

[193019] Diseño de una metaheurística GRASP en un ambiente *Flow Shop* determinístico biobjetivo.

Kerly Gisell Galindo Medina^{a,c}, Jennifer Alejandra Ramos Fajardo^{a,c}, María Juliana Yunez Charry^{a,c},

Ing. Eliana María González Neira, PhD^{b,c},

^aEstudiante de Ingeniería Industrial

^bProfesor, Director del Proyecto de Grado, Departamento de Ingeniería Industrial

^cPontificia Universidad Javeriana, Bogotá, Colombia

Abstract

Scheduling is an analytic tool for making decisions that has acquired an important role in manufacturing and services industries. Is the solid foundation of industrial development. For this study, we propose to solve the bi-objective scheduling problem in a deterministic *Blocking Permutation Flow Shop* (BPFS), that minimizes the *makespan* and the total *tardiness*. Flow Shop environments are present in many industries such as chemistry, petrochemicals, assembly, food, circuits, among others. In addition, the analysis of Blocking (due to the limited buffer) and the minimization of two objectives simultaneously make this application closer to real environments. *Makespan* seeks a better use of machines and *Tardiness* a higher level of customer services. In this proposal, GRASP metaheuristic will be considered as a multiobjective solution with two different approaches: *Pareto Archived Evolution Strategy* (PAES), to find the Pareto border between the two objectives and the lexicographical order. The results show that the execution time is a decisive factor to define the approach of the objective function, taking in count that this is a NP-Hard problem. For this experiment, the largest instances tend to get better results with the lexicographic ordering while the small ones find it with GRASP-PAES.

Keywords: Scheduling, Flow Shop (FS), GRASP, makespan, tardiness, permutation, blocking, Pareto border, lexicographical order, PAES.

1. Justificación y planteamiento del problema

El tiempo es uno de los recursos más valiosos para las empresas pues es escaso y agotable. La programación de la producción o *scheduling* surge para aprovechar la cantidad limitada de tiempo (Wiers, 1997). En épocas pasadas, las empresas y fábricas no tenían un control preciso sobre su producción, en cuanto a tiempos de procesamiento, tiempos de ciclo, tiempos de inicialización y finalización de los trabajos, cómo debían realizarse los mismos, niveles de prioridad, tardanzas en los pedidos, entre otros. De esta manera, la programación de la producción (o *scheduling* en inglés) se define como la asignación de recursos disponibles a lo largo del tiempo para realizar un conjunto de tareas (Clewett & Baker, 1977).

Sin embargo, hablar de programación de la producción implica considerar la configuración del ambiente productivo que se analizará. Los problemas de programación de la producción se pueden clasificar de acuerdo con la configuración de las máquinas en el taller en: una máquina (*single machine*), máquinas paralelas (*parallel machine*), *flow shop* (FS), *job shop* y *open shop*. Un FS consiste en m máquinas en serie, por las cuales deben pasar n trabajos en el mismo orden. Es decir, primero a la máquina 1, luego a la máquina 2 y, así sucesivamente, hasta la máquina m . Esto se traduce en un espacio de soluciones de $n!^m$. Un caso especial del FS es cuando la secuencia de procesamiento de todos los trabajos en todas las operaciones es la misma, lo cual se denomina *permutation flow shop* (PFS), reduciendo el espacio de soluciones a $n!$. Estos problemas son aplicados a industrias como la química, petroquímica, acero inoxidable, ensamble de circuitos electrónicos, ensamble de automóviles, entre otras (Anjana, Sridharan, &

Ram Kumar, 2019; Hosseini & Tavakkoli-Moghaddam, 2013). Se ha demostrado que el PFS para 3 o más máquinas con la minimización del *makespan* es *NP-hard* (Garey, Johnson, & Sethi, 1976).

De acuerdo con la revisión de la literatura de los últimos años, realizada por el equipo de trabajo, se puede observar que aproximadamente, el 40% de los artículos estudia la minimización del *makespan* como función objetivo, mientras que el *flow time* cuenta con 12% y el *tardiness* con 6%. En cuanto a funciones multiobjetivo, el 5% del total minimiza *makespan* y *flow time*, mientras que el 1% hace referencia al *makespan* y *tardiness*. Por otro lado, en cuanto a los fenómenos presentados, se demuestra que, alrededor del 23% de las investigaciones se enfatiza en *permutation*, el 12% se refiere a *blocking* y el 11% a *no-wait*. Finalmente, en cuanto a los métodos de solución, se exhibe que, a grosso modo, el 45,5% de los documentos utiliza heurísticas, el 46,42% emplea metaheurísticas y el 8,08% ejecuta métodos exactos. Es por esto que, para el presente estudio, se ha seleccionado el “*blocking permutation flow shop scheduling problem*” (BPFS), con función multiobjetivo que permita minimizar el valor esperado del *makespan* y de la tardanza total (*tardiness*).

La característica de bloqueo (*blocking*) se puede presentar cuando existe una capacidad limitada de almacenamiento entre dos estaciones, lo cual es común en la industria, aún más cuando las operaciones del proceso productivo se encuentran ubicadas en serie, como corresponde al ambiente productivo FS. Supongamos que el almacenamiento (*buffer*) entre la máquina *i* y la máquina *i + 1* ha colmado su capacidad de trabajos en espera para la máquina *i + 1*. Entonces, si la máquina *i* acaba de terminar de procesar un trabajo, no lo puede pasar al buffer dado que este está lleno. Por lo tanto, el trabajo debe quedarse en la máquina *i* esperando que en el buffer se libere una posición, implicando que dicha máquina no puede recibir el siguiente trabajo que debe procesar. Es decir que se bloquea el procesamiento de la máquina *i*.

Ahora bien, en cuanto a funciones objetivo, se evidencia que pocas investigaciones se enfocan en problemas multiobjetivo. Sin embargo, es común que en la industria se manejen varios objetivos que suelen ir en contravía, por lo que el estudio de estos problemas ha adquirido gran relevancia (Yenisey & Yagmahan, 2014). Diversas investigaciones han abordado los problemas multiobjetivo con enfoques a priori o a posteriori. Los enfoques a priori son aquellos en los que se asignan pesos a cada objetivo. Los enfoques a posteriori involucran un conjunto de soluciones no dominadas, o soluciones óptimas de Pareto (Pasupathy, Rajendran, & Suresh, 2006). Para encontrar las soluciones no dominadas (frontera de Pareto) es adecuado utilizar algoritmos como el *Pareto Archived Evolution Strategy* (PAES) propuesta por (Knowles & Corne, 2000) cuando se utilizan metaheurísticas de tipo constructivo, o técnicas como MOGA o NGSa -II para algoritmos evolutivos como el algoritmo genético (Minella, Ruiz, & Ciavotta, 2008).

Por otra parte, considerando que los PFS son NP-duro (*NP-hard*), la solución con métodos exactos sólo tiene sentido para instancias pequeñas, ya que para instancias medianas y grandes los tiempos computacionales de resolución no serían razonables. Es por ello que se avala el uso de procedimientos heurísticos y metaheurísticos para resolver este tipo de problemas. Como se observó en la revisión de literatura realizada por el grupo, la metaheurística *GRASP*, propuesta por (Resende, 1998), ha sido poco utilizada para resolver problemas FS, pero, así mismo, ha demostrado obtener resultados altamente competitivos para estos, cubriendo todo el frente de Pareto y obteniendo una buena distribución del espacio objetivo (Arroyo & De Souza Pereira, 2011).

Teniendo en cuenta que: i) existen pocos trabajos en la literatura que hayan estudiado el PFS con bloqueo, situación que es común en la industria; ii) pocos estudios han considerado múltiples objetivos; iii) la entrega a tiempo a los clientes es un indicador de nivel de servicio y; iv) la metaheurística *GRASP* ha obtenido buenos resultados para problemas de programación de la producción, entonces la pregunta de investigación es: ¿cómo diseñar una metaheurística *GRASP* para minimizar la tardanza total y el *makespan*, comparando la frontera de Pareto y el método de ordenamiento lexicográfico, en un PFS con bloqueo?

2. Antecedentes

Dado que el problema a resolver es la minimización de la tardanza y el *makespan* en un ambiente FS, se realizó una revisión sistemática de la literatura utilizando como palabras clave *Flowshop* “OR” *Flow Shop* “AND” *Scheduling Problems*, para los años 2015 a 2019 y *Flowshop Multiobjective*, de 2009 a 2019, en las bases de datos Proquest y Scopus. Se consideraron como criterios de exclusión que: el artículo no sea exclusivamente de programación de la producción, el documento esté incompleto o que el título contenga conceptos como *Cyclic hybrid flow shop*, *Hybrid flow shop*, *Dynamic flexible*, *Interval shop*, *Parallel Machines*, *Concurrent Scheduling* o *Recirculation*. De esta forma, se estudiaron 113 artículos para posicionar la presente investigación respecto al estado del arte.

Así mismo, es importante presentar dos escenarios que, normalmente, hacen referencia a uno de los fenómenos más comunes dentro de un problema de FS. El primero, considera las capacidades del *buffer* entre máquinas consecutivas como “ilimitadas”, dado que el tamaño de los productos que se están trabajando es pequeño. Esto hace posible que se almacenen grandes cantidades de producto entre máquinas. Por otra parte, el segundo escenario aparece cuando los productos son físicamente grandes, lo que disminuye la capacidad del *buffer* entre dos máquinas, limitándola y teniendo como posible consecuencia un bloqueo. Este fenómeno ocurre cuando el *buffer* se encuentra lleno y la máquina no puede liberar un trabajo después de completar su procesamiento (Pinedo, 2016).

Por otro lado, muchos de estos escenarios trabajan con el criterio del *makespan* que es equivalente al tiempo de finalización del último trabajo para dejar el sistema (Pinedo, 2016). Sin embargo, el concepto ha sido ampliamente estudiado y se evidencia una necesidad de complementarlo para obtener como resultado un modelo multiobjetivo. De esta forma, se introduce el concepto de tardanza (*tardiness*), referente al tiempo que se tarda en entregar un pedido, después de su fecha de vencimiento (*due date*).

Con base en lo anterior, es de destacar que, al dar solución a un problema multiobjetivo, se genera un mayor impacto ya que cada una de las tareas contribuye a los componentes de dicho objetivo. Durante la última década, se han desarrollado diversos estudios resolviendo problemas multiobjetivo. (Sha & Hung Lin, 2009) afirman que la mayoría de los estudios de programación de FS se encuentran centrados en problemas con un único objetivo que podría ser optimizado de forma independiente. Además, las decisiones de programación no sólo implican considerar más de un objetivo, sino también de minimizar el conflicto entre dos o más. Es así como se evidencia el desarrollo de enfoques que incluyen metaheurísticas y la memética para reducir la complejidad y la eficiencia de las soluciones. La literatura sobre la programación de FS con múltiples objetivos se puede dividir en dos grupos: los enfoques a priori con pesos asignados a cada objetivo; y los enfoques a posteriori que involucran un conjunto de soluciones no dominadas (Pasupathy et al., 2006).

Al trabajar con múltiples objetivos, se incrementa la dificultad del problema por lo que muchos estudios se enfocan en métodos heurísticos o metaheurísticos, entre los que se encuentran, principalmente, el algoritmo genético (GA) con el que se han evidenciado resultados exitosos en términos de soluciones eficientes a gran escala (Hosseini & Tavakkoli-Moghaddam, 2013). También, está el *simulated annealing* (SA) que corresponde a un procedimiento de búsqueda de vecindario, el cual ha obtenido buenos resultados para problemas *NP-hard*, y se ha aplicado de manera efectiva y eficiente en problemas de minimización de tardanza (Dhingra & Chandna, 2010). A su vez, se encuentra el *particle swarm optimization* (PSO) que es utilizado para problemas de optimización continuos sin restricciones y ha sido empleado en gran variedad de problemas multiobjetivo en FS gracias a su fácil implementación, convergencia rápida y que requiere de pocos parámetros (Liu, Zhao, Ou'Yang, & Yang, 2011). Finalmente, está el procedimiento de búsqueda adaptativa aleatoria (GRASP) para funciones multiobjetivo (MO-GRASP) que realiza una búsqueda de buenas soluciones no dominadas y se guía por la optimización de una función de utilidad lineal ponderada, la cual se ejecuta iterativamente con diferentes escalarizaciones ponderadas y con diferentes variaciones de los vectores de peso, para cubrir todo el frente de Pareto y obtener una buena distribución en el espacio objetivo (Arroyo & De Souza Pereira, 2011).

Teniendo en cuenta lo anterior, para estos problemas que plantean funciones multiobjetivo, se ha comprobado que existen diversas técnicas para abordarlas como la frontera de Pareto, la regla de ordenamiento lexicográfico, entre otras. En la frontera de Pareto se trabajan los objetivos simultáneamente, seleccionando las soluciones no dominadas. Por su parte, el ordenamiento lexicográfico permite evaluar los objetivos uno a uno, según su importancia absoluta (Ciatera Díaz, 2015).

Gracias a la revisión de literatura realizada por el equipo de trabajo, se evidencian algunas investigaciones que abordan la función multiobjetivo a partir del método de frontera de Pareto. Así, (Hanoun & Nahavandi, 2012), para la minimización del costo del material que se gasta como criterio dominante y la tardanza, proponen una heurística *greedy* junto con un algoritmo SA para cada objetivo, respectivamente. Por otra parte, el indicador que se emplea como criterio de referencia corresponde al RPD (desviación porcentual relativa). Por su parte (Hosseini & Tavakkoli-Moghaddam, 2013) minimizan el *idle time* total y la desviación media de los datos comunes con tiempos de llegada dinámicos. Para esto, se presentan dos algoritmos: MOGA y MOSA. Dado que se pueden obtener soluciones similares a partir de los algoritmos SA y GA, se decide realizar la modificación para aplicarlos a un problema multiobjetivo y, finalmente, compararlos. También, se encontraron investigaciones con condiciones similares a las del problema propuesto en este proyecto. (Lebbar, El Abbassi, Jabri, El Barkany, & Darcherif, 2018) proponen un FSP con permutación y bloqueo, para minimizar el *makespan* y el tiempo de finalización total, al cual se le da solución por medio de un algoritmo genético basado en NGSa-II, que utiliza un enfoque de búsqueda basado una población en el que más de una solución participa en una iteración y se desarrolla una nueva población de soluciones en cada

ejecución. A su vez, (Anjana et al., 2019) implementan cuatro metaheurísticas de algoritmo genético de clasificación no dominado (NSGA II, NSGA II híbrido, DPSO y DPSO híbrido), para la minimización del valor del *makespan* y la tardanza media. Estas técnicas multiobjetivo son eficientes, ya que optimizan los objetivos de un problema de manera simultánea sin ser influenciadas por alguna solución adicional y sin restricción alguna. Desde otro punto, (Guanlong, Shuning, & Mei, 2016) presentan un optimizador de búsqueda de grupo discreto para el *multiobjective permutation flow shop problem* (MOPFSP) con el fin de minimizar el *makespan* y el *flowtime*. Esto fue desarrollado por medio de un diagrama de Gantt de BFSP con cuatro trabajos y tres máquinas. Luego, se implementó el MDGSO considerando tres roles con un mecanismo de actualización específico. Los métodos de solución se encuentran basados en la heurística *Nawas - Enscore - Ham* (NEH) y similares. Finalmente, (Shao, Pi, & Shao, 2019) tienen en estudio un *multiobjective blocking flow shop problem* (MOBFSP) el cual se soluciona con un algoritmo de optimización multiobjetivo discreto de ondas de agua MODWWO para minimizar el *flowtime* total y temporal, de manera simultánea.

En cuanto a ordenamiento lexicográfico, (Yang & Liu, 2018) proponen un algoritmo híbrido de optimización *Gray Wolf* multiobjetivo para un BFS con el fin de minimizar el factor difuso y maximizar el índice de acuerdo con el promedio. Este algoritmo emplea un mecanismo exhaustivo para la selección del líder y el proceso de búsqueda está guiado por α , β y δ , que son seguidos concomitantemente por el ω lobos. Por su parte, (Caio Soares de Araújo & Fuchigami, 2018), para el PFS con tiempos de configuración independientes y diferentes fechas de lanzamiento del trabajo, proponen un modelo de programación lineal entero mixto en donde, primero, se minimiza el *makespan* y, luego, el *flow time*, analizando la dependencia entre las variables involucradas en el problema y la trascendencia de los criterios.

Por último, en otros estudios, se han utilizado métodos de abordaje distintos como la técnica de ponderación para trabajar las funciones objetivo. (Dhingra & Chandna, 2010) proponen una solución para problemas multiobjetivo de FS, específicamente para minimizar el tiempo de alistamiento que dependen de la secuencia (SDST, por sus siglas en inglés), y, por otra parte, minimizar la suma de la tardanza ponderada y el *makespan* de manera simultánea. Para la solución de este, los autores propusieron seis algoritmos de *Hybrid Simulating Annealing* (HSA) basados en NEH, los cuales fueron modificados para la programación eficiente en un FS multiobjetivo con SDST. Para esto, se realizaron pruebas con problemas de hasta 200 tareas con 20 máquinas y se utilizó un índice de mejora porcentual, el cual fue definido para hacer la comparación de los diferentes algoritmos propuestos. Por otra parte, (Lebbar et al., 2018) proponen y evalúan el rendimiento computacional del modelo MILP para BFSP multimáquinas con fechas de lanzamiento de trabajos, donde el objetivo es minimizar la suma ponderada de tardanza máxima y el *makespan*. Los autores utilizaron métodos que consisten en asignar un conjunto finito de trabajos que se van a realizar, secuencialmente, en un conjunto de máquinas en el mismo enrutamiento de proceso.

En conclusión, posterior a la revisión de la literatura, se evidencia que los problemas multiobjetivo estudiados, en su mayoría, dan solución al PFS, para diferentes casos. Por el contrario, el caso con menos información es el BPFS con función multiobjetivo referente a minimizar el valor esperado del *makespan* y del *tardiness*. Adicionalmente, se evidencia la carencia de la utilización de la metaheurística GRASP como método de solución a este tipo de problemas, a pesar de obtener resultados ampliamente competitivos y eficientes. Es así como se observa la viabilidad para desarrollar este estudio entorno a un problema de PFS con fenómeno *blocking*, contrastando la frontera de Pareto con la regla de ordenamiento lexicográfico para abordar la función objetivo.

3. Objetivos

Diseñar una metaheurística multiobjetivo GRASP para solucionar BPFS, que considera como criterios a minimizar el makespan y la tardanza total.

- Plantear el modelo matemático del problema BPFS.
- Diseñar e implementar la metaheurística GRASP para obtener la solución óptima teniendo en cuenta el método de ordenamiento lexicográfico para abordar la función biobjetivo.
- Diseñar e implementar la metaheurística GRASP hibridizada con el algoritmo PAES para obtener las fronteras de Pareto entre el *makespan* y la tardanza total en el problema BPFS.
- Evaluar el desempeño de la metaheurística GRASP-Pareto y GRASP-Lexicográfico contra el modelo matemático para instancias pequeñas.
- Evaluar el desempeño de la metaheurística GRASP-Pareto y GRASP-Lexicográfico contra el mejor resultado obtenido por el modelo matemático después de dos horas de ejecución, para instancias medianas y grandes.

- Comparar el abordaje de la función biobjetivo a través de la frontera de Pareto en contraste con el ordenamiento lexicográfico.

4. Metodología

4.1. Modelamiento Matemático

En esta sección, se presentan los modelos matemáticos del BPFs para el caso de ordenamiento lexicográfico. Dado que se quieren presentar ambos ordenamientos lexicográficos (minimizar *makespan* después de haber minimizado tardanza *MinMakespan/MinTardiness* y minimizar la tardanza dada la minimización del *makespan* *MinTardiness/MinMakespan*), se hace necesario plantear cuatro modelos matemáticos: dos modelos uniobjetivo individuales con las mismas restricciones propias del BPFs (uno que minimiza la tardanza y otro que minimiza el *makespan*) y, luego, dos modelos a los que se adiciona como restricción la no superación de la tardanza y del *makespan* respectivamente, obtenidos de los dos modelos individuales planteados en primera instancia, modificando la función objetivo. Es decir, en el modelo lexicográfico *MinMakespan/MinTardiness*, se minimiza en primera instancia la tardanza y, posteriormente, ésta se pone como restricción adicional a la minimización del *makespan*. En el segundo modelo lexicográfico *MinTardiness/MinMakespan*, se minimiza primero el *makespan* y, luego, éste se pone como restricción a la minimización de la tardanza.

La notación utilizada en todos los modelos es la siguiente:

Para los conjuntos, se determinan:

- I: Máquinas {1..M}
- J: Trabajos {1..N}
- K: Posiciones de la secuencia {1..N}

Dentro de los parámetros, se encuentran:

- D_j : *Due dates* de los trabajos.
- P_{ij} : Tiempos de procesamiento de cada trabajo en cada máquina.
- M: Número muy grande.

En cuanto a las variables de decisión, se establecen:

- X_{jk} : {1 Si el trabajo es procesado en esa posición; 0 de lo contrario}
- C_{ik} : Tiempo de terminación de un trabajo en una posición.
- T_j : Tardanza de un trabajo.
- H_{ik} : Tiempo de inicio de un trabajo en una posición y en una máquina.
- F_j : Tiempo de finalización de un trabajo.
- CMax: Makespan.
- TT: Tardanza total.

Así, el modelo matemático individual de *makespan* es el siguiente:

Función Objetivo:

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

Restricciones:

$$\sum_{j=1}^n X_{j,k} = 1; \quad \forall k \in K \quad (2)$$

$$\sum_{k=1}^n X_{j,k} = 1; \quad \forall j \in J \quad (3)$$

$$Cmax \geq C_{i,k}; \quad \forall i \in I, \forall k \in K \quad (4)$$

$$C_{i,k} \geq C_{i+1,k-1}; \quad \forall i \in I, k \in K, k > 1 \quad (5)$$

$$H_{i,k} \geq C_{i-1,k}; \quad \forall i \in I, k \in K \quad (6)$$

$$C_{i,k} \geq H_{i,k} + \sum_{j=1}^n P_{i,j} * X_{j,k}; \quad \forall i \in I, k \in K \quad (7)$$

$$H_{i,k} \geq C_{i,k-1}; \forall i \in I, \forall k \in K, k > 1 \quad (8)$$

$$C_{i,k} \geq H_{i+1,k}; \forall i \in I, \forall k \in K \quad (9)$$

$$X_{j,k} \in \{0,1\}; \forall j \in J, \forall k \in K \quad (10)$$

$$C_{i,k} \geq 0; \forall i \in I, \forall k \in K \quad (11)$$

$$H_{i,k} \geq 0; \forall i \in I, \forall k \in K \quad (12)$$

$$Cmax \geq 0 \quad (13)$$

Para el caso de minimización de *makespan*, la función objetivo (1) corresponde a minimizar dicha variable. Para el caso de las restricciones, en la (2) se asegura que cada posición de la secuencia debe procesar sólo un único trabajo, así como en la (3), en donde cada trabajo se debe procesar en una única posición de la secuencia. A su vez, la restricción (4) garantiza que, para cada máquina y para cada posición de la secuencia, el *makespan* sea mayor o igual que el tiempo de terminación del trabajo. Además, la restricción (5) garantiza que el tiempo de terminación de un trabajo sea mayor al tiempo en que termina el trabajo de la posición anterior, en la siguiente máquina. En la restricción (6), garantiza que el tiempo de inicio de un trabajo en una posición sea mayor o igual al tiempo de terminación del trabajo en la máquina anterior, para cada máquina y para cada posición de la secuencia, desde la máquina 2 en adelante y, en la restricción (7), se asegura que, para cada máquina y para cada posición de la secuencia, el tiempo de terminación de un trabajo, en una posición, en una máquina, es igual a la suma de su tiempo de inicio y el tiempo de procesamiento, si ese trabajo se realiza en esa posición. Finalmente, en la restricción (8) se garantiza que, para cada máquina y para cada posición en la secuencia, el tiempo de inicio de un trabajo debe ser mayor o igual al tiempo de terminación del trabajo de la posición anterior en una máquina, mientras que la restricción (9) avala que el tiempo de terminación de un trabajo, en una posición y en una máquina, debe ser mayor o igual al tiempo de inicio de ese trabajo, en la misma posición, en la siguiente máquina. [A través de estas dos restricciones, se garantiza que el buffer no es contemplado en el modelo, es decir que es igual a cero, permitiendo que no exista bloqueo pues un trabajo inicia su procesamiento tan pronto esté disponible la siguiente máquina. Este funcionamiento aplica tanto para los modelos matemáticos, como para las metaheurísticas.](#) Por último, en las restricciones finales, de la (10) a la (13), se garantiza la no negatividad de las variables de decisión.

Por su parte, el modelo matemático individual de tardanza es el siguiente:

Función Objetivo:

$$Min Z = \sum_{j=1}^n T_j \quad (14)$$

Restricciones:

$$\sum_{j=1}^n X_{j,k} = 1; \forall k \in K \quad (15)$$

$$\sum_{k=1}^n X_{j,k} = 1; \forall j \in J \quad (16)$$

$$Cmax \geq C_{i,k}; \forall i \in I, \forall k \in K \quad (17)$$

$$H_{i,k} \geq C_{i,k-1}; \forall i \in I, \forall k \in K \quad (18)$$

$$C_{i,k} \geq C_{i+1,k-1}; \forall i \in I, \forall k \in K, k > 1 \quad (19)$$

$$H_{i,k} \geq C_{i-1,k}; \forall i \in I, \forall k \in K \quad (20)$$

$$C_{i,k} \geq H_{i,k} + \sum_{j=1}^n P_{i,j} * X_{j,k}; \forall i \in I, \forall k \in K \quad (21)$$

$$H_{i,k} \geq C_{i,k-1}; \forall i \in I, \forall k \in K, k > 1 \quad (22)$$

$$C_{i,k} \geq H_{i+1,k}; \forall i \in I, \forall k \in K \quad (23)$$

$$F_j \geq C_{i,k} - M * (1 - X_{j,k}); \forall j \in J, \forall k \in K \quad (24)$$

$$F_j \leq C_{i,k} + M * (1 - X_{j,k}); \forall j \in J, \forall k \in K \quad (25)$$

$$T_j \geq F_j - D_j; \forall j \in J, \quad (26)$$

$$X_{j,k} \in \{0,1\}; \forall j \in J, \forall k \in K \quad (27)$$

$$C_{i,k} \geq 0; \forall i \in I, \forall k \in K \quad (28)$$

$$T_j \geq 0; \forall j \in J \quad (29)$$

$$H_{i,k} \geq 0; \forall i \in I, \forall k \in K \quad (30)$$

$$F_j \geq 0; \forall j \in J \quad (31)$$

$$Cmax \geq 0 \quad (32)$$

$$TT \geq 0 \quad (33)$$

En este caso, la función objetivo (14) corresponde a minimizar la tardanza total. Desde esta manera, el modelo mantiene el funcionamiento igual al del modelo de minimización de *makespan*, es decir, desde la restricción (15) hasta la (23), y se añaden tres restricciones correspondientes a garantizar que el tiempo de finalización de un trabajo sea mayor o igual al tiempo de terminación de ese trabajo, en una máquina y en una posición, referente a la restricción (24). Así, cuando el trabajo no se realiza en esa posición (X_{jk}), no se asigna tiempo de terminación. Además, a través de la restricción (25) se avala que el tiempo de finalización de un trabajo sea el tiempo de terminación de ese trabajo, en una máquina y en una posición y, por lo tanto, cuando X_{jk} es igual a 1, el tiempo de finalización se limita a ser el mismo tiempo de finalización de la máquina, para ese trabajo, en esa posición. Finalmente, la restricción (26) garantiza que la tardanza total de un trabajo sea mayor o igual a la diferencia entre el tiempo de terminación del trabajo y su *due date*. Por último, en las restricciones finales, de la (27) a la (33) se garantiza la no negatividad de las variables de decisión.

En consecuencia, a partir de los resultados óptimos de *tardiness* y *makespan* que se obtengan de cada instancia para los modelos individuales, se define el valor de la restricción en los modelos lexicográficos con restricción asociada *MinMakespan/MinTardiness* y *MinTardiness/MinMakespan*, respectivamente. En el modelo *MinMakespan/MinTardiness*, se incluyen las restricciones referentes al tiempo de finalización de cada trabajo (F_j), al cálculo de la tardanza de cada trabajo (T_j) y a la tardanza total (TT), de tal manera que sea posible limitar la tardanza según el Z obtenido en el modelo individual de *tardiness*. Por su parte, para el modelo *MinTardiness/MinMakespan*, se coloca como restricción adicional el *makespan*. Estos modelos se presentan a continuación.

El modelo matemático *MinMakespan/MinTardiness*, contempla las mismas restricciones del modelo individual de *makespan* (ecuaciones (2) a la (13)), agregando cinco restricciones adicionales que son las ecuaciones (35) a la (39):

Función Objetivo:

$$\text{Min } Z = C_{max} \quad (34)$$

Restricciones:

$$F_j \geq C_{i,k} - M * (1 - X_{j,k}); \quad \forall j \in J, \forall k \in K \quad (35)$$

$$F_j \leq C_{i,k} + M * (1 - X_{j,k}); \quad \forall j \in J, \forall k \in K \quad (36)$$

$$T_j \geq F_j - D_j; \quad \forall j \in J \quad (37)$$

$$TT = \sum_{j=1}^n T_j; \quad \forall j \in J \quad (38)$$

$$TT \leq Z \text{ de la FO de MinTardanza} \quad (39)$$

Para el caso de minimización de *makespan*, la función objetivo (34) corresponde a minimizar dicha variable. En cuanto a las restricciones adicionales, a través de la (35), se garantiza que el tiempo de finalización de un trabajo sea mayor o igual al tiempo de terminación de ese trabajo, en esa máquina y en esa posición. Así, cuando el trabajo no se realiza en esa posición (X_{jk}), no se asigna tiempo de terminación, mientras que en la restricción (36), se avala que el tiempo de finalización de un trabajo sea el tiempo de terminación de ese trabajo, en una máquina y en una posición y, por lo tanto, cuando X_{jk} es igual a 1, el tiempo de finalización se limita a ser el mismo tiempo de finalización de la máquina, para ese trabajo, en esa posición. A través de la restricción (37), se calcula la tardanza de cada trabajo, mientras que con la restricción (38), se calcula la tardanza total. Finalmente, en la restricción (39), se asegura que el valor de la tardanza total calculada sea menor o igual que el valor de la tardanza en el modelo individual (Z), previamente realizado, es decir, menor al valor máximo de la restricción.

Por otro lado, el modelo matemático *MinTardiness/MinMakespan* contempla las mismas restricciones del modelo individual de *tardiness* (ecuaciones (15) a (33)), agregando una restricción adicional correspondiente a:

Función Objetivo:

$$\text{Min } Z = \sum_{j=1}^n T_j \quad (40)$$

Restricciones:

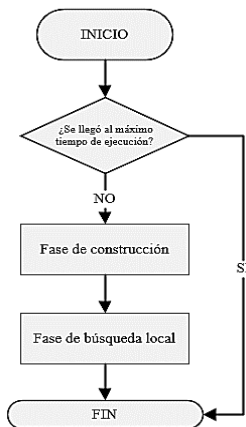
$$C_{max} \leq Z \text{ de la FO de MinMakespan} \quad (41)$$

En este caso, la restricción (41), se asegura que el valor del *makespan* calculado sea menor o igual que el valor del *makespan* en el modelo individual (Z), previamente realizado, es decir, menor al valor máximo de la restricción.

4.2. Metaheurística GRASP

La metaheurística GRASP es un algoritmo voraz e iterativo, desarrollado por los autores (Feo & Resende, 1995), en donde se proporciona una solución por cada iteración del problema la cual es comparada con la mejor solución obtenida hasta el momento. Esta, se encuentra compuesta por dos fases: la fase de construcción y la fase de búsqueda local. La primera fase corresponde a la construcción de la solución inicial haciendo uso de una función de utilidad (*greedy function*), que se usa para medir la contribución de cada trabajo candidato a la solución parcial (no secuenciado aún). Una vez se calcula la función de utilidad, se construye una lista de trabajos restringida (RCL, por sus siglas en inglés) que contiene los trabajos con el $\alpha\%$ de los mejores valores de función de utilidad. De esta RCL, se selecciona aleatoriamente un trabajo para ser adicionado a la secuencia. Este procedimiento se sigue hasta que todos los trabajos sean secuenciados. En la segunda fase, se desarrolla un algoritmo de búsqueda local que permite llegar a un óptimo local partiendo de la solución obtenida en la fase de construcción. Esto implica definir un vecindario de búsqueda e iterar en esa búsqueda hasta que se cumpla el criterio de parada seleccionado. Para el caso de esta investigación, se ha seleccionado como criterio de parada un tiempo máximo de ejecución que dependerá del tamaño de la instancia, y se define como $t_{max} = t \times \text{número de trabajos} \times \text{número de máquinas}$, en donde t_{max} es el tiempo máximo de ejecución del algoritmo, en segundos, y t es un factor multiplicativo, en segundos. El valor de t será parametrizado, posteriormente, en el tuneo de la metaheurística. En la Figura 1. Funcionamiento General Metaheurística GRASP., se detalla el funcionamiento de la metaheurística. Para visualizar el diagrama con mayor detalle, ver Anexo 1.

Figura 1. Funcionamiento General Metaheurística GRASP.



Considerando que la presente investigación corresponde a un BPFS determinístico biobjetivo, con dos abordajes distintos de la función objetivo –regla de ordenamiento lexicográfico y frontera de Pareto–, es necesario explicar la metaheurística para cada uno de esos enfoques, las cuales se presentan a continuación.

4.2.1. Metaheurística GRASP para la solución de un BPFS biobjetivo a través de la regla de ordenamiento lexicográfico

Como se menciona anteriormente, la regla de ordenamiento lexicográfico para dos objetivos propone la minimización por prioridad de los mismos, en donde el primer objetivo que se minimiza se convierte en la restricción del segundo objetivo. De esta manera, se realizan cuatro (4) metaheurísticas, correspondientes a minimización de *makespan*, minimización de *tardiness* y las de restricciones asociadas *MinMakespan/MinTardiness* y *MinTardiness/MinMakespan*:

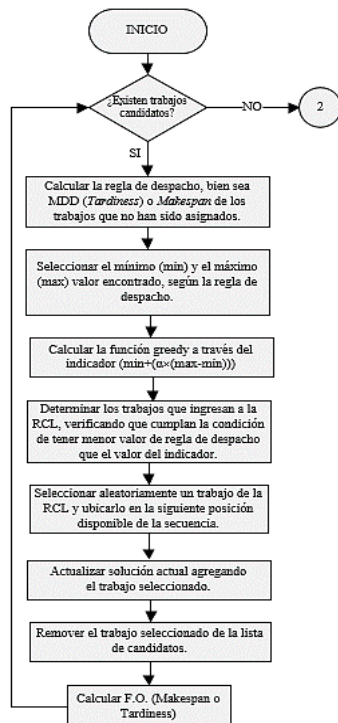
➤ Fase de Construcción

La fase de construcción consiste en determinar la secuencia inicial a partir de los parámetros establecidos. De esta manera, requiere de una función *greedy* que debe ser definida dependiendo de la función objetivo a minimizar en el primer nivel del lexicográfico, para obtener una solución inicial favorable. Así, en cada iteración de esta etapa, sólo un elemento, en este caso un trabajo, se agrega a la solución dependiendo de dicha función.

En primera instancia, para los modelos individuales, se encuentra el caso de minimización de *makespan* y *tardiness*, de manera independiente. Así, para la minimización del *makespan*, la fase de construcción inicia con el cálculo de dicha variable para cada uno de los trabajos candidatos a asignar, generando una lista preliminar con todos los valores encontrados. De este modo, se deduce que la función *greedy* a utilizar es el valor del *makespan* parcial al incluir cada trabajo no secuenciado en la última posición de la secuencia parcial bajo construcción. Posteriormente, se determina tanto el mayor como el menor valor de dicha variable dentro de la lista previamente mencionada y se conforma la RCL con los trabajos candidatos que cuenten con el $\alpha\%$ de mejores valores de función de utilidad, es decir, aquellos cuyo *makespan* sea menor a $(\minMakespan + (\alpha\% \times (\maxMakespan - \minMakespan)))$. Finalmente, se selecciona un trabajo aleatoriamente de dicho subconjunto RCL y se asigna a la siguiente posición disponible de la secuencia. Además, es relevante aclarar que, cada vez que se asigna un trabajo a una posición en la secuencia, se vuelve a generar la lista preliminar de trabajos candidatos, omitiendo los trabajos anteriormente secuenciados y se vuelve a calcular la función de utilidad de cada uno siguiendo los pasos ya mencionados hasta que se terminen de secuenciar todos los trabajos. Una vez todos los trabajos han sido secuenciados inicia la fase de búsqueda local.

Por su parte, para la minimización de la tardanza total, la función de utilidad corresponde a la adaptación de la regla de despacho *Modified Due Date* (MDD) para el problema FS. Esta regla de despacho es el resultado de la combinación de las reglas *Shortest Processing Time* (SPT) y *Earliest Due Date* (EDD), la cual tiene como objetivo programar los trabajos de menor a mayor lapso de tiempo para ser entregados. Para determinar el valor del MDD, se tiene en cuenta el máximo entre la suma de los tiempos de procesamiento de cada trabajo j en todas las máquinas ($\sum P_{ij}$) y el *due date* (dd_j) de un trabajo j menos el *makespan* parcial que lleva la secuencia actual, es decir, con los trabajos que ya se han asignado. Una vez se calcula el MDD para cada trabajo candidato, se ejecutan los mismos pasos previamente explicados en el algoritmo de construcción para la minimización del *makespan*. En otras palabras, esto es: generar la lista preliminar con el valor del MDD de cada trabajo candidato, determinar el mayor y el menor valor del MDD, calcular el indicador, construir la RCL con los valores de MDD menores al indicador, seleccionar aleatoriamente un trabajo de la RCL y asignarlo a la siguiente posición disponible en la secuencia. Nuevamente, existen tantas RCL como posiciones a asignar y el ciclo se ejecuta hasta que todos los trabajos estén asignados a una posición en la secuencia. En el momento en el que no existan más trabajos candidatos, se procede con la búsqueda local. En la Figura 2. Fase de Construcción - Metaheurística GRASP con abordaje de la función objetivo a través de la regla de ordenamiento lexicográfico para los modelos individuales y modelos con restricción asociada. se presenta el funcionamiento detallado de esta fase. Para visualizar el diagrama con mayor detalle, ver Anexo 1.

Figura 2. Fase de Construcción - Metaheurística GRASP con abordaje de la función objetivo a través de la regla de ordenamiento lexicográfico para los modelos individuales y modelos con restricción asociada.

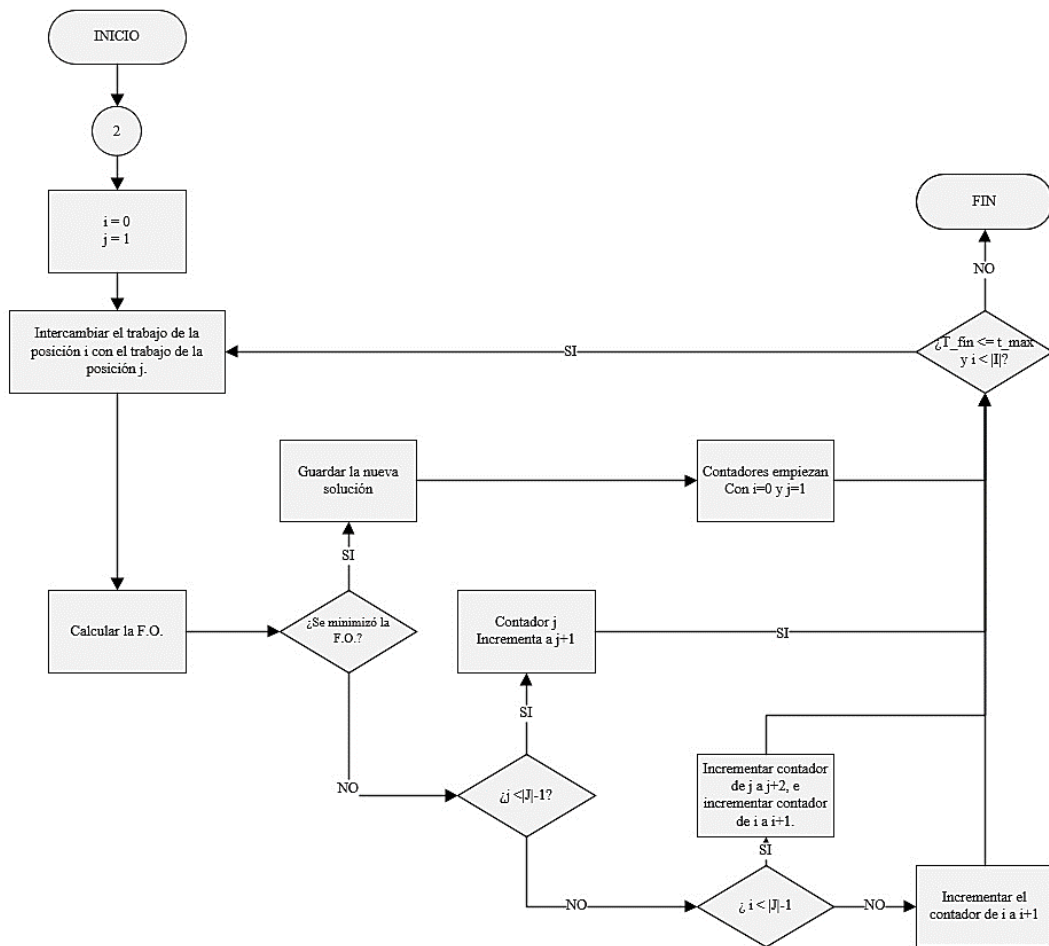


Finalmente, para los modelos con restricción asociada, es decir *MinMakespan/MinTardiness* y *MinTardiness/MinMakespan*, la fase de construcción se desarrolla como en los modelos individuales de la variable asociada a la restricción. Es decir, para el caso de *MinMakespan/MinTardiness*, la fase de construcción se compone de los pasos del modelo lexicográfico individual de *tardiness* y, para el algoritmo *MinTardiness/MinMakespan*, la secuencia inicial de la fase de construcción se obtiene del funcionamiento del modelo lexicográfico individual de *makespan*.

➤ *Fase de Búsqueda Local*

La fase de búsqueda local, para el caso del problema de programación de la producción planteado, para los modelos lexicográficos individuales, se basa en los intercambios 2-opt originalmente propuestos para el problema TSP por (Croes, 1958). La adaptación de estos movimientos en un problema de programación de la producción implica el intercambio de las posiciones en la secuencia de dos trabajos. El ciclo inicia intercambiando las dos primeras posiciones y así va avanzando hasta hacer todos los intercambios posibles. Sin embargo, cada vez que hay un intercambio que mejora la función objetivo el ciclo vuelve a reiniciar desde las dos primeras posiciones de la secuencia. Cuando un intercambio no mejora, se continúa con el siguiente intercambio. En la Figura 3. Fase de Búsqueda Local - Metaheurística GRASP con abordaje de la función objetivo a través de la regla de ordenamiento lexicográfico para los modelos individuales., se presenta el funcionamiento detallado de esta fase. Para visualizar el diagrama con mayor detalle, ver Anexo 1.

Figura 3. Fase de Búsqueda Local - Metaheurística GRASP con abordaje de la función objetivo a través de la regla de ordenamiento lexicográfico para los modelos individuales.



Ahora bien, para los modelos lexicográficos con restricción asociada, *MinMakespan/MinTardiness* y *MinTardiness/MinMakespan*, se ejecuta el mismo procedimiento, pero el algoritmo tiene en cuenta el valor de la restricción asociada. Por lo tanto, un intercambio entre pares es aceptado siempre y cuando el valor de la función

objetivo de la modificación sea menor que la función objetivo actual y el valor de la restricción de la variable adicional se cumple, es decir que, en la mutación, el valor de la variable asociada sea menor o igual que el valor previamente encontrado de la restricción.



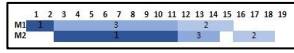
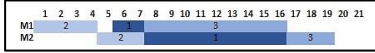

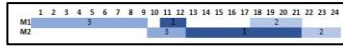
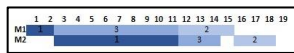
➤ *Ejemplo*

Modelos Individuales – Ordenamiento Lexicográfico

En la Tabla 1. Ejemplos de funcionamiento de los modelos de ordenamiento lexicográficos individuales., se presenta un ejemplo de los modelos de GRASP individuales de ordenamiento lexicográfico para la misma instancia de ejemplo de tres trabajos que se ha utilizado a lo largo del documento.

Tabla 1. Ejemplos de funcionamiento de los modelos de ordenamiento lexicográficos individuales.


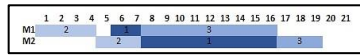
DATOS:		Tiempos de Procesamiento				Trabajos		Due Dates							
Número de Máquinas = 2.		Máquinas	Trabajos	1	2	3	1	3							
Número de Trabajos = 3.				2	4	9	2	4							
Alpha = 0,2.				9	3	3	3	5							
Tiempo = 1 segundo.															
Minimización de Makespan					Minimización de Tardiness										
Fase de Construcción: 1. Se calcula el <i>makespan</i> de cada trabajo. Trabajo 1 = 2 + 9 = 11 Trabajo 2 = 4 + 3 = 7 Trabajo 3 = 9 + 3 = 12 2. Se determina el mayor y el menor <i>makespan</i> . Máximo = 12 Mínimo = 7 3. Se evalúa la función <i>greedy</i> . $(min + (\alpha \times (max - min))) = 7 + (0,2 \times (12 - 7))$ $= 7 + (0,2 \times 5)$ $= 7 + 1$ $= 8$ 4. Entran a la RCL los valores de <i>makespan</i> que sean menores o iguales a 8. Entonces, la RCL, en este caso, está conformada solamente por el trabajo 2 y, este, se asigna a la posición #1 de la secuencia. 5. Se calcula nuevamente el <i>makespan</i> de los trabajos restantes y se determina el mínimo y el máximo valor. Trabajo 1 = 2 + 9 = 11 (Mínimo) Trabajo 3 = 9 + 3 = 12 (Máximo) 6. Se calcula la función <i>greedy</i> . $(min + (\alpha \times (max - min))) = 11 + (0,2 \times (12 - 11))$ $= 11 + (0,2 \times 1)$ $= 11 + 0,2$ $= 11,2$ 7. Entran a la RCL los valores de <i>makespan</i> que sean menores o iguales a 11,2. Entonces, la RCL, en este caso, está conformada solamente por el trabajo 1 y, este, se asigna a la posición #2 de la secuencia. 8. El trabajo #3 se asigna a la posición #3 de la secuencia, dado que no existen más trabajos. 9. Resultados: Secuencia = 2 1 3 <i>Makespan</i> Total = 19 (Ver Gantt) <div style="text-align: center;"> </div> Tardanza = (7 - 4) + (16 - 3) + (19 - 5) = 30					Fase de Construcción: 1. Se calcula el MDD de cada trabajo. MDD Trabajo 1 = 3 - 0 = 3 MDD Trabajo 2 = 4 - 0 = 4 MDD Trabajo 3 = 5 - 0 = 5 2. Se determina el mayor y el menor MDD. Máximo = 5 Mínimo = 3 3. Se evalúa la función <i>greedy</i> . $(min + (\alpha \times (max - min))) = 3 + (0,2 \times (5 - 3))$ $= 3 + (0,2 \times 2)$ $= 3 + 0,4$ $= 3,4$ 4. Entran a la RCL los valores de MDD que sean menores o iguales a 3,4. Entonces, la RCL, en este caso, está conformada solamente por el trabajo 1 y, este, se asigna a la posición #1 de la secuencia. El <i>makespan</i> parcial corresponde a 11. 5. Se calcula nuevamente el MDD de los trabajos restantes y se determina el mínimo y el máximo valor. MDD Trabajo 2 = 4 - 11 = -7 MDD Trabajo 3 = 5 - 11 = -6 Como los valores son negativos, se selecciona la suma de los tiempos de procesamiento de un trabajo en todas las máquinas, para determinar el valor mínimo y máximo del listado de MDD. MDD Trabajo 2 = 4 + 3 = 7 (Mínimo) MDD Trabajo 3 = 9 + 3 = 12 (Máximo) 6. Se calcula la función <i>greedy</i> . $(min + (\alpha \times (max - min))) = 7 + (0,2 \times (12 - 7))$ $= 7 + (0,2 \times 5)$ $= 7 + 1$ $= 8$ 7. Entran a la RCL los valores de <i>makespan</i> que sean menores o iguales a 8. Entonces, la RCL, en este caso, está conformada solamente por el trabajo 2 y, este, se asigna a la posición #2 de la secuencia. El <i>makespan</i> parcial es 14. 8. El trabajo #3 se asigna a la posición #3 de la secuencia, dado que no existen más trabajos y el <i>makespan</i> total corresponde a 23. 9. Resultados: Secuencia = 1 2 3 <i>Makespan</i> Total = 23 (Ver Gantt) <div style="text-align: center;"> </div> Tardanza = (11 - 3) + (14 - 4) + (23 - 5) = 36										
Fase de Búsqueda Local: 1. Contadores inician en i = 0 y j = 1. 2. Intercambios: <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">i = Trabajo 2 j = Trabajo 1</td> <td> Intercambio #1: Nueva Secuencia = 1 2 3 Nuevo <i>Makespan</i> = 23 (Ver Gantt) <div style="text-align: center;"> </div> Como el valor del <i>makespan</i> es mayor al <i>makespan</i> de la secuencia actual, entonces no se acepta el intercambio. </td> </tr> <tr> <td>i = Trabajo 2 j = Trabajo 3</td> <td> Intercambio #2: Nueva Secuencia = 2 3 1 </td> </tr> </table>					i = Trabajo 2 j = Trabajo 1	Intercambio #1: Nueva Secuencia = 1 2 3 Nuevo <i>Makespan</i> = 23 (Ver Gantt) <div style="text-align: center;"> </div> Como el valor del <i>makespan</i> es mayor al <i>makespan</i> de la secuencia actual, entonces no se acepta el intercambio.	i = Trabajo 2 j = Trabajo 3	Intercambio #2: Nueva Secuencia = 2 3 1	Fase de Búsqueda Local: 1. Contadores inician en i = 0 y j = 1. 2. Intercambios: <table border="1" style="width: 100%;"> <tr> <td style="width: 20%;">i = Trabajo 1 j = Trabajo 2</td> <td> Intercambio #1: Nueva Secuencia = 2 1 3 Nuevo <i>Makespan</i> = 25 (Ver Gantt) Nueva Tardanza = (7 - 4) + (16 - 3) + (25 - 5) = 36 </td> </tr> </table>					i = Trabajo 1 j = Trabajo 2	Intercambio #1: Nueva Secuencia = 2 1 3 Nuevo <i>Makespan</i> = 25 (Ver Gantt) Nueva Tardanza = (7 - 4) + (16 - 3) + (25 - 5) = 36
i = Trabajo 2 j = Trabajo 1	Intercambio #1: Nueva Secuencia = 1 2 3 Nuevo <i>Makespan</i> = 23 (Ver Gantt) <div style="text-align: center;"> </div> Como el valor del <i>makespan</i> es mayor al <i>makespan</i> de la secuencia actual, entonces no se acepta el intercambio.														
i = Trabajo 2 j = Trabajo 3	Intercambio #2: Nueva Secuencia = 2 3 1														
i = Trabajo 1 j = Trabajo 2	Intercambio #1: Nueva Secuencia = 2 1 3 Nuevo <i>Makespan</i> = 25 (Ver Gantt) Nueva Tardanza = (7 - 4) + (16 - 3) + (25 - 5) = 36														


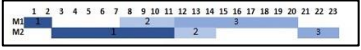
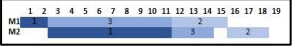
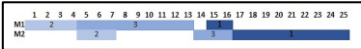
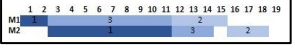

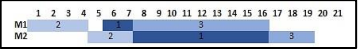
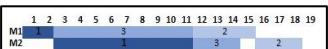
	<p>Nuevo <i>Makespan</i> = 25 (Ver Gantt)</p>  <p>Como el valor del <i>makespan</i> es mayor al <i>makespan</i> de la secuencia actual, entonces no se acepta el intercambio.</p>		 <p>Como el valor de la tardanza es igual a la tardanza de la secuencia actual, entonces no se acepta el intercambio.</p>
<p>$i =$ Trabajo 1 $j =$ Trabajo 3</p>	<p>Ya se evaluó esta modificación y, como no existen más posibles intercambios, la fase de búsqueda local termina.</p>	<p>$i =$ Trabajo 1 $j =$ Trabajo 3</p>	<p>Intercambio #2: Nueva Secuencia = 1 3 2 Nuevo <i>Makespan</i> = 18 (Ver Gantt) Nueva Tardanza = $(11 - 3) + (14 - 5) + (18 - 4) = 31$</p>  <p>Como el valor de la tardanza es menor a la tardanza de la secuencia actual, entonces se acepta el intercambio.</p>
<p>3. Resultados: Secuencia = 2 1 3 <i>Makespan</i> Total = 19 (Ver Gantt)</p>  <p>Tardanza = 30</p>		<p>Nuevos Resultados</p>	<p>Los resultados y la nueva secuencia actual, sobre los cuales se van a determinar los intercambios son: Nueva Secuencia = 1 3 2 <i>Makespan</i> = 18 (Ver Gantt)</p>  <p>Tardanza = 31</p>
		<p>$i =$ Trabajo 1 $j =$ Trabajo 3</p>	<p>Intercambio #3: Nueva Secuencia = 3 1 2 Nuevo <i>Makespan</i> = 24 (Ver Gantt) Nueva Tardanza = $(12 - 5) + (21 - 3) + (24 - 4) = 45$</p>  <p>Como el valor de la tardanza es mayor a la tardanza de la secuencia actual, entonces no se acepta el intercambio.</p>
		<p>$i =$ Trabajo 1 $j =$ Trabajo 2</p>	<p>Ya se evaluó esta modificación y, como no existen más posibles intercambios, la fase de búsqueda local termina.</p> <p>3. Resultados: Secuencia = 1 3 2 <i>Makespan</i> = 18 (Ver Gantt)</p>  <p>Tardanza = 31</p>

Modelos Con Restricción Asociada – Ordenamiento Lexicográfico

En la Tabla 2. Ejemplos de funcionamiento de los modelos de ordenamiento lexicográficos con restricción asociada., se presenta el ejemplo de los modelos de GRASP con restricción asociada para la misma instancia de ejemplo de tres trabajos que se ha utilizado a lo largo del documento, partiendo de los resultados del ejemplo presentado en la Tabla 1.

Tabla 2. Ejemplos de funcionamiento de los modelos de ordenamiento lexicográficos con restricción asociada.

<p>DATOS:</p> <p>Número de Máquinas = 2. Número de Trabajos = 3. Alpha = 0,2. Tiempo = 1 segundo.</p>	<table border="1" data-bbox="505 1493 878 1598"> <thead> <tr> <th colspan="2"></th> <th colspan="3">Tiempos de Procesamiento</th> </tr> <tr> <th>Máquinas</th> <th>Trabajos</th> <th>1</th> <th>2</th> <th>3</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>2</td> <td>4</td> <td>9</td> </tr> <tr> <td>2</td> <td>2</td> <td>9</td> <td>3</td> <td>3</td> </tr> </tbody> </table> <table border="1" data-bbox="894 1493 1146 1598"> <thead> <tr> <th>Trabajos</th> <th>Due Dates</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>3</td> </tr> <tr> <td>2</td> <td>4</td> </tr> <tr> <td>3</td> <td>5</td> </tr> </tbody> </table>			Tiempos de Procesamiento			Máquinas	Trabajos	1	2	3	1	1	2	4	9	2	2	9	3	3	Trabajos	Due Dates	1	3	2	4	3	5
		Tiempos de Procesamiento																											
Máquinas	Trabajos	1	2	3																									
1	1	2	4	9																									
2	2	9	3	3																									
Trabajos	Due Dates																												
1	3																												
2	4																												
3	5																												
<p>Minimización de <i>Makespan</i> con restricción de Tardiness</p> <p>Fase de Construcción: 1. Resultados Modelo Individual – Lexicográfico <i>Tardiness</i>. Secuencia = 1 2 3 <i>Makespan</i> Total = 23 (Ver Gantt)</p>  <p>Tardanza = $(11 - 3) + (14 - 4) + (23 - 5) = 36$</p> <p>Fase de Búsqueda Local: 1. Contadores inician en $i = 0$ y $j = 1$.</p>	<p>Minimización de Tardiness con restricción de <i>Makespan</i></p> <p>Fase de Construcción: 1. Resultados Modelo Individual – Lexicográfico <i>Makespan</i>: Secuencia = 2 1 3 <i>Makespan</i> Total = 19 (Ver Gantt)</p>  <p>Tardanza = $(7 - 4) + (16 - 3) + (19 - 5) = 30$</p> <p>Fase de Búsqueda Local: 1. Contadores inician en $i = 0$ y $j = 1$.</p>																												

<p>2. El valor de la restricción asociada, referente a la tardanza es 36. Este valor es el umbral que permite determinar si se acepta o no un intercambio, teniendo en cuenta si la función objetivo mejora respecto a la solución actual.</p>	<p>2. El valor de la restricción asociada, referente al <i>makespan</i> es 19. Este valor es el umbral que permite determinar si se acepta o no un intercambio, teniendo en cuenta si la función objetivo mejora respecto a la solución actual.</p>
<p>3. Intercambios:</p> <p>i = Trabajo 1 j = Trabajo 2</p> <p>Intercambio #1: Nueva Secuencia = 2 1 3 Nuevo <i>Makespan</i> = 25 (Ver Gantt)</p>  <p>Nueva Tardanza = $(7 - 4) + (16 - 3) + (25 - 5)$ = 36</p> <p>Como el valor del <i>makespan</i> es mayor al <i>makespan</i> de la secuencia actual y el valor de la tardanza es igual al valor de la restricción, entonces no se acepta el intercambio.</p>	<p>3. Intercambios:</p> <p>i = Trabajo 2 j = Trabajo 1</p> <p>Intercambio #1: Nueva Secuencia = 1 2 3 Nuevo <i>Makespan</i> = 23 (Ver Gantt) Nueva Tardanza = $(11 - 3) + (14 - 4) + (23 - 5)$ = 36</p>  <p>Como el valor de la tardanza es mayor a la tardanza de la secuencia actual y el valor del <i>makespan</i> es mayor al valor de la restricción, entonces no se acepta el intercambio.</p>
<p>i = Trabajo 2 j = Trabajo 3</p> <p>Intercambio #2: Nueva Secuencia = 1 3 2 Nuevo <i>Makespan</i> = 18 (Ver Gantt) Nueva Tardanza = $(11 - 3) + (14 - 5) + (18 - 4)$ = 31</p>  <p>Como el valor del <i>makespan</i> es menor al <i>makespan</i> de la secuencia actual y el valor de la tardanza es menor que el valor de la restricción, entonces se acepta el intercambio.</p>	<p>i = Trabajo 2 j = Trabajo 3</p> <p>Intercambio #2: Nueva Secuencia = 2 3 1 Nuevo <i>Makespan</i> = 25 (Ver Gantt) Nueva Tardanza = $(7 - 4) + (16 - 5) + (25 - 3)$ = 36</p>  <p>Como el valor de la tardanza es mayor a la tardanza de la secuencia actual y el valor del <i>makespan</i> es mayor al valor de la restricción, entonces no se acepta el intercambio.</p>
<p>Nuevos Resultados</p> <p>Los resultados y la nueva secuencia actual, sobre los cuales se van a determinar los intercambios son: Nueva Secuencia = 1 3 2 <i>Makespan</i> = 18 (Ver Gantt)</p>  <p>Tardanza = 31</p> <p>*El valor de la restricción sigue siendo el mismo, es decir, 36.</p>	<p>i = Trabajo 1 j = Trabajo 3</p> <p>Ya se evaluó esta modificación y, como no existen más posibles intercambios, la fase de búsqueda local termina.</p>
<p>i = Trabajo 1 j = Trabajo 3</p> <p>Intercambio #3: Nueva Secuencia = 3 1 2 Nuevo <i>Makespan</i> = 24 (Ver Gantt) Nueva Tardanza = $(12 - 5) + (21 - 3) + (24 - 4)$ = 45</p>  <p>Como el valor del <i>makespan</i> es mayor al <i>makespan</i> de la secuencia actual y el valor de la tardanza es igual al valor de la restricción, entonces no se acepta el intercambio.</p>	<p>4. Resultados:</p> <p>Secuencia = 2 1 3 <i>Makespan</i> Total = 19 (Ver Gantt)</p>  <p>Tardanza = 30</p>
<p>i = Trabajo 1 j = Trabajo 2</p> <p>Ya se evaluó esta modificación y, como no existen más posibles intercambios, la fase de búsqueda local termina.</p> <p>4. Resultados: Nueva Secuencia = 1 3 2 <i>Makespan</i> = 18 (Ver Gantt)</p>  <p>Tardanza = 31</p>	

4.2.2. Metaheurística GRASP para la solución de un BPFs biobjetivo híbrida con el algoritmo PAES para obtener la frontera de Pareto de los dos objetivos (GRASP-PAES)

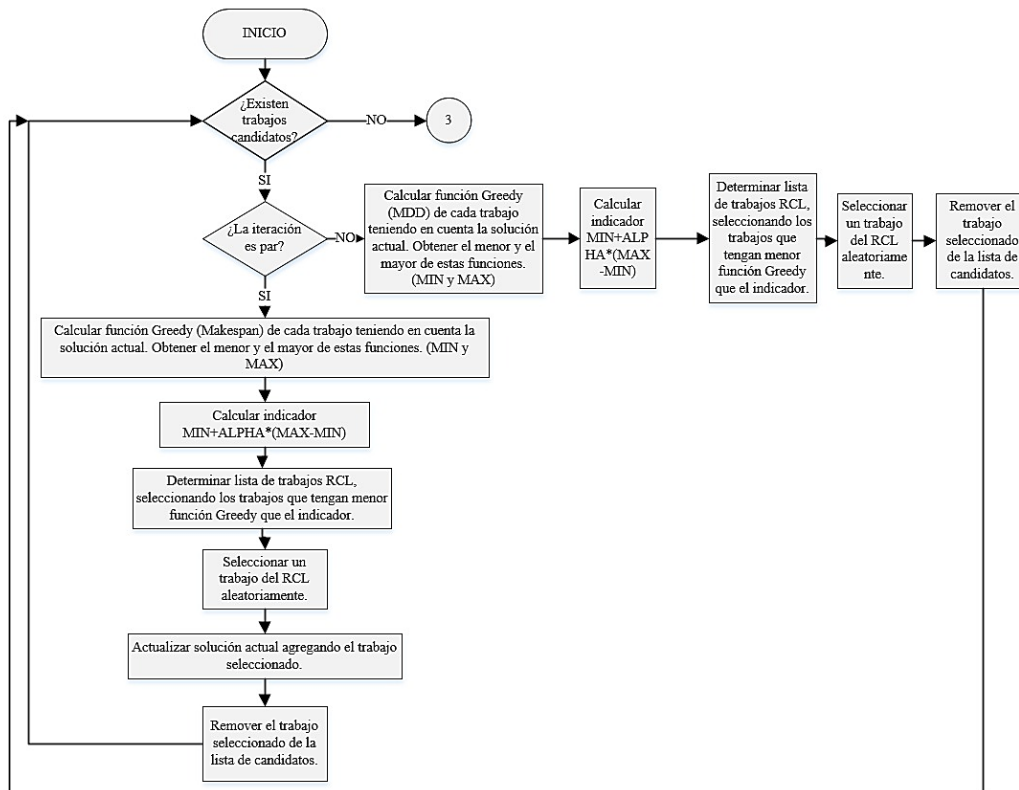
Para diseñar la metaheurística GRASP que permitiera obtener la frontera de Pareto de los objetivos seleccionados, se hibridizó GRASP con la metodología *Pareto Archived Evolution Strategy* (PAES), la cual fue desarrollada como un método de búsqueda local para la generación de la frontera de Pareto en problemas multiobjetivo. A continuación, se explican las dos fases del GRASP-PAES propuesto (Knowles & Corne, 2000).

➤ Fase de Construcción

Para la fase de construcción del GRASP-PAES se determinan las mismas funciones *greedy* definidas para los dos procedimientos lexicográficos. Sin embargo, dado que este algoritmo contempla las dos funciones objetivo

simultáneamente, es necesario hacer uso del método *Sequential Combined*, en donde se selecciona una función *greedy* diferente por cada uno de los pasos de la fase de construcción (Martí, Campos, Resende, & Duarte, 2015). De esta manera, se intercalan dichas funciones correspondientes a la minimización del *makespan* y de la tardanza en la construcción de la secuencia inicial. Es decir que, que para los pasos impares de la construcción se utiliza como función de utilidad la regla MDD y para los pasos pares de la construcción se utiliza como función de utilidad el cálculo del *makespan* de la secuencia parcial. En la Figura 4. Fase de Construcción - Metaheurística GRASP hibridizada con el algoritmo PAES., se observa el funcionamiento detallado de esta fase. Para visualizar el diagrama con mayor detalle, ver Anexo 1.

Figura 4. Fase de Construcción - Metaheurística GRASP hibridizada con el algoritmo PAES.



➤ Fase de Búsqueda Local

La fase de búsqueda local busca mejorar n funciones objetivo a partir de la información obtenida en la fase de construcción. Para esto, se evalúa la metodología PAES, cuyo objetivo, en este caso, corresponde a minimizar el valor de *makespan* y *tardiness* simultáneamente, obteniendo diversas soluciones que sean no dominadas y comparables entre sí. De esta forma, la solución obtenida en la fase de construcción se denominará C y se incorpora ésta al archivo de Pareto. Luego, se hace una mutación de S_C , la cual se denominará S_M , intercambiando dos posiciones de S_C y se calculan ambas funciones objetivo de S_M . Posteriormente, se verifican las dominancias entre las soluciones S_C y S_M de la siguiente manera:

- **Si C domina a M :** Siendo F_1 el *makespan* y F_2 la tardanza, si se cumplen alguna de las dos siguientes condiciones, entonces C domina a M y por tanto M se descarta:

$$\text{Si } [F_1(C) < F_1(M) \text{ y } F_2(C) \leq F_2(M)] \quad \text{ó} \quad \text{Si } [F_1(C) \leq F_1(M) \text{ y } F_2(C) < F_2(M)]$$

- **Si M domina a C :** Si se cumplen alguna de las dos condiciones siguientes entonces M domina a C y, por tanto, se hace la solución $C=M$:

$$\text{Si } [F_1(M) < F_1(C) \text{ y } F_2(M) \leq F_2(C)] \quad \text{ó} \quad \text{Si } [F_1(M) \leq F_1(C) \text{ y } F_2(M) < F_2(C)]$$

- Si M es dominada por alguna de las soluciones que se encuentran hasta el momento dentro del archivo de Pareto (S_{Arch}), entonces M se descarta:

$$\text{Si } [F_1(S_{Arch}) < F_1(M) \text{ y } F_2(S_{Arch}) \leq F_2(M)] \quad \text{ó} \quad \text{Si } [F_1(S_{Arch}) \leq F_1(M) \text{ y } F_2(S_{Arch}) < F_2(M)]$$

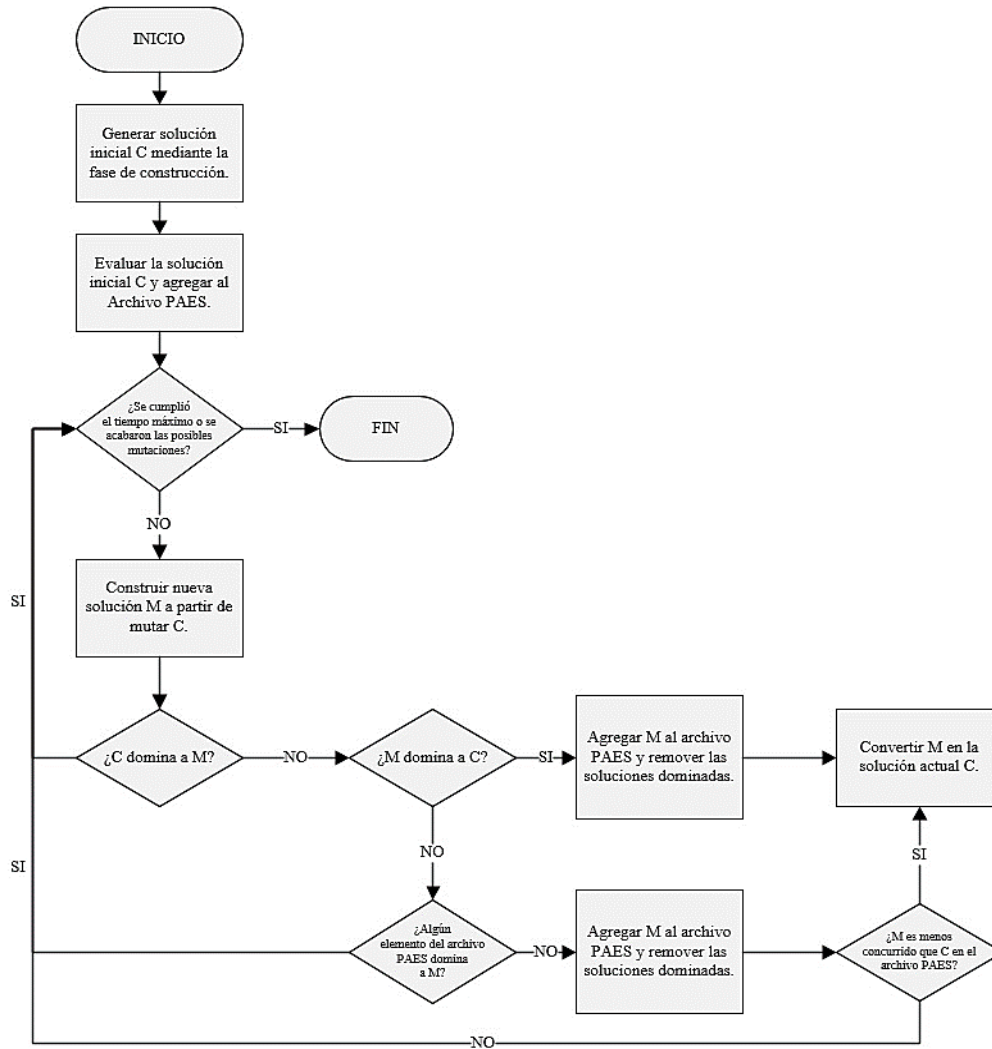
Si no se cumple ninguna de estas condiciones, previamente mencionadas, entonces se procede a calcular la concurrencia *crowding distance*, que representa una medida de densidad de población de acuerdo con la distancia euclidiana normalizada de las funciones objetivo (Fu & Wen, 2017). Esta medida, se calcula a partir de los valores máximos y mínimos evaluados en ambas funciones objetivo con la siguiente ecuación (42):

$$\Delta_i = \sum_{j=1}^{n_o} \frac{d_j^i}{\Delta f_j} \quad (42)$$

En donde d_j^i es la diferencia absoluta entre el valor inmediatamente superior y el inmediatamente inferior de la función objetivo evaluada, y Δf_j es la diferencia absoluta entre el máximo y el mínimo de los valores evaluados para el objetivo j .

Por otra parte, si la solución que se está evaluando es un extremo superior entonces el superior de éste, es su mismo valor y el inferior de éste, es el inmediatamente inferior, mientras que, para el extremo inferior, el valor inferior a éste es, también, su mismo valor y, el valor superior es el inmediatamente superior. Finalmente, si la concurrencia de C es mayor, entonces M pasa a ser la solución actual y, por lo tanto, se reinician los contadores a $i=0$ y $j=1$, se genera una nueva solución M y se comienzan a hacer intercambios entre posiciones para evaluar dominancias. En la Figura 5, se presenta el funcionamiento detallado de esta fase. En la Figura 5. Fase de Búsqueda Local - Metaheurística GRASP con abordaje de la función objetivo a través de la metodología PAES., se observa el funcionamiento detallado de esta fase. Para visualizar el diagrama con mayor detalle, ver Anexo 1.

Figura 5. Fase de Búsqueda Local - Metaheurística GRASP con abordaje de la función objetivo a través de la metodología PAES.



➤ *Ejemplo*

En la Tabla 3. Ejemplos de funcionamiento del GRASP hibridizado con el algoritmo PAES., la secuenciación con GRASP-PAES para la misma instancia de ejemplo que se explicó en el lexicográfico.

Tabla 3. Ejemplos de funcionamiento del GRASP hibridizado con el algoritmo PAES.

DATOS:							
Número de Máquinas = 2.	Tiempos de Procesamiento					Trabajos	
Número de Trabajos = 3.	Trabajos	1	2	3	Due Dates		
Alpha = 0,2.	Máquinas	1	2	3	1	3	
Tiempo = 1 segundo.	1	2	4	9	2	4	
	2	9	3	3	3	5	
Función bioobjetivo para minimización de Makespan y Tardiness							
Fase de Construcción:							
<i>Posición #1:</i> Como es una posición impar, se hace uso del modelo lexicográfico individual de la tardanza.							
1. Se calcula el MDD de cada trabajo.							
MDD Trabajo 1 = 3 - 0 = 3							
MDD Trabajo 2 = 4 - 0 = 4							
MDD Trabajo 3 = 5 - 0 = 5							
2. Se determina el mayor y el menor MDD.							
Máximo = 5							
Mínimo = 3							
3. Se evalúa la función <i>greedy</i> .							
$(min + (\alpha \times (max - min))) = 3 + (0,2 \times (5 - 3))$							
$= 3 + (0,2 \times 2)$							
$= 3 + 0,4$							

$$= 3,4$$

4. Entran a la RCL los valores de MDD que sean menores o iguales a 3,4. Entonces, la RCL, en este caso, está conformada solamente por el trabajo 1 y, este, se asigna a la posición #1 de la secuencia. El *makespan* parcial corresponde a 11.

Posición #2: Como es una posición par, se hace uso del modelo lexicográfico individual del *makespan*.

1. Se calcula el *makespan* de cada trabajo, omitiendo el trabajo que tiene una posición asignada en la secuencia.

$$\begin{aligned} \text{Trabajo 2} &= 4 + 3 = 7 \\ \text{Trabajo 3} &= 9 + 3 = 12 \end{aligned}$$

2. Se determina el mayor y el menor *makespan*.

$$\begin{aligned} \text{Máximo} &= 12 \\ \text{Mínimo} &= 7 \end{aligned}$$

3. Se evalúa la función *greedy*.

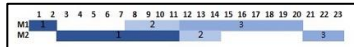
$$\begin{aligned} (\min + (\alpha \times (\max - \min))) &= 7 + (0,2 \times (12 - 7)) \\ &= 7 + (0,2 \times 5) \\ &= 7 + 1 \\ &= 8 \end{aligned}$$

4. Entran a la RCL los valores de *makespan* que sean menores o iguales a 8. Entonces, la RCL, en este caso, está conformada solamente por el trabajo 2 y, este, se asigna a la posición #2 de la secuencia.

5. El trabajo 3 se asigna a la posición #3 de la secuencia, dado que no existen más trabajos.

6. Resultados:

$$\begin{aligned} \text{Secuencia} &= 1 \ 2 \ 3 \\ \text{Makespan Total} &= 23 \text{ (Ver Gantt)} \end{aligned}$$



$$\text{Tardanza} = (11 - 3) + (14 - 4) + (23 - 5) = 36$$

Fase de Búsqueda Local:

1. Contadores inician en $i = 0$ y $j = 1$.

2. Intercambios:

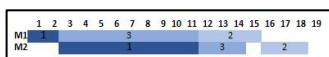
$i = \text{Trabajo 1}$ $j = \text{Trabajo 2}$	<p>Intercambio #1:</p> <p>Nueva Secuencia = 2 1 3 Nuevo <i>Makespan</i> = 25 (Ver Gantt) Nueva Tardanza = $(7 - 4) + (16 - 3) + (25 - 5)$ $= 36$</p>	<p>Dominancia C sobre M:</p> <table border="1"> <thead> <tr> <th></th> <th>Solución C</th> <th>Solución M</th> </tr> </thead> <tbody> <tr> <td><i>Makespan</i></td> <td>23</td> <td>25</td> </tr> <tr> <td><i>Tardiness</i></td> <td>36</td> <td>36</td> </tr> </tbody> </table> <p>Entonces: $23 < 25$ y $36 \leq 36$ (Se cumple) $23 \leq 25$ y $36 < 36$ (No se cumple)</p> <p>Como una de las dos condiciones se cumple, entonces M se descarta y se continúan los intercambios.</p>		Solución C	Solución M	<i>Makespan</i>	23	25	<i>Tardiness</i>	36	36
	Solución C	Solución M									
<i>Makespan</i>	23	25									
<i>Tardiness</i>	36	36									
$i = \text{Trabajo 1}$ $j = \text{Trabajo 3}$	<p>Intercambio #2:</p> <p>Nueva Secuencia = 1 3 2 Nuevo <i>Makespan</i> = 18 (Ver Gantt) Nueva Tardanza = $(11 - 3) + (14 - 5) + (18 - 4)$ $= 31$</p>	<p>Dominancia C sobre M:</p> <table border="1"> <thead> <tr> <th></th> <th>Solución C</th> <th>Solución M</th> </tr> </thead> <tbody> <tr> <td><i>Makespan</i></td> <td>23</td> <td>18</td> </tr> <tr> <td><i>Tardiness</i></td> <td>36</td> <td>31</td> </tr> </tbody> </table> <p>Entonces: $23 < 18$ y $36 \leq 31$ (No se cumple) $23 \leq 18$ y $36 < 31$ (No se cumple)</p> <p>Como no se cumple la dominancia de C sobre M, entonces se procede a verificar la dominancia de M sobre C. Entonces:</p> $18 < 23 \text{ y } 31 \leq 36 \text{ (Se cumple)}$ $18 \leq 23 \text{ y } 31 < 36 \text{ (Se cumple)}$ <p>Como las dos condiciones se cumplen, entonces M se acepta, pasa a ser C y se agrega al archivo de Pareto.</p>		Solución C	Solución M	<i>Makespan</i>	23	18	<i>Tardiness</i>	36	31
	Solución C	Solución M									
<i>Makespan</i>	23	18									
<i>Tardiness</i>	36	31									
<p>Nuevos Resultados</p>	<p>Los resultados y la nueva secuencia actual, sobre los cuales se van a determinar los intercambios son:</p> <p>Nueva Secuencia = 1 3 2 <i>Makespan</i> = 18 (Ver Gantt) Tardanza = 31</p>										
$i = \text{Trabajo 1}$ $j = \text{Trabajo 3}$	<p>Intercambio #3:</p> <p>Nueva Secuencia = 3 1 2 Nuevo <i>Makespan</i> = 24 (Ver Gantt) Nueva Tardanza = $(12 - 5) + (21 - 3) + (24 - 4)$ $= 45$</p>	<p>Dominancia C sobre M:</p> <table border="1"> <thead> <tr> <th></th> <th>Solución C</th> <th>Solución M</th> </tr> </thead> <tbody> <tr> <td><i>Makespan</i></td> <td>18</td> <td>24</td> </tr> <tr> <td><i>Tardiness</i></td> <td>31</td> <td>45</td> </tr> </tbody> </table> <p>Entonces: $18 < 24$ y $31 \leq 45$ (Se cumple) $18 \leq 24$ y $31 < 45$ (Se cumple)</p> <p>Como las dos condiciones se cumplen, entonces M se descarta.</p>		Solución C	Solución M	<i>Makespan</i>	18	24	<i>Tardiness</i>	31	45
	Solución C	Solución M									
<i>Makespan</i>	18	24									
<i>Tardiness</i>	31	45									
$i = \text{Trabajo 3}$ $j = \text{Trabajo 2}$	<p>Ya se evaluó esta modificación y, como no existen más posibles intercambios, la fase de búsqueda local termina.</p>										

3. Resultados:

$$\text{Nueva Secuencia} = 1 \ 3 \ 2$$

Makespan = 18 (Ver Gantt)

Tardanza = 31



Esta es la única solución encontrada y, por lo tanto, la única que se encuentra en la frontera de Pareto.

5. Resultados

5.1. Resultados de los modelos matemáticos

Los modelos matemáticos fueron implementados en Gusek y NEOS, y probados para instancias pequeñas creadas de acuerdo con los procedimientos presentados por (Minella et al., 2008). De esta manera, la creación de éstas surge dado que las instancias *benchmark* de la literatura existentes para FS tienen 20 o más trabajos y 5 o más máquinas, para las cuales el modelo matemático uniobjetivo del BPPS no arroja resultados ni en Gusek ni en NEOS, por falta de memoria. Así, se crearon un total de 84 instancias pequeñas en las que los tiempos de procesamiento de las máquinas y los *due dates* se generaron de forma aleatoria a partir de una distribución uniforme entre 0 y 99, y una distribución uniforme entre $P(1 - T - R/2)$ y $P(1 - T + R/2)$ respectivamente, siendo T el factor de retardo, R el rango de fechas de vencimiento y P la suma de todos los tiempos de procesamiento. Las 84 instancias provienen de las siguientes combinaciones: $T=\{0.2; 0.6\}$, $R=\{0.2;1\}$, número de máquinas $M=\{2;3;4\}$ y número de trabajos $N=\{5;6;7;8;9;10;15\}$.

De esta forma, se corrieron los modelos individuales y modelos con restricción asociada del lexicográfico para cada una de las 84 instancias pequeñas creadas, así como las instancias *benchmark*, limitando el tiempo en Gusek a un máximo de 7200 segundos. En la Tabla 4. Promedios de los resultados obtenidos con los modelos matemáticos., se presentan los resultados promedio para cada tamaño de instancia (NxM) de *tardiness* y *makespan* de cada uno de los modelos individuales, así como los resultados de *makespan* del modelo *MinMakespan/MinTardiness* y de *tardiness* del modelo *MinTardiness/MinMakespan*. También, se presentan los tiempos de ejecución. Finalmente, en el Anexo 2 y en el Anexo 3, se muestran los resultados independientes de cada una de las 84 instancias pequeñas, más las 110 de *Taillard*, tanto para los modelos individuales como para los modelos con restricción asociada.

Tabla 4. Promedios de los resultados obtenidos con los modelos matemáticos.

Instancia	Modelos Individuales				Modelos Con Restricción Asociada					
	Makespan	Tiempo Ejecución (s)	Tardiness	Tiempo Ejecución (s)	Makespan SA Tardiness			Tardiness SA Makespan		
					Makespan	Min Tardiness	Tiempo Ejecución (s)	Tardiness	Min Makespan	Tiempo Ejecución (s)
5x2	325,50	0,30	57,75	0,65	328,75	57,75	0,35	76,75	325,50	0,36
5x3	397,75	0,29	37,25	0,56	399,50	37,25	0,33	24,50	397,75	0,41
5x4	428,00	0,34	38,25	0,71	439,75	38,25	0,38	735,50	428,00	0,45
6x2	305,50	0,34	100,25	0,71	306,50	100,25	0,35	562,25	305,50	0,66
6x3	425,00	0,34	89,50	0,89	451,25	89,50	0,34	147,75	400,25	0,93
6x4	539,75	0,52	129,25	0,74	541,00	129,25	0,51	185,00	539,75	1,02
7x2	388,00	0,46	184,75	75,89	393,25	184,75	0,46	267,75	388,00	88,20
7x3	442,25	0,41	30,00	3,19	442,75	30,00	0,46	91,25	442,25	5,33
7x4	566,75	0,96	0,00	2,84	581,25	0,00	1,04	1239,50	566,75	7,13
8x2	468,50	0,80	71,25	12,09	477,75	71,25	0,83	1028,25	468,50	163,55
8x3	494,00	1,54	132,75	11,99	495,25	132,75	1,71	1105,00	494,00	59,46
8x4	597,75	1,99	43,50	7,14	602,25	43,50	2,13	1498,50	597,75	82,58
9x2	522,50	1,26	78,75	193,24	524,75	78,75	1,53	1331,75	522,50	2756,81
9x3	578,50	6,63	49,50	168,66	601,25	86,00	7,56	86,00	578,50	3,17
9x4	628,00	8,42	45,25	90,30	632,75	45,25	10,76	169,75	628,00	3,33
10x2	529,75	8,25	154,25	125,44	532,75	154,25	6,07	281,25	529,75	10,28
10x3	673,25	18,10	133,50	259,95	708,25	133,50	20,16	229,50	673,25	24,35
10x4	757,75	27,19	153,75	1866,87	796,50	153,75	25,01	284,25	757,75	16,75
15x2	634,00	192,84	45,25	537,91	794,25	45,25	220,54	102,00	634,00	593,72
15x3	926,25	251,25	0,00	642,34	937,50	0,00	304,31	248,50	926,25	704,20

15x4	1029,00	454,78	30,00	724,45	1063,75	30,00	412,97	472,75	1029,00	880,74
20x5	1602,8	7203,29	6197,5	7205,03	1654,5	6197,5	7201,72	7162	1602,8	7201,39
20x10	2258,6	7203,65	5101,9	7206,06	2266	5101,9	7203,42	5301,33	2258,60	7202
20x20	3445,8	7205,19	6028,4	7205,32	6028,4	6028,4	7205,32	4224,0	3445,8	7204,95
50x5	3946,1	7215,55	69380,8	7186,79	-	69380,8	7202,6	-	3946,1	7206,36
50x10	5597,4	7215,55	89348	7217,28	5744,5	89348	7239,96	-	5597,4	7218,49
50x20	8051	7242,75	-	7244,42	-	-	-	-	8051	7257,8
100x5	-	7235,64	-	7216,37	-	-	-	-	-	-
100x10	-	7239,9	-	7218,63	-	-	-	-	-	-
100x20	-	7229,43	-	7222,96	-	-	-	-	-	-
200x10	-	7244,27	-	7209,20	-	-	-	-	-	-
200x20	-	7236,19	-	7213,85	-	-	-	-	-	-

De acuerdo con los resultados obtenidos, para el caso del *makespan*, los valores óptimos encontrados en los modelos con restricción asociada, es decir, cuando se evalúa la minimización del *makespan* sujeto a la restricción de la tardanza mínima, tienden a variar mínimamente respecto a los modelos individuales. Sin embargo, cuando se minimiza la tardanza, posterior a la minimización del *makespan* (restricción asociada de *makespan*), la tardanza crece con respecto al modelo individual de *tardiness*. Esto se da porque la tardanza mínima tiene mayores valores de *makespan* en promedio. Finalmente, en algunos pocos casos como las instancias 20x20, el valor de la tardanza en el modelo individual es mayor al valor de la tardanza en el modelo sujeto a *makespan* debido a que la solución que se encuentra, en el modelo con restricción asociada, corresponde a la mejor solución obtenida hasta el tiempo límite de 7200 segundos más no a la solución óptima.

5.2. Resultados de la metaheurística GRASP

Los GRASP propuestos, tanto el enfoque de ordenamiento lexicográfico como el GRASP-PAES, fueron ejecutados para las 84 instancias pequeñas creadas, así como para 110 instancias *benchmark* medianas y grandes tomadas de (Ciavotta, Minella, & Ruiz, 2013), asumiendo tiempos de *setup* en cero.

Inicialmente, con el fin de establecer los valores fijos de los parámetros, correspondientes al *alpha* y el tiempo máximo de corrida, se realizaron unas pruebas estadísticas que permitieron identificar si dichos factores eran significativos para las respectivas funciones objetivo y, además, asignar valores predeterminados para desarrollar el experimento, parametrizándolo, y, así, garantizar los mejores resultados posibles. Para esto, la metaheurística GRASP con abordaje de ordenamiento lexicográfico y la hibridizada con PAES evaluó cinco instancias aleatorias del total de instancias, ejecutando cada una de ellas 5 veces, asegurando que existiera variación en los tamaños de las instancias *benchmark*, desde 5M 20N hasta 20M 200N.

Una vez definidas las instancias de prueba, se determinaron los valores a evaluar para cada uno de los parámetros. Así, para el *alpha* se estableció 0.2 y 0.4, lo que permite garantizar voracidad al momento de elegir un trabajo para conformar la RCL, pues, entre más cercano sea al 1, mayor aleatoriedad va a tener la elección. Por otra parte, en cuanto al criterio correspondiente al tiempo máximo de ejecución, para las metaheurísticas GRASP con abordaje de ordenamiento lexicográfico, se concretó que los modelos individuales evaluarán el factor multiplicativo *t* de tiempo en 0.001, 0.005, 0.01 y 0.05 segundos, de tal forma que no hicieran una búsqueda local exhaustiva, permitiendo encontrar mejoras en los modelos con restricción asociada. Una vez hallada la solución, la variable de respuesta obtenida para realizar las pruebas estadísticas (ANOVAS), corresponde al valor de la función objetivo, que, a su vez, representa el valor de la restricción en los modelos *MinMakespan/MinTardiness* y *MinTardiness/MinMakespan*. Finalmente, para estos modelos, el factor multiplicativo de tiempo contempló como valores a evaluar 1.0, 1.5 y 2.0 segundos, manteniendo la misma variable de respuesta para el experimento estadístico. Por su parte, en el caso de la metaheurística GRASP hibridizada con PAES, se hacen evaluación de pruebas estadísticas con valores del factor multiplicativo de tiempo correspondientes a 1.0, 1.5, 2.0, 3.0 y 4.0 segundos, y la variable de respuesta hace referencia al *Mean Ideal Distance* (MID), siendo esta la medida que representa la cercanía entre la solución de Pareto y el punto ideal (0,0) (Karimi, Zandieh, & Karamooz, 2010), presentada a partir de la siguiente ecuación (43):

$$MID = \frac{\sum_{i=1}^n C_i}{n} \quad (43)$$

en donde C_i es

$$C_i = \sqrt{f_{1i}^2 + f_{2i}^2} \quad (44)$$

y f_{1i} y f_{2i} son los valores ordinales (primero, segundo, tercero...) de la solución no dominada de la primera y segunda función objetivo, respectivamente.

Una vez definidos estos valores a evaluar, se procedieron a realizar los procedimientos estadísticos (ANOVAS), iniciando con pruebas paramétricas y, finalmente, ejecutando pruebas no paramétricas. A continuación, se presentan los resultados encontrados para cada una de las metaheurísticas. Para ver el detalle de la parametrización, ver Anexo 4.

5.2.1. GRASP Ordenamiento Lexicográfico – Modelos Individuales

Makespan

A través de un análisis de varianza, se procede a evaluar el efecto de los factores “Alpha”, “Tiempo” e “Iteraciones”, así como sus interacciones, de tal forma que sea posible confirmar la significancia de estos, con respecto a la minimización del *makespan*. En la Tabla 5. Análisis de varianza – Minimización *Makespan*., se presentan los resultados obtenidos y, en la Tabla 6. Resumen del modelo – Minimización *Makespan*., el resumen del modelo.

Tabla 5. Análisis de varianza – Minimización *Makespan*.

<i>Fuente</i>	<i>GL</i>	<i>SC Ajust.</i>	<i>MC Ajust.</i>	<i>Valor F</i>	<i>Valor p</i>
T MAX	3	19375785	6458595	151,88	0,000
ALPHA	1	15887885	15887885	373,62	0,000
INSTANCIA	4	15971510839	3992877710	93897,28	0,000
T MAX*ALPHA	3	3188118	1062706	24,99	0,000
T MAX*INSTANCIA	12	6692611	557718	13,12	0,000
ALPHA*INSTANCIA	4	28538207	7134552	167,78	0,000
T MAX*ALPHA*INSTANCIA	12	3460501	288375	6,78	0,000
Error	160	6803823	42524		
Total	199	16055457768			

Tabla 6. Resumen del modelo – Minimización *Makespan*.

<i>S</i>	<i>R-cuad.</i>	<i>R-cuad.</i> <i>(ajustado)</i>	<i>R-cuad.</i> <i>(pred)</i>
206,213	99,96%	99,95%	99,93%

Posterior a este paso, se realizan las pruebas estadísticas para comprobar los tres supuestos del diseño de experimentos. No obstante, las pruebas de Levene y Kolmogorov-Smirnov obtuvieron valores-p menores a 0.05, indicando el incumplimiento de la homogeneidad de varianzas y normalidad, respectivamente. Por ello, se procede a realizar la prueba no paramétrica de Friedman, la cual no tiene estos supuestos, con el fin de corroborar los resultados del ANOVA (ver Tabla 7. Prueba de Friedman – Minimización *Makespan*.). Los resultados de la prueba de Friedman indican, con un valor-p < 0.001, que la combinación $Alpha = 0.20$, con un factor multiplicativo de tiempo $t = 0.50$, es el que mejores resultados obtiene para la función objetivo.

Tabla 7. Prueba de *Friedman* – Minimización *Makespan*.

<i>Alpha Tiempo</i>	<i>N</i>	<i>Mediana</i>	<i>Suma de clasificaciones</i>
0,2_0,001	5	7236,38	30,0
0,2_0,005	5	6984,63	21,0
0,2_0,010	5	6948,88	14,0
0,2_0,050	5	6541,00	8,0
0,4_0,001	5	7625,00	40,0
0,4_0,005	5	7320,75	32,0
0,4_0,010	5	6988,13	21,0
0,4_0,050	5	6657,25	14,0
General	40	7037,75	

Prueba

<i>GL</i>	<i>Chi-cuadrada</i>	<i>Valor p</i>
7	27,07	0,000

Tardiness

A través de un análisis de varianza, se procede a evaluar el efecto de los factores “Alpha”, “Tiempo” e “Iteraciones”, así como sus interacciones, de tal forma que sea posible confirmar la significancia de estos, con respecto a la minimización del *tardiness*. En la Tabla 8, se presentan los resultados obtenidos y, en la Tabla 9, el resumen del modelo.

Tabla 8. Análisis de varianza – Minimización *Tardiness*.

<i>Fuente</i>	<i>GL</i>	<i>SC Ajust.</i>	<i>MC Ajust.</i>	<i>Valor F</i>	<i>Valor p</i>
T MAX	3	1,48305E+11	49435082708	88,28	0,000
ALPHA	1	14672067691	14672067691	26,20	0,000
INSTANCIA	4	1,63389E+14	4,08473E+13	72942,17	0,000
ALPHA*T MAX	3	2440996706	813665569	1,45	0,229
T MAX*INSTANCIA	12	2,38477E+11	19873080853	35,49	0,000
ALPHA*INSTANCIA	4	30473673586	7618418396	13,60	0,000
ALPHA*T MAX*INSTANCIA	12	9668706927	805725577	1,44	0,153
Error	160	89599337290	559995858		
Total	199	1,63923E+14			

Tabla 9. Resumen del modelo – Minimización *Tardiness*.

<i>S</i>	<i>R-cuad.</i>	<i>R-cuad. (ajustado)</i>	<i>R-cuad. (pred)</i>
23664,2	99,95%	99,93%	99,91%

Posterior a este paso, se realizan las pruebas estadísticas para comprobar los tres supuestos del diseño de experimentos. No obstante, las pruebas de Levene y Kolmogorov-Smirnov obtuvieron valores-p menores a 0.05, indicando el incumplimiento de la homogeneidad de varianzas y normalidad, respectivamente. Por ello, se procede a realizar la prueba no paramétrica de Friedman, la cual no tiene estos supuestos, con el fin de corroborar los resultados del ANOVA (ver Tabla 10). Los resultados de la prueba de Friedman indican, con un valor-p < 0.001, que la combinación *Alpha* = 0.20, con un factor multiplicativo de tiempo *t* = 0.50, es el que mejores resultados obtiene para la función objetivo.

Tabla 10. Prueba de *Friedman* – Minimización *Tardiness*.

<i>Alpha</i>	<i>Tiempo</i>	<i>N</i>	<i>Mediana</i>	<i>Suma de clasificaciones</i>
0,2	0,001	5	93611,4	34,0
0,2	0,005	5	85092,0	24,0
0,2	0,010	5	79554,4	16,0
0,2	0,050	5	65592,6	6,0
0,4	0,001	5	99576,6	39,0
0,4	0,005	5	88363,6	29,0
0,4	0,010	5	82034,4	23,0
0,4	0,050	5	69827,0	9,0
General		40	82956,5	

Prueba

<i>GL</i>	<i>Chi-cuadrada</i>	<i>Valor p</i>
7	31,53	0,000

5.2.2. GRASP Ordenamiento Lexicográfico – Modelos con Restricción Asociada

MinMakespan/MinTardiness

A través de un análisis de varianza, se procede a evaluar el efecto de los factores “Alpha”, “Tiempo” e “Iteraciones”, así como sus interacciones, de tal forma que sea posible confirmar la significancia de estos, con respecto a la función objetivo. En la Tabla 11, se presentan los resultados obtenidos y, en la Tabla 12, el resumen del modelo.

Tabla 11. Análisis de varianza – *MinMakespan/MinTardiness*.

<i>Fuente</i>	<i>GL</i>	<i>SC Ajust.</i>	<i>MC Ajust.</i>	<i>Valor F</i>	<i>Valor p</i>
ALPHA	1	476017	476017	4,03	0,047
T MAX	2	636497	318249	2,70	0,072
INSTANCIA	4	9395560034	2348890008	19895,15	0,000
ALPHA*T MAX	2	195927	97963	0,83	0,439
ALPHA*INSTANCIA	4	2025265	506316	4,29	0,003
T MAX*INSTANCIA	8	2146479	268310	2,27	0,027
ALPHA*T	8	468466	58558	0,50	0,857
MAX*INSTANCIA					
Error	120	14167612	118063		
Total	149	9415676297			

Tabla 12. Resumen del modelo – *MinMakespan/MinTardiness*.

<i>S</i>	<i>R-cuad.</i>	<i>R-cuad. (ajustado)</i>	<i>R-cuad. (pred)</i>
343,604	99,85%	99,81%	99,76%

Posterior a este paso, se realizan las pruebas estadísticas para comprobar los tres supuestos del diseño de experimentos. No obstante, las pruebas de Levene y Kolmogorov-Smirnov obtuvieron valores-p menores a 0.05, indicando el incumplimiento de la homogeneidad de varianzas y normalidad, respectivamente. Por ello, se procede a realizar la prueba no paramétrica de Friedman, la cual no tiene estos supuestos, con el fin de corroborar los resultados del ANOVA (ver Tabla 13). Dado que no existe diferencia significativa en los resultados, es posible escoger cualquier valor de parametrización. Por lo tanto, se seleccionó el mismo *Alpha* obtenido en las parametrizaciones anteriores, $\text{Alpha} = 0.2$, y se seleccionó el factor multiplicativo de tiempo más alto ($t = 2.0$) para darle a la metaheurística el mayor tiempo posible, que permita obtener mejores resultados (pues, desde el punto de vista descriptivo, este valor obtuvo el menor promedio de función objetivo).

Tabla 13. Prueba de *Friedman* – *MinMakespan/MinTardiness*.

<i>Alpha</i>	<i>Tiempo</i>	<i>N</i>	<i>Mediana</i>	<i>Suma de clasificaciones</i>
0,2	1,0	5	6402,17	20,5
0,2	1,5	5	6428,83	23,0
0,2	2,0	5	6389,33	20,0
0,4	1,0	5	6392,17	15,0
0,4	1,5	5	6385,83	14,5
0,4	2,0	5	6383,67	12,0
General		30	6397,00	

MinTardiness/MinMakespan

A través de un análisis de varianza, se procede a evaluar el efecto de los factores “Alpha”, “Tiempo” e “Iteraciones”, así como sus interacciones, de tal forma que sea posible confirmar la significancia de estos, con respecto a la función objetivo. En la Tabla 14, se presentan los resultados obtenidos y, en la Tabla 15, el resumen del modelo.

Tabla 14. Análisis de varianza – *MinTardiness/MinMakespan*.

<i>Fuente</i>	<i>GL</i>	<i>SC Ajust.</i>	<i>MC Ajust.</i>	<i>Valor F</i>	<i>Valor p</i>
ALPHA	1	1561751840	1561751840	5,08	0,026
T MAX	2	5197350984	2598675492	8,45	0,000
INSTANCIA	4	8,50931E+13	2,12733E+13	69200,90	0,000
ALPHA*T MAX	2	665099135	332549567	1,08	0,342
ALPHA*INSTANCIA	4	4372754696	1093188674	3,56	0,009
T MAX*INSTANCIA	8	15327714559	1915964320	6,23	0,000
ALPHA*T MAX*INSTANCIA	8	1354478596	169309824	0,55	0,816
Error	120	36889608726	307413406		
Total	149	8,51585E+13			

Tabla 15. Resumen del modelo – *MinTardiness/MinMakespan*.

<i>S</i>	<i>R-cuad.</i>	<i>R-cuad. (ajustado)</i>	<i>R-cuad. (pred)</i>
17533,2	99,96%	99,95%	99,93%

Posterior a este paso, se realizan las pruebas estadísticas para comprobar los tres supuestos del diseño de experimentos. No obstante, las pruebas de Levene y Kolmogorov-Smirnov obtuvieron valores-p menores a 0.05, indicando el incumplimiento de la homogeneidad de varianzas y normalidad, respectivamente. Por ello, se procede a realizar la prueba no paramétrica de Friedman, la cual no tiene estos supuestos, con el fin de corroborar los resultados del ANOVA (ver Tabla 16). Dado que no existe diferencia significativa en los resultados, es posible escoger cualquier valor de parametrización. Por lo tanto, se seleccionó el mismo Alpha obtenido en las parametrizaciones anteriores, $Alpha = 0.2$, y se seleccionó el factor multiplicativo de tiempo más alto ($t = 2.0$) para darle a la metaheurística el mayor tiempo posible, que permita obtener mejores resultados (pues, desde el punto de vista descriptivo, este valor obtuvo el menor promedio de función objetivo).

Tabla 16. Prueba de *Friedman* – *MinTardiness/MinMakespan*.

<i>Alpha</i>	<i>Tiempo</i>	<i>N</i>	<i>Mediana</i>	<i>Suma de clasificaciones</i>
0,2	1,0	5	63326,3	19,0
0,2	1,5	5	62476,4	14,0
0,2	2,0	5	62660,8	12,0
0,4	1,0	5	65310,8	22,0
0,4	1,5	5	63540,8	19,0
0,4	2,0	5	63888,6	19,0
General		30	63533,9	

5.2.3. GRASP hibridizado con PAES

A través de un análisis de varianza, se procede a evaluar el efecto de los factores “Alpha”, “Tiempo” e “Iteraciones”, así como sus interacciones, de tal forma que sea posible confirmar la significancia de estos, con respecto al resultado del MID calculado. En la Tabla 17, se presentan los resultados obtenidos y, en la Tabla 18, el resumen del modelo.

Tabla 17. Análisis de varianza – *GRASP/PAES*.

<i>Fuente</i>	<i>GL</i>	<i>SC Ajust.</i>	<i>MC Ajust.</i>	<i>Valor F</i>	<i>Valor p</i>
ALPHA	1	6930075619	6930075619	15,23	0,000
TIEMPO	3	17828057140	5942685713	13,06	0,000
INSTANCIA	4	5,10427E+13	1,27607E+13	28039,69	0,000
ALPHA*TIEMPO	3	2454301532	818100511	1,80	0,163
ALPHA*INSTANCIA	4	21687211794	5421802949	11,91	0,000
TIEMPO*INSTANCIA	12	59542089928	4961840827	10,90	0,000
ALPHA*TIEMPO*INSTANCIA	12	8601705162	716808764	1,58	0,139
Error	40	18203717828	455092946		
Total	79	5,11779E+13			

Tabla 18. Resumen del modelo – GRASP/PAES.

<i>S</i>	<i>R-cuad.</i>	<i>R-cuad. (ajustado)</i>	<i>R-cuad. (pred)</i>
21332,9	99,96%	99,93%	99,86%

Posterior a este paso, se realizan las pruebas estadísticas para comprobar los tres supuestos del diseño de experimentos. No obstante, las pruebas de Levene y Kolmogorov-Smirnov obtuvieron valores-p menores a 0.05, indicando el incumplimiento de la homogeneidad de varianzas y normalidad, respectivamente. Por ello, se procede a realizar la prueba no paramétrica de Friedman, la cual no tiene estos supuestos, con el fin de corroborar los resultados del ANOVA (ver Tabla 19). Los resultados de la prueba de Friedman indican, con un valor-p < 0.001, que la combinación *Alpha* = 0.20, con un factor multiplicativo de tiempo *t* = 4.0, es el que mejores resultados obtiene para la función objetivo.

Tabla 19. Prueba de Friedman – GRASP/PAES.

<i>AlphaTiempo</i>	<i>N</i>	<i>Mediana</i>	<i>Suma de clasificaciones</i>
0,2 1,0	5	69228,4	27,0
0,2 1,5	5	65801,7	16,0
0,2 3,0	5	65040,7	17,0
0,2 4,0	5	65033,7	12,0
0,4 1,0	5	69644,3	37,0
0,4 1,5	5	68563,7	31,0
0,4 3,0	5	66908,9	24,0
0,4 4,0	5	65589,2	16,0
General	40	66976,3	

<i>GL</i>	<i>Chi-cuadrada</i>	<i>Valor p</i>
7	17,67	0,014

5.2.4. Comparación de los modelos individuales de metaheurística GRASP-Lexicográfico y el modelo matemático

En este apartado, se presenta la comparación de los resultados de los modelos matemáticos individuales respecto a los resultados obtenidos con los modelos individuales de la metaheurística GRASP-Lexicográfico. Para visualizar los resultados obtenidos con las metaheurísticas, ver Anexo 5. Tablas de Resultados.

Tabla 20. Comparación entre los modelos matemáticos individuales y los modelos individuales del GRASP Lexicográfico.

<i>Instancia</i>	<i>Modelos Individuales Gusek</i>		<i>GRASP Ordenamiento Lexicográfico (Modelos Individuales)</i>		<i>Diferencia Porcentual</i>	
	<i>Makespan</i>	<i>Tardiness</i>	<i>Makespan</i>	<i>Tardiness</i>	<i>Makespan</i>	<i>Tardiness</i>
5x2	325,50	57,75	334,80	61,95	2,86	7,27
5x3	397,75	37,25	398,05	37,25	0,08	0,00
5x4	428,00	38,25	444,90	38,25	3,95	0,00
6x2	305,50	100,25	324,90	100,25	6,35	0,00
6x3	425,00	89,50	434,95	93,75	2,34	4,75
6x4	539,75	129,25	560,65	133,65	3,87	3,40
7x2	388,00	184,75	400,85	184,75	3,31	0,00
7x3	442,25	30,00	479,15	37,75	8,34	25,83
7x4	566,75	0,00	596,70	8,9	5,28	-
8x2	468,50	71,25	480,40	75,85	2,54	6,46
8x3	494,00	132,75	510,40	163,55	3,32	23,20
8x4	597,75	43,50	621,70	43,5	4,01	0,00

9x2	522,50	78,75	527,90	98,6	1,03	25,21
9x3	578,50	49,50	598,30	0	3,42	-
9x4	628,00	45,25	651,80	45,95	3,79	1,55
10x2	529,75	154,25	540,05	157,95	1,94	2,40
10x3	673,25	133,50	692,30	163,65	2,83	22,58
10x4	757,75	153,75	775,95	167,7	2,40	9,07
15x2	634,00	45,25	802,15	68,9	26,52	52,27
15x3	926,25	0,00	962,65	5,35	3,93	-
15x4	1029,00	30,00	1083,80	30	5,33	0,00
20x5	1602,8	6197,5	1583,80	5019,8	-1,19	-19,00
20x10	2258,6	5101,9	2125,66	2424,6	-5,89	-52,48
20x20	3445,8	6028,4	3163,80	1296,78	-8,18	-78,49
50x5	3946,1	69380,8	3561,48	55874,16	-9,75	-19,47
50x10	5597,4	89348	4743,70	61439,46	-15,25	-31,24
50x20	8051	-	6602,34	66802,5	-17,99	-
100x5	-	-	7350,00	284166,98	-	-
100x10	-	-	9689,44	284166,98	-	-
100x20	-	-	13084,06	428040,88	-	-
200x10	-	-	19245,46	1669929,5	-	-
200x20	-	-	25473,80	2147552,56	-	-

A partir de los resultados encontrados, se evidencia que, en las instancias pequeñas, se presenta una diferencia porcentual menor al 10% entre los valores obtenidos del GRASP con ordenamiento lexicográfico en sus modelos individuales, con respecto a los modelos matemáticos de Gusek. Además, a través de la metaheurística, la variable de *makespan* mejora en dicho tamaño. Por otra parte, en cuanto a las instancias grandes, las diferencias porcentuales empiezan a ser negativas debido a que, a través del GRASP, se encuentra un mejor resultado, en menos tiempo, evidenciando así la capacidad de este para procesar instancias de mayor tamaño, mientras que, en caso de correr estas instancias en Gusek o con ayuda de CMD, se necesitaría una gran cantidad de tiempo para tratar de encontrar un resultado óptimo. Ver Anexo 5. Tablas de Resultados.

5.2.5. Resultados Metaheurísticas GRASP-Lexicográfico y GRASP-PAES

En la Tabla 21, se presentan los resultados obtenidos del experimento general, teniendo en cuenta los parámetros establecidos, en donde el GRASP con abordaje de ordenamiento lexicográfico contempla *Alpha* de 0.2 y factor multiplicativo de tiempo $t = 2.0$ segundos, mientras que el GRASP-PAES utiliza *Alpha* de 0.2 y factor multiplicativo de tiempo $t = 4.0$ segundos. Para visualizar los resultados obtenidos con las metaheurísticas, ver Anexo 5.

Tabla 21. Promedios Resultados Metaheurísticas.

Instancia	GRASP Lexicográfico (Con Restricción Asociada)								GRASP/PAES				Tiempo Ejecución
	Makespan SA Tardiness				Tardiness SA Makespan				Extremo de la frontera con mejor <i>Makespan</i>		Extremo de la frontera con mejor <i>Tardiness</i>		
	Makespan	Tardiness	Tiempo Ejecución (s)	Min Tardiness	Makespan	Tardiness	Tiempo Ejecución (s)	Min Makespan	Makespan	Tardiness	Makespan	Tardiness	
5x2	343,55	61,95	0,0032	61,95	334,80	100,35	0,0031	334,80	332,75	92,25	350	57,75	0,2900
5x3	397,75	37,25	0,0059	37,25	397,75	37,25	0,0053	398,05	414,5	77,75	419,75	67	0,1875
5x4	493,05	38,25	0,0050	38,25	444,90	63,85	0,0075	444,90	446	75,75	505,00	39	0,2050
6x2	321,90	100,25	0,0076	100,25	324,90	126,30	0,0076	324,90	324	127,75	341,50	102	0,2275
6x3	475,00	93,75	0,0079	93,75	434,95	141,25	0,0075	434,95	441,5	212,50	455,50	96	0,3000
6x4	552,95	133,65	0,0098	133,65	560,65	168,90	0,0089	560,65	554,25	145,50	555,00	129	0,3300
7x2	408,95	184,75	0,0105	184,75	400,85	277,80	0,0095	400,85	397	251,75	415,50	231	0,6550
7x3	521,50	37,75	0,0096	37,75	479,15	83,90	0,0130	479,15	485,25	71,50	486,75	42	0,6225
7x4	624,85	8,90	0,0156	8,90	596,70	88,25	0,0143	596,70	608,5	33,50	634,25	10	0,6050
8x2	499,00	75,85	0,0161	75,85	480,40	230,60	0,0138	480,40	480	174,50	498,25	88	0,7900
8x3	532,80	163,55	0,0178	163,55	509,50	223,65	0,0213	510,40	503	243,75	514,50	177	1,0725
8x4	642,40	43,50	0,0259	43,50	621,70	100,65	0,0220	621,70	603	182,00	655,50	44	0,9425
9x2	538,25	98,60	0,0206	98,60	527,90	235,25	0,0202	527,90	530,25	168,50	550,25	97	1,4300
9x3	609,70	0,00	0,0320	0,00	597,10	64,70	0,0298	598,30	599,25	61,75	600,00	0	1,2825
9x4	712,30	45,95	0,0348	45,95	651,80	223,65	0,0311	651,80	645,5	153,00	690,25	45	1,7375
10x2	553,80	157,95	0,0300	157,95	539,45	324,75	0,0329	540,05	549	186,25	554,25	154	1,2525
10x3	745,15	163,65	0,0322	163,65	691,50	256,60	0,0424	692,30	696,5	386,50	725,75	134	1,6375
10x4	827,25	167,70	0,0446	167,70	775,70	348,30	0,0440	775,95	807,5	3772,50	833,75	256	1,7525
15x2	816,25	68,90	0,1494	68,90	802,15	291,55	0,1419	802,15	803	60,75	803,00	61	4,1825
15x3	978,05	5,35	0,1631	5,35	962,20	280,65	0,1741	962,65	1054	117,25	1054,00	117	8,5900
15x4	1114,05	30,00	0,2518	30,00	1083,80	152,10	0,2240	1083,80	1111,5	30,00	1111,50	30	5,4175
20x5	1646,56	5029,80	0,5860	5019,80	1582,28	6145,92	0,4879	1583,80	1598,9	5592,10	1640,9	5207	8,2720
20x10	2321,32	2439,84	1,3531	2424,60	2125,44	4979,76	0,9457	2125,66	2129,5	5116,90	2314,5	3034	49,2330
20x20	3592,24	1296,78	2,5561	1296,78	3163,80	5279,88	1,9930	3163,80	3222,3	5144,80	3533,5	2322	83,6010
50x5	3518,70	55027,18	34,2446	55874,16	3531,66	56049,92	27,2734	3561,48	3505,6	56389,70	3566,3	55652	505,8040
50x10	4683,86	61072,48	44,7588	61439,46	4682,46	62431,80	53,3919	4743,70	4617,6	61795,50	4695,1	61175	731,7070
50x20	6513,02	66044,52	124,4001	66802,50	6503,48	67642,96	131,3114	6602,34	6454,3	69046,70	6650,7	66767	2195,5140
100x5	7890,20	255492,96	998,3260	284166,98	6754,08	248198,52	795,5866	7350,00	7243,8	266952,90	7278,4	264420	2001,2580
100x10	7890,20	255492,96	998,3260	284166,98	8824,12	305539,04	1441,4211	9689,44	9416,6	324437,50	9429,4	323653	4002,3400
100x20	11781,20	378111,26	3229,9565	428040,88	11951,28	364315,54	3816,9373	13084,06	12556,1	391240,70	12600,9	389068	8004,6000
200x10	17885,70	1429189,4	5405,5505	1669929,5	18322,92	1462449,90	4261,9201	19245,46	19374,4	1542402,90	19387,5	1540599	8016,3920
200x20	23734,32	1818283,2	10307,5451	2147552,36	24276,42	1867348,62	8815,5146	25473,80	25418,4	1954953,70	25446,3	1950128	19254,5030

A partir de los promedios de los resultados obtenidos, es posible evidenciar que el 34,375% del total de los tamaños, que involucra desde las instancias pequeñas creadas hasta las más grandes de *Taillard*, se ve favorecido por el GRASP con abordaje de ordenamiento lexicográfico, en donde se evidencia una tendencia a tener alta participación en las de mayor tamaño, es decir, de 100 trabajos en adelante. Finalmente, el 25% de los tamaños se ve favorecido por el GRASP-PAES, en donde la tendencia corresponde a tener participación en los tamaños grandes de las instancias pequeñas creadas y en los tamaños más pequeños de las instancias *benchmark*.

Ahora bien, con respecto al GRASP con abordaje de ordenamiento lexicográfico, se evidencia que todos los tamaños de las instancias tienen una diferencia menor o igual al 1%, al contrastar los resultados de los modelos con restricción asociada respecto a los mejores valores de los extremos de la frontera de Pareto, para cada una de las variables involucradas (*makespan* y *tardiness*). Esto se obtiene al calcular la diferencia porcentual tomando como referencia el GRASP hibridizado con PAES. Además, de la totalidad de tamaños, el 31,25% no encuentra una solución mejor a la de los límites inferiores de la frontera, mientras que, por el contrario, el porcentaje restante, si encuentra un mejor resultado. Por otra parte, se observa que, para las instancias de mayor tamaño, es decir de 100 trabajos en adelante, los valores obtenidos con el GRASP de ordenamiento lexicográfico son mejores, en su totalidad, respecto al GRASP hibridizado con PAES.

Por otra parte, se evidencia que el 15,625% del total de los resultados del lexicográfico mejoran en ambas funciones objetivo al extremo correspondiente de la frontera de Pareto obtenida por el GRASP-PAES, presentándose la mayoría de estos casos en las instancias de 100 trabajos en adelante. Por ello, es posible concluir que, en caso de utilizar un factor multiplicativo de tiempo más grande y, por ende, un mayor tiempo límite de ejecución del algoritmo GRASP-PAES, es probable que se encuentren resultados tan buenos como los que arroja el GRASP con ordenamiento lexicográfico en las instancias de gran tamaño.

6. Limitaciones, conclusiones y recomendaciones.

En este proyecto se resuelve el problema de programación de la producción en un ambiente *Blocking Permutation Flow Shop* (BPFS) para minimizar la tardanza total y el *makespan*. En primera instancia se propusieron los modelos matemáticos individuales de *makespan* y tardanza, para posteriormente incluirlos como restricciones de los modelos matemáticos lexicográficos. El modelo matemático fue resuelto para instancias pequeñas. Posteriormente se propuso como método de solución para instancias medianas y grandes una metaheurística GRASP con dos enfoques: el enfoque de ordenamiento lexicográfico y el enfoque de la frontera de Pareto logrado a través de la hibridización del GRASP con la metodología Pareto Archived Evolution Strategy (PAES).

Ahora bien, el modelo matemático fue implementado en Gusek y NEOS, logrando resolver de manera óptima 84 instancias pequeñas creadas para probar los modelos. Sin embargo, dado que el problema planteado es *NP Hard*, para las instancias de 10 y 15 trabajos ni en Gusek, ni en NEOS se logró obtener respuesta óptima. Por ello, para estas instancias, se programó, posteriormente, en Gusek un tiempo límite de 7200 segundos (2 horas), a través del uso de la consola de Microsoft (CMD), de tal forma que se obtuviera la mejor solución factible encontrada hasta ese momento. Este tiempo límite de 2 horas permitió obtener resultados de soluciones factibles para algunas instancias de 50 trabajos, pero no para instancias de mayor tamaño.

Por otro lado, a través del contraste entre el modelo matemático y la metaheurística, ambos evaluando los modelos individuales, se evidencia una amplia ventaja en el tiempo de corrida, ya que, el GRASP-Lexicográfico permite encontrar soluciones óptimas en menor tiempo, sin importar el tamaño de la instancia, mientras que Gusek tarda más y, para las instancias grandes, no encuentra un valor óptimo a menos que se límite el modelo, generando el mejor resultado hasta el momento.

Además, la metaheurística GRASP, en sus dos enfoques, fue implementada para las 84 instancias pequeñas y para las 110 instancias grandes *benchmark* de *Taillard*. Los resultados de la metaheurística en sus dos enfoques fueron comparados entre sí obteniendo que las instancias más grandes obtienen mejores resultados por medio del GRASP con ordenamiento lexicográfico y que las instancias medianas se veían mejor favorecidas por el GRASP hibridizado con PAES. Por tanto, es probable que al utilizar un factor multiplicativo mayor para limitar el tiempo de procesamiento de cada iteración del GRASP-PAES, se obtengan resultados tan buenos como los que se obtienen en el lexicográfico para las instancias más grandes.

Por otra parte, se compararon los dos enfoques del GRASP con ordenamiento lexicográfico para los modelos con restricción asociada respecto al mejor valor de la frontera de Pareto, para cada una de las variables involucradas

(*makespan* y tardanza). Este contraste evidenció que todos los resultados obtenidos a través del ordenamiento lexicográfico son similares respecto al mejor valor de los extremos de las fronteras, para todos los tamaños de las instancias, en al menos alguna de las funciones objetivo evaluadas. Finalmente, las instancias grandes (de 100 trabajos en adelante) obtienen mejores resultados con el abordaje de ordenamiento lexicográfico, lo que permite afirmar que, en caso de aumentar el factor multiplicativo de tiempo, asociado al tiempo máximo de ejecución del GRASP híbrido con PAES, es posible obtener mejores resultados, equiparando los valores hallados con el ordenamiento lexicográfico.

Como futuros trabajos se recomienda probar otras funciones de utilidad dentro del GRASP y explorar otras estrategias de intercambio para el PAES. En este proyecto, la estrategia dentro del PAES fue realizar intercambios de posiciones entre pares de trabajos, pero podrían probarse otras tales como inserción o intercambio entre tres posiciones en la secuencia. Adicionalmente, se sugiere agregar tiempos de alistamiento a este problema y probar adicionando otra función objetivo. Finalmente, se pueden implementar estos dos métodos en otros problemas de programación de la producción u otros problemas combinatorios.

7. Anexos

En este apartado, se relaciona el listado de anexos complementarios al documento.

- Anexo 1. Flujogramas.
- Anexo 2. Resultados Instancias de Prueba Modelos Individuales Gusek.
- Anexo 3. Resultados Instancias de Prueba Modelos Con Restricción Variable Asociada Gusek.
- Anexo 4. Parametrización.
- Anexo 5. Tablas de Resultados.

8. Glosario

Blocking: Es un fenómeno que ocurre si un *flow shop* tiene la capacidad limitada de buffer entre dos máquinas consecutivas. Luego, si el buffer se encuentra lleno la máquina anterior a este no tiene permitido entregar un trabajo completo (Pinedo, 2016).

Flow shop: Ambiente de programación con m máquinas en serie en donde todos los trabajos deben pasar por cada una de ellas, de tal manera que cada una siga la misma ruta (Pinedo, 2016).

GRASP: El codicioso procedimiento de búsqueda adaptativa, es un algoritmo de aproximación, cuyo objetivo es diseñar un procedimiento de solución que funcione bien bajo varios criterios de optimización (González-Neira & Montoya-Torres, 2017)

Makespan: Dado que C_j es el tiempo de terminación de un trabajo. El *makespan* se define como C_{\max} de (C_1, C_2, \dots, C_n) . Siendo así, el equivalente al tiempo de terminación del último trabajo que abandona el sistema (Pinedo, 2016)

Metaheurística: Son algoritmos estocásticos principalmente utilizados para resolver problemas de tipo NP-hard, inspirados en el comportamiento de la naturaleza como técnicas de búsqueda local o estrategias de alto nivel que examinan un conjunto de soluciones al mismo tiempo (Boumediene, Houbad, Bessenouci, & Ghomri, 2018)

NP-hard: Hace referencia a que toda la clase NP polinomialmente se reduce a P, siendo P el espacio que contiene todos los problemas de decisión (Pinedo, 2016)

Permutation: Es una restricción que se puede presentar cuando las máquinas operan de acuerdo a la disciplina FIFO (First in, first out), lo que implica que el orden o permutación en el que pasan los trabajos se mantiene en el sistema (Pinedo, 2016)

Tardanza: Tiempo de terminación de las diferentes tareas después de su respectiva fecha de vencimiento. (Pinedo, 2016)

Referencias

- Anjana, V., Sridharan, R., & Ram Kumar, P. N. (2019). Metaheuristics for solving a multi-objective flow shop scheduling problem with sequence-dependent setup times. *Journal of Scheduling*, (2006). <https://doi.org/10.1007/s10951-019-00610-0>
- Arroyo, J. E. C., & De Souza Pereira, A. A. (2011). A GRASP heuristic for the multi-objective permutation flowshop scheduling problem. *International Journal of Advanced Manufacturing Technology*, 55(5–8), 741–753. <https://doi.org/10.1007/s00170-010-3100-x>
- Caio Soares de Araújo, & Fuchigami, H. Y. (2018). A COMPUTATIONAL STUDY OF NEW PRIORITY RULES FOR FLOW SHOP PROBLEM WITH SETUP TIMES AND DIFFERENT RELEASE DATES. *Journal of Chemical Information and Modeling*, 53(9), 1689–1699. <https://doi.org/10.1017/CBO9781107415324.004>
- Ciatera Díaz, D. (2015). Modelo de Optimización Multiobjetivo para Evaluación de Eficiencia en una Empresa de Servicios Eléctricos. *Departamento de Ingeniería Industrial, Ingeniería*, 91. Retrieved from [http://repositoriodigital.ucsc.cl/bitstream/handle/25022009/133/David Ciatera Díaz.pdf?sequence=1&isAllowed=y](http://repositoriodigital.ucsc.cl/bitstream/handle/25022009/133/David%20Ciatera%20D%C3%ADaz.pdf?sequence=1&isAllowed=y)
- Ciavotta, M., Minella, G., & Ruiz, R. (2013). Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study. *European Journal of Operational Research*, 227(2), 301–313. <https://doi.org/10.1016/j.ejor.2012.12.031>
- Clewett, A. J., & Baker, K. R. (1977). Introduction to Sequencing and Scheduling. *Operational Research Quarterly (1970-1977)*. <https://doi.org/10.2307/3009192>
- Croes, G. A. (1958). A method for solving traveling-salesman Problems. *Operations Research*, 6(6), 791–812. <https://doi.org/10.1287/opre.6.6.791>
- Dhingra, A., & Chandna, P. (2010). Multi-objective flow shop scheduling using hybrid simulated annealing. *Measuring Business Excellence*, 14(3), 30–41. <https://doi.org/10.1108/13683041011074191>
- 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>
- Fu, J., & Wen, X. H. (2017). Model-based multi-objective optimization methods for efficient management of subsurface flow. *Society of Petroleum Engineers - SPE Reservoir Simulation Conference 2017*, (September 2017), 159–178. <https://doi.org/10.2118/182598-pa>
- Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The Complexity of Flowshop and Jobshop Scheduling. *Mathematics of Operations Research*, 1(2), 117–129. <https://doi.org/10.1287/moor.1.2.117>
- Guanlong, D., Shuning, Z., & Mei, Z. (2016). A discrete group search optimizer for blocking flow shop multi-objective scheduling. *Advances in Mechanical Engineering*, 8(8), 1–9. <https://doi.org/10.1177/1687814016664262>
- Hanoun, S., & Nahavandi, S. (2012). A greedy heuristic and simulated annealing approach for a bicriteria flowshop scheduling problem with precedence constraints-a practical manufacturing case. *International Journal of Advanced Manufacturing Technology*, 60(9–12), 1087–1098. <https://doi.org/10.1007/s00170-011-3650-6>
- Hosseini, N., & Tavakkoli-Moghaddam, R. (2013). Two meta-heuristics for solving a new two-machine flowshop scheduling problem with the learning effect and dynamic arrivals. *International Journal of Advanced Manufacturing Technology*, 65(5–8), 771–786. <https://doi.org/10.1007/s00170-012-4216-y>
- Karimi, N., Zandieh, M., & Karamooz, H. R. (2010). Bi-objective group scheduling in hybrid flexible flowshop: A multi-phase approach. *Expert Systems with Applications*, 37(6), 4024–4032. <https://doi.org/10.1016/j.eswa.2009.09.005>
- Knowles, J. D., & Corne, D. W. (2000). Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2), 149–172. <https://doi.org/10.1162/106365600568167>
- Lebbar, G., El Abbassi, I., Jabri, A., El Barkany, A., & Darcherif, M. (2018). Multi-criteria blocking flow shop scheduling problems: Formulation and performance analysis. *Advances in Production Engineering And Management*, 13(2), 136–146. <https://doi.org/10.14743/apem2018.2.279>
- Liu, L. L., Zhao, G. P., Ou'Yang, S. S., & Yang, Y. J. (2011). Integrating theory of constraints and particle swarm optimization in order planning and scheduling for machine tool production. *International Journal of Advanced Manufacturing Technology*, 57(1–4), 285–296. <https://doi.org/10.1007/s00170-011-3294-6>
- Martí, R., Campos, V., Resende, M. G. C., & Duarte, A. (2015). Multiobjective GRASP with path relinking. *European Journal of Operational Research*, 240(1), 54–71. <https://doi.org/10.1016/j.ejor.2014.06.042>
- Minella, G., Ruiz, R., & Ciavotta, M. (2008). A Review and Evaluation of Multiobjective Algorithms for the Flowshop Scheduling Problem. *INFORMS Journal on Computing*, 20(3), 451–471. <https://doi.org/10.1287/ijoc.1070.0258>
- Pasupathy, T., Rajendran, C., & Suresh, R. K. (2006). A multi-objective genetic algorithm for scheduling in flow shops to minimize the

makespan and total flow time of jobs. *International Journal of Advanced Manufacturing Technology*.
<https://doi.org/10.1007/s00170-004-2249-6>

Pinedo, M. L. (2016). *Scheduling*.

Resende, M. G. C. (1998). *Greedy randomized adaptive search procedures (grasp)*. 1–11.

Sha, D. Y., & Hung Lin, H. (2009). A particle swarm optimization for multi-objective flowshop scheduling. *International Journal of Advanced Manufacturing Technology*, 45(7–8), 749–758. <https://doi.org/10.1007/s00170-009-1970-6>

Shao, Z., Pi, D., & Shao, W. (2019). A novel multi-objective discrete water wave optimization for solving multi-objective blocking flow-shop scheduling problem. *Knowledge-Based Systems*, 165, 110–131. <https://doi.org/10.1016/j.knosys.2018.11.021>

Wiers, V. C. S. (1997). *Human-Computer Interaction in Production Scheduling*.

Yang, Z., & Liu, C. (2018). A hybrid multi-objective gray wolf optimization algorithm for a fuzzy blocking flow shop scheduling problem. *Advances in Mechanical Engineering*, 10(3), 1–13. <https://doi.org/10.1177/1687814018765535>

Yenisey, M. M., & Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega*, 45, 119–135. <https://doi.org/10.1016/J.OMEGA.2013.07.004>