithms. Proceedings of an international conference on genetic algorithms and their applications, 92-101.

Solari & Ocampo. (2006). Aplicación de algoritmos genéticos en un sistema multiagente de planificación en una industria manufacturera. XXXII Conferencia Latinoamericana de Informática (CLEI).

Toro-Ocampo, E., Restrepo-Grisales, Y. & Granada-Echeverri, M. (2006). Algoritmo genético modificado aplicado al problema de secuenciamiento de tareas en sistemas de producción lineal-fow shop. *Scientia Et Technica*, XII(30), 285-290.

Zandieh & Karimi. (2011). An adaptive multi-population genetic algorithm to solve the multi-objective group scheduling problem in hybrid flexible flowshop with sequence-dependent setup times. *Jorunal of Intelligent Manufacturing*, 22(6), 979-989.

Zhang, M. (2019). Flexible flow shop scheduling problem with setup times and blocking constraint via genetic algorithm and simulation. IOP Conf. Series: Materials Science and Engineering, 637.

Zobolas, Tarantilis & Ioannou. (2009). Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm. *Computers & Operations Research*, 36, 1249-1267.

Cómo citar:

Espitia-Mendez, J. A. & Mendoza-Rojas, G. L. (2021). Metodología basada en un algoritmo genético para programar la producción de una empresa del sector textil. *Ingeniería Investigación y Tecnología*, 22 (04), 1-16. https://doi.org/10.22201/fi.25940732e.2021.22.4.032