Facultad de Ingeniería

INGENIERÍA INDUSTRIAL

Trabajo de Grado – Segundo Semestre 2020

[201018] *Job shop* estocástico con minimización del valor esperado del *maximum lateness*

Gabriel Forero 1[a,c] , María José Ocampo 2[a,c] , Andrea Rivera 3[a,c],

Eliana María González Neira

[a]*Estudiante de Ingeniería Industrial*
[b]*Profesor, Director del Proyecto de Grado, Departamento de Ingeniería Industrial*
[c]*Pontificia Universidad Javeriana, Bogotá, Colombia*

Abstract

The drawbacks that programming in job -shop environment imply, refer to a notorious difficulty for its resolution due to its NP-hard nature. However, the research has grown in the late years because of its constant use in manufacturing industries. According to studies, most of the research has approached the job shop scheduling through a deterministic approach. Nevertheless, real industrial environments are subject to random events as: machinery faults, maintenance duration, processing duration, enlistment times, availability times, among many others. In this project, a stochastic job shop that minimizes the expected maximum lateness is addressed. The problem consider sequence dependent setup times, and the stochastic events are machine breakdowns. To solve the problem a simheuristic approach is proposed. The simheuristic Hybridizes a tabu search algorithm with a Monte Carlo simulation.

The problem was solved in three phases: Firstly, a mixed integer linear programming model was designed for the deterministic counterpart of the JSSP studied. Secondly, the meta-heuristic tabu search was designed to solving large instances of the deterministic problem. Thirdly, the simheuristic was designed and implemented to minimize the expected maximum lateness value, considering stochastic machine breakdowns.

For the simheuristic designing, stochastic variables were generated: times between failures and repair times, following exponential and log-normal distributions. To generate their respective parameters [expected value (μ) and standard deviation (σ)], the mean time to repair was found (MTTR Mean Time to Repair), out of the total mean time between breakdowns. Four different variation coefficient values were proposed (0%, 5%, 10% and 15%), them being: 0% for the deterministic case and 5%, 10% and 15% for stochastic events, to calculate the (σ) in log-normal distribution. On the other hand, a simulation was performed to calculate the expected objective function. The simheuristic was firstly parametrized through an experimental design considering different tabu list sizes and number of iterations without improvement.

With the generated parametrization, another computational experiment was executed for a total of 554 instances of different sizes. First, the performance of the simheuristic, for small instances, was evaluated in comparison with the simulation of optimal solutions obtained with the mathematical model. Results show that the simheuristic improves the results of simulations of the model in a 37% for 4x4 instances and in an 11% for 6x6 instances, demonstrating that the simheuristic is better than a deterministic mathematical model simulated. Additionally, the simheuristic performance was evaluated, for large instances, in comparison with the simulation of EDD dispatching rule sequences. Results show that the average improvement is 28% in log-normal distribution and 10% for exponential distribution.

Palabras claves: stochastic jop shop, maximum lateness, breakdowns, simheuristic, tabu search.

1. Justificación y planteamiento del problema

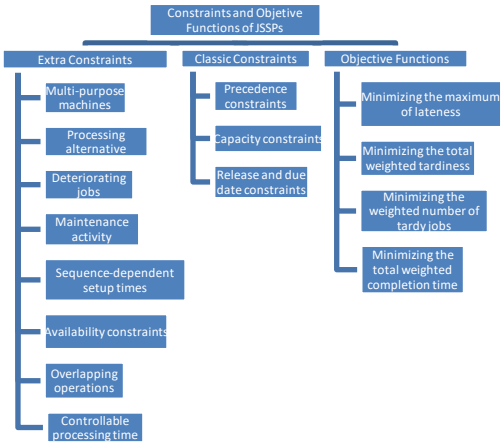
En el mundo dinámico y competitivo en el que se vive en la actualidad, la búsqueda de la excelencia es una necesidad para sobrevivir en los mercados. La globalización y la revolución tecnológica del último siglo ha

obligado a las empresas a mejorar todos sus procesos internos. Bajo esta premisa, día a día nuevas metodologías y paradigmas aparecen en el panorama con el fin de hacer mejor las cosas. Debido a lo anterior, hoy en día la producción ha sido cada vez más presionada para buscar reducir los tiempos, los residuos, la mano de obra o los costos, entre muchas otras cosas (Hill, 2011). En adición, las demandas del mercado han mutado y ahora se busca la especialización de los productos según las necesidades específicas del cliente. Por esto se ha hecho necesario para las empresas buscar programar la producción debidamente y así seguir siendo competitivos con la gran oferta del mercado.

La programación de la producción consiste en la asignación de recursos a las operaciones para lograr uno o varios objetivos. Esta se empezó a estudiar seriamente en ambientes de manufactura a mediados del siglo XVIII a comienzos del siglo XIX con los trabajos de Gantt (1919) y otros pioneros como Smith (1956), Johnson (1954) y Jackson (1956). Una de las clasificaciones existentes de los problemas de programación de la producción está dada por la configuración de las máquinas en el ambiente productivo. Se encuentran así ambientes de una sola máquina, máquinas en paralelo, *flow shop, job shop* y *open shop*. En particular, el problema de programación de la producción en un ambiente tipo *job shop* (JSSP por sus siglas en inglés) es aquel en el que se programan *n* trabajos en *m* máquinas que hacen operaciones distintas, cada trabajo tiene una única ruta predeterminada por las *m* máquinas la cual es independiente de la ruta de los otros trabajos y cada máquina sólo puede procesar un trabajo a la vez (Jain & Meeran, 1999; Lazar, 2012). Este tipo de configuración se presenta en industrias como aeroespacial (Chiang & Torng, 2016), productora de herramientas para maquinaria (Kadipasaoglu et al., 1999), productora de equipos de construcción (Jacobs & Lauer, 1994), entre otras.

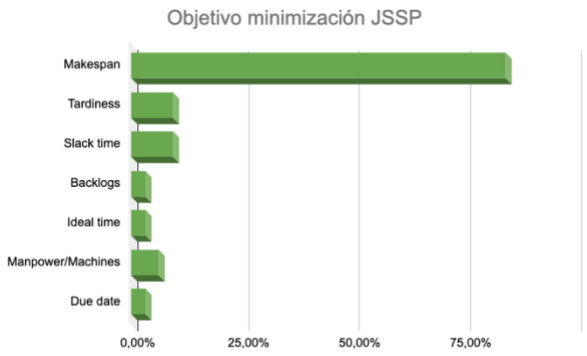
La revisión de la literatura recientemente presentada por Abdolrazzagh y Adbullah (2017) resume en la Ilustración 1 las características estudiadas en los problemas de JSSP analizados, así como las funciones objetivo consideradas. En particular, los autores mencionan que las restricciones clásicas que se han tenido en cuenta en los problemas JSSP están relacionadas con precedencia, capacidad y la inclusión de tiempos de *release* y *due dates*. No obstante, existen otras restricciones que se han estudiado en menor cantidad debido a la complejidad que agregan al problema.

Ilustración 1 Funciones Objetivo y Variables empleadas en problemas Job shop. Fuente: Job shop Scheduling: Classification, Constraints and Objective Functions; Absolrazzagh, M and Abdullah, S. (2017).



Adicionalmente, Abdolrazzagh y Adbullah (2017) concluyen que las funciones objetivo más estudiadas han sido *makespan*, tardanza ponderada, retraso máximo, entre otras. Al respecto, luego de la revisión de la literatura más reciente realizada para este proyecto, se encontró que el 84,88% de los estudios se centraron en minimizar el *makespan*. Mientras que solo el 9,38 %, 6,25% y el 3,13% corresponden respectivamente a la minimización de la tardanza ponderada, retrasos en el proceso y eventos asociados a las máquinas (Ilustración 2)

Ilustración 2 Objetivos minimización de JSSP Fuente: Autoría propia



Por otro lado, la mayoría de los estudios realizado en JSSP han considerado todos los parámetros determinísticos (Mohan et al., 2019). No obstante, los entornos industriales son intrínsecamente complejos con la ocurrencia de eventos reales de comportamiento estocástico tales como averías de máquinas, tiempo de mantenimiento, tiempos de procesamientos variables, entre otras, y no son tenidas en cuentas en los problemas analizados (Meng et al., 2020). Por ello, la importancia de estudiar el JSSP estocástico (SJSSP).

Unos de los métodos en auge utilizados en la literatura para resolver problemas de optimización estocásticos son las simheurísticas, que consisten en una hibridación entre las metaheurísticas y la simulación (A. A. Juan et al., 2015). La ventaja de implementar una simheurística, es que permite verificar la solución obtenida mediante una simulación de la misma teniendo en cuenta las variables estocásticas definidas previamente. Esto permite llegar a soluciones más adecuadas para el contexto del problema dado que se tiene en cuenta la incertidumbre. Las simheurísticas han sido implementadas exitosamente para la solución de problemas de ruteo (González-Martín et al., 2016; Gruler et al., 2018) y programación de la producción (Gonzalez-Neira et al., 2017).

Considerando que las simheurísticas son una hibridación entre las metaheurísticas y la simulación, es importante mencionar las metaheurísticas que han sido utilizadas para resolver problemas *Job Shop* en la literatura: *alternative graph* (Meloni et al., 2004), *simulated annealing* (Naderi et al., 2010), (Baykasoglu, 2002), *tabu research* (Baykasoglu, 2002) e *Iterated Greedy* (IG)(Pranzo & Pacciarelli, 2016), hibridación entre *artificial inmune y simulated annealing* (Roshan et al., 2013), o el algoritmo NHGASA que es un híbrido entre algoritmos genéticos (GA) y *simulated annealing* (Shahsavari-Pour & Ghasemishabankareh, 2013).

Para este proyecto se va a utilizar una metaheurística de tipo *Tabu search*. Una de las ventajas del algoritmo de búsqueda Tabú (TS por sus siglas en inglés)- según Abido (2002). “es la robustez de los parámetros y de la solución inicial. En adición, TS se caracteriza por su habilidad para evitar atrapamientos en soluciones optimas locales y en prevenir ciclos debido al uso de memoria flexible para la búsqueda histórica” .

En consecuencia, de lo anterior, el presente proyecto pretende resolver el problema del SJSSP mediante una simheurística, usando como metaheurística la Búsqueda Tabú, para minimizar el *maximun lateness* considerando los daños en las máquinas como variables independientes de tipo estocástico. El *maximun lateness* está relacionado con la fecha de entrega a los clientes, el cual implica minimizar la peor violación de los *due dates* (Pinedo, 2016) y es una medida más general que el *makespan* y recomendada para ambientes *make-to-order* (Botta-Genoulaz, 2000). Por otro lado, la variable de tipo estocástico (daños en las máquinas) se modelará con dos distribuciones de probabilidad para el desarrollo de esta propuesta: Log-normal y Exponencial. La primera de estas distribuciones es importante ya que no tiene en cuenta valores probabilísticos negativos lo cual para el tiempo de averías de máquinas es deseable. La distribución exponencial es una función de tiempo que ocurre hasta que se produce un fallo, por esta razón es una distribución apta para caracterizar la variable.

Considerando todos los aspectos mencionados, se formula la siguiente pregunta:

¿Cómo se debe estructurar y diseñar una simheurística con búsqueda tabú para programar la producción en un JSSP estocástico que minimice el valor esperado del *maximum lateness* considerando tiempos de alistamiento dependientes de la secuencia determinísticos y daños estocásticos en las máquinas?

2. Antecedentes

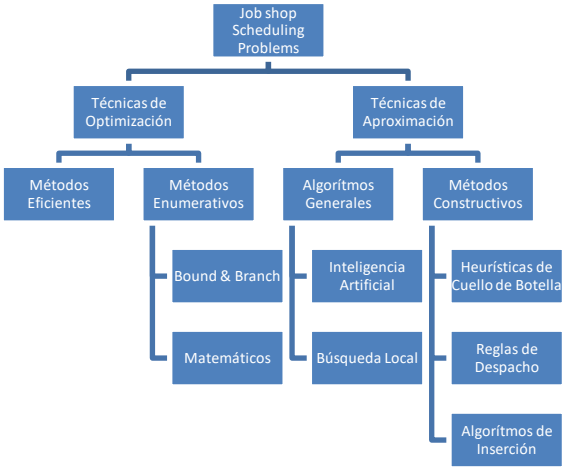
Considerando que se pretende resolver un JSSP estocástico, esta sección se divide en 4 partes. La primera revisa los JSSP haciendo énfasis en aquellos que han considerado información determinística de entrada. La segunda subsección hace referencia a los estudios que se han hecho sobre JSSP estocásticos. La tercera sección menciona estudios recientes que se han hecho sobre simheurísticas.

2.1 *JSSP* determinísticos

Desde los años cincuenta para adelante inicio el interés por expandir los modelos teóricos para la resolución de problemas de *Job shop* (Abdolrazzagh & Adbullah, 2017). "No se sabe con certeza quién fue el primero en hablar en el ámbito académico de JSSP. Sin embargo, estudios como la utilización del Álgebra Booleana de Akers y Friedman (1955) o la generalización del algoritmo de *Flow Shop* de Johnson a los JSSP por parte de Jackson en 1956" Jain y Meeran (1999) son algunos de los primeros estudios académicos del JSSP, el objetivo de dichos estudios se resumía en el cumplimiento de todos los trabajos en el menor tiempo posible (Jain & Meeran, 1999).

A partir de las primeras investigaciones, la literatura frente a la resolución de problemas *Job shop* tuvo gran crecimiento utilizando técnicas muy innovadoras. Estas van desde simples reglas de despacho, hasta sofisticados algoritmos, heurísticas completas y metaheurísticas que buscan resultados más exactos. La Ilustración 3 muestra las diferentes técnicas que se han implementado.

Ilustración 3 Técnicas Fuente: Autoría propia



En optimización se han utilizado métodos exactos para la resolución de JSSP clásicos (Zhang et al., 2019). Es decir, se construyen con base en datos de entrada con los que se puedan establecer una serie de reglas preliminares que determinen el flujo del proceso para obtener una solución óptima global. Sin embargo, su tiempo de ejecución es impredecible y para problemas más grandes requerirían de tiempos computacionales muy extensos (Gmys et al., 2020). Uno de los métodos exactos más utilizados en *Job shop* es el algoritmo Branch & Bound (Gmys et al., 2020) . Por otro lado, están los modelos matemáticos, entre los más utilizados están *mixed integer linear programming (MILP)* de Manne (1960), *Langrangian Relaxation* de Fisher (1973) y *integer linear programming"* de Li et al., (2018). Estos modelos buscan resolver problemas de instancias de tamaños limitados (Gromicho et al., 2012).

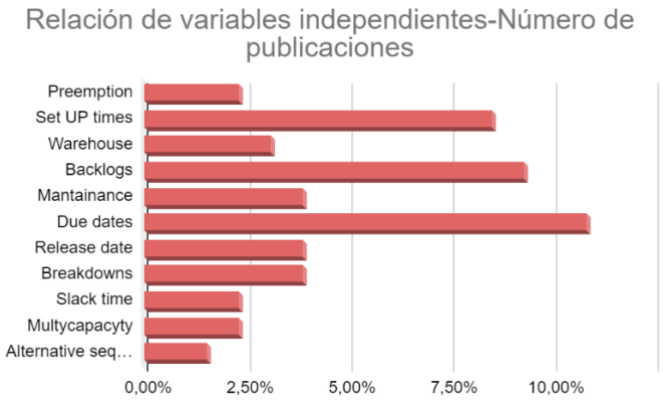
Debido a que los métodos exactos requieren altos tiempos computacionales para instancias medianas a grandes, se han desarrollado métodos de aproximación que, si bien no encuentran un óptimo global, buscan óptimos locales que se aproximen en cierto porcentaje a éste. De esta manera, los métodos de aproximación encuentran soluciones de alta calidad dentro de un tiempo de cálculo razonablemente corto (Fernandez-Viagas et al, 2017). Además, se pueden utilizar las soluciones óptimas como el estándar de referencia para desarrollar métodos aproximados (Li et al., 2018). Estos modelos pueden garantizar la resolución de problemas de instancias más grandes encontrando una solución óptima más rápido que los métodos exactos (Jain & Meeran, 1999).

Entre los métodos de aproximación están los métodos constructivos donde la mayoría de las heurísticas se basan en reglas de despacho (Jain & Meeran, 1999). También están los métodos iterativos, que en diferentes fases, buscan acercarse a una solución óptima, seleccionándola ya sea de una lista aleatoria o con una función de probabilidad esto se conoce como Greedy Randomized Adaptive Search Procedure (GRASP) (Bierwirth & Kuhpfahl, 2017; Feo & Resende, 1995).

En general, en la literatura *Job shop* se encuentra que la mayoría de estudios han minimizado el *makespan* y el *tardiness* (Ilustración 2). El *makespan* se define como el tiempo utilizado hasta la terminación de todos los trabajos del sistema. Por otro lado, *tardiness* está relacionado con los *due dates*, y busca medir el retraso que tienen los trabajos frente a su fecha de entrega presupuestada (Pinedo, 2016).

En cuanto a las restricciones que se tuvieron en cuenta en los problemas JSSP revisados para el presente proyecto, de 40 trabajos recientes, se encontró que características como *mantainance, breakdowns y slack time*, en conjunto, solo representan un 10,06% de las referencias encontradas. En la Ilustración 4 se puede observar que las variables independientes más representativas fue *set up times* el cual se tuvo en cuenta en el 8,53% de las referencias consultadas.

Ilustración 4 Relación de variables independientes-Número de publicaciones Fuente: Autoría propia



2.2 JSSP estocásticos

La mayoría de los problemas de toma de decisiones son modelos de optimización combinatoria que pueden modelarse con condiciones inciertas (estocásticas) (A. A. Juan et al., 2015). Por lo tanto, los métodos de aproximaciones estocásticas son métodos para optimizar o controlar sistemas donde la relación entre las variables y los factores controlables de un proceso y su modelo analítico son desconocidos, estos problemas se conocen como Job shop estocástico (SJSSP) (Olguin Tiznado et al., 2011).

Banerjee (1965) y Makino (1965) fueron los primeros en realizar estudios en ambientes de producción específicamente en *flow shop* con tiempos estocásticos. El primero solucionó un problema de una sola máquina con tiempos de procesamiento aleatorios siguiendo una distribución de probabilidad conocida, con el objetivo de minimizar la probabilidad máxima de retraso. El segundo realizo el estudio con 2 y 3 máquinas considerando distribuciones exponenciales y k-Erlang para los tiempos de procesamiento, minimizando el *makespan* (A. A. Juan et al., 2015).

Ahora bien, D. Lei (2011) realizó un estudio para SJSSP por medio de un algoritmo genético (GA) teniendo en cuentas varios objetivos. Para la realización del estudio, se tuvieron en cuenta tiempos de procesamiento exponenciales, buscando minimizar el *makespan* y el *total tardiness*. En este caso, los tiempos de procesamiento se modelaron con una distribución exponencial. El autor recomienda que a futuro se consideren los daños estocásticos en las máquinas (objetivo del presente proyecto) (D. Lei, 2011).

D. M. Lei (2012) abordó también el problema de SJSSP sujeto a *breakdowns*, proponiendo también GA. El estudio tuvo en cuenta tiempos de procesamiento como variables aleatorias en una distribución normal buscando minimizar el *makespan.* Para medir la factibilidad del diseño propuesto comparó el GA con otra metaheurística conocida como *particle swarm optimization* (PSO) mostrando que GA tiene mejor desempeño que PSO. En cuanto a la restricción de *breakdowns* tuvo en cuenta las siguientes condiciones: la ocurrencia de los *breakdowns*, la reparación de la máquina y el tratamiento de las operaciones interrumpidas (D. M. Lei, 2012).

Azadeh et al. (2012) proponen un algoritmo hibrido el cual se evalúa su desempeño en una simulación de eventos discretos y *artificial neural networks* (ANN), para encontrar la solución de un JSSP. El objetivo del algoritmo propuesto es seleccionar la mejor regla de despacho que se asignará a cada máquina con el objetivo de minimizar el *makespan.* Tuvieron en cuentan tiempos de procesamiento aleatorios medidos en tres distribuciones de probabilidad: normal, exponencial y uniforme, donde la variabilidad de la desviación estándar y la media son las mismas para cada máquina (Azadeh et al., 2012).

2.3 Simheurísticas aplicadas a problemas de optimización

La simulación de Monte Carlo usa un muestreo probabilístico mediante la generación de variables aleatorias, con el objetivo de construir nuevos estados del sistema. Es por esta razón que diversos estudios utilizan este modelo unido a las metaheurísticas para encontrar mejores soluciones. Por ejemplo, Almeder y Hartl (2013) realizaron una simheurística para un ambiente de *flow shop* estocástico en la industria metalúrgica. Los autores propusieron una metaheurística llamada *neighborhood local search* en la cual la función objetivo se evaluaba mediante la simulación de Monte Carlo (Juan et al, 2015).

A. Juan et al, (2014) utilizaron una simheurística para resolver el problema de permutación en un ambiente de producción *flow shop* con tiempos de procesamiento estocástico. Esto lo lograron combinando la simulación de Monte Carlo con la metaheurística *Iterated local Search* (ILS). Utilizaron el *makespan* como criterio de minimización buscando soluciones de alta calidad con bajo tiempo de procesamiento computacional. Este es un problema Np- hard que tiene varias aplicaciones en la vida real como lo es: industria de fabricación y producción, sector de servicios, industria de bioquímica y farmacéutica y el sector de la informática y telecomunicaciones.

No obstante, las simheurísticas también tienen aplicación en problemas de optimización de ruteo (Guimarans et al., 2018). Estos autores buscaron resolver el problema de 2L-VRP el cual es una extensión al problema clásico de ruteo, en el cual los artículos o productos de los clientes no son aplicables. En este estudio la variable estocástica es el tiempo de ruteo que tienen los vehículos a sus puntos de destino y para resolverlo utilizaron una simulación de Monte Carlo con la metaheurística ILS.

Otro ejemplo, del problema de ruteo es resuelto por A. a. Juan et al. (2014). En esta investigación se busca resolver el problema de enrutamiento de inventario estocástico con desabastecimientos. El objetivo es minimizar los costos de inventario para cada periodo, para lograr esto utilizaron una simulación de Monte Carlo hibridizada con heurísticas.

3. Objetivos:

Proponer una simheurística que involucre la metaheurística Tabú Search para la programación de la producción en un Job shop estocástico considerando tiempos estocásticos de daños en las máquinas para minimizar el maximum lateness esperado.

- Plantear el modelo matemático del Job shop determinístico para la minimización del *maximum lateness*
- Diseñar e implementar la simheurística para minimizar el valor esperado del *maximum lateness* considerando daños estocásticos en las máquinas
- Comparar el desempeño de la simheurística para el caso determinístico contra los resultados del modelo matemático simulado en instancias pequeñas
- Comparar el desempeño de la simheurística para el caso estocástico contra la simulación de la regla de despacho EDD considerando las distribuciones de probabilidad Log-normal y exponencial en los daños en las máquinas

3.1. Declaración de diseño

Diseñar e implementar en un lenguaje de programación una simheurística para la programación en la producción en un Job shop clásico considerando tiempos estocásticos de daños en las máquinas para minimizar el *maximum lateness* esperado, considerando tiempos de alistamiento y procesamiento dependientes de la secuencia determinísticos y daños en las máquinas estocásticos.

3.2. Requerimientos esperados de diseño

La simheurística planteada en el presente proyecto permitirá incluir diferentes distribuciones de probabilidad de los daños en las máquinas. Para fines del proyecto se espera probar las siguientes distribuciones:

3.2.1 Distribución exponencial

Se dice que una variable aleatoria Y tiene una distribución exponencial con parámetro $\beta > 0$ si y sólo si la función de densidad de Y es:

$$F(y) = \begin{cases} \frac{1}{\beta} e^{-\frac{y}{\beta}} \text{ si } 0 \leq y < \infty \\ 0, \ en\ otro\ punto \end{cases}$$

3.2.2 Distribución Log-normal

Se dice que una variable aleatoria Y tiene una distribución Log-normal parámetro μ_y y σ_y si y sólo si la función de densidad de Y es:

$$F(y) = \frac{1}{\sqrt{2\pi}\sigma_y} e^{-\frac{1}{2}\left(\frac{\ln(y)-\mu_y}{\sigma_y}\right)}, si\ 0 \leq y < \infty$$

Por otro lado, se espera:

- Al menos mejorar un 5% los resultados de la regla de despacho EDD (*Earliest Due Date*) en promedio en las instancias evaluadas
- El tiempo de ejecución sea máximo de dos horas

3.3. Restricciones de diseño

- Los tiempos de *setup* son determinísticos
- Todos los trabajos están disponibles desde el inicio y se conocen sus tiempos de procesamiento (determinísticos)
- No se permite *preemption* es decir el procesamiento de los trabajos no pueden ser interrumpidos
- La ruta de cada trabajo por las máquinas es conocida desde el inicio
- Una máquina sólo procesa un trabajo a la vez
- Los *buffers* entre máquinas tienen capacidad ilimitada
- Cada trabajo consiste de *M* operaciones

- Cuando una máquina se daña mientras un trabajo está siendo procesado, se repara primero la máquina y una vez reparada el trabajo continúa su tiempo de procesamiento faltante en dicha máquina
- Todos los trabajos están disponibles desde el tiempo cero

4. Modelo matemático del JSSP determinístico

Este apartado está enfocado en desarrollar el primer objetivo el cual fue plantear un modelo matemático para un ambiente *Job shop* considerando tiempos de alistamiento dependientes de la secuencia determinísticos para la minimización del *maximum lateness*. Por consiguiente, en primer lugar, se propuso un modelo de programación lineal entera mixta (MILP, por sus siglas en inglés). Posteriormente se evaluó el rendimiento del modelo MILP en instancias pequeñas (2x3 y 4x4) como para instancias grandes (6x6 y 10x10) y por medio de diagramas de Gantt se comprobaron las soluciones factibles. Ahora bien, el modelo de JSSP está caracterizado por un conjunto de J trabajos $\{0, \dots, |J|\}$ que tienen que ser procesados por un conjunto de máquinas $I\{1, \dots, |I|\}$. Cada trabajo tiene una ruta predefinida de procesamiento en las máquinas. Cada máquina puede procesar solo un trabajo a la vez y cada trabajo puede ser procesado por solo una máquina a la vez. Para el modelo se tuvieron en cuenta los supuestos mencionados en el aparto 3.3 del presente documento. Para presentar el modelo matemático se necesitan las siguientes notaciones:

Conjuntos:
I: Máquinas $\{1, \dots, |I|\}$
J: Trabajos $\{0, \dots, |J|\}$

Parámetros:
- P_{ji}: Es el tiempo requerido de operación de un trabajo $j \in J$ en la máquina $i \in I$
- D_j: Es la fecha de vencimiento o fecha en que el trabajo $j \in J$ debe terminar todas las operaciones, generalmente por un compromiso preestablecido con un cliente externo y/o interno.
- $Ruta_{jil}$: 1 si el trabajo $j \in J$ visita la máquina $i \in I$ y después la máquina $l \in I$. 0 de lo contrario.
- Q: un número muy grande
- S_{kji}: tiempo de alistamiento del trabajo $j \in J$ en la máquina $i \in I$ que es procesado inmediatamente después del trabajo $k \in J$

Variables:
- X_{kji}: Toma el valor de 1 si el trabajo $k \in J$ es procesado inmediatamente después del trabajo $j \in J$ en la maquina $i \in I$. 0 de lo contrario
- C_{ij}: El tiempo en el que se completó el trabajo $j \in J$ en la maquina $i \in I$
- L_j: Diferencia entre el tiempo de terminación del trabajo $j \in J$ y tiempo de entrega del trabajo $j \in J$
- $MaxL$: Maximo de lateness
- F_j: Tiempo de terminación del trabajo $j \in J$

Es importante aclarar que se introdujo un trabajo ficticio 0 que precede el primer trabajo en las máquinas. Teniendo en cuenta lo anterior, el problema se modela de la siguiente manera:

$$\text{Min } Z = MaxL \quad (1)$$

Sujeto a:

$$MaxL \geq L_j \quad \forall j \in J, j \geq 1 \quad (2)$$

$$L_j = F_j - d_j \quad \forall j \in J, j \geq 1 \quad (3)$$

$$\sum_{k=1,k\neq j} X_{kji} = 1 \quad \forall i \in I, \forall j \in J, j \geq 1. \quad (4)$$

$$\sum_{j\in J,k\neq j} X_{kij} \leq 1 \quad \forall i \in I, \forall k \in J. \quad (5)$$

$$\sum_{j\in J,k\neq j} X_{0ji} = 1 \quad \forall i \in I \quad (6)$$

$$X_{kji} + X_{jki} \leq 1 \quad \forall k, j \in J, \forall i \in I, j \geq 1, j \leq |J| - 1, k > j \quad (7)$$

$$C_{ji} \geq C_{jl} + P_{ji} + \sum_{k\in J,k\neq j} X_{kji} S_{kji} - Q\left(1 - Ruta_{jil}\right) \forall i \in I, \forall j \in J, \forall l \in I \cup \{0\}, j \geq 1, i \neq l \quad (8)$$

$$C_{ji} \geq C_{ki} + P_{ji} + S_{kji} - Q\left(1 - X_{kji}\right) \quad \forall i \in I, \forall j \in J, \forall k \in J, j \geq 1, k \geq 1, j \neq k \quad (9)$$

$$F_j \geq C_{ji} \quad \forall j \in J, \forall i \in I \tag{10}$$
$$C_{ji} \geq 0 \quad \forall i \in I, \forall j \in J, j \geq 1 \tag{11}$$
$$X_{kji} \in \{0,1\} \quad \forall k, j \in J, j \neq k \tag{12}$$

La función objetivo (1) consiste en minimizar el *maximum lateness*. El conjunto de restricciones (2) permite calcular el "máximo retraso" el cual va a ser igual al mayor retraso del total de trabajos. La ecuación (3) describe el retraso del trabajo j. El conjunto de restricciones (4) asegura que en cada máquina cada trabajo es programado sólo una vez. El conjunto (5) asegura que en cada máquina cada trabajo tiene a lo sumo un sucesor. La ecuación (6) especifica que en cada máquina debe tener haber un trabajo que inicia la secuencia. La ecuación (7) describe que un trabajo no puede ser predecesor y sucesor al mismo tiempo de otro trabajo en la misma máquina. El conjunto de restricciones (8) asegura que un trabajo j no empiece una operación hasta que no haya terminado la anterior. El conjunto (9) asegura que en una misma máquina dos trabajos no pueden ser procesados a la vez. En la ecuación (10) calcula el tiempo de terminación de un trabajo después de haber pasado por todas las máquinas. El conjunto (11) indica la no negatividad de los tiempos de terminación de todos los trabajos en todas las máquinas.

5. Metaheurística Búsqueda Tabú (caso determinístico)

Como paso preliminar al planteamiento de la simheurística se realizó el diseño de la metaheurística y su posterior evaluación de desempeño en comparación con los resultados del modelo matemático. Los componentes principales a tener en cuenta fueron la solución inicial, cálculo de la función objetivo, espacio de búsqueda y estructura del vecindario.

5.1 Solución inicial

La solución inicial de la Búsqueda Tabú consiste en encontrar una primera solución factible que no necesariamente sea la óptima. La solución factible propuesta cumple con todas las restricciones del JSSP teniendo en cuenta que se programa con horarios *non-delay* (horarios sin demoras), es decir ningún recurso se mantiene ocioso si este puede empezar alguna actividad.

A partir las rutas de las operaciones determinadas para cada trabajo se genera la solución inicial de la siguiente manera:

(1) Se toma la operación de cada trabajo que sea candidata para ser asignada. Las operaciones cuyos predecesores han sido programados son candidatos a la máquina correspondiente.

(2) En cada etapa puede haber más de un candidato para cada máquina. En general, para una máquina j dada en la etapa k, puede ocurrir uno de los siguientes tres casos:

Caso 1: No hay ningún candidato para la máquina en la etapa.
Caso 2: Solo hay un candidato para la máquina j en la etapa k. En este caso dicho candidato es asignado en la secuencia de la máquina inmediatamente posterior.
Caso 3: Si hay más de un trabajo candidato por procesar en la misma máquina j de la etapa k. Dado que se sólo puede procesarse un trabajo a la vez, se debe tomar la decisión de asignación para dicha etapa a partir de una regla de despacho. En este caso se debe escoger primero el que tenga el mínimo tiempo de procesamiento (SPT Smallest Processing Tiempo), este entendido como la suma del tiempo de procesamiento y el tiempo de alistamiento.

(3) Hacer esto con todas las posiciones de las rutas hasta asignar todas las operaciones de los trabajos a cada una de sus máquinas respectivas.

Es importante aclarar dos aspectos: i) Para este caso, se escogió el SPT como regla de despacho para el desempate de operaciones debido a su buen rendimiento para minimizar el *maximum lateness* de acuerdo con los estudios de Vinod y Sridharan (2011) y Su et al, (1998).

5.2 Espacio de búsqueda y estructura del vecindario

5.2.1 Ejemplo ilustrativo

A manera ilustrativa se presenta el siguiente ejemplo de un problema de secuenciación de tareas en un ambiente JSSP con 6 máquinas y 6 trabajos. El objetivo es encontrar el orden para secuenciar las operaciones por cada máquina de tal forma que se minimice el *maximum lateness* de la ejecución de los trabajos. Los tiempos de procesamiento y los *due dates* dados se encuentran en la Tabla 1 y los tiempos de alistamiento se encuentran en la Tabla 2. En la Tabla 3 se contemplan, las rutas de cada uno de los trabajos.

Tabla 1 Due dates y tiempos de procesamiento

	Tiempos de procesamiento						
Trabajo	Due Date	M1	M2	M3	M4	M5	M6
1	39	1	6	6	3	3	7
2	58	10	8	4	10	10	5
3	42	5	9	4	1	8	7
4	50	3	9	5	5	9	4
5	53	1	3	3	5	9	4
6	40	4	10	3	1	9	3

Tabla 2 Tiempos de alistamiento

	Máquina 1						Máquina 2						Máquina 3						Máquina 4						Máquina 5						Máquina 6					
	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6
1	5	7	9	9	10	8	8	7	8	8	2	4	4	10	6	2	7	10	2	4	8	2	6	5	3	7	1	8	3	2	9	10	8	2	6	6
2	3	3	5	1	10	1	2	4	8	6	9	9	1	7	10	2	2	2	5	10	7	8	6	8	4	9	6	6	8	4	1	7	3	3	2	1
3	7	5	7	9	4	8	7	9	5	7	8	4	5	3	9	4	5	7	5	6	8	3	3	4	8	7	2	9	6	9	9	5	9	3	10	6
4	5	4	6	8	1	8	2	6	1	10	8	4	3	10	8	7	9	7	10	5	2	7	1	2	2	9	6	10	5	5	4	7	7	4	1	2
5	2	9	9	1	8	4	6	6	#	6	2	5	3	7	10	2	9	8	3	5	3	10	2	7	7	2	3	6	6	10	6	4	6	3	10	4
6	4	4	8	6	10	4	6	6	2	2	9	1	7	2	4	2	2	10	8	9	6	9	5	10	7	8	5	6	8	3	6	1	4	4	1	2

Tabla 3 Ruta para cada trabajo

Rutas						
Job						
1	3	1	2	4	6	5
2	2	3	5	6	1	4
3	3	4	6	1	2	5
4	2	1	3	4	5	6
5	3	2	5	6	1	4
6	2	4	6	1	5	3

Como se mencionaba anteriormente, se asume que es necesario construir una solución inicial para el problema en consideración. Esta solución se compone a través de escoger, para cada máquina, una ruta de trabajos seleccionando siempre, entre los trabajos que están disponibles en el momento, el que tenga el menor tiempo de producción, compuesto por la sumatoria entre el tiempo de procesamiento y su respectivo tiempo de alistamiento. A continuación, se hace una breve explicación de lo anterior:

Como orden de ejecución se comienza por planear la ruta de la primera máquina. Es importante rescatar, que la restricción de disponibilidad permite asignar trabajos a máquinas solo si el trabajo ya puede hacerse en esa

máquina, siguiendo la ruta preestablecida para su producción y si la máquina está disponible. De esta forma se puede evidenciar en la Tabla 4 que, de momento, no existe ningún trabajo que pueda iniciar en la máquina 1. Por tal razón, siguiendo el algoritmo, se procede a evaluar la segunda máquina, donde los trabajos 2, 4 y 6 están disponibles para su debida asignación. Siguiendo con lo propuesto, la ruta para estos primeros trabajos seria 0-2-4-6, teniendo en cuenta que cada vez que se asigna un trabajo su tiempo de alistamiento cambia, al ser este tiempo dependiente de la secuencia.

Tabla 4 Ejemplo ilustrativo solución inicial

Máquina 1			**Máquina 2**	**RUTA: 2-4-6**			
	Disponibilidad	**Tiempo (TP+TA)**		**Disponibilidad**	**Iteración 1**	**Iteración 2**	**Iteración 3**
Trabajo 1	NO	N/A	**Trabajo 1**	NO	N/A	N/A	N/A
Trabajo 2	NO	N/A	**Trabajo 2**	SI	8	N/A	N/A
Trabajo 3	NO	N/A	**Trabajo 3**	NO	N/A	N/A	N/A
Trabajo 4	NO	N/A	**Trabajo 4**	SI	9	15	N/A
Trabajo 5	NO	N/A	**Trabajo 5**	No	N/A	N/A	N/A
Trabajo 6	NO	N/A	**Trabajo 6**	SI	10	19	14

La Tabla 5 contiene la ruta de la solución inicial. Para determinar la función objetivo es importante tener en cuenta, tanto la disponibilidad de la máquina como la del trabajo. Es decir, aunque la máquina esté disponible puede que el trabajo no lo esté, debido a que este puede estar pendiente por la realización de operaciones anteriores en otras máquinas. El valor de la función objetivo para este caso es de 73.

Tabla 5 Secuencia generada solución inicial

Rutas Solución Inicial						
Mq						
1	1	4	3	6	5	2
2	2	4	6	5	1	3
3	5	3	1	2	4	6
4	3	6	1	4	5	2
5	5	2	4	6	1	3
6	6	3	5	2	1	4

Teniendo presente la solución inicial, el paso a seguir es utilizar el método TS para construir “soluciones adyacentes”, a partir de intercambiar la posición de dos trabajos en la solución anterior, hasta contemplar todas las posibles combinaciones que mejoren la solución actual. En el problema ejemplificado, un intercambio consiste en el cambio de la posición en la secuencia de dos tareas en cada una de las máquinas, como se ilustra en la Tabla 6. De esta forma, para cada intercambio existe un valor de movimiento para la función objetivo. Con el intercambio ejemplificado la función objetivo obtiene un valor de 78.

Tabla 6 Intercambio 1

Intercambio 1						
Máquina	Trabajo					
1	4	1	3	6	5	2
2	2	4	6	5	1	3
3	5	3	1	2	4	
4	3	6	1	4	5	2
5	5	2	4	6	1	3
6	6	3	5	2	1	4

El método TS utiliza una lista tabú (FT) para simplificar el proceso de permutación. En esta lista se guardan los intercambios que han sido evaluados con anterioridad en cada una de las máquinas, por un número determinado de intercambios; de tal forma que no sean evaluados nuevamente si ya se ha evidenciado que este intercambio no mejora la función objetivo. En el ejemplo, como el intercambio en la
Tabla 6 (4,1) no se mejora la función objetivo, se realiza el siguiente intercambio en la máquina, y el intercambio (4,1) es guardado en la FT. Cada vez que un intercambio, en cualquiera de las máquinas, optimiza la solución del problema estas listas se reinician con el fin de volver a evaluar intercambios que tal vez, antes no mejoraban la solución, pero después de un intercambio, sí lo hacen. No obstante, si el número de iteraciones sin mejora es igual al FT, este intercambio vuelve a ser factible reiniciando su FT en 0. Siguiendo con el ejemplo, con el segundo intercambio de pares (4,3) la función objetivo ahora es de 74, lo que hace que la solución mejore, por consiguiente, la FT se reinicia.

6. Simheurística

Los tiempos de daños entre máquinas y los tiempos de reparación fueron probados con las distribuciones exponencial y Log-normal en las diferentes instancias. Para utilizar dichas distribuciones de probabilidad es importante saber los valores que tendrán sus parámetros: Valor esperado (μ) y desviación estándar (σ). Con el fin de asignar de manera lógica dichos parámetros, se presenta el procedimiento de obtención a partir del uso del parámetro *Ag* utilizado por Gholami et al. (2008) el cual denota la proporción de tiempo medio que una máquina se encuentra en reparación (MTTR *Mean Time To Repair*) de su tiempo total medio entre daños (MTBF *Mean Time Between Failures*) (ver ecuación (12)).

$$Ag = \frac{MTTR}{MTBF+MTTR} \qquad (12)$$

Para los datos experimentales se utilizaron 3 valores de Ag, equidistantes entre sí, que son 0.05, 0.1 y 0.15. Basado en el mismo autor el valor de MTTR se tomó como 0,1x $\bar{P}$, siendo $\bar{P}$ el tiempo promedio de procesamiento de cada máquina, lo cual implica que el tiempo de reparación toma un 10% del tiempo promedio de procesamiento. De la ecuación (12) se puede despejar el valor de MTBF tal como se presenta en las ecuaciones (13 y (14).

$$MTBF = \frac{MTTR(1-Ag)}{Ag} \qquad (13)$$

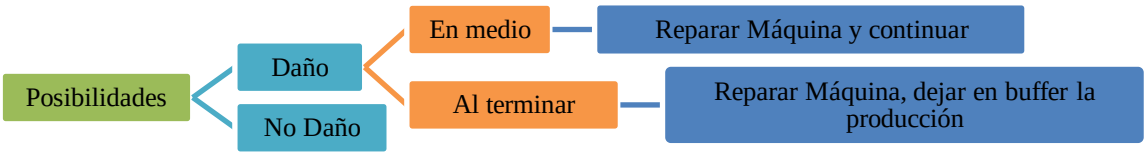
$$MTBF = \frac{0{,}1*\bar{P}(1-Ag)}{Ag} \qquad (14)$$

Los valores obtenidos con los procedimientos anteriores para MTTR y MTBF son los valores esperados que se utilizaran como μ en las distribuciones para los tiempos de reparación y tiempo de daño entre las máquinas respectivamente.

Es importante aclarar, que cuando hay un daño en la máquina, esta debe ser reparada inmediatamente como se muestra en la Ilustración 5. Hay dos posibilidades, que se dañe durante el procesamiento de un trabajo o al

terminar. Cuando ocurre durante el procesamiento de un trabajo, se repara la máquina, y luego se continua con el tiempo de procesamiento del trabajo restante, es decir, la máquina no genera daño en el producto. Por lo tanto, cuando hay un fallo, se hace un corrimiento a la derecha en el tiempo de las operaciones siguientes al daño de la máquina.

Ilustración 5 Árbol de posibilidades cuando hay daños en las máquinas Fuente: Autoría propia



Ahora bien, se proponen cuatro diferentes valores de coeficientes de variación (0%, 5%, 10% y 15%) para el caso de la distribución Log-normal, ya que la exponencial siempre tiene coeficiente de variación 1. El coeficiente de variación 0% se utilizó para el caso determinístico (pues es equivalente) y 5%,10% y 15% para los eventos estocásticos. Teniendo en cuenta que el coeficiente de variación se calcula de acuerdo con la ecuación (15), al fijar valores para el coeficiente de variación, y al ya haber determinado el valor esperado, se puede despejar la desviación estándar a utilizar en la fórmula de la distribución.

$$CV = \frac{\sigma}{\mu} \qquad (15)$$

Se procede a realizar la simulación para calcular la función objetivo esperada. En primer lugar, se simuló la función objetivo de la solución inicial y de la regla de despacho EDD, luego se realiza la simheurística utilizando como entrada la secuencia obtenida en la solución inicial. A continuación, en cinco pasos se explica cómo se calcula el valor esperado del *máximum lateness*, utilizando como referencia a Framinan y Perez-Gonzalez, (2015).

1. Configurar el contador de la simulación en 0. N=0
2. Configurar la suma de los *lateness* a 0. $SumL_{max} = 0$
3. Establecer la suma de cuadrados del *lateness* a 0. $SumSL_{max} = 0$
4. Ejecutar:

(a) Ejecutar una simulación para obtener una muestra de L_{max}
(b) Actualizar el contador de simulaciones N=N + 1
(c) Actualizar la suma del *lateness*. $SumL_{max} = SumL_{max} + L_{max}$
(d) Actualizar la suma de cuadrados del *lateness* $SumSL_{max} = SumSL_{max} + L^2{}_{max}$
(e) Calcular $\overline{L_{max}} := \frac{SumL_{max}}{N}$ y s: $= \sqrt{\frac{SumSL_{max} - N*(\bar{L}_{max})^2}{N-1}}$, mientras que $\frac{s*t_{N-1},\ \alpha/2}{\bar{L}_{max}\sqrt{N}} > p$

5. Imprimir $\overline{L_{max}}$

Con el fin de estimar el valor esperado del *máximum lateness* $\overline{L_{max}}$ se utiliza un porcentaje de error (precisión) del $p = 2\%$ alrededor de la media con un nivel de confianza del 95%. Para lograr esto, la mitad de la anchura del intervalo de confianza para E $[[L_{max}]$ con un nivel de significancia de $(1-\alpha)\% = 95\%$ está dado por $t_{1-\alpha/2,N-1}\frac{s}{\sqrt{N}}$, donde s es la desviación estándar muestra el *máximum lateness* y $\alpha/2$ es el área de distribución de la t-Student con N-1 grados de libertad en el intervalo.

7. Experimentos computacionales

7.1. Instancias

En esta sección se propone un conjunto de evaluaciones empíricas para la simheurística planteada. Se tomó el *benchmark* de instancias disponibles para JSSP propuesta en la Escuela de Ingenierías Industriales de la

Universidad de Valladolid[1]. En todas las instancias de referencia según Oddi et al. (2010), los tiempos de alistamientos S_{kji} y los tiempos de procesamiento P_{ji} en cada máquina son valores calculados aleatoriamente en el intervalo [1,200]. Por otro lado, las fechas de vencimiento (*due dates*) de los trabajos D_j se distribuyen uniformemente en un intervalo *I* el cual tiene los siguientes parámetros: el valor medio μ (16) donde τ denota el porcentaje de trabajos que se espera que lleguen tarde y $E(C_{max})$ es el *makespan* esperado. Se calcularon para todas las instancias los siguientes valores de τ de 0,3 y 0,6 correspondientes a fechas de vencimientos flexibles. El segundo parámetro, R determina el rango de *I* cuyos límites están definidos en la ecuación (17), con valores de R de 0,5 y 2,5, siendo diferentes niveles de variación de las fechas de vencimiento. De esta manera se tuvieron 4 diferentes combinaciones de τ y R que permitieron generar 4 combinaciones diferentes de fechas de entrega para cada conjunto de datos de tiempos de procesamiento, trabajos y máquinas.

$$\mu = (1-\tau)E(C_{max}) \qquad (16)$$

$$\text{I}= [\mu(1-\tfrac{R}{2}), \mu\left(1+\tfrac{R}{2}\right)] \qquad (17)$$

7.2. Parametrización

Para realizar la parametrización se tuvo en cuenta el criterio de parada (CP) y el tamaño de la lista tabu (FT) como factores principales para realizar el experimento computacional. Para el factor FT se probaron los valores 10, 30 y 70, utilizados por Vilcot y Billaut (2008). Para generar los CP se seleccionó una instancia mediana (6x6), se corrió varias veces con diferentes valores de CP para los valores de FT mencionados anteriormente, se buscó el punto de convergencia, y de esta manera se establecieron límites inferiores y superiores para el factor CP, quedando los siguientes niveles 100, 250, 400, 600, 800 y 1000.

Con estos factores se ejecutaron 24 instancias cada una con sus variaciones de diferentes niveles de *due date* mencionados en el apartado 7.1, (6x6, 20x5, 20x15, 15x5, 15x10, 10x10). Cada instancia se corrió dos veces con los CP y FT mencionados, donde se buscaba la variable de respuesta a partir de una medida relativa de la diferencia porcentual del valor esperado de la función objetivo dada por la simheurística contra el valor esperado de la función objetivo dada por la implementación de la regla de despacho EDD. Para garantizar la parametrización adecuada se realizó una ANOVA por separado para las instancias pequeñas, medianas y grandes, con el fin de buscar la mejor combinación de los parámetros CP y FT para cada tamaño de instancia.

Para las intancias grandes, se realizó la prueba de ANOVA, donde se cumplieron los tres supuestos (normalidad, homocedasticidad e independencia) como se puede observar en la Tabla 7 al ser valor-p > 0,05. Por esta razón se se utilizó la prueba de Tukey para seleccionar la mejor combinación de los parámetros FT y CP para resolver las instancias grandes. La combinación seleccionada fue CP de 3000 y FT de 10.

Tabla 7 Resumen Anova lineal- Instancias grandes

		valor-p	
Instancias grandes	**Homogeneidad de varianza(Levene)**	0,419	
	Normalidad (Anderson-Darley)	0,253	
	Significancia factores	**Factor**	**valor-p**
		CP	0,99
		FT	0,000
	Prueba Tukey (Modelo Lin Gen)	**FT**	**Media**
		10	0,014318
		30	-0,0005579
		70	-0,0026591

[1] https://www.eii.uva.es/elena/Elena%20Perez%20Vazquez_archivos/files_optimaJSSP/jobshop1.txt. recuperado el 30 de septiembre, 2020.

Se ejecutó también el ANOVA para las instancias pequeñas y medianas. Al analizar los supuestos , la homogeneidad de varianzas y la normalidad no se cumplieron con valores valor-p< 0.05 en tanto que la independencia sí se cumplió. Por consiguiente, se procedió a realizar un método estadístico no paramétrico que hace la correlación de la variable de respuesta sin los supuestos del ANOVA. Esta prueba se conoce como (*ANOVA-type statistic*), propuesto por Brunner et al. (1997) y realizada en el programa R. Este método ordena la variable de respuesta en un ranking que toma el valor de *pd*, con intervalos de confianza normales desde el punto de vista de la función *logit*. Para encontrar la parametrización adecuada, se debe mirar el mayor *pd* buscando conformar subgrupos de intervalos con el objetivo que no se traslapen, para así buscar el mejor tratamiento, que arrojaría la parametrización final. Dado esto, si el valor-p es mayor a 0,05, los factores evaluados no serían significativos, como se muestra a modo de resumen en la Tabla 8.

Tabla 8 Resultados Anova type Statistic pequeñas y medianas instancias

Anova type statistic						
			Statistic	**df1**	**df2**	**valor-p**
Instancias pequeñas	4x4	**FT**	0,08104523	1.000000	39.59902	9.058547e-01
	6x6	**CP**	43.70372591	1.000000	39.59902	6,865428e-08
Instancias medianas	10x10 10x5 15x10	**FT**	1.5809301	1.968114	195.354	0.20874180
	15x5 20x5	**CP**	35.0779462	4.648266	195.354	0.000000e+00

En la Tabla 9 se presentan los resultados para las instancias medianas, arrojando que los parámetros para realizar el experimento computacional es con FT igual a 30 y CP 1000. Para las instancias pequeñas (4x4) y (6x6), fueron FT igual 15 y CP 50, FT igual 5 y CP 50 respectivamente como se observa en la Tabla 10.

Tabla 9 Parametrización instancias medianas

	FT	**CP**	**Size**	**pd**	**Var**	**L.Normal**	**U.Normal**	**L.Logit**	**U.Logit**
2	30	100	18	0.1159979	0.1218264	0.07799245	0.1540034	0.08305695	0.1597278
3	70	100	18	0.1400892	0.2303809	0.08782562	0.1923527	0.09548749	0.2008969
1	10	100	18	0.1902435	0.5432023	0.10999139	0.2704956	0.12245612	0.2834355
4	10	250	18	0.3844307	0.8621578	0.28332658	0.4855349	0.28945635	0.4891163
6	70	250	18	0.4015775	15.770.297	0.26483752	0.5383175	0.27529842	0.5424256
5	30	250	18	0.4776235	0.8364775	0.37803644	0.5772105	0.38019668	0.5767831
7	10	400	18	0.4892833	10.375.851	0.37836899	0.6001975	0.38066313	0.5989249
9	70	400	18	0.5241770	12.905.067	0.40048094	0.6478730	0.40151409	0.6439906
8	30	400	18	0.5603567	0.9656059	0.45335868	0.6673546	0.45221850	0.6630532
12	70	600	18	0.5703875	13.155.761	0.44549582	0.6952792	0.44368131	0.6884965
10	10	600	18	0.5953361	0.7701618	0.49977819	0.6908940	0.49735427	0.6862663
13	10	800	18	0.5961077	0.7617551	0.50107275	0.6911426	0.49863739	0.6865408
16	10	1000	18	0.6153121	0.8297062	0.51612896	0.7144952	0.51266682	0.7086252
15	70	800	18	0.6414609	11.579.198	0.52429134	0.7586305	0.51805536	0.7486016
18	70	1000	18	0.6540638	11.575.814	0.53691134	0.7712162	0.52976039	0.7603717
11	30	600	18	0.6677812	0.9616912	0.56100035	0.7745621	0.55399996	0.7648557
14	30	800	18	0.6677812	0.9616912	0.56100035	0.7745621	0.55399996	0.7648557
17	30	1000	18	0.7079904	0.9776571	0.60032681	0.8156540	0.59022020	0.8031994

Tabla 10 Parametrización instancias pequeñas

	Instancia	FT	CP	Size	pd	Var	L.Normal	U.Normal	L.Logit	U.Logit
6	6x6	15	10	6	0.1215278	0.13993056	0.03512293	0.2079326	0.05800952	0.2370910
4	6x6	10	10	6	0.1574074	0.06342593	0.09923524	0.2155796	0.10752741	0.2246027
2	6x6	5	10	6	0.1828704	0.10887346	0.10665495	0.2590858	0.11846311	0.2715100
10	6x6	10	50	6	0.3449074	0.36018519	0.20628138	0.4835334	0.22182962	0.4930091
12	6x6	15	50	6	0.3622685	0.41685957	0.21313450	0.5114025	0.22951541	0.5199859
8	6x6	5	50	6	0.3981481	0.05023148	0.34637914	0.4499172	0.34768564	0.4508724
	Instancia	**FT**	**CP**	**Size**	**pd**	**Var**	**L.Normal**	**U.Normal**	**L.Logit**	**U.Logit**
1	4x4	5	10	6	0.6180556	0.09425154	0.54714252	0.6889686	0.54510313	0.6860468
5	4x4	15	10	6	0.6388889	0.18888889	0.53850014	0.7392776	0.53380212	0.7321728
3	4x4	10	10	6	0.6562500	0.22638889	0.54634704	0.7661530	0.53977535	0.7565425
7	4x4	5	50	6	0.8344907	0.10123457	0.76099769	0.9079838	0.74756798	0.8956600
9	4x4	10	50	6	0.8356481	0.14174383	0.74868527	0.9226110	0.72968114	0.9054575
11	4x4	15	50	6	0.8495370	0.29895833	0.72324174	0.9758323	0.67763688	0.9381395

7.3. Resultados simheurística vs las soluciones simuladas de Gusek

El objetivo de este apartado es realizar una comparación entre los resultados de la simheurística contra los resultados de Gusek simulado. Para esto, se ejecutaron las instancias 4x4 y 6x6, dado que al ejecutar instancias más grandes no se llegó a ninguna solución en Gusek en un tiempo máximo de ejecución de 2 horas. Para realizar los cálculos del porcentaje de mejora se usó la ecuación (18).:

$$\%\ de\ mejora = \frac{(FOGusek - FOSimheuristica)}{FOGusek} * 100 \qquad (18)$$

Los resultados del porcentaje de mejora muestran que que la simheurística genera mejores soluciones en promedio que el Gusek simulado, tanto para las instancias 4x4 como para la 6x6. La mejora fue de un 39% y un 18% respectivamente (Tabla 11). Esto demuestra que la metodología propuesta para generar soluciones de un ambiente de producción Job Shop, teniendo en cuenta los daños en las máquinas, es mejor que el modelo matemático determinístico simulado con los daños en las máquinas.

Tabla 11 Resultados simheurística vs Gusek

Instancia	Tao	R	Distribución	FO Gusek	Mejora
4x4	0,3	0,5	Exponencial	18,3	-9%
			Log-normal	19,8	0%
	0,3	2,5	Exponencial	4,0	75%
			Log-normal	3,0	174%
	0,6	0,5	Exponencial	31,3	18%
			Log-normal	30,0	14%
	0,6	2,5	Exponencial	14,7	32%
			Log-normal	14,0	5%
	Total porcentaje de mejora				39%
6x6	0,3	0,5	Exponencial	69,0	23%
			Log-normal	69,0	30%
	0,3	2,5	Exponencial	19,0	4%
			Log-normal	17,0	35%
	0,6	0,5	Exponencial	82,5	13%
			Log-normal	81,0	12%
	0,6	2,5	Exponencial	70,0	22%
			Log-normal	68,0	3%
	Total porcentaje de mejora				18%

7.4. Resultados simheurística vs resultados de EDD simulados

El objetivo es comparar los resultados de la simheurística con la secuencia simulada dada por la regla de despacho EDD, para determinar el porcentaje de mejoramiento. En la Tabla 12, se puede observar que se tuvo en cuenta el tamaño de la instancia (pequeñas, medianas y grandes) y el tipo de distribución (exponencial y Log-normal) para realizar un posterior análisis de acuerdo con estos parámetros. Para realizar los cálculos del porcentaje de mejora se usó la ecuación (19).

$$\%\ de\ mejora = \frac{(FOEDD - FOSimheuristica)}{FOEDD} * 100 \qquad (19)$$

Tabla 12 Resultados simheurística vs EDD

		Mejora
Pequeñas		
4x4	Exponencial	26%
	Log-normal	48%
6x6	Exponencial	6%
	Log-normal	23%
Medianas		
10x5	Exponencial	28%
	Log-normal	37%
10x10 A	Exponencial	14%
	Log-normal	24%
10x10F	Exponencial	7%
	Log-normal	26%
15x5	Exponencial	19%
	Log-normal	26%
15x10	Exponencial	11%
	Log-normal	29%
20x5	Exponencial	-6%
	Log-normal	19%
Grandes		
20x15	Exponencial	1%
	Log-normal	19%

La Tabla 13 resume los resultados, en promedio, del porcentaje de mejora por tamaño de instancia y distribución. Para las instancias pequeñas el porcentaje de mejora fue de un 16% para la distribución exponencial, un 35% para la distribución Log-normal, teniendo como mejora total un 26%. En cuanto a las instancias medianas el porcentaje de mejora fue de un 12% para la distribución exponencial, un 27% para la distribución Log-normal, siendo de un 19% la mejora promedio total. Finalmente, las instancias grandes tuvieron un porcentaje de mejora de un 1% para la distribución exponencial y un 19% para la distribución Log-normal, siendo de un 10% en total. Por lo tanto, a medida que aumenta el tamaño de la instancia disminuyen los porcentajes de mejora de las soluciones dadas por la regla EDD para las dos distribuciones.

Tabla 13 Promedio resultados % de mejora simheurística vs EDD

Instancia	EDD	Gusek	Exponencial	Log-normal
Pequeña	26%	28%	16%	35%
Mediana	19%	-	12%	27%
Grande	10%	-	1%	19%
Total	18%	28%	10%	27%

7.5. Incidencia de las distribuciones en la simheurística

En este apartado se pretende determinar el efecto que tienen las distribuciones de probabilidad en la función objetivo. Con este fin se presentarán a continuación dos pruebas. La primera de estas busca encontrar cómo influyen los resultados de la función objetivo para las mismas instancias con el mismo coeficiente de variación (CV=1) pero con distintas distribuciones de probabilidad (Log-Normal y exponencial). La segunda prueba tiene como objetivo encontrar cómo en diferentes tamaños de instancia al aumentar el coeficiente de variación este puede afectar en la función objetivo esperada, tomando como distribución la Log-normal.

Para la primera prueba se utilizaron las instancias que se presentan en la

Tabla 14 con un CV = 1 para ambas distribuciones.

Tabla 14 Influencia de la distribución en los valores esperados

	Observacion 1				Observación 2			
Tamaño	**Valor**	**Distribución**	**Valor**	**Distribución**	**Valor**	**Distribución**	**Valor**	**Distribución**
6x6	51.00	Log-normal	46.83	Exponencial	52.00	Log-normal	52.17	Exponencial
10x5	630.75	Log-normal	618.00	Exponencial	656.00	Log-normal	645.75	Exponencial
10x10A	1404.25	Log-normal	1270.50	Exponencial	1438.42	Log-normal	1408.33	Exponencial
10x10F	1377.75	Log-normal	1413.67	Exponencial	1365.25	Log-normal	1375.25	Exponencial
15X5	824.00	Log-normal	815.08	Exponencial	838.25	Log-normal	825.25	Exponencial
15X10	1194.33	Log-normal	1485.75	Exponencial	1282.50	Log-normal	1457.17	Exponencial
20X5	1684.58	Log-normal	1879.83	Exponencial	1728.17	Log-normal	1889.83	Exponencial
20X15	3115.33	Log-normal	3400.08	Exponencial	3471.92	Log-normal	3402.92	Exponencial

Para buscar si estos factores son significativos se procedió a hacer una ANOVA de un factor. Sin embargo, los supuestos de normalidad y homogeneidad no se cumplen, por lo cual se utilizó la *ANOVA-type statistic* en el programa R. Los resultados de esta prueba no paramétrica muestran que tanto el tamaño de instancia como el tipo de distribución tienen un valor-p por debajo de 0,05, lo cual indica que ambos factores tienen efecto significativo en la función objetivo. Se puede evidenciar que las instancias, al ser corridas con una distribución exponencial, tienden a tener un menor valor esperado del *maximum lateness* que al ser corridas con una distribución Log-normal.

En contraste, para la segunda prueba se utilizaron los tamaños de instancia que se muestran en la Tabla 15.

Tabla 15 Relación del tamaño de instancia y CV en el valor esperado

Log-normal								
Tamaño Instancia	**CV**	**FO Promedio**	**Tamaño Instancia**	**CV**	**FO Promedio**	**Tamaño Instancia**	**CV**	**FO Promedio**
4x4	5%	15,5825	**10x10 A**	5%	1174,833333	**15x10**	5%	1295,33
4x4	10%	13,08325	**10x10 A**	10%	1149,166667	**15x10**	10%	1266,75
4x4	15%	13,75	**10x10 A**	15%	1241,583333	**15x10**	15%	1332,17
6x6	5%	46,24975	**10x10 F**	5%	1256,75	**20x5**	5%	1574,92
6x6	10%	49,99925	**10x10 F**	10%	1377,75	**20x5**	10%	1684,50
6x6	15%	51,416	**10x10 F**	15%	1359,5	**20x5**	15%	1752,33
10x5	5%	575	**15x5**	5%	688,5833333	**20x15**	5%	3378,42
10x5	10%	630,75	**15x5**	10%	619,0000833	**20x15**	10%	3415,33
10x5	15%	609	**15x5**	15%	693,4999167	**20x15**	15%	3494,75

Siguiendo los mismos pasos que en la primera prueba, se encontró que estos datos no cumplen con los supuestos del ANOVA, por lo cual se utilizó nuevamente el *ANOVA-type statistic* para generar los resultados. Se obtuvo que los factores analizados en esta prueba, CV y Tamaño de instancia, sí eran significativos al tener valores de valor-p menores a 0,05. Se puede evidenciar que a medida que crece el coeficiente de variación la función objetivo crece independientemente del tamaño de instancia.

En la Tabla 16 se presenta un resumen de todos los resultados de la experimentación realizada en el proyecto.

Tabla 16 Resultados promedio finales

		Tiempo Promedio	**EDD promedio**	**Mejora**	**Gusek promedio**	**Mejora**
Pequeñas						
4x4	Exp	591,58	17,33	26%	17	29%
	Log-normal	115,92	15,19	48%	17	48%
6x6	Exp	916,83	52,17	6%	60	15%
	Log-normal	130,39	59,83	23%	59	20%
Tiempo promedio		**438,68**	**% mejora promedio total**	26%	**% mejora promedio**	28%
Medianas						
10x5	Exp	1501,25	882,17	28%		
	Log-normal	377,06	887,44	37%		
10x10 A	Exp	864,75	1561,33	14%		
	Log-normal	489,47	1556,39	24%		
10x10F	Exp	918,79	1506,58	7%		
	Log-normal	1264,61	1451,36	26%		
15x5	Exp	2159,46	1012,92	19%		
	Log-normal	507,08	911,72	26%		
15x10	Exp	1034,92	1649,08	11%		
	Log-normal	422,28	1917,61	29%		
20x5	Exp	1802,25	1632,92	-6%		
	Log-normal	771,06	1620,86	19%		
Tiempo promedio		**1009,41**	**% mejora promedio total**	19%		
Grandes						
20x15	Exp	1385,29	3447,25	1%		
	Log-normal	1811,64	3428,06	19%		
Tiempo promedio		**1598,47**	**% mejora promedio total**	10%		

8. Limitaciones, conclusiones y recomendaciones.

La programación de la producción en un ambiente Job Shop es bastante compleja, por todas las variables que se ven involucradas en el sistema como lo son: tiempos de procesamiento, *due dates*, ruta que debe cumplir cada trabajo y daños en las máquinas. Adicionalmente, en la industria real algunas de estas variables pueden tener comportamiento aleatorio, por lo que utilizar valores estáticos representa una solución en las mejores condiciones, caso que pocas veces pasa. Por esta razón, realizar una programación en la cual alguna de las variables sea dinámica o esté en constante cambio complica aún más el problema, y es ahí donde un modelo de Job shop estocástico (SJSSP por sus siglas en inglés) cobra importancia.

En este caso el objetivo es diseñar una simheurística que permita realizar la programación del JSSP teniendo en cuenta la aleatoriedad en los daños de las máquinas. Para lograrlo se diseñó una solución inicial factible, y posteriormente, con las distribuciones exponencial y Log-normal, ésta última bajo diferentes coeficientes de variación, se generaron las realizaciones de dichos tiempos, en cada corrida de la simulación de Monte Carlo, bajo un mismo valor esperado en cada máquina. La simheurística fue parametrizada en el tamaño de lista y el número de iteraciones sin mejora para los tres tipos de tamaño de instancia (pequeña, mediana y grande). Esta parametrización se realizó evaluando el mejor desempeño de la simheurística en comparación con los resultados simulados del modelo matemático (a un tiempo máximo de ejecución de 2 horas), en instancias pequeñas, y con los resultados simulados de la regla de despacho EDD en instancias grandes.

Los porcentajes promedio de mejoramiento del *maximum lateness* esperado fueron: en las instancias pequeñas, un 28% en comparación con la simulación de la mejor solución encontrada por el modelo matemático y; en las medianas y grandes un mejoramiento promedio de 19% y 10% respectivamente, en comparación con la simulación de las soluciones dadas por la regla de despacho EDD, demostrando un muy buen desempeño de la simheurística. Es importante notar que los tiempos de ejecución de la simheurística son en promedio de 7 minutos para las instancias pequeñas, 16 minutos y 30 segundos para las medianas, y 26 minutos y 40 segundos para las grandes, en tanto que el modelo en Gusek para instancias pequeñas tomó menos de 2 horas y se simuló la mejor solución encontrada hasta el momento y en instancias medianas y grandes el modelo no había obtenido ninguna solución factible luego del tiempo límite de 2 horas de ejecución.

Una vez parametrizada la simheurística y habiendo demostrado su buen desempeño, se realizaron dos tipos de pruebas. Una primera prueba para la distribución exponencial y Log-normal, ambas con coeficiente de variación 1, con el fin de determinar el efecto de dos tipos de distribuciones bajo los mismos parámetros de medias y coeficientes de variación en los tiempos entre daños y tiempos de reparación en el valor esperado de la función objetivo. Una segunda prueba, bajo la distribución Log-normal, para determinar el efecto de diferentes coeficientes de variación en el valor esperado del *maximum lateness*. En la primera prueba se concluyó que con la distribución exponencial se obtienen valores esperados menores de la función objetivo que con la distribución Log-normal, a pesar de tener el mismo coeficiente de variación y media. En la segunda prueba se concluyó que en la medida que crece el coeficiente de variación el valor esperado de la función objetivo a aumentando, pero además que la secuencia final en cada coeficiente de variación no es la misma. Estos dos resultados demuestran la importancia de ajustar adecuadamente las distribuciones de probabilidad de los tiempos entre daños y los tiempos de reparación para obtener soluciones adecuadas y además que no es adecuado solucionar el problema determinístico e implementar dicha solución.

Como limitación se tiene que, a pesar de que los tiempos de ejecución de la metaheurística son menores a una hora, la Simulación de Monte Carlo hace que el tiempo de ejecución de la simheurística sea mayor en comparación con el caso determinístico (metaheusítica), por lo cual sería importante buscar maneras más eficientes de programar dicha simulación.

Como trabajos futuros se sugiere probar diferentes tipos de solución inicial de la búsqueda tabú tales como adaptaciones de las reglas de despacho *critical ratio*, *apparent tardiness cost* o *modified due date*, para establecer cuál de ellas permite lograr un mejor desempeño. Otra recomendación importante a tener en cuenta es la implementación de otra metaheurística combinada con simulación de Monte Carlo, como podría ser algoritmo genético, GRASP o *iterated local search*. También se sugiere agregar otros parámetros con comportamiento aleatorio como los tiempos de procesamiento y los tiempos de alistamiento. Finalmente, se sugiere explorar el uso de otras distribuciones de probabilidad para modelar el comportamiento aleatorio de los tiempos entre daños y reparación de las máquinas.

9. Glosario

***Lateness* (Lj = Cj − dj):** El lapso de la terminación del trabajo con la planeada inicialmente. (Pinedo, 2016)

***Makespan* (Cmax)**: Es el tiempo equivalente hasta el momento de finalización del último trabajo para abandonar el sistema. (Pinedo, 2016)

***Tardiness* (Tj = max(Cj − dj , 0)):** Lapso de tiempo que se demora en terminar el trabajo después de cumplir su tiempo de terminación. (Pinedo, 2016) ***Preemtion* (prmp):** Implica que no es necesario mantener un trabajo en una máquina, una vez iniciado, hasta su finalización. (Pinedo, 2016)

***Due dates*:** Representa el tiempo o fecha de terminación. (Pinedo, 2016)

***Breakdowns*:** Periodos de tiempo que las máquinas no están disponibles(Pinedo, 2016)

***Slack time*:** La diferencia entre el posible último tiempo de finalización y el tiempo de finalización más pronto. (Pinedo, 2016)

Retraso máximo (Lmax): La máxima tardanza del sistema, mide la máxima violación de los tiempos de terminación.(Pinedo, 2016)

Heurísticas: un procedimiento para el que se tiene un alto grado de confianza en que encuentra soluciones de alta calidad con un coste computacional razonable, aunque no se garantice llegar al más optimo o su factibilidad, e incluso, en algunos casos, no se llegue a establecer lo cerca que se está de dicha situación. Se usa el calificativo heurístico en contraposición a exacto. (Orlando de Antonio Suárez, 2011)

Metaheurísticas: son una clase de métodos aproximados que están diseñados para resolver problemas de difícil optimización combinatoria, en los que los heurísticos clásicos no son efectivos (Orlando de Antonio Suárez, 2011)

Simheuristica: Extensión de las metaheurísticas para tratar los problemas de optimización combinatoria estocástica. (A. A. Juan et al., 2015)

Mixed integer linear programming: Es una técnica que permite modelar y resolver problemas cuya característica principal es que el conjunto de soluciones factibles es discreto.

Greedy Randomized Adaptive Search Procedure (GRASP): es una metaheurística iterativa multiarranque que en cada iteración realiza dos fases perfectamente definidas. La primera de ellas construye una solución factible al problema, adicionando uno a uno los elementos según un criterio de utilidad asignado a cada elemento. La segunda fase optimiza mediante la exploración del vecindario de soluciones hasta caer en un óptimo local. (Caballero-Villalobos & Alvarado-Valencia, 2010)

10. Referencias

Abdolrazzagh-Nezhad, M., & Abdullah, S. (2017). Job Shop Scheduling: Classification, Constraints and Objective Functionas. *World Academy of Science, Engineering and Technology*, *Vol:11*, *No*.

Abdolrazzagh, M., & Adbullah, S. (2017). Job Shop Scheduling: Classification, Constraints and Objective Funtions. *World Academy OfScience-Engeneering and Technology*, *11*,*No.4-20*.

Abido, M. A. (2002). Optimal Power Flow Using Tabu Search Algorithm. *Electric Power Components and Systems*, *30*(5), 469–483. https://doi.org/10.1080/15325000252888425

Akers, S. B., & Friedman, J. (1955). A Non-Numerical Approach to Production Scheduling Problems.

Journal of the Operations Research Society of America, *3*(4), 429–442. https://doi.org/10.1287/opre.3.4.429

Almeder, C., & Hartl, R. F. (2013). A metaheuristic optimization approach for a real-world stochastic flexible flow shop problem with limited buffer. *International Journal of Production Economics*, *145*(1), 88–95. https://doi.org/10.1016/j.ijpe.2012.09.014

Azadeh, A., Negahban, A., & Moghaddam, M. (2012). A hybrid computer simulation-artificial neural network algorithm for optimisation of dispatching rule selection in stochastic job shop scheduling problems. *International Journal of Production Research*, *50*(2), 551–566. https://doi.org/10.1080/00207543.2010.539281

Banerjee, B. P. (1965). Single Facility Sequencing with Random Execution Times. *Operations Research*, *13*(3), 358–364. https://doi.org/https://doi.org/10.1287/opre.13.3.358

Baykasoglu, A. (2002). Linguistic-based meta-heuristic optimization model for flexible job shop scheduling. *International Journal of Production Research*, *40*(17), 4523–4543. https://doi.org/10.1080/00207540210147043

Bierwirth, C., & Kuhpfahl, J. (2017). Extended GRASP for the job shop scheduling problem with total weighted tardiness objective. *European Journal of Operational Research*, *261*(3), 835–848. https://doi.org/10.1016/j.ejor.2017.03.030

Botta-Genoulaz, V. (2000). Hybrid flow shop scheduling with precedence constraints and time lags to minimize maximum lateness. *International Journal of Production Economics*, *64*(1–3), 101–111. https://doi.org/10.1016/S0925-5273(99)00048-1

Brunner, E., Dette, H., & Munk, A. (1997). Box-Type Approximations in Nonparametric Factorial Designs. *Journal of the American Statistical Association*, *92*(440), 1494–1502. https://doi.org/10.1080/01621459.1997.10473671

Caballero-Villalobos, J. P., & Alvarado-Valencia, J. A. (2010). Greedy randomized adaptive search procedure (GRASP), una alternative valiosa en la minimización de la tardanza total ponderada en una máquina. *Ingenieria y Universidad*, *14*(2), 275–295.

Chiang, P. H., & Torng, C. C. (2016). A production planning and optimisation of multi-mode job shop scheduling problem for an avionics manufacturing plant. *International Journal of Manufacturing Technology and Management*, *30*(3/4), 179. https://doi.org/10.1504/IJMTM.2016.077813

Feo, T. A., & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, *6*(2), 109–133. https://doi.org/10.1007/BF01096763

Fernandez-Viagas, V., Ruiz, R., & Framinan, J. M. (2017). A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation. *European Journal of Operational Research*, *257*(3), 707–721. https://doi.org/10.1016/j.ejor.2016.09.055

Fisher, M. L. (1973). *Optimal Solution of Scheduling Problems Using Lagrange Multipliers: Part II*. https://doi.org/10.1007/978-3-642-80784-8_20

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

Gantt, H. L. (1919). *Organizing for work*. Harcourt, Brace and Howe.

Gholami, M., Zandieh, M., & Alem-Tabriz, A. (2008). Scheduling hybrid flow shop with sequence-dependent

setup times and machines with random breakdowns. *The International Journal of Advanced Manufacturing Technology*, *42*(1–2), 189–201. https://doi.org/10.1007/s00170-008-1577-3

Gmys, J., Mezmaz, M., Melab, N., & Tuyttens, D. (2020). A computationally efficient Branch-and-Bound algorithm for the permutation flow-shop scheduling problem. *European Journal of Operational Research*, *xxxx*. https://doi.org/10.1016/j.ejor.2020.01.039

González-Martín, S., Juan, A., Riera, D., Elizondo, M., & Ramos-González, J. (2016). A Simheuristic algorithm for solving the Arc Routing Problem with Stochastic Demands. *Journal of Simulation*, *12*. https://doi.org/10.1057/jos.2016.11

Gonzalez-Neira, E. M., Ferone, D., Hatami, S., & Juan, A. A. (2017). A biased-randomized simheuristic for the distributed assembly permutation flowshop problem with stochastic processing times. *Simulation Modelling Practice and Theory*, *79*, 23–36. https://doi.org/10.1016/j.simpat.2017.09.001

Gromicho, J. A. S., Van Hoorn, J. J., Saldanha-Da-Gama, F., & Timmer, G. T. (2012). Solving the job-shop scheduling problem optimally by dynamic programming. *Computers and Operations Research*, *39*(12), 2968–2977. https://doi.org/10.1016/j.cor.2012.02.024

Gruler, A., Panadero, J., de Armas, J., Moreno Pérez, J. A., & Juan, A. A. (2018). Combining variable neighborhood search with simulation for the inventory routing problem with stochastic demands and stock-outs. *Computers & Industrial Engineering*, *123*(November 2016), 278–288. https://doi.org/10.1016/j.cie.2018.06.036

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

Hill, C. (2011). Negocios Internacionales. In *Corporate environmental orientation: Conceptualization and the case of Andean exporters* (Vol. 30). https://doi.org/10.1039/C4NJ00351A

Jackson, J. R. (1956). An extension of Johnson's results on job IDT scheduling. *Naval Research Logistics Quarterly*, *3*(3), 201–203. https://doi.org/10.1002/nav.3800030307

Jacobs, L. W., & Lauer, J. (1994). DSS for Job Shop Machine Scheduling. *Industrial Management & Data Systems*, *94*(4), 15–23. https://doi.org/10.1108/02635579410059455

Jain, A. S., & Meeran, S. (1999). Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research*. https://doi.org/10.1016/S0377-2217(98)00113-1

Johnson, S. M. (1954). Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*. https://doi.org/10.1002/nav.3800010110

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

Juan, A. A., Faulin, J., Grasman, S. E., Rabe, M., & Figueira, G. (2015). A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives*, *2*, 62–72. https://doi.org/10.1016/j.orp.2015.03.001

Juan, A. a., Grasman, S. E., Caceres-Cruz, J., & Bektaş, T. (2014). A simheuristic algorithm for the Single-Period Stochastic Inventory-Routing Problem with stock-outs. *Simulation Modelling Practice and Theory*, *46*, 40–52. https://doi.org/10.1016/j.simpat.2013.11.008

Kadipasaoglu, S. N., Peixoto, J. L., & Khumawala, B. M. (1999). Global manufacturing practices: an

empirical evaluation. *Industrial Management & Data Systems*, *99*(3), 101–108. https://doi.org/10.1108/02635579910370652

Lazar, I. (2012). Review on solving the job-shop scheduling problem: recent development and trends. *Transfer Inovacii*, *23*, 55–60.

Lei, D. (2011). Simplified multi-objective genetic algorithms for stochastic job shop scheduling. *Applied Soft Computing Journal*, *11*(8), 4991–4996. https://doi.org/10.1016/j.asoc.2011.06.001

Lei, D. M. (2012). Minimizing makespan for scheduling stochastic job shop with random breakdown. *Applied Mathematics and Computation*, *218*(24), 11851–11858. https://doi.org/10.1016/j.amc.2012.04.091

Li, J. qing, Sang, H. yan, Han, Y. yan, Wang, C. gang, & Gao, K. zhou. (2018). Efficient multi-objective optimization algorithm for hybrid flow shop scheduling problems with setup energy consumptions. *Journal of Cleaner Production*, *181*, 584–598. https://doi.org/10.1016/j.jclepro.2018.02.004

Makino, T. (1965). On a scheduling problem. *Journal of the Operations Research Society Japan*, *8*, 32–44.

Manne, A. S. (1960). On the Job-Shop Scheduling Problem. *Operations Research*. https://doi.org/10.1287/opre.8.2.219

Meloni, C., Pacciarelli, D., & Pranzo, M. (2004). A Rollout Metaheuristic for Job Shop Scheduling Problems. *Annals of Operations Research*, *131*(1), 215–235. https://doi.org/10.1023/B:ANOR.0000039520.24932.4b

Meng, L., Zhang, C., Ren, Y., Zhang, B., & Lv, C. (2020). Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem. *Computers and Industrial Engineering*, *142*(February), 106347. https://doi.org/10.1016/j.cie.2020.106347

Mohan, J., Lanka, K., & Rao, A. N. (2019). A review of dynamic job shop scheduling techniques. *Procedia Manufacturing*, *30*, 34–39. https://doi.org/10.1016/j.promfg.2019.02.006

Naderi, B., Tavakkoli-Moghaddam, R., & Khalili, M. (2010). Electromagnetism-like mechanism and simulated annealing algorithms for flowshop scheduling problems minimizing the total weighted tardiness and makespan. *Knowledge-Based Systems*, *23*(2), 77–85. https://doi.org/10.1016/j.knosys.2009.06.002

Oddi, A., Rasconi, R., Cesta, A., & Smith, S. F. (2010). Job shop scheduling with setup times: Exploring the applicability of a constraint-based iterative sampling approach. *CEUR Workshop Proceedings*, *616*(Rcra), 1–15.

Olguin Tiznado, J. E., Garcia Martinez, R., Camargo Wilson, C., & Lopez Barreras, J. A. (2011). Stochastic approximation algorithm for industrial process optimisation. *Ingenieria e Investigacion*, *31*(3), 100–111.

Orlando de Antonio Suárez. (2011). Una aproximación a la heuristica y metaheuristicas. *Universidad Antonio Nariño*, 8.

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

Pranzo, M., & Pacciarelli, D. (2016). An iterated greedy metaheuristic for the blocking job shop scheduling problem. *Journal of Heuristics*, *22*(4), 587–611. https://doi.org/10.1007/s10732-014-9279-5

Roshan, S., Jooibari, M., Teimouri, R., Asgharzadeh-Ahmadi, G., Falahati-Naghibi, M., & Sohrabpoor, H. (2013). Optimization of friction stir welding process of AA7075 aluminum alloy to achieve desirable

mechanical properties using ANFIS models and simulated annealing algorithm. *The International Journal of Advanced Manufacturing Technology*, *69*. https://doi.org/10.1007/s00170-013-5131-6

Shahsavari-Pour, N., & Ghasemishabankareh, B. (2013). A novel hybrid meta-heuristic algorithm for solving multi objective flexible job shop scheduling. *Journal of Manufacturing Systems*, *32*(4), 771–780. https://doi.org/10.1016/J.JMSY.2013.04.015

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, *3*(1–2), 59–66. https://doi.org/10.1002/nav.3800030106

Su, L. H., Chang, P. C., & Lee, E. S. (1998). A heuristic for scheduling general job shops to minimize maximum lateness. *Mathematical and Computer Modelling*, *27*(1), 1–15. https://doi.org/10.1016/S0895-7177(97)00250-1

Vilcot, G., & Billaut, J. C. (2008). A tabu search and a genetic algorithm for solving a bicriteria general job shop scheduling problem. *European Journal of Operational Research*, *190*(2), 398–411. https://doi.org/10.1016/j.ejor.2007.06.039

Vinod, V., & Sridharan, R. (2011). Simulation modeling and analysis of due-date assignment methods and scheduling decision rules in a dynamic job shop production system. *International Journal of Production Economics*, *129*(1), 127–146. https://doi.org/10.1016/j.ijpe.2010.08.017

Zhang, J., Ding, G., Zou, Y., Qin, S., & Fu, J. (2019). Review of job shop scheduling research and its new perspectives under Industry 4.0. *Journal of Intelligent Manufacturing*, *30*(4), 1809–1830. https://doi.org/10.1007/s10845-017-1350-2