Facultad de Ingeniería

INGENIERÍA INDUSTRIAL

Trabajo de Grado – Segundo Semestre 2021

Trabajo de grado en modalidad de aplicación

Distributed permutation flow shop estocástico para minimizar la tardanza esperada

Juan Camilo Orjuela 1[a,c], Juan Daniel Sánchez 2[a,c] , Natalia Sánchez 3[a,c], Carlos Eduardo Walteros 4[a,c]

Eliana María González Neira [b,c]

[a] *Estudiante de Ingeniería Industrial*

[b] *Profesor, Director del Trabajo de Grado, Departamento de Ingeniería Industrial*

[c] *Pontificia Universidad Javeriana, Bogotá, Colombia*

Resumen

En los últimos años la programación de la producción -Scheduling- ha tomado fuerza y relevancia dentro de las organizaciones por su impacto en el logro de los objetivos de la organización. En un mundo cada vez más globalizado, la producción y la gestión de los recursos pueden irse extendiendo desde una sola fábrica, hasta una red compleja de producción distribuida. Este artículo estudia el problema del Distributed Permutation Flowshop (DPFSP) estocástico que minimiza el valor esperado de la tardanza. Para la solución del problema, se propone una simheurística basada en un algoritmo genético hibridizado con simulación de Monte Carlo. A su vez, para evaluar el desempeño de la simheurística propuesta, se utilizaron 92 instancias extraídas de la literatura, en las cuales se pudo mejorar en un 100% la solución en las instancias cortas, en cuanto a las instancias largas, en el 88% de las corridas realizadas se encontró una mejor solución con la simheuristica, sin embargo se destaca que en el 100% de las instancias se encontraron como mínimo 6 soluciones donde la FO disminuye con la simheuristica, habiendo realizado 9 corridas por instancia. Al comparar la simheuristica implementada, junto con el modelo matemático y la regla de despacho ATC, en los resultados estadísticos obtenidos, se pudo identificar que la tardanza esperada se pudo mejorar hasta en un 20% comparada con los otros métodos ya mencionados.

1. Planteamiento del problema

La programación de la producción (scheduling, en inglés) consiste en la asignación de los recursos a las tareas en el tiempo para lograr uno o varios objetivos (Pinedo, 2016). Dentro de los problemas de programación de la producción más estudiados se encuentra el Flowshop (FSP, por sus siglas en inglés) que se consiste en la programación de un conjunto n de trabajos $j \in \{1,\ldots,n\}$ que tienen que ser procesados en un grupo de máquinas (flujo $i \in \{1, 2,\ldots,k\}$) en serie. Es decir, primero los trabajos son procesados en la máquina 1, luego en la 2 y así sucesivamente. En este sistema de producción cada trabajo puede procesarse en una sola máquina a la vez. Así mismo, una sola maquina puede procesar cada una de las tareas individuales. Se asume que los trabajos son realizados desde el comienzo hasta el final del proceso, sin interrupciones, una vez que se asignan a una máquina. También, se asume que las máquinas

están disponibles durante todo el horizonte de la programación y no sufren averías ni daños. Para finalizar, se asume que los trabajos están disponibles para su procesamiento desde el comienzo del período de programación y que el período de tiempo tiene duración suficiente para completar la tarea en cada una de las máquinas.

Debido a la globalización, las empresas se vieron en la necesidad de mejorar la capacidad de respuesta a los cambios y a ser más competitivas internacionalmente, por lo cual muchas empresas pasaron de la producción tradicional en una sola planta a la producción en varias plantas (Chan & Chung, 2013). Por esta razón, una variante del FSP que se comenzó a estudiar en el año 2010 y que ha tomado alta importancia es el Distributed Permutation Flowshop Problem (DPFSP). En el DPFSP se tienen n trabajos para ser procesados en una de las f fábricas idénticas, en cada una de las cuales hay m máquinas en serie, y todas las fábricas son capaces de procesar todos los trabajos de manera independiente. Cada trabajo debe ser procesado en una sola fábrica. La secuencia de los trabajos en todas las máquinas de una fábrica debe ser la misma (permutation) (J. P. Huang et al., 2020).

El DPFSP es un problema que empezó a ser estudiado en el año 2010, por (Naderi & Ruiz, 2010). Desde entonces, los estudios en este problema han venido creciendo, aunque comparativamente con el FS clásico hay menor cantidad de investigaciones. De las investigaciones encontradas se puede observar que la mayoría consideran el problema determinístico uni-objetivo, donde el makespan es el objetivo principal (Jing et al., 2021). No obstante, el análisis de otras funciones objetivo, relacionadas con las fechas de entrega de los trabajos es muy útil para mejorar el nivel de servicio al cliente (Fu et al., 2019).

En los entornos de producción reales se pueden encontrar variables sujetas a aleatoriedad como los tiempos de procesamiento, tiempos de alistamiento, daños de máquinas, tiempo de reparación de máquinas, llegada de los trabajos entre otros. Este es un aspecto que no ha sido muy estudiado en el DPFSP. De hecho, de los 70 artículos encontrados en la literatura, desde el año 2010, solo tres cuentan con parámetros sujetos a aleatoriedad que son el de (Fu et al., 2019), (Jing et al., 2021) y el de (Rifai et al., 2016a). Particularmente, (Fu et al., 2019) consideraron los tiempos de procesamiento y alistamiento como parámetros estocásticos que seguían una distribución de probabilidad uniforme en un intervalo entre 1 y 99. Para resolver el problema los autores propusieron 2 métodos: MOALNS con simulación de Monte Carlo y NSGA-2 con simulación de Monte Carlo para compararlos posteriormente (Rifai et al., 2016a).

Uno de los métodos de solución recientes para abordar problemas combinatorios estocásticos son las simheurísticas (Juan et al., 2015). Éstas combinan metaheurísticas con simulación en una sola herramienta para proporcionar soluciones rápidas a los problemas estocásticos. Ejemplos de implementación de simheurísticas se encuentran en diversos problemas de ruteo de vehículos (Guimarans et al., 2018; Latorre-Biel et al., 2021), localización de plantas (de Armas et al., 2017), y programación de la producción (González-Neira et al., 2021; González-Neira & Montoya-Torres, 2019; Hatami et al., 2018; Villarinho et al., 2021), mostrando buenos resultados.

Un aspecto importante para la aplicación de las simheurísticas es tener la distribución de probabilidad de los parámetros bajo incertidumbre. Para esta propuesta se utilizará la distribución de probabilidad Log-Normal ya que representa una elección más natural que la distribución normal para modelar los tiempos de procesamiento que son no negativos y que ha sido probada en varios artículos (Baker & Altheimer, 2012; Framinan & Perez-Gonzalez, 2015; Juan et al., 2014).

Considerando los elementos anteriores, se plantea la siguiente pregunta de investigación: ¿cómo diseñar una simheurística para resolver un DPFSP estocástico que minimice el valor esperado de la tardanza, donde los tiempos de procesamiento de los trabajos son estocásticos?

2. Antecedentes

Se realizó una revisión sistemática de la literatura sobre el Distibuted Permutation Flowshop Problem (DPFSP) en las bases de datos de Scopus e ISI Web of Science. La cadena de búsqueda fue (“distributed permutation flowshop” OR “distributed permutation flow shop”) realizada en los campos de título, resumen y/o palabras clave,

desde el año 2001 hasta la actualidad. Como criterios de exclusión se consideraron la eliminación de artículos que no estuvieran en inglés y/o aquellos que cuyo objetivo específico no fuera realizar la programación de la producción. Teniendo en cuenta estos aspectos se encontraron 70 artículos relacionados con este tema, de los cuales se seleccionaron 48 debido a que no presentan variaciones significativas.

Desde que se comenzó a estudiar el DPFSP la función objetivo más estudiada ha sido el makespan. Sin embargo, recientemente la minimización de la tardanza o los problemas justo a tiempo (que minimizan tardanza y adelanto) han despertado gran interés por parte de los investigadores, debido a la importancia del cumplimiento en la entrega como una medida de nivel de servicio. Es de resaltar que el 79% de los trabajos encontrados en la literatura en DPFSP tiene en cuenta funciones objetivo relacionados con las fechas de entrega.

En cuanto a los métodos de solución, se encontró en repetidas ocasiones el algoritmo greedy y algunas variantes de este como en (Chen et al., 2021). También se ve en algunos estudios la implementación algoritmos de búsqueda local (Fernandez-Viagas & Framinan, 2014) y (Gao et al., 2012). En otros trabajos se encontraron métodos adaptativos como algoritmos genéticos (J. Huang et al., 2019), colonia de abejas (Pan et al., 2018)y finalmente heurísticas propias de cada autor (Companys & Ribas, 2016; S. Y. Wang et al., 2013) .

Con relación al DPFSP estocástico o bajo incertidumbre, sólo se encontraron 3 artículos que lo han estudiado. En primera medida, en la investigación de (Fu et al., 2019) resolvieron el DPFSP estocástico para minimizar el valor esperado del makespan y el valor esperado del consumo de energía sujeto a una probabilidad de que la tardanza esperada no supere determinado valor. Los parámetros estocásticos fueron los tiempos de procesamiento los cuales fueron modelados a través de distribuciones de probabilidad. Como método de solución propusieron un brain storm optimization algorithm que incorporaba la simulación y lo compararon contra un NSGAII y un MOEA mostrando mejores resultados que esas dos simheurísticas. (Fu et al., 2019). (K. Wang et al., 2016) estudiaron el DPFSP con minimización de makespan y daños en las máquinas como parámetro bajo incertidumbre. Los autores propusieron un hybrid estimation of distribution algorithm basado en lógica difusa para resolver el problema y compararon su método con un algoritmo genético y un estimation of distribution algorithm clásico demostrando mejores resultados. Por último, (Jing et al., 2021) resolvieron un DPFSP robusto que minimiza el makespan modelando los tiempos de procesamiento a través de intervalos. Como métodos de solución propusieron un Iterated Greedy Algorithm y un Iterated Local Search, los cuales fueron evaluados en comparación con otros cinco algoritmos de la literatura mostrando mejor desempeño. Todo se reúne en un procedimiento de destrucción con tamaños dinámicos que facilite la exploración del IGA. En ambos casos se utilizan múltiples métodos de búsqueda local para producir soluciones factibles, prometedoras y realistas mediante la explotación de diferentes áreas de búsqueda.

En la Tabla 1 se presentan los artículos de DPFSP encontrados en la literatura con sus principales características:

ART.	*ENLACE*	*CARACTERÍSTICAS ESPECIALES*				*OBJETIVO*		*OBJETIVOS*					*MÉTODO DE SOLUCIÓN*			Distribuciones de probabilidad	números difusos	Intervalos	Método de Solución
		Set-up Time	Release Time	Machine Elegibility	Limited Buffers	Uni	Multi	Makespan	Flowtime	Tardiness	Earliness	Otros	Modelo Matemático	Heurística	Meta Heurística				
1	(Fathollahi-Fard et al., 2021)	X					X	X				Salaries		X					Taguchi method
2	(Chen et al., 2021)	Blocking				X		X						X					Iterated greedy (IG)
3	(Y. Z. Li, Pan, Gao, et al., 2021)						X		X			Total energy consumption		X					
4	(Rossi & Nagano, 2021)	X				X		X						X					greedy algorithm
5	(Jing et al., 2021)	X				X		X							X			X	The objective function is the robust makespan which contemplates expected makespan and standard deviation of makespan
6	(Shao et al., 2021)	Blocking				X		X						X					efficient iterated greedy (EIG)
7	(Y. Z. Li, Pan, Li, et al., 2021)	X				X			X				X	X					Adaptive Iterated Greedy (AIG) / Referenced Local Search (RLS)
8	(Mao et al., 2021)			X		X		X						X					Multi-start iterated greedy (MSIG)
9	(Lu et al., 2021)			X			X	X				Negative social impact (NSI), and total energy consumption (TEC)	X	X					Taguchi method
10	(J. P. Huang et al., 2021)	x				X		X							X				Bee colony (DABC) algorithm.
11	(J. P. Huang et al., 2020)	Blocking				X		X					X						iterated greedy algorithm with a restart scheme (IGR)

ART.	*ENLACE*	*CARACTERÍSTICAS ESPECIALES*				*OBJETIVO*		*OBJETIVOS*					*MÉTODO DE SOLUCIÓN*			Distribuciones de probabilidad	números difusos	Intervalos	Método de Solución
		Set-up Time	Release Time	Machine Elegibility	Limited Buffers	Uni	Multi	Makespan	Flowtime	Tardiness	Earliness	Otros	Modelo Matemático	Heurística	Meta Heurística				
12	(Hamzadayı, 2020)	X				X		X					X						Hibridación de la heurística NEH y el modelo matemático
13	(G. Wang et al., 2020)	X	X				X	X						X					Algoritmo multi-whale sharm
14	(Mao et al., 2020b)					X			X					X					Iterated greedy (IG).
15	(Chen et al., 2020)					X		X						X					Heurística NEH
16	(Mao et al., 2020a)		X			X		X						X					Algoritmo artficial bee colony
17	(J. J. Wang & Wang, 2020)				X		X	X						X					Heurística NEH
18	(Meng et al., 2019)					X		X						X					Iterated greedy (IG). Artificial bee colony
19	(J. Huang et al., n.d.)					X			X				X						Hybrid genetic algorithm
20	(G. Wang et al., 2019)				X		X	X						X					Algoritmo multi-whale sharm
21	(W. Li et al., 2019)		X			X		X						X					Iterated greedy (IG)
22	(Fu et al., 2019)				X		X	X		X				X		X			BI-objective, expected makespan and expected energy consumption. Simulación de Montecarlo
23	(Q. K. Pan et al., 2019)	X				X			X					X	X				Algoritmo artficial bee colony, heurística NEH

ART.	ENLACE	CARACTERÍSTICAS ESPECIALES				OBJETIVO		OBJETIVOS					MÉTODO DE SOLUCIÓN			Distribuciones de probabilidad	números difusos	Intervalos	Método de Solución
		Set-up Time	Release Time	Machine Elegibility	Limited Buffers	Uni	Multi	Makespan	Flowtime	Tardiness	Earliness	Otros	Modelo Matemático	Heurística	Meta Heurística				
24	(Ruiz et al., 2019)					X		X					X						Iterated greedy (IG)
25	(J. Q. Pan et al., 2018)	X				X			X					X					Algoritmo DABC
26	(Fernandez-Viagas et al., 2018)	X				X			X					X					Límite inferior del tiempo total de finalización
27	(Bargaoui, Belkahla Driss, et al., 2017)				X	X		X							X				METODO TAGUCHIY
28	(Ying et al., 2017)					X		X						X					Algoritmo codicioso de referencia iterado
29	(Duan et al., 2017)	X				X						tiempo máximo de finalización de cada fábrica		X					EDAPrMA
30	(Komaki & Malakooti, 2017)					X		X							X				matriz de tiempo y distancia de finalización
31	(Ribas et al., 2017)	X				X		X						X					algoritmo codicioso iterado
32	(Deng & Wang, 2017)	X					x	x		X			X						método de Taguchi
33	(Bargaoui, Driss, et al., 2017)				X	X		X							X				CROMAS
34	(Z. Li et al., 2016)	X				X			X				X						Método basado en NEH
35	(J. Wang et al., 2016)	X				X		X					X						(DOE) de Taguchi
36	(Duan et al., 2016)					X						tiempo máximo de finalización de cada fábrica.		X					EDA

ART.	ENLACE	CARACTERÍSTICAS ESPECIALES				OBJETIVO		OBJETIVOS					MÉTODO DE SOLUCIÓN			Distribuciones de probabilidad	Números difusos	Intervalos	Método de Solución
		Set-up Time	Release Time	Machine Elegibility	Limited Buffers	Uni	Multi	Makespan	Flowtime	Tardiness	Earliness	Otros	Modelo Matemático	Heurística	Meta Heurística				
37	(Rifai et al., 2016)			x	X		X	X		X		Costo total de producción	X						algoritmo de MOALNS
38	(Companys & Ribas, 2016)	Blocking					x	x	x					x					HPF2, RC1_1, RC1_m, RC2…….
39	(K. Wang et al., 2017)	x					x	x	x						x		X		FL-HEDA hibridado con un GA
40	(Fernandez-Viagas & Framinan, 2014)					x		x						x					NEH2 y busqueda local
41	(Naderi & Ruiz, 2014)					x		x							x				Scatter search con busqueda local
42	(Xu et al., 2014)					x		x						x					HIA
43	(Deng et al., 2014)					x		x							x				comptetitive memetic algorithm
44	(S. Y. Wang et al., 2013)					x		x						x					EDA
45	(Lin et al., 2013)					x		x							x				Grredy
46	(Gao et al., 2012					x		x							x				Tabu
47	(Liu & Gao, 2010)					x		x							x				EM
48	(Naderi & Ruiz, 2010)					x		x					x	x					MILP

Tabla 1: Tabla de Antecedentes

Autoría propia

Considerando que la mayoría de los trabajos en DPFSP han sido determinísticos para minimizar el makespan, este trabajo pretende resolver el DPFSP con tiempos estocásticos de procesamiento para la minimización de la tardanza esperada, a través de una simheurística.

3. Objetivos

Objetivo General:

Diseñar una simheurística para resolver un DPFSP estocástico, que minimice el valor esperado de la tardanza, donde los tiempos de procesamiento de los trabajos son estocásticos.

Objetivo Específicos:

- Diseñar el modelo matemático para resolver el DPFSP determinístico que minimice la tardanza total.
- Diseñar una simheurística para resolver el DPFSP estocástico para minimizar la tardanza esperada.
- Evaluar el desempeño de la simheurística en comparación con la solución simulada del modelo matemático en instancias pequeñas.
- Evaluar el desempeño de la simheurística en comparación con la solución simulada de la regla de despacho Apparent Tardiness Cost (ATC).

4. Modelo Matemático del DPFSP determinístico que minimiza la tardanza total

El siguiente modelo matemático se basa en el modelo planteado por (Ruiz et al., 2019). Las modificaciones realizadas al modelo corresponden a la inclusión de la tardanza de los trabajos como variable de decisión, el cambio en la función objetivo por la tardanza total, y algunas ligeras modificaciones en dos restricciones para facilitar la interpretación del resultado del modelo en los aspectos relacionados con la secuencia de trabajos en cada fábrica.

Conjuntos:

I: Máquinas

J: Trabajos

F: Fábricas

Variables de decisión:

$X_{j,f,k}$: Variable binaria que toma el valor de 1 si el trabajo $\forall j \in J$ es procesado en la fábrica $\forall f \in F$ finmediatamente después del trabajo k. 0 de lo contrario, donde $j \neq k$.

$Y_{j,f}$:Variable binaria que toma el valor de 1 si el trabajo $\forall j \in J$ es procesado en la fábrica $\forall f \in$. 0 de lo contrario.

$C_{j,i}$: Variable continua para el tiempo de finalización del trabajo $\forall j \in J$ en la máquina $\forall i \in I$.

T_j: Tardanza del trabajo $\forall j \in J$.

Parámetros:

Parámetros	Descripción
$P_{j,i}$	Tiempo de procesamiento del trabajo $j \in J$ en la máquina $i \in I$
D_j	*Due date* del trabajo $j \in J$
M	Número positivo suficientemente grande

Figure 1: Tabla de Parámetros
Autoría propia

Función Objetivo:

$$min\ z = \sum_{j \in J} T_j \quad (1)$$

s.a.:

Restricciones:

$$\sum_{k \in J \cup 0, k \neq j} X_{kjf} = Y_{kf} \quad \forall j \in J, \forall f \in F \quad (2)$$

$$\sum_{f \in F} Y_{jf} = 1 \quad \forall j \in J \quad (3)$$

$$\sum_{k \in J \cup 0, k \neq j} X_{kjf} = Y_{jf} \quad \forall j \in J, \forall f \in F \quad (4)$$

$$\sum_{j \in J} X_{0,j,f} = 1 \quad \forall f \in F \quad (5)$$

$$\sum_{k \in J} X_{k,0,f} = 1 \quad \forall f \in F \quad (6)$$

$$C_{ji} \geq C_{j(i-1)} + P_{ji} \quad \forall j \in J, \forall i \in I \quad (7)$$

$$C_{ji} \geq C_{ki} + P_{ji} + \left(X_{kif} - 1\right) * M \quad \forall k \in J, \forall j \in J, \forall i \in I, j \neq k \quad (8)$$

$$C_{ji} - D_j \leq T_j \quad \forall j \in J, \forall i \in I \quad (9)$$

$$C_{j,i} \geq 0 \quad \forall j \in J, \forall i \in I \quad (10)$$

$$T_j \geq 0 \quad \forall j \in J \quad (11)$$

$$X_{k,j,f} \in \{0,1\} \quad \forall k \in J, \forall j \in J, \forall i \in I, j \neq k \quad (12)$$

$$Y_{j,f} \in \{0,1\} \quad \forall j \in J, \forall f \in F \quad (13)$$

En las ecuaciones anteriores empezamos por la función objetivo (1) que tiene como propósito la minimización de la tardanza total. El conjunto de restricciones (2) asegura que cada trabajo tiene solamente un trabajo sucesor inmediato de otro trabajo en la fábrica en la que es procesado. El conjunto de restricciones (3) asegura que cada trabajo debe asignarse exactamente a una fábrica, el conjunto de restricciones (4) asegura que cada trabajo tiene solamente un trabajo antecesor inmediato de otro trabajo en la fábrica en la que es procesado. El conjunto de restricciones (5) asegura que en cada fábrica sólo empieza un trabajo, el conjunto de restricciones (6) asegura que en cada fábrica sólo termina un trabajo. El conjunto de restricciones (7) el tiempo de terminación de un trabajo es mayor o igual que el tiempo de terminación del mismo trabajo en la máquina anterior más el tiempo de procesamiento en dicha máquina, El conjunto de restricciones (8) asegura que un trabajo sólo puede empezar a ser procesado en una máquina hasta que el anterior en procesamiento haya terminado. El conjunto de restricciones (9) calcula la tardanza de los trabajos. El conjunto de restricciones (10) y (11) que asegura la no negatividad de las variables y el conjunto de restricciones (12) y (13) definen los dominios de las variables de decisión binarias.

La confiabilidad y eficiencia del modelo matemático se verificó a través de su implementación en el software Gusek. En éste se realizaron las corridas de las 36 instancias cortas (tamaño explicado en la sección 6.1). Para cada una de las instancias se realizaron los diagramas de Gantt (Figura 2) correspondientes de acuerdo con las secuencias resultantes de las variables binarias. Adicionalmente se compararon los tiempos de inicio y de terminación ilustrados en los diagramas de Gantt contra los resultados que se obtenían de modelo matemático (Tabla 2), verificando su coincidencia y asegurando así ningún tipo de cruce entre la secuencia de los trabajos.

A continuación, se toma la instancia con tamaño F=2, M=2 Y T=4

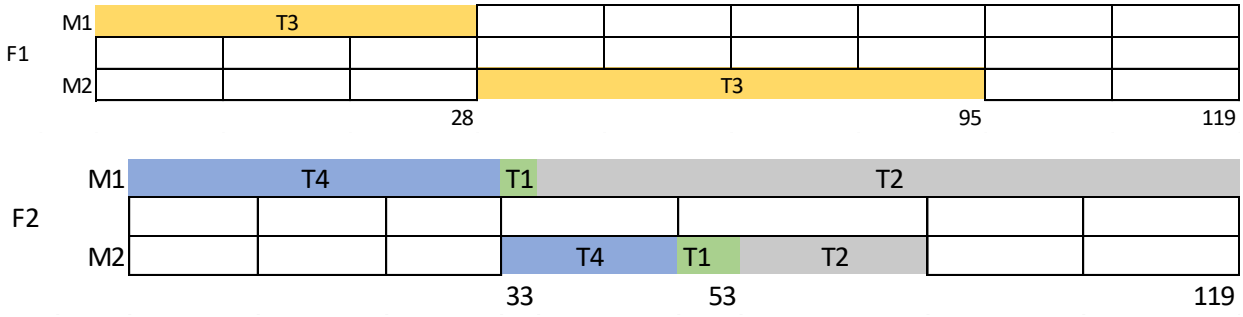

Figura 2: Diagrama de Gantt para instancia corta

Autoría propia

Tarea	Finalización	Fábrica
3	95	1

Tarea	Finalización	Fábrica
4	49	2
1	44	2
2	119	2

Tabla 2: Resultados modelo matemático

Autoría propia

5. Simheurística (algoritmo genético hibridizado con simulación de Monte Carlo)

Con el fin dar solución al problema, se implementa un algoritmo genético hibridizado con simulación de Monte Carlo. Los algoritmos genéticos se basan en la mecánica de selección natural y de la genética natural. Según menciona Golberg, el algoritmo genético combina la supervivencia del más apto por medio del intercambio de información de manera aleatoria con el fin de construir un algoritmo que tenga las mejores características de adaptación que la generación anterior. (Golberg et al., 1989)

Considerando los elementos de un algoritmo genético y de una simheurística esta sección se presenta seis subsecciones que corresponden a la definición del cromosoma, población inicial, función fitness y simulación de Monte Carlo, operador de cruce, operador de mutación y, proceso de selección.

5.1 Definición del Cromosoma

El cromosoma es un vector en el que se representa la secuencia de trabajos en cada fábrica. La secuencia de trabajos entre fábricas está separada por el número cero. De esta manera, primero se coloca la secuencia de procesamiento de los trabajos en la fábrica 1, luego el número cero, posteriormente la secuencia de procesamiento de los trabajos en la fábrica 2, luego el número cero, y así sucesivamente hasta haber colocado la secuencia de procesamiento en todas las fábricas y la última casilla se debe registrar el valor de la tardanza esperada. En la Figura 2 se puede observar un ejemplo con 10 trabajos y 3 fábricas. En dicho ejemplo, en la fábrica 1 se procesa primero el trabajo 5, luego el 4 y finalmente el 3. En la fábrica 2, se procesa primero el trabajo 2, luego el 1, luego el 9 y finalmente el 10. En la fábrica tres la secuencia de procesamiento es 7, 6 y 8.

5	4	3	0	2	1	9	10	0	7	6	8	0

Figura *3*: Definición del cromosoma
Autoría propia

5.2 Población inicial

Para la población inicial se generan todos los cromosomas de forma aleatoria. Los cromosomas se crean enviándolos a una subrutina donde para cada trabajo se asigna una fábrica de manera aleatoria, esto se va modificando según el operador de mutación y cruce.

```
switch (fabricas)
{
    case 2:for(int i=0;i<trabajos;i++)
    {
        aleatorio=rand()%2;
        if(aleatorio==0)
        {
            vecaux(i)=1;
        }
        else
        {
            vecaux(i)=2;
        }
    }
break;
}
```

Figura 4: Definición de la población inicial
Autoría propia

5.3 Función fitness y simulación de Monte Carlo

La función fitness corresponde a la tardanza esperada de cada cromosoma. Es aquí donde se integra la Simulación de Monte Carlo al algoritmo genético para convertirlo en una simheurística. La tardanza se calcula de forma tradicional utilizando tiempos de procesamiento modificados por la simulación de Monte Carlo, este procedimiento se repite 100 veces por cada corrida de cada instancia para finalmente calcular un promedio de las tardanzas obtenidas. Se adapta de la literatura el número específico de repeticiones por cada corrida (100) teniendo en cuenta que fueron utilizadas las iteraciones sin mejora continuas y, después de revisar en la misma, se encontró que fue lo más común en cada uno de los artículos, después de ser probados otros valores, claramente. El hecho de utilizar iteraciones sin mejora continuas puede hacer que con una instancia se hagan 101 corridas como mínimo, sin límite superior, por lo que se puede llegar a más de 1500 iteraciones en el caso que el algoritmo siga encontrando mejores soluciones. Además, el costo computacional (en recursos como el tiempo) que requieren realizar más de esas cien repeticiones es considerable.

Para calcular la tardanza esperada, se tienen en cuenta el tiempo de procesamiento del trabajo i en la máquina j, el tiempo de finalización del trabajo i-1 en la máquina j y/o el tiempo de finalización del trabajo i en la máquina j-1. Tomando el valor más alto de estos últimos y sumando el tiempo de procesamiento, de esta manera se repite este proceso para todos los trabajos en todas las máquinas. Finalmente se compara el tiempo de finalización de cada trabajo en la última máquina donde fue procesado con el Due Date obteniendo así una tardanza por tarea, las cuales se suman para obtener así la tardanza esperada.

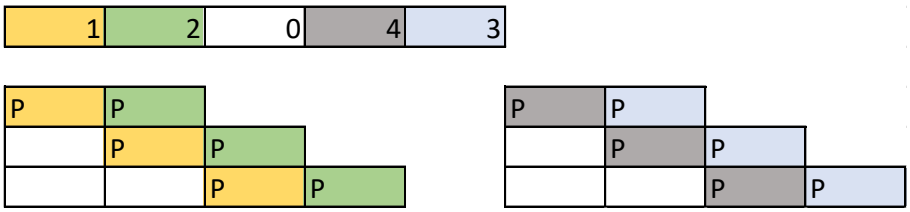

Figura 5: Cromosoma y diagrama de Gantt

Autoría propia

$$\mu_{ij} = ln(P_{ij}) - \frac{1}{2} ln\left(1 + \frac{(\{P_{ij}\} * CV)^2}{\{P_{ij}\}^2}\right)$$

Ecuación #1: Miu para Monte Carlo. (Juan et al., 2014)

$$\sigma_{ij} = \left|\sqrt{ln\left(1 + \frac{(\{P_{ij}\} * CV)^2}{\{P_{ij}\}^2}\right)}\right|$$

Ecuación #2: Sigma para Monte Carlo. (Juan et al., 2014)

En las ecuaciones #1 y #2 (Juan et al., 2014) encontramos los parámetros que utilizaremos en la distribución Log Normal donde μ es la media de los datos y σ la desviación estándar, dentro de las ecuaciones P_{ij} representa el tiempo de procesamiento de cada trabajo i en cada máquina j y CV representa los coeficientes de variación. Se genera un número randómico determinado por la distribución utilizando μ y σ como parámetros de entrada los cuales usan un coeficiente de variación de 0.1, 0.4 y 0.7 de acuerdo con la corrida que se esté realizando de la instancia.

5.4 Operador de cruce

En primer lugar, se seleccionan dos padres al azar de la población. Siendo $\boldsymbol{p_c = 100\%}$ la probabilidad de cruce**,** se ejecuta la siguiente operación: se selecciona un punto aleatorio de corte en el primer padre y los trabajos del lado izquierdo se copian en el mismo orden en el primer hijo, los trabajos faltantes se toman del segundo padre respetando la secuencia en la que se encuentran. El mismo proceso se hace para generar el segundo hijo. Es decir, se selecciona un punto de corte al azar del segundo padre y se copia la secuencia de los trabajos a la izquierda del punto de corte en el segundo hijo. Luego del primer padre se toman los trabajos restantes respetando la secuencia en la que se encuentran. En la Figura 5 se observa un ejemplo de los dos hijos resultantes de dos padres seleccionados al azar. Los trabajos se asignan aleatoriamente a cada fabrica para formar el cromosoma, de manera que cuando ya se encuentran registrados en el cromosoma los trabajos asignados a cada fábrica se separan por un cero para registrar en seguida los trabajos asignados a la siguiente fábrica, siendo una fila de números del 1 a la cantidad de trabajos que tenga asignados, separados por una cantidad de ceros que es determinado por el número de fábricas -1, y al final se registra su tardanza.

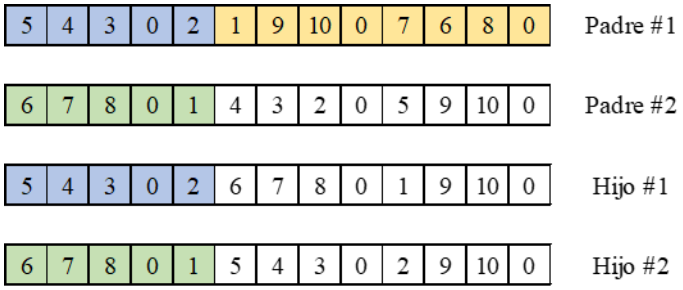

Figura 6: Operador de cruce

Autoría propia

5.5 Operador de mutación

El operador de mutación consiste en seleccionar dos posiciones aleatorias del cromosoma e intercambiar los trabajos en esas dos posiciones. Para cada hijo obtenido en el proceso de cruce, el operador de mutación es aplicado de manera independiente con una probabilidad pequeña de mutación $\boldsymbol{p_m = 20\%}$. La forma en que el Operador de Mutación funciona comienza al seleccionar una posición aleatoria del cromosoma entre 0 y trabajos+fabricas-1, donde no se encuentre un 0 en el interior. Luego se selecciona otra posición del cromosoma entre 0 y trabajos+fábricas-1 donde no se encuentre un 0 en el interior y dicha posición sea obligatoriamente diferente a la anterior y finalmente se intercambia el contenido de estas dos posiciones sin importar si es en la misma fábrica o en fábricas diferentes. En la Figura 6 se presenta la mutación realizada a un hijo. Aleatoriamente se seleccionaron las posición 2 de la primera fábrica y la posición 4 de la segunda fábrica y se intercambiaron los trabajos en dichas posiciones.

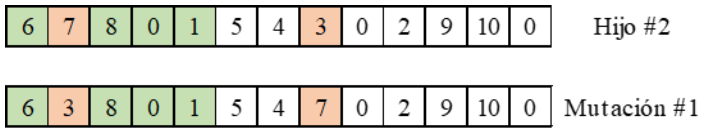

Figura 7: Operador de mutación

Autoría propia

5.6 Proceso de selección

Una vez realizados los procesos de cruce y mutación se deben ordenar todos los cromosomas del mejor fitness al peor fitness. Cuando ya se tienen ordenados se seleccionan los N mejores cromosomas que corresponden al tamaño de la población, y éstos pasan a la siguiente generación. En cada generación se guarda el mejor resultado obtenido para poder realizar el conteo de la cantidad de generaciones sin mejora que haya, y así dar por terminado el algoritmo.

6. Experimentos computacionales

6.1. Instancias

Las instancias utilizadas para comprobar la efectividad del algoritmo genético fueron tomadas de (SISTEMA DE OPTIMIZACIÓN APLICADA, 2021). El conjunto original de instancias en este repositorio contiene 420 instancias pequeñas y un 720 instancias grandes. Éstas cuentan con varias combinaciones de n trabajos y m máquinas, donde n={20,50,100} y m={5,10,20} con un número de fábricas desde 2 hasta 7 para las instancias grandes. Para las instancias pequeñas la combinación es la siguiente, n={4,6,8} y m={2,3,4,5} con fábricas desde 2 hasta 4. De cada combinación de n, m y f los autores generaron 10 instancias diferentes.

Para realizar las pruebas de este proyecto, se seleccionó al azar una instancia de cada combinación de n, m y f. De esta manera se seleccionaron 36 instancias pequeñas y 54 instancias grandes. Dado que las instancias del repositorio no tienen *due dates*, éstos fueron generados utilizando el procedimiento propuesto por (Rifai et al., 2016b) que consiste en genera un número aleatorio de una distribución uniforme, tal como se presenta en la ecuación #3:

$$td_j = U(X * \left(1 - \Omega - \frac{Z}{2}\right), X * \left(1 - \Omega + \frac{Z}{2}\right))$$

Ecuación #3: Distribución uniforme para calcular los Due date

Donde X es el valor máximo de la suma de cada trabajo en cada máquina, Ω es el factor de tardanza y Z es el intervalo de fechas de vencimiento que fueron tomados en (Vallada et al., 2008) teniendo valores para Ω={0.1, 0.2, 0.3, 0.4, 0.5} y Z={0.8, 1, 1.2, 1.4, 1.6, 1.8} para las instancias grandes, y de Ω={0.2, 0.4, 0.6} y Z={0.2, 0.6, 1} para las cortas.

6.2. Parametrización de la simheurística

Se seleccionaron 5 instancias al azar de todo el conjunto de instancias *benchmark* para hacer la parametrización de la simheurística. Estas instancias fueron ejecutadas con el modelo matemático cuya solución fue simulada con Monte Carlo y también fueron ejecutadas con la simheurística, con el objetivo de obtener el GAP y a través de este seleccionar la mejor combinación de parámetros de la simheurística. Para ello se realizó un diseño experimental en el que se tomaron como factores "Tamaño de Población", "Iteraciones sin mejora", "Probabilidad de mutación" e "Instancias". En la Tabla #2 se presentan los niveles de cada uno de los factores los cuales fueron elegidos a través de experimentos previos o con soporte en la literatura. Cada instancia en cada combinación de los factores fue ejecutada 3 veces, de tal manera que se obtuvieron para el ANOVA un total de 2025 observaciones. Se probaron los supuestos de normalidad, homocedasticidad e independencia de los residuos, pero la normalidad y la homocedasticidad no se cumplieron (valores-p < 0.001). Por ello se realizó la prueba no paramétrica de diseños factoriales denominada ANOVA-Type Statistic (Brunner et al., 2012)

FACTOR	Niveles				
Tamaño de población	20	60	100		
Iteraciones Sin Mejora	20	60	100		
Probabilidad de Mutación	0,1	0,2	0,3		
Instancias	1	2	3	4	5

Tabla 3: Factores del ANOVA

Autoría propia

Por medio de la herramienta de ingeniería RStudio se obtuvieron los resultados respectivos del ANOVA-Type Statistic (Tabla #3) que indican que los factores que tienen efecto significativo son "Tamaño de Población" e "Iteraciones sin mejora". Con una confianza del 95% se concluye que en los rankings de los intervalos de confianza para cada solución de las instancias la mejor combinación corresponde a "Tamaño de Población" = 100 y "Número de iteraciones sin mejora" = 100. quedado que la "probabilidad de mutación" no tuvo efecto significativo (entre los dos niveles probados) se eligió al azar uno de los dos niveles que fue el de "probabilidad de mutación" =20%

ANOVA.Type.Statistic:	Statistic	df1	df2	p-Value
TamanoPoblacion	35,9604	1,9639	149,7917	0
ItSinMejora	74,7803	1,9282	149,7917	0
TamanoPoblacion:ItSinMejora	0,5318	3,7129	149,7917	0,6993
Instancia	1713,466	2,7206	149,7917	0
TamanoPoblacion:Instancia	1,6329	5,1285	149,7917	0,1529
ItSinMejora:Instancia	2,5091	5,2047	149,7917	0,0306
TamanoPoblacion:ItSinMejora:Instancia	0,9227	9,3763	149,7917	0,5099

Tabla 4: ANOVA Type Statistic

Autoría propia

6.3. Comparación de simheurística vs. soluciones simuladas del modelo matemático en instancias pequeñas

El desempeño de la simheurística fue evaluado en primer lugar para instancias pequeñas, una vez se obtuvo la solución del modelo matemático para cada instancia, esta fue simulada con Monte Carlo para obtener el valor esperado de la tardanza. Posteriormente se comparó para cada instancia el valor esperado de la tardanza obtenido con la simheurística contra el valor esperado de la tardanza dado por la simulación de la solución del modelo matemático. En la ecuación #4 se presenta la fórmula del GAP.

$$GAP = \frac{Tardanza\ Simh - Tardanza\ Mod.Mate}{Tardanza\ Mod.Mate}$$

Ecuación #4: GAP para instancias pequeñas

Donde *Tardanza Simh* es la tardanza esperada obtenida por la simheurística, y *Tardanza Mod. Mate*, es la tardanza esperada obtenida al simular la solución dada por el modelo matemático. El GAP promedio fue de -0.270 y todos los valores obtenidos fueron negativos, demostrando así que en todos los casos es mejor implementar la simheurística

para resolver el problema estocástico que quedarse con la solución determinística (ver Tabla 4). Adicionalmente, todos estos datos se tuvieron en cuenta como la variable de respuesta "GAP" en el diseño de experimentos que se realizó para poder obtener la mejor combinación de parámetros en el algoritmo genético.

Utilizando el GAP obtenido para cada una de las instancias corridas cada una 3 veces con cada uno de los coeficientes de variación, se realizó un estadístico junto con los conjuntos de fábricas, trabajos y máquinas obteniendo los siguientes resultados:

CV	PROMEDIO	INTERVALOS DE CONFIANZA	
0.1	-0.20	-0.07	-0.04
0.4	-0.30	-0.25	-0.16
0.7	-0.30	-0.30	-0.21
FABRICAS	**PROMEDIO**	**INTERVALOS DE CONFIANZA**	
2	-0.43	-0.47	-0.30
3	-0.22	-0.23	-0.17
4	-0.14	-0.11	-0.07
TRABAJOS	**PROMEDIO**	**INTERVALOS DE CONFIANZA**	
4	-0.14	-0.15	-0.10
6	-0.23	-0.20	-0.11
8	-0.42	-0.49	-0.35
MAQUINAS	**PROMEDIO**	**INTERVALOS DE CONFIANZA**	
2	-0.21	-0.22	-0.15
3	-0.25	-0.23	-0.11
4	-0.24	-0.21	-0.10
5	-0.36	-0.31	-0.18

Tabla 5: Tabla de estadísticos para instancias cortas

Autoría propia

Para los intervalos de confianza, se realizaron las pruebas para comprobar los supuestos de normalidad, homocedasticidad e independencia de los residuos, pero con valores $p < 0.001$ no se cumplieron, por lo tanto, se procedió a realizarlos haciendo uso del test de modo. Se puede observar en los resultados que, para coeficientes de variación (máquinas, trabajos) en cuanto mayor sea el valor de estos, mejor será el GAP, pero en las fábricas es lo contrario, entre menor sea el número de fábricas el GAP será mucho mejor.

6.4. Comparación de simheurística vs. Soluciones simuladas de regla de despacho en instancias grandes

Se realizó también la evaluación del desempeño de la simheurística en comparación con la simulación de las soluciones dadas por la regla de despacho ATC, en instancias grandes. En la ecuación #5 se presenta la fórmula del del porcentaje de mejoramiento.

$$\%Mejoramiento = \frac{Tardanza\ Simh - Tardanza\ EsperadaATC}{TardanzaEsperada\ ATC}$$

Ecuación #5: % de mejoramiento para instancias grandes

CV	PROMEDIO	INTERVALOS DE CONFIANZA	
0.1	-18.02%	-19.22%	-15.49%
0.4	-8.62%	-9.05%	-6.04%
0.7	-3.13%	-3.22%	-1.51%
FABRICAS	**PROMEDIO**	**INTERVALOS DE CONFIANZA**	
2	-7.42%	-5.04%	-5.04%
3	-9.28%	-5.42%	-5.42%
4	-10.18%	-5.36%	-5.36%
5	-11.01%	-6.44%	-6.44%
6	-11.00%	-6.23%	-6.23%
7	-10.91%	-6.51%	-6.51%
MAQUINAS	**PROMEDIO**	**INTERVALOS DE CONFIANZA**	
5	-7.81%	-8.44%	-5.71%
10	-10.82%	-11.49%	-8.21%
20	-11.14%	-11.33%	-7.32%
TRABAJOS	**PROMEDIO**	**INTERVALOS DE CONFIANZA**	
20	-13.63%	-14.65%	-11.41%
50	-11.58%	-9.51%	-7.09%
100	-4.14%	-4.63%	-1.26%

Tabla 6: Tabla de estadísticos para instancias largas

Autoría propia

Para los intervalos de confianza, que se obtuvieron de correr 3 veces cada una de las instancias con cada uno de los coeficientes de variación, con un total de 486 corridas en total. Se realizaron las pruebas para comprobar los supuestos de normalidad, homocedasticidad e independencia de los residuos, pero con un (valores-p < 0.001) no se cumplieron, por lo tanto, se procedió a realizarlos haciendo uso del test de modo. Donde se puede determinar que la simheuristica que se realizó puede mejorar la tardanza en hasta un 19% mejor que la regla de despacho ATC. En los diferentes valores en lo que se realizaron corridas, para los coeficientes de variación y los trabajos el mejoramiento es mejor cuando el valor es menor, de forma contraria para fábricas y máquinas el mejoramiento en mayor cuando se tienen valores altos de estos.

6.5. Comportamiento de la tardanza bajo diferentes coeficientes de variación de los tiempos de procesamiento

Se evaluó a través de un ANOVA el comportamiento de la tardanza esperada bajo diferentes coeficientes de variación en todas las instancias. Los factores analizados fueron factores “Coeficiente de Variación” y las “Instancias”. Se realizó la comprobación de los supuestos (normalidad, homocedasticidad e independencia de los residuos) obteniendo el incumplimiento de la normalidad y la homocedasticidad con valores-p < 0.001. Por tanto se realizó un MANOVA no paramétrico con el que se pudo concluir que los coeficientes de variación tienen un efecto significativo en el valor esperado de la tardanza, y como es de esperarse las instancias por sí mismas también lo tienen (ver Tabla #6).

	Df	SumsOfSqs	MeanSqs	F.Model	R2	PVALUE
CV	2	1,318	0,6591	1894,1	0,01224	0,0
INSTANCIA	53	99,454	1,87648	5392,5	0,92336	0,0
CV:INSTANCIA	106	6,823	0,06437	185,0	0,06335	0,0
Residuals	324	0,113	0,00035		0,00105	
Total	485	107,708			1,00000	

Tabla 7: ANOVA Type Statistic

Autoría propia

Adicional a esto, se halla la mediana de la función objetivo por cada uno de los coeficientes de variación. Con los valores de mediana para 0.1 de 9254, 0.4 de 13287 y 0.7 de 20428.5 se concluye que a mayor valor de coeficiente de variación mayor es la tardanza esperada. Esto reafirma la importancia de aplicar la simheurística para resolver el problema estocástico, ya que la solución que minimiza la tardanza a diferentes coeficientes de variación es distinta, indicando que no es adecuado resolver el problema determinístico ante la presencia de incertidumbre.

7. Conclusiones

- En el GAP para todas las instancias pequeñas comparando la simheuristica con el modelo matemático, se evidenció que en el 100% de los casos se encontró una mejor solución y función objetivo más baja en la simheuristica cumpliendo así con este objetivo que se buscaba este trabajo.

- Para el porcentaje de mejoramiento en las instancias largas en las que se realizó la comparación entre las simheuristica y la heurística (regla de despacho ATC). Para el 100% de las instancias se obtuvieron como mínimo 6 de 9 resultados cuyos valores eran más favorables en las corridas de la simheuristica, obteniendo así una mejora frente a la heurística en el 88% del total de las corridas realizadas. En los casos donde la heurística era mejor se presentaba en situaciones particulares de las condiciones de corridas en las que se les atribuye mayormente a los coeficientes de variación, ya que como se mencionó anteriormente, a mayor valor de coeficiente de variación, mayor sería el valor de la tardanza esperada.

- Dentro de la simheuristica se destaca el hecho de haber utilizado como parámetro de salida las iteraciones sin mejora consecutivas, ya que en la literatura tradicional se utiliza un número especifico de iteraciones lo que, a diferencia de estos, permitió superar mínimos locales y llegar a una solución más optima incurriendo en unos tiempos de procesamiento mayores.

Bibliografía

Chan, H. K., & Chung, S. H. (2013). Optimisation approaches for distributed scheduling problems. International Journal of Production Research, 51(9), 2571–2577. https://doi.org/10.1080/00207543.2012.755345

Baker, K. R., & Altheimer, D. (2012). Heuristic solution methods for the stochastic flow shop problem. *European Journal of Operational Research*, *216*(1), 172–177. https://doi.org/10.1016/j.ejor.2011.07.021

Basha, I. M., Ibrahim, A. H., Abd El-Azim, A. N., & Abd El, A. N. (2016). Profitability Optimization of Construction Project Using Genetic Algorithm Cash Flow Model. *American Journal of Civil Engineering and Architecture, Vol. 4, 2016, Pages 17-27*, *4*(1), 17–27. https://doi.org/10.12691/AJCEA-4-1-3

Brunner, E., Dette, H., & Munk, A. (2012). Box-Type Approximations in Nonparametric Factorial Designs. *Https://Doi.Org/10.1080/01621459.1997.10473671*, *92*(440), 1494–1502. https://doi.org/10.1080/01621459.1997.10473671

Chan, H. K., & Chung, S. H. (2013). Optimisation approaches for distributed scheduling problems. *International Journal of Production Research*, *51*(9), 2571–2577. https://doi.org/10.1080/00207543.2012.755345

Chen, S., Pan, Q. K., & Gao, L. (2021). Production scheduling for blocking flowshop in distributed environment using effective heuristics and iterated greedy algorithm. *Robotics and Computer-Integrated Manufacturing*, *71*, 102155. https://doi.org/10.1016/J.RCIM.2021.102155

Companys, R., & Ribas, I. (2016). Efficient constructive procedures for the distributed blocking flow shop scheduling problem. *Proceedings of 2015 International Conference on Industrial Engineering and Systems Management, IEEE IESM 2015*, 92–98. https://doi.org/10.1109/IESM.2015.7380142

de Armas, J., Juan, A. A., Marquès, J. M., & Pedroso, J. P. (2017). Solving the deterministic and stochastic uncapacitated facility location problem: from a heuristic to a simheuristic. *Journal of the Operational Research Society*, *68*(10), 1161–1176. https://doi.org/10.1057/s41274-016-0155-6

Fernandez-Viagas, V., & Framinan, J. M. (2014). A bounded-search iterated greedy algorithm for the distributed permutation flowshop scheduling problem. *Http://Dx.Doi.Org.Ezproxy.Javeriana.Edu.Co/10.1080/00207543.2014.948578*, *53*(4), 1111–1123. https://doi.org/10.1080/00207543.2014.948578

Framinan, J. M., & Perez-Gonzalez, P. (2015). On heuristic solutions for the stochastic flowshop scheduling problem. *European Journal of Operational Research*, *246*(2), 413–420. https://doi.org/10.1016/j.ejor.2015.05.006

Fu, Y., Tian, G., Fathollahi-Fard, A. M., Ahmadi, A., & Zhang, C. (2019). Stochastic multi-objective modelling and optimization of an energy-conscious distributed permutation flow shop scheduling problem with the total tardiness constraint. *Journal of Cleaner Production*, *226*, 515–525. https://doi.org/10.1016/J.JCLEPRO.2019.04.046

González-Neira, E. M., & Montoya-Torres, J. R. (2019). A simheuristic for bi-objective stochastic permutation flow shop scheduling problem. *Journal of Project Management*, *4*, 57–80. https://doi.org/10.5267/j.jpm.2019.1.003

Gonzalez-Neira, E. M., Montoya-Torres, J. R., & Jimenez, J.-F. (2021). A Multicriteria Simheuristic Approach for Solving a Stochastic Permutation Flow Shop Scheduling Problem. *Algorithms*, *14*(7), 210. https://doi.org/10.3390/a14070210

Guimarans, D., Dominguez, O., Panadero, J., & Juan, A. A. (2018). A simheuristic approach for the two-dimensional vehicle routing problem with stochastic travel times. *Simulation Modelling Practice and Theory*, *89*. https://doi.org/10.1016/j.simpat.2018.09.004

Hatami, S., Calvet, L., Fernández-Viagas, V., Framiñán, J. M., & Juan, A. A. (2018). A simheuristic algorithm to set up starting times in the stochastic parallel flowshop problem. *Simulation Modelling Practice and Theory*, *86*(April), 55–71. https://doi.org/10.1016/j.simpat.2018.04.005

Huang, J. P., Pan, Q. K., & Gao, L. (2020). An effective iterated greedy method for the distributed permutation flowshop scheduling problem with sequence-dependent setup times. *Swarm and Evolutionary Computation*, *59*, 100742. https://doi.org/10.1016/J.SWEVO.2020.100742

Huang, J., Pan, Q., & Chen, Q. (n.d.). *A Hybrid Genetic Algorithm for the Distributed Permutation Flowshop Scheduling Problem with Sequence-Dependent Setup Times*. https://doi.org/10.1088/1757-899X/646/1/012037

Jing, X. L., Pan, Q. K., & Gao, L. (2021). Local search-based metaheuristics for the robust distributed permutation flowshop problem. *Applied Soft Computing*, *105*, 107247. https://doi.org/10.1016/J.ASOC.2021.107247

Juan, A. A., Barrios, B. B., Vallada, E., Riera, D., & Jorba, J. (2014). A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times. *Simulation Modelling Practice and Theory*, *46*, 101–117. https://doi.org/10.1016/j.simpat.2014.02.005

Latorre-Biel, J. I., Ferone, D., Juan, A. A., & Faulin, J. (2021). Combining simheuristics with Petri nets for solving the stochastic vehicle routing problem with correlated demands. *Expert Systems with Applications*, *168*, 114240. https://doi.org/10.1016/j.eswa.2020.114240

Naderi, B., & Ruiz, R. (2010). The distributed permutation flowshop scheduling problem. *Computers & Operations Research*, *37*(4), 754–768. https://doi.org/10.1016/j.cor.2009.06.019

Pan, J. Q., Zou, W. Q., & Duan, J. H. (2018). A discrete artificial bee colony for distributed permutation flowshop scheduling problem with total flow time minimization. *Chinese Control Conference, CCC*, *2018-July*, 8379–8383. https://doi.org/10.23919/CHICC.2018.8482716

Pinedo, M. L. (2016). Scheduling. In *Scheduling* (5 th Editi). Springer International Publishing. https://doi.org/10.1007/978-3-319-26580-3

Rifai, A. P., Nguyen, H. T., & Dawal, S. Z. M. (2016a). Multi-objective adaptive large neighborhood search for distributed reentrant permutation flow shop scheduling. *Applied Soft Computing*, *40*, 42–57. https://doi.org/10.1016/J.ASOC.2015.11.034

Rifai, A. P., Nguyen, H. T., & Dawal, S. Z. M. (2016b). Multi-objective adaptive large neighborhood search for distributed reentrant permutation flow shop scheduling. *Applied Soft Computing Journal*, *40*, 42–57. https://doi.org/10.1016/J.ASOC.2015.11.034

Ruiz, R., Pan, Q. K., & Naderi, B. (2019). Iterated Greedy methods for the distributed permutation flowshop scheduling problem. *Omega*, *83*, 213–222. https://doi.org/10.1016/J.OMEGA.2018.03.004

SISTEMA DE OPTIMIZACIÓN APLICADA. (2021). *Instancias de problemas*. ITI. http://soa.iti.es/instancias-problemas

Vallada, E., Ruiz, R., & Minella, G. (2008). Minimising total tardiness in the m-machine flowshop problem: A review and evaluation of heuristics and metaheuristics. *Computers and Operations Research*, *35*(4), 1350–1373. https://doi.org/10.1016/J.COR.2006.08.016

Villarinho, P. A., Panadero, J., Pessoa, L. S., Juan, A. A., & Oliveira, F. L. C. (2021). A simheuristic algorithm for the stochastic permutation flow-shop problem with delivery dates and cumulative payoffs. *International Transactions in Operational Research*, *28*(2), 716–737. https://doi.org/10.1111/itor.12862

Wang, K., Huang, Y., & Qin, H. (2016). A fuzzy logic-based hybrid estimation of distribution algorithm for distributed permutation flowshop scheduling problems under machine breakdown. *Journal of the Operational Research Society*, *67*(1), 68–82. https://doi.org/10.1057/jors.2015.50

Wang, S. Y., Wang, L., Liu, M., & Xu, Y. (2013). An effective estimation of distribution algorithm for solving the distributed permutation flow-shop scheduling problem. *International Journal of Production Economics*, *145*(1), 387–396. https://doi.org/10.1016/J.IJPE.2013.05.004