UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dpto. de Estadística e Investigación Operativa Aplicadas y Calidad

Optimización de una línea de soldadura multimodelo en el sector del automóvil: heurísticas para determinar la secuencia de producción

Trabajo Fin de Máster

Máster Universitario en Ingeniería de Análisis de Datos, Mejora de Procesos y Toma de Decisiones

AUTOR/A: Vizcaino Hilario, Judith

Tutor/a: Vallada Regalado, Eva

Cotutor/a externo: YEPES BORRERO, JUAN CAMILO

CURSO ACADÉMICO: 2022/2023

Agradecimientos

Quiero agradecer a mi familia por su apoyo incondicional y a mis compañeros/amigos del máster por hacer el camino más ameno. También quiero agradaecer a ValgrAI (*Valencian Graduate School and Research Network for Artificial Intelligence* y a la Generalitat Valenciana por su apoyo económico.

Resumen

Uno de los principales retos de la industria consiste en la optimización inteligente de sus líneas de producción con el propósito de incrementar la productividad, mejorar la calidad y reducir costes, así como ajustarse a los tiempos de ciclo preestablecidos.

Por este motivo, el presente Trabajo Fin de Máster tiene como objetivo realizar una revisión bibliográfica en profundidad acerca de los métodos de programación de la producción más trabajados en la literatura. En particular, centraremos nuestro estudio en el Problema de Permutaciones Flow Shop, cuyo principal alcance consiste en obtener la secuencia óptima de un conjunto de trabajos que deben ser procesados siguiendo el mismo orden por un conjunto de máquinas, de manera que el tiempo de completación de los trabajos sea mínimo. Así mismo, se intentará ampliar el alcance de la funcionalidad incluyendo en la función objetivo otros medibles que resultan ser de especial importancia para el mundo industrial, cuya optimización permite, además de aumentar la eficiencia de las líneas de producción y reducir los costes asociados, mejorar la eficiencia energética de las máquinas.

Se debe tener en cuenta que el Problema de Permutaciones de Flow Shop forma parte del conjunto de problemas NP-hard, caracterizados por su alto coste computacional. Por este motivo, como parte de la revisión bibliográfica, se estudiará la posibilidad de aplicar algoritmos heurísticos y metaheurísticos con el objetivo de obtener soluciones en tiempos razonables. Mediante una comparación numérica, concluiremos cuál sería, en términos de tiempos de computación y en calidad de la solución, el mejor algoritmo a ser considerado e implementado.

Cabe recalcar que el proyecto surge a raíz de una necesidad en la planta de carrocerías de una factoría del sector del automóvil donde se ha generado un creciente interés por determinar la manera óptima de secuenciar la producción multimodelo en una línea de soldadura. En particular, se pretende minimizar tanto el tiempo de ciclo como los tiempos de espera y de bloqueo de las estaciones que provocan períodos de inactividad de la maquinaria y, por tanto, pérdidas por desaprovechamiento.

Por este motivo, el algoritmo propuesto será implementado en dicha línea de soldadura de la factoría objeto de estudio, la cual cuenta con 8 estaciones de trabajo diferenciadas y por la cual son procesados 68 variantes de modelos diferentes. La simulación a partir de datos pasados reales nos permite obtener la producción real de un determinado rango de tiempo y secuenciar de manera inteligente los trabajos a procesar.

Índice general

Capítulo 1

Introducción

Una línea de producción, fabricación o montaje puede ser definida como el conjunto de operaciones secuenciales que se establecen en una factoría a través de las cuales se obtiene el producto final construído. De esta manera, cada máquina u operario es responsable de realizar uno o más trabajos pertenecientes al conjunto de operaciones que deben ser realizados sobre el producto, el cual va pasando por todas ellas de la manera secuencial determinada. El principal escenario para el cual se desarrollaron las líneas de montaje es el de producción en masa de productos más o menos estandarizados y muy similares, con pocas variantes o modelos. Sin embargo, muchos sistemas de producción han debido de adaptar sus productos a la demanda de los clientes, permitiendo así un alcance individualizado del producto final, con multitud de opciones, variantes y particularidades. Este nuevo sistema de producción donde se contempla la posibilidad de adaptar las peculiaridades del producto al cliente y donde surgen múltiples variantes recibe el nombre de Sistema Multimodelo. Este tipo de sistema es el más común en las líneas de producción reales. En estos casos, es necesario preestablecer un mix o secuencia de los productos en función de su similitud o variabilidad respecto los adyacentes. Desde hace años, este problema ha sido de gran interés para los investigadores, por lo que es fácil encontrar en la literatura científico-técnica propuestas de algoritmos de optimización que tratan de reprogramar las configuraciones de las líneas de montaje. Sin embargo, sólo en un porcentaje muy pequeño de empresas realmente se llevan a cabo algoritmos matemáticos para configurar y programar la producción. Este tipo de procedimientos son conocidos como ALB (*Assembly Line Balancing*).

A parte de la configuración de la línea y la variedad de peculiaridades de los productos fabricados, otro gran problema en la industria manufacturera son los cuellos de botella. En muchas ocasiones, el cuello de botella es causado por el bloqueo de líneas posteriores y la no disponibilidad de las anteriores. En este caso particular, el cuello de botella estaría influenciado por una situación en la que el próximo trabajo no ha terminado de procesarse en la estación anterior, lo que llamamos tiempo de espera, o la siguiente estación no ha terminado de procesar el trabajo anterior (tiempo de bloqueo). El objetivo final sería detectar el cuello de botella en una línea de producción en tiempo real, lo que permitiría priorizar tareas de mantenimiento preventivo o, si es posible, evitar estos vacíos de producción al secuenciar los trabajos.

1.1. Motivación

La semilla de la presente investigación surge a raíz de una necesidad en la planta de carrocerías de una factoría del sector del automóvil, donde se ha generado un creciente interés por determinar la manera óptima de secuenciar los productos a fabricar. Con el añadido de tratarse de una línea de soldadura multimodelo, se ha generado un creciente problema de ineficiencia y desaprovechamiento de la línea, causado por los tiempos de espera y de bloqueo en las estaciones.

Además, otro de los alcances es aprovechar datos en tiempo real y algoritmos automáticos para obtener una secuencia de fabricación que se alinee con el estado actual de la línea de producción. Este enfoque permite una mejor capacidad de respuesta a los cambios, minimiza los cuellos de botella y maximiza la eficiencia al asegurarse de que cada trabajo se programe adecuadamente según las condiciones en tiempo real de la línea de producción.

1.2. Objetivos

En términos generales, el objetivo de la investigación es encontrar el mejor algoritmo de secuenciación que se adapte al estado en tiempo real de la línea de producción mediante la comparación de varios algoritmos. Esto tiene como objetivo minimizar los tiempos de espera y bloqueo, maximizar la capacidad de producción y minimizar tiempo de computación del algoritmo. El propósito es optimizar la programación y secuenciación de tareas en la línea de producción, reduciendo cualquier retraso o tiempo inactivo innecesario. En definitiva, se tratará de mejorar la productividad general y la eficiencia operativa, al tiempo que maximiza el rendimiento de la línea de producción.

Para alcanzar nuestro objetivo, hemos recurrido a uno de los problemas más estudiados en la literatura de procesos de fabricación. El Problema de Programación de Flujo de Trabajo de Permutación (PFSP, por sus siglas en inglés) es un desafío común en los procesos de producción y cadenas de suministro que se ocupa de optimizar la secuencia de trabajos que se procesarán en un conjunto de máquinas, generalmente con el objetivo de minimizar el tiempo total de finalización. Se supone que todos los trabajos siguen el mismo orden en cada máquina, sin procesos paralelos. Además, el problema asume que cada trabajo puede ser procesado por solo una máquina a la vez, sin interrupciones.

De acuerdo a lo comentado, se establecen como objetivos del presente trabajo los siguientes aspectos a desarrollar:

- Introducir el *Permutation Flow Shop Scheduling Problem* y presentar la formulación matemática de este problema.
- Introducir el carácter multicriterio de este tipo de problemas para modelizar con mayor precisión las casuísticas y necesidades reales de una línea de producción, así como la definición matemática de los objetivos planteados.
- Revisar en la literatura algoritmos heurísticos propuestos para la resolución del *Permutation Flow Shop Scheduling Problem* y analizar los alcances en cada caso. Plantear ejemplos sencillos para comprender su funcionamiento en exactitud.

- Implementar computacionalmente los algoritmos.
- Calibrar los hiperparámetros de los algoritmos en caso de ser necesario mediante simulaciones con instancias de diferentes tamaños. Analizar los resultados y sacar conclusiones sobre la aleatorización de los métodos.
- Comparar numéricamente los algoritmos propuestos mediante simulaciones con instancias de diferentes magnitudes para concluir qué algortimo conviene utilizar en cada caso.
- Introducir un caso de estudio real en la línea de carrocerías de Ford Almussafes. Modelizar matemáticamente la situación de una línea de soldadura y aplicar los métodos heurísticos propuestos.

En particular, a continuación definimos de qué manera se va a lograr el alcance definido. Para ello, definimos la estructura general del trabajo y los contenidos que se desarrollarán en cada sección.

- **Capítulo 2**: En esta sección se realiza una revisión bibliográfica sobre el problema a tratar. Se introduce el problema *Permutation FLow Shop Scheduling Problem*, sus variaciones y se modeliza matemáticamente el problema y las funciones objetivo desarrolladas.
- **Capítulo 3**: En este capítulo se realiza una revisión bibliográfica sobre algoritmos heurísticos propuestos en la literatura y se adapta sus funcionalidades a los requirimientos deseados. Además, se realiza la implementación en RStudio de los algoritmos desarrollados y se explican las particularidades añadidas.
- **Capítulo 4**: En este capítulo se realizan experimentos computacionales con instancias de ejemplo de diferentes tamaños con el objetivo de, por una parte, calibrar los hiperparámetros de los algoritmos y, por otra parte, comparar númericamente qué algoritmo nos proporciona mejores soluciones en diferentes situaciones. Así mismo, se define la función objetivo en base a los diferentes criterios a considerar.
- **Capítulo 5**: En este último capítulo se introduce el caso de estudio a analizar, sus alcances, sus objetivos y el contexto industrial que lo envuelve. Se formula matemáticamente el problema *Permutation FLow Shop Scheduling Problem*, particularizando a la línea de soldadura estudiada.

Capítulo 2

Descripción del problema y revisión bibliográfica

2.1. Introducción al *Permutation Flow Shop Scheduling Problem*

Un problema frecuente en los procesos productivos y cadenas de suministro es el de secuenciar de manera óptima los trabajos a realizar en un conjunto de máquinas determinadas.

El problema que se encarga de abordar esta cuestión es el de *Flowshop Scheduling Problem*, denotado por FSP. El FSP se caracteriza por un conjunto de trabajos que deben ser procesados por un conjunto de máquinas de manera que todos los trabajos visiten todas las máquinas siguiendo el mismo orden, es decir, con la misma ruta. En particular, existe una variante del problema en la que se supone que el orden de los trabajos es el mismo para todas las máquinas. Esta variante recibe el nombre de *Permutation Flowshop Scheduling Problem* (PFSP).

Por lo general, el propósito del PFSP es encontrar la mejor secuencia de trabajos a procesar tal que el tiempo de completación total sea mínimo, es decir, el tiempo que tarda el último trabajo en finalizar su proceso. Este valor recibe el nombre de *makespan*, se denota por C_{max} y corresponde a la función objetivo del problema de optimización. Sin embargo, en función del problema que se quiera abordar, el problema puede tomar diferentes funciones objetivo.

Además, en el PFSP debemos considerar una serie de hipótesis:

- Cada trabajo puede ser únicamente procesado por una máquina al mismo tiempo, es decir, las máquinas no pueden procesar en paralelo un mismo trabajo.
- Un trabajo sólo puede ser procesado en la máquina siguiente una vez haya terminado de ser procesado en la máquina anterior.
- De manera análoga, si un trabajo ha terminado su procesamiento en una máquina pero la máquina siguiente sigue ocupada, debe esperar a que la siguiente máquina acabe de procesar el anterior trabajo.
- Una máquina puede procesar un único trabajo a la vez.
- El procesamiento de un trabajo en una máquina no puede ser interrumpido.

- Todos los trabajos son independientes entre ellos.
- Todos los trabajos están disponibles para su procesado en el instante inicial.
- La disponibilidad de las máquinas es continua.
- Si una máquina está procesando un trabajo determinado, el siguiente trabajo secuenciado puede esperar a que la máquina termine el procesado del anterior.
- Cada máquina procesa todos los trabajos en el mismo orden. Por tanto, una vez se ha determinado la secuencia de trabajos específica para la primera máquina, ésta se mantiene intacta para las máquinas faltantes.

Por otra parte, es importante tener en cuenta las consideraciones influyentes en un ambiente de producción industrial. Un factor clave durante un proceso productivo en el que trabajos diferentes son procesados de manera consecutiva por una misma máquina, son los conocidos tiempos de setup. Podemos definir el tiempo de setup como el tiempo necesario por cada máquina entre dos trabajos consecutivos, es decir, la duración del proceso de reajuste de la maquinaria para cambiar de un trabajo a otro. Esta nueva variante del problema es conocida como *Permutation Flowshop Scheduling Problem with Setups* (PFS-S).

2.2. Definición formal del PFS

A continuación, definimos formalmente el Problema de Secuenciación de Flowshop.
Como ya hemos comentado, el *Permutation Flowshop Scheduling Problem* trata de encontrar la secuencia de n trabajos que son procesados en el mismo orden por m máquinas en función de una serie de medidas y condiciones.

En primer lugar, consideramos los datos de entrada del problema.

- Sea $N = \{1, \ldots, n\}$ un conjunto de n trabajos. Denotaremos los trabajos por i, k, $l \in \{1, \ldots, n\}$.
- Sea $M = \{1, \ldots, m\}$ un conjunto de m máquinas. Sin pérdida de generalidad, suponemos que la ruta es $1 - 2 - \cdots - m$, es decir, se asume que todos los trabajos siguen la misma ruta por las máquinas. Denotaremos las máquinas por $j \in \{1, \ldots, m\}$.
- Sea $p_{ij} \in \mathbb{Z}^+$ las unidades de tiempo necesarias para procesar el trabajo i en la máquina j para $i = 1, \ldots, n$ y $j = 1, \ldots, m$.
  Podemos considerar la matriz $\mathbb{P}$ como la matriz que nos da la información acerca de los tiempos de proceso de cada trabajo (columnas) en cada máquina (filas) y viene dada por:

$$\mathbb{P} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \ddots & p_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nm} \end{bmatrix}$$

En definitiva, el Problema de Secuenciación de *Flow Shop* parte de un conjunto de trabajos, los cuales quiere secuenciar de manera que sean procesados por todas las máquinas en el mismo orden y de forma que una máquina no procese más de dos trabajos a la vez.

2.3. Objetivos del PFS

El objetivo del problema más estudiado es la minimización del tiempo de completación del último trabajo a ser procesado, conocido como *makespan* y denotado por C_{max}. Sin embargo, la mayor parte de problemas estudiados en líneas de producción de la industria del mundo real se abarca la optimización de más de un medible. Por este motivo, es necesario definir una variante del PFS-S que permite incluir en la función objetivo múltiples criterios.

Algunos de los objetivos más estudiados en literatura del problema son: el tiempo de completación del último trabajo o *makespan*; el tiempo de flujo o *flowtime*; el retraso o adelanto respecto la fecha de entrega de una tarea (*tardiness* o *earliness* respectivamente); el tiempo de desaprovechamiento de una máquina disponible (*waiting time* o *idle time*); además de la combinación de todos ellos.

A continuación, es necesario definir formalmente los parámetros y conceptos que nos ayudarán a definir los medibles de los problemas *flow shop* multicriterio y la respectiva notación que vamos a utilizar en cada caso. De esta manera, en función de los intereses de producción seremos capaces de modelizar los medibles a optimizar.

Suponemos que pretendemos modelizar formalmente un problema *flow shop* que cuenta con n trabajos a procesar por m máquinas. Sea π una solución factible cualquiera de la secuencia de los n trabajos.

- p_{ij}: Tiempo de proceso del trabajo i en la máquina j, $i = 1, \ldots, n$, $j = 1, \ldots, m$.
- p_i: Tiempo total de proceso del trabajo i una vez ha sido procesada por todas las m máquinas, $i = 1, \ldots, n$.
- $W_{ij}(\pi)$: Tiempo de espera del trabajo i en la máquina j en la solución factible π, $i = 1, \ldots, n$, $j = 1, \ldots, m$.
- $W_i(\pi)$: Tiempo de espera total del trabajo i en la solución π.
- r_i: Tiempo de indisponibilidad del trabajo i, es decir, el tiempo que debe transcurir hasta que el trabajo i puede empezar a ser procesado por la primera máquina, $i = 1, \ldots, n$.
- $C_i(\pi)$: Tiempo de completación del trabajo i en la solución π, $i = 1, \ldots, n$.
- $F_i(\pi)$: Tiempo de flujo del trabajo i en la solución π, $i = 1, \ldots, n$.
- d_i: Fecha de entrega del trabajo i, $i = 1, \ldots, n$.
- w_i: Peso del trabajo i, $i = 1, \ldots, n$.
- $L_i(\pi)$: Retraso del trabajo i en la solución π, es decir, la diferencia entre la completación del trabajo y su fecha de entrega especulada.

$$L_i(\pi) = C_i(\pi) - d_i, \;\; i = 1, \ldots, n$$

- $T_i(\pi)$: Tardanza del trabajo i en la solución π. Es equivalente al retraso pero en este caso se tiene en cuenta el hecho de que el tiempo de completación supere la fecha de entrega.

$$T_i(\pi) = \text{máx}\{L_i(\pi), 0\}, \;\; i = 1, \ldots, n$$

- $E_i(\pi)$: Anticipación del trabajo i en la solución π. Es equivalente al retardo pero en este caso de aquellos trabajos cuya fecha de completación es anterior a la fecha de entrega, es decir, con retardo negativo.

$$E_i(\pi) = max\{-L_i(\pi), 0\}, \;\; i = 1, \ldots, n$$

- $U_i(\pi)$: Variable auxiliar binaria que nos indica si el trabajo i tiene tardanza o no en la solución π.

$$U_i(\pi) = \begin{cases} 1 \;\; si \;\; T_i(\pi) > 0 \\ 0 \;\; si \;\; T_i(p_i) = 0 \end{cases}$$

Notemos que p_{ij}, p_i, r_i, d_i y w_i son parámetros predefinidos o conocidos al inicio del problema y que se mantienen constantes a lo largo de la resolución, independientemente de cuál sea la secuencia de los trabajos. Sin embargo, el resto de parámetros en realidad varían en función de la solución π que se esté tomando en cada momento, puesto que dependen fuertemente de esta decisión.
A su vez, estos parámetros, constantes o continuos, están relacionados entre sí a partir de las expresiones que comentaremos a continuación.
Por una parte, es evidente que el tiempo total del proceso de un trabajo corresponde a la suma de los tiempos de proceso de ese mismo trabajo en cada una de las máquinas, es decir,

$$p_i = \sum_{j=1}^{m} p_{ij}, \;\; \forall i = 1, \ldots, n.$$

Por otra parte, el tiempo total de espera de un trabajo será la suma de los tiempos de espera de ese trabajo en las diferentes máquinas. Por tanto,

$$W_i(\pi) = \sum_{j=1}^{m} W_{ij}(\pi), \;\; \forall i = 1, \ldots, n.$$

Así mismo, el tiempo de completación de un trabajo debe tener en cuenta tanto el tiempo de procesado total de ese trabajo como el tiempo de indisponibilidad y los tiempos de espera que se producen durante el proceso. En definitiva,

$$C_i(\pi) = r_i + p_i + W_i(\pi), \;\; \forall i = 1, \ldots, n.$$

Por último, el tiempo de flujo de un determinado trabajo viene unívocamente condicionado por su tiempo total de procesado y su tiempo de espera. Además, el tiempo de flujo es en definitiva la diferencia entre el tiempo de completación y el tiempo de indisponibilidad.

$$F_i(\pi) = p_i + W_i(\pi) = C_i(\pi) - r_i$$

2.3.1. Funciones objetivo del PFS

Como ya hemos comentado anteriormente, el objetivo más estudiado en la literatura del *Flow shop* es la minimización del tiempo de completación del último trabajo en ser procesado. Este medible recibe el nombre de *makespan* y es denotado por C_{max}. En definitiva, el tiempo de completación total de una solución factible π vendría dado por:

$$C_{max}(\pi) = \max_{i=1,\ldots,n} C_i(\pi)$$

La minimización del *makespan* permite maximizar la utilización y la eficiencia de la línea de producción.

Sin embargo, en función de las necesidades o prioridades de la industria se deben tomar otros medibles en cuenta a la hora de definir la función objetivo del problema. En particular, podemos clasificar los objetivos de los problemas de planificación de la producción en 3 grandes grupos. Por una parte, tenemos los objetivos basados en el tiempo de completación, como serían el propio C_{max} o el tiempo de flujo F. Por otra parte, tenemos los objetivos basados en la fecha de entrega, es decir, los retrasos L o las tardanzas T. Por último, podemos considerar también objetivos basados en inventorio o en costes.

Así mismo, podemos tomar como función objetivo variantes de algunos ya definidos, donde entrará en juego el peso que se le quiere dar a cada trabajo o a cada máquina en función de su importancia en el proceso. Algunos ejemplos a destacar serían el tiempo total de completación (C), el tiempo de completación medio ($\bar{C}$), el tiempo total de completación ponderado ($\bar{C}^w$, el tiempo de flujo total (F), el tiempo de flujo medio ($\bar{F}$), el tiempo de flujo medio ($\bar{F}^w$), la máxima tardanza (T_{max}), la tardanza total (T), la tardanza media ($\bar{T}$), la tardanza total ponderada (T^w), la tardanza media ponderada ($\bar{T}^w$), la máxima anticipación (E_{max}), la anticipación total (E), la anticipación media ($\bar{E}$), la anticipación total ponderada E^w, la anticipación media ponderada ($\bar{E}^w$), el número de los trabajos tardíos (n_T), el tiempo de inactividad (I), la varianza en el tiempo de completación (ctv).

A continuación, veamos cómo se definen algunos de los más estudiados a partir de combinaciones, estadísticos o derivados de los conceptos y parámetros definidos anteriormente.

- **Tiempo total de completación**

$$C(\pi) = \sum_{i=1}^{n} C_i(\pi)$$

- **Tiempo medio de completación**

$$\bar{C}(\pi) = \frac{1}{n} \sum_{i=1}^{n} C_i(\pi)$$

- **Tiempo de completación ponderado**

$$C^w(\pi) = \sum_{i=1}^{n} w_i C_i(\pi)$$

- **Tiempo de completación medio ponderado**

$$\bar{C}^w(\pi) = \frac{1}{n} \sum_{i=1}^{n} w_i C_i(\pi)$$

- **Tiempo de flujo total**

$$F(\pi) = \sum_{i=1}^{n}(C_i(\pi) - r_i) = \sum_{i=1}^{n} F_i$$

- **Tardanza total**

$$T(\pi) = \sum_{i=1}^{n} T_i(\pi)$$

- **Anticipación total**

$$E(\pi) = \sum_{i=1}^{n} E_i(\pi)$$

- **Tardanza máxima**

$$T_{max}(\pi) = \max_{i} T_i(\pi)$$

- **Número de trabajos tardíos**

$$n_T(\pi) = \sum_{i=1}^{n} U_i(\pi)$$

- **Tiempo de inactividad (*Idle time*) de la máquina** j

$$I_j = \max_{i=1,\ldots,n} C_{ij}(\pi) - \sum_{i=1,\ldots,n} p_{ij}$$

- **Tiempo total de inactividad de las máquinas**

$$I = \sum_{j=1}^{m} I_j$$

- **Varianza en el tiempo de completación**

$$ctv = \frac{1}{n}\sum_{i=1}^{n}(C_i(\pi) - \bar{F}(\pi))^2$$

En definitiva, además del *makespan*, pueden ser tomados como función objetivo muchos otros medibles en función de cuál sea nuestra principal prioridad con la programación de la producción. Además, existen heurísticas desarrolladas en la literatura que tienen en cuenta la optimización de varios medibles de forma simultánea. La combinación de algunas de estas funciones objetivos nos permiten aumentar la eficiencia de la línea de producción, disminuir significativamente los costes de producción o incrementar la tasa de utilización de la maquinaria.

En el Anexo A de este documento podemos consultar la implementación en RStudio del cálculo de las funciones objetivo que consideraremos y que serán utilizadas en la implementación de los algoritmos heurísticos para evaluar su valor.

Capítulo 3

Algoritmos propuestos

En el presente apartado procedemos a realizar una breve revisión de algunos de los diferentes métodos de resolución propuestos en la literatura para abarcar el problema *Flow Shop*. Además, se propone un algoritmo propio que resuelve el problema teniendo en cuenta aspectos no considerados por los algoritmos ya desarrollados.

Existe un amplio abanico de propuestas de métodos de resolución del Problema de *Flow Shop* con un único objetivo. Entre ellos encontramos métodos exactos, heurísticos y metaheurísticos. Sin embargo, teniendo en cuenta que la mayoría de procesos industriales involucran múltiples medibles a optimizar, vamos a centrar nuestro estudio en la investigación de métodos de resolución para problemas Flowshop.

Debemos tener en cuenta que hay dos tipos de métodos utilizados para resolver el *Flow Shop Scheduling Problem*. Por una parte, disponemos de métodos exactos, los cuales permiten obtener la solución óptima del problema. Sin embargo, debido a la complejidad del problema de permutación Flow Shop multiobjetivo, estos métodos resultan ser muy ineficientes cuando se trata resolver un problema de gran tamaño, puesto que la complejidad computacional aumenta exponencialmente. De hecho, si tenemos n trabajos, el número de posibles soluciones es de $n!$, independientemente del número de máquinas. Así mismo, no existen algoritmos exactos sencillos que ofrezcan la solución óptima para un problema de grandes dimensiones, más allá del caso en que disponemos de 2 máquinas únicamente (*2-machine problem*). De esta manera, se pueden utilizar algoritmos de programación lineal como el branch and bound, pero resultan muy poco efectivos cuando se trata de problemas de gran o medio tamaño, que son los que nos encontramos en la industria real. Por este motivo, a lo largo de los años se han desarrollado métodos heurísticos y metaheurísticos para resolver este problema de carácter NP-hard [2] y permitir por tanto resolver problemas de la vida real de mayor tamaño sin obtener un coste computacional demasiado elevado y de manera bastante sencilla.

3.1. Algoritmo NEH

El Algoritmo NEH [5] es uno de los heurísticos más populares utilizados en la literatura para resolver el *Flowshop Scheduling Problem*. Las siglas del nombre que toma el algoritmo vienen dadas por los nombres de sus creadores: Nawaz, Enscore y Ham. Fue creado en 1983. Se trata de una heurística constructiva con mejora que utiliza ideas del Branch and Bound, puesto que la idea general del algoritmo es que en cada iteración toma

la decisión siguiente en función de la evaluación de cuál sería la secuencia que optimiza la función objetivo. Al tratarse de una heurística, no nos asegura la solución óptima, simplemente una aproximación bastante cercana a ella dentro del conjunto de soluciones factibles. El objetivo del método es resolver el problema de secuenciación de n tabajos en m máquinas atendiendo la minimización del tiempo de completación del último trabajo en la última máquina, conocido por *makespan* y denotado por C_{max}.

La popularidad de este método de resolución es en parte debida a la calidad de las soluciones en la mayoría de las instancias [5], pero también a la simplicidad y entendimiento del algoritmo, así como a su corto tiempo de computación. Sin embargo, en la literatura se encuentran multitud de propuestas de mejora y análisis de algunos aspectos del comportamiento del algoritmo, puesto que es cierto que el algoritmo omite centrar su búsqueda en ciertas características que realmente mejoran la eficiencia de la solución. Por este motivo, es frecuente encontrar en la literatura propuestas de heurísticos para resolver el *Flowshop Scheduling Problem* que toman como punto de partida una solución obtenida con el algoritmo NEH y tratan de mejorarla, centrando la función objetivo en alguno de los aspectos que dicho algoritmo no tiene en cuenta.

A continuación, procedemos a describir los pasos del algoritmo NEH [6].

1. En primer lugar, calculamos el tiempo de proceso total de cada trabajo, es decir, la suma de los tiempos de proceso de cada trabajo en cada máquina.

$$C_i = \sum_{j=1}^{m} p_{ij}, \quad i = 1, \ldots, n.$$

2. A continuación, generamos la primera secuencia inicial basada en el orden decreciente de estos tiempos de proceso calculados, es decir, ordenamos los trabajos de mayor a menor valor de C_i.

3. Elegimos los dos primeros trabajos de la secuencia inicial y obtenemos las dos posibles secuencias ($i_1 - i_2$ o $i_2 - i_1$). De esta manera, nos quedamos con la secuencia parcial con mínimo *makespan*. Esta será la secuencia incumbente.

4. El siguiente trabajo de la lista ordenada que todavía no ha sido insertado en la secuencia parcial, es insertado en todas las posibles posiciones que puede tomar, evaluando su *makespan* en cada una de ellas y quedándonos así con la posición que lo minimice. De manera iterativa, realizamos la misma operación con los $n - 3$ trabajos restantes hasta que hayan sido todos ellos asignados a una posición específica.

Notamos que, tal y como habíamos comentado por su semejanza con el *Branch and Bound*, en cada iteración se toma la decisión en función de qué alternativa supone una mejora para la función objetivo.

Ejemplo 1. Consideramos una línea de producción constituída por $n = 4$ trabajos y $m = 3$ máquinas. Los tiempos de proceso de cada trabajo en cada máquina vienen resumidos en la matriz 4×3 de la Figura 3.1.

El problema consiste en determinar una secuencia de los 4 trabajos idéntica para todas las máquinas tal que el tiempo de completación del último trabajo en la última máquina sea mínimo. En definitiva, la función objetivo consiste en minimizar lo que conocemos por *makespan*.

Para ello, vamos a recurrir a aplicar el algoritmo heurístico NEH.

Job / Machine	1	2	3
1	5	9	2
2	7	8	3
3	4	6	5
4	6	10	4

Figura 3.1: Tiempos de proceso ejemplo de cada trabajo en cada máquina para $n = 4$ trabajos y $m = 3$ máquinas

El primer paso del algoritmo consiste en calcular el tiempo total de completación de cada trabajo, es decir, la suma de todos los tiempos de proceso en todas las máquinas para cada trabajo. En particular, tenemos que calcular:

$$C_i = \sum_{j=1}^{m} p_{ij}, \quad \forall i = 1, \ldots, n.$$

En nuestro caso particular, tenemos

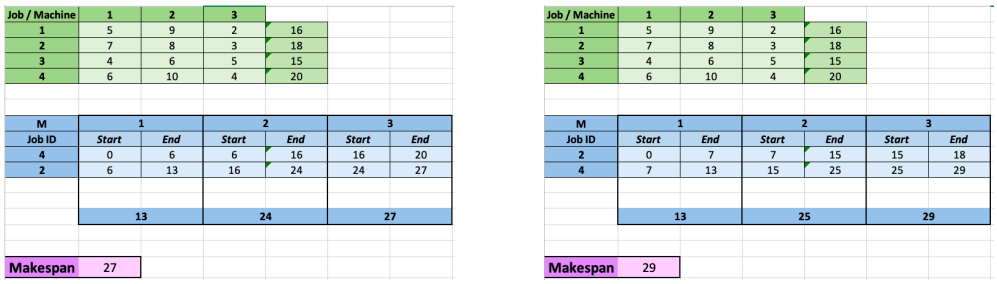
$$C_1 = \sum_{i=1}^{3} p_{i1} = 5 + 9 + 2 = 16, \quad C_2 = \sum_{i=1}^{3} p_{i2} = 7 + 8 + 3 = 18,$$
$$C_3 = \sum_{i=1}^{3} p_{i3} = 4 + 6 + 5 = 15, \quad C_4 = \sum_{i=1}^{3} p_{i4} = 6 + 10 + 4 = 20.$$

Una vez tenemos calculado el tiempo de completación total de cada trabajo procedemos a ordenar los trabajos según orden no creciente de su tiempo total de completación. De esta manera, tenemos la siguiente ordenación: $\{4, 2, 1, 3\}$. Elegimos los dos primeros trabajos de la lista ordenada y formamos las dos posibles secuencias. En este caso, las dos posibles secuencias parciales formadas por los 2 primeros trabajos de la lista ordenada serían $\{4, 2\}$ y $\{2, 4\}$.

Evaluamos el C_{max} de ambas posibles secuencias parciales y nos quedamos con aquella que minimice el *makespan*. La secuencia parcial $\{2, 4\}$ tiene un *makespan* asociado de 29. Sin embargo, la secuencia parcial $\{4, 2\}$ tiene un *makespan* asociado de 27. Por tanto, tomaremos como base la secuencia parcial $\{4, 2\}$ porque tiene menor valor de C_{max}. En las Figuras 3.2 podemos ver los tiempos de inicialización y finalización de cada trabajo en cada máquina, así como el *makespan* asociado a cada secuencia parcial respectivamente.

3.1.1. Implementación del método

La implementación del algoritmo NEH se ha desarrollado mediante RStudio. En el Anexo B podemos consultar el código que será utilizado posteriormente para obtener las simulaciones numéricas pertinentes.



Job / Machine	1	2	3	
1	5	9	2	16
2	7	8	3	18
3	4	6	5	15
4	6	10	4	20

M	1		2		3	
Job ID	Start	End	Start	End	Start	End
4	0	6	6	16	16	20
2	6	13	16	24	24	27
	13		24		27	

Makespan	27

(a)

Job / Machine	1	2	3	
1	5	9	2	16
2	7	8	3	18
3	4	6	5	15
4	6	10	4	20

M	1		2		3	
Job ID	Start	End	Start	End	Start	End
2	0	7	7	15	15	18
4	7	13	15	25	25	29
	13		25		29	

Makespan	29

(b)

Figura 3.2

3.2. Algoritmo 1

El método heurístico que desarrollaremos a continuación fue llevado a cabo por Ho and Chang en [3] y propone un método heurístico para el problema de permutación de *Flow Shop* de n trabajos y m máquinas que toma como objetivos el tiempo de completación del último trabajo (*makespan*), el tiempo total de flujo (*total Flow Time*) y el tiempo total de inactividad (*Total Idle Time*).

En particular, el artículo objeto de estudio propone un método heurístico mejorado para la resolución del problema *Flow Shop*.

Las soluciones proporcionadas por otras heurísticas suelen presentar huecos vacíos entre trabajos sucesivos, de manera que se producen tiempos de inactividad en las máquinas correspondientes a los tiempos de espera entre que el próximo trabajo a procesar en una determinada máquina se acaba de procesar en la máquina anterior. Esto provoca pérdidas importantes de productividad, por lo que la minimización de estos tiempos se ha convertido en uno de los grandes retos de la industria. De esta manera, el método desarrollado trata de minimizar los huecos entre trabajos sucesivos que se generan en las soluciones ofrecidas por otros métodos heurísticos, tratando de no incrementar significativamente el coste computacional.

A continuación, procedemos a definir formalmente el algoritmo heurístico.

Consideremos un problema de permutación *Flow Shop* con m máquinas y n trabajos a procesar. Suponemos las hipótesis características de un problema *flow shop* de permutación multiobjetivo ya definidas en anterioridad.

En primer lugar, cabe recordar que p_{ij} representa el tiempo de procesado del trabajo i en la máquina j, $\forall i = 1, \ldots, n, \forall j = 1, \ldots, m$.
Cabe recordar también que el objetivo del problema es determinar la secuencia de trabajos que optimice determinadas medidas de desempeño de la producción. En particular, en este caso trataremos de minimizar el *makespan*, el tiempo de flujo medio y el tiempo de inactividad de la maquinaria (*Idle Time*). Aunque intuitivamente estos tres medibles presenten una alta correlación, es necesario considerar todos ellos a la hora de definir nuestra función objetivo, ya que podríamos encontrarnos con ejemplos en los que la secuencia obtenida presentara un valor bastante bueno de uno de los medibles pero tuviera un desempeño bastante pésimo en el

resto.

Como habíamos comentado, en algunas de las soluciones que nos propocionan la mayoría de heurísticas, notamos la existencia de espacios temporales vacíos entre que un trabajo acaba de ser procesado en una máquina y puede pasar a ser procesado por la siguiente, puesto que la máquina estaba procesando el trabajo consecutivamente anterior.

Consideremos pues un problema de m máquinas y n trabajos.

Definimos g_{ij} como la diferencia temporal entre que el trabajo i termina de ser procesado en la máquina j y empieza a ser procesado en la máquina $j+1$. De esta manera, el tiempo de completación del trabajo i vendrá dado por la siguiente expresión en la que, además de los tiempos de procesado en cada máquina, se tiene en cuenta los tiempos de espera y bloqueo de las máquinas. Suponemos que no tenemos indisponibilidad de trabajos.

$$C_i = \sum_{j=1}^{m} p_{ij} + \sum_{j=1}^{m-1} g_{ij}, \quad \forall i = 1, \ldots, n$$

Por tanto, el tiempo de completación (*makespan*) C_{max} que habíamos definido como

$$C_{max} = \max_{i=1,\ldots,n} C_i,$$

quedará como sigue:

$$C_{max} = \max_{i=1,\ldots,n} \sum_{j=1}^{m} p_{ij} + \sum_{j=1}^{m-1} g_{ij}$$

De esta manera, como los primeros j términos de C_i, correspondientes a los tiempos de procesado, son constantes, (p_ij constantes $\forall i = 1, \ldots, n$, $\forall j = 1, \ldots, m$), si queremos minimizar el tiempo de completación *makespan* debemos de optar por minimizar la suma de estos vacíos temporales de maquinaria, es decir, el sumatorio de los g_{ij}. En definitiva, la heurística que se propone tiene como principal objetivo minimizar estos vacíos.

El primer paso, consiste en definir un medible que cuantifique de alguna manera la longitud de estos huecos vacíos. Por este motivo, definimos d_{ik}^j como la diferencia entre el tiempo de proceso del trabajo i en la máquina j y el trabajo k en la máquina $j+1$.

$$d_{ik}^j = p_{i,j+1} - p_{kj}, \quad \forall i, k = 1, \ldots, n : i \neq k, \forall j = 1, \ldots, m-1$$

En el caso de que el trabajo i preceda el trabajo k en la secuencia, en función del signo de d_{ik}^k podemos deducir el comportamiento del trabajo k o de la máquina $j+1$. En particular:

- Si $d_{ik}^j < 0$, entonces el trabajo k debe esperar en la máquina $k+1$ al menos d_{ik}^k unidades de tiempo hasta que el trabajo i finalice en dicha máquina. Esto es porque un valor positivo de d_{ik}^j implica que el tiempo de procesado del trabajo i en la máquina $j+1$ es superior que el tiempo de procesado del trabajo k en la máquina j, por lo que el trabajo k tendría que esperar en la máquina j hasta que la siguiente (máquina $j+1$) quedará habilitada.

- Si $d_{ik}^j < 0$, entonces existe tiempo de inactividad en la máquina $j+1$ entre los trabajos i y k. Esto es porque un valor negativo de d_{ik}^j implica que el tiempo de proceso del trabajo i en la máquina $j+1$ es inferior al tiempo de proceso del trabajo k en la máquina j, por lo que la máquina $j+1$ se mantendrá inactiva una vez ha terminado de procesar el trabajo i esperando a que el trabajo k termine de ser procesado por la máquina anterior.
- Si $d_{ik}^j = 0$, no existe ni tiempo de inactividad ni tiempo de bloqueo. Estamos frente al caso óptimo. De hecho, lo que busca el algoritmo es precisamente que los valores de d_{ik}^j sean lo más cercanos a 0 posibles.

Atendiendo a nuestro objetivo de minimizar la duración de los huecos vacíos con el propósito de reducir el *makespan*, el procedimiento a seguir sería el siguiente. Por una parte, los pares de trabajos con los huecos asociados más negativos, deben colocarse preferiblemente al final de la secuencia. Esto es porque si acumulamos los valores negativos de d_{ik}^j al final, se consigue contrarrestar los valores positivos acumulados al principio de la secuencia, mientras que si se colocaran al principio de la secuencia, con una alta probabilidad resultarían ser tiempos de inactividad de la maquinaria, es decir, *idle time*. De manera análoga, los pares de trabajos con los huecos vacíos más positivos deberían ser colocados al principio de la secuencia, puesto que podrán ser compensados por los huecos negativos que se producirán debido a trabajos posteriores.

Posteriormente, debemos definir un nuevo factor asociado a los valores negativos de d_{ik}^j y que nos ayudará a contrarrestar su influencia y evitar tiempos de inactividad.

$$factor(j) = \frac{1{,}0 - 0{,}1}{m-2} \cdot (m-j-1) + 0{,}1, \quad \forall j = 1, \ldots, m-1.$$

De esta manera, los factores definidos para cada máquina nos permiten asignar pesos más elevados a las j menores ($factor(j)$ tiende a 1), mientras que se le asignan pesos menores a mayor j ($factor(j)$ tiende a 0). Estos pesos se asignan para obligar a que los valores de d_{ik}^j más negativos estén asociados a máquinas en las primeras etapas de procesamiento, es decir, valores pequeños de j. Por tanto, esto permite aumentar la efectividad de reducir los huecos positivos acumulados. De hecho, para $j = 1$, tenemos que $factor(1) = 1{,}0$. En cambio, para $j = m-1$, $factor(m-1) = 0{,}1$. En cuanto a las máquinas intermedias, se utiliza una interpolación lineal entre $1{,}0$ y $0{,}1$.

Una vez hemos definido las diferencias de tiempos de procesado de trabajos consecutivos en máquinas consecutivas y el factor que permite contrarrestar los valores negativos, estamos en condiciones de definir la función de descuento como sigue:

$$\delta_{ik}^j = \begin{cases} factor(j) & si\, d_{ik}^j < 0 \\ 1 & \text{en otro caso} \end{cases} \tag{3.1}$$

Si combinamos los valores de d_{ik}^j de cada una de las máquinas con el valor de la función de descuento, obtenemos un único valor revisado que representa los huecos vacíos d_{ik}^j $\forall i, k = 1, \ldots, n$, $\forall j = 1, \ldots, m-1$ teniendo en cuenta el descuento de los valores negativos. Denotaremos por d_{ik}^R el valor revisado de estos vacíos temporales.

$$d_{ik}^R = \sum_{j=1}^{m-1} d_{ik}^j \delta_{ik}^j, \qquad \forall i, k = 1, \ldots, n$$

En definitiva, para aplicar la heurística definida debemos seguir los siguientes pasos:

1. Obtenemos una primera solución inicial mediante cualquier método heurístico conocido. Esta primera solución se denominará solución titular y se utilizará como punto de partida del algoritmo.

2. Construimos los valores revisados de $d_{ik}^R \ \forall i, k = 1, \ldots, n$. Para ello, primero deberemos obtener para cada una de las m máquinas el valor de d_{ik}^j para cada par de trabajos i, k. Una vez tenemos ese valor calculado, en función de su signo obtendremos el valor del factor de contrarrestado de valores negativos y su función asociada.

3. Suponemos el conjunto de índices l con $l = 1, \ldots, n$ para representar las posiciones de los trabajos en la solución titular inicialmente obtenida. De esta manera, P_l nos indica el trabajo que se encuentra en la posición l en la solución titular.

4. Fijamos $a = 1$ y $b = n$.

5. Tomamos X como el mayor valor de $d_{P_a P_l}^R$, donde l está entre a y b, es decir,

$$X := \underset{l \in (a,b]}{\text{máx}} \, d_{P_a P_l}^R \tag{3.2}$$

   Tomamos $u := l$, es decir, u representa la posición del trabajo P_l con el mayor valor de $d_{P_a P_l}^R$.

6. De manera análoga, tomamos Y como el menor valor de $d_{P_l P_b}^R$, donde l está entre a y b, es decir,

$$Y := \underset{l \in [a,b)}{\text{mín}} \, d_{P_l P_b}^R \tag{3.3}$$

   Tomamos $v := l$, es decir, v representa la posición de la máquina P_l con el menor valor de $d_{P_l P_b}^R$.

7. Si $X < 0$, $Y > 0$ y $|X| \leq |Y|$, entonces vamos al Paso 10.

8. Si $X < 0$, $Y > 0$ y $|X| > |Y|$, entonces vamos al Paso 11.

9. Si $|X| > |Y|$, entonces vamos al Paso 10. En caso contrario, vamos al Paso 11.

10. Tomamos $a = a + 1$ e intercambiamos los trabajos en las posiciones a y u. A continuación, vamos al Paso 12.

11. Tomamos $b = b - 1$ e intercambiamos los trabajos en las posiciones b y v.

12. Si la secuencia nueva mejora el valor de la función objetivo respecto la secuencia titular, entonces la nueva secuencia pasa a ser la secuencia titular a partir de este momento. En otro caso, volveríamos a la solución titular original.

13. Si $b = a + 2$, entonces termina el algoritmo. En otro caso, volvemos al Paso 5 y repetimos los pasos pertinentes hasta alcanzar el criterio de parada definido.

Job / Machine	1	2	3
1	5	9	2
2	7	8	3
3	4	6	5
4	6	10	4

Figura 3.3: Tiempos de proceso ejemplo de cada trabajo en cada máquina para $n = 4$ trabajos y $m = 3$ máquinas

Ejemplo 2. Tomamos como ejemplo los mismos tiempos de proceso que habíamos empleado para ilustrar el funcionamiento del algoritmo NEH. Recordemos que contábamos con 4 trabajos a ser procesados en 3 máquinas. Los tiempos de proceso de cada trabajo en cada máquina vienen dados en la Figura 3.3. Siguiendo la notación definida, en este caso tenemos $n = 4$ trabajos y $m = 3$ máquinas.

En primer lugar, obtenemos una primera solución inicial. En este caso, la hemos obtenido mediante una secuencia aleatoria de los 4 trabajos. En particular, la solución titular random viene dada por: $\{4, 2, 1, 3\}$. El tiempo de completación asociado a esta secuencia es de $C_{max} = 44$. En la Figura 3.4 observamos el cálculo del tiempo de completación de cada trabajo tomando como secuencia la solución titular dada, así como los instantes de inicialización y finalización de cada trabajo en cada máquina.

M	1		2		3	
Job ID	*Start*	*End*	*Start*	*End*	*Start*	*End*
4	0	6	6	16	16	20
2	6	13	16	24	24	27
1	13	18	24	33	33	35
3	18	22	33	39	39	44
	22		39		44	

Makespan	44

Figura 3.4: Tiempos de completación solución titular

Siguiendo los pasos del algoritmo, debemos calcular primeramente el valor de las diferencias d_{ik}^{j} para cada par de trabajos y para todas las máquinas menos una, es decir, $\forall i, k = 1, \ldots, n \; i \neq k$ y $\forall j = 1, \ldots, m-1$. En este caso, como tenemos únicamente 3 máquinas y 4 trabajos, se obtienen 2 matrices de 4×4. En la Figura 3.5 observamos los valores de estas diferencias. Notemos que los valores de las diagonales no se contabilizan.

A continuación, construímos los valores revisados de estas diferencias. Para ello, primeramente es necesario calcular los factores de contrarrestado asociados a cada máquina. En este caso,

$$\begin{aligned} factor(1) &= \frac{1{,}0 - 0{,}1}{3-2} \cdot (3-1-1) + 0{,}1 = 1 \\ factor(2) &= \frac{1{,}0 - 0{,}1}{3-2} \cdot (3-2-1) + 0{,}1 = 0{,}1 \end{aligned} \tag{3.4}$$

De esta manera, en función del signo de las diferencias calculadas en la Figura 3.5 y haciendo uso de los factores de contrarrestado asociados a cada máquina calculados en (3.4), obtenemos el valor de la función de descuento para cada máquina y para cada par de trabajos en la Figura 3.6 según la condición (3.1).

Diferencias tiempo de proceso

Máquina 1

Job ID	1	2	3	4
1	-	2	5	3
2	3	-	4	2
3	1	-1	-	0
4	5	3	6	-
				1

Máquina 2

Job ID	1	2	3	4
1	-	-6	-4	-8
2	-6	-	-3	-7
3	-4	-3	-	-5
4	-5	-4	-2	-
				0,1

Figura 3.5: Diferencias de los tiempos de proceso de los trabajos 2 a 2 en las máquinas 1 y 2 respectivamente

Función de descuento

Máquina 1

Job ID	1	2	3	4
1	-	1	1	1
2	1	-	1	1
3	1	1	-	1
4	1	1	1	-

Máquina 2

Job ID	1	2	3	4
1	-	0,1	0,1	0,1
2	0,1	-	0,1	0,1
3	0,1	0,1	-	0,1
4	0,1	0,1	0,1	-

Figura 3.6: Función de descuento asociada a cada máquina y a cada par de trabajo

Combinando la función de descuento con las diferencias según (3.2), obtenemos las diferencias revisadas de la Figura 3.7.

Diferencias revisadas

Job ID	1	2	3	4
1	-	1,4	4,6	2,2
2	2,4	-	3,7	1,3
3	0,6	-1,3	-	-0,5
4	4,5	2,6	5,8	-

Figura 3.7: Diferencias revisadas de los trabajos 2 a 2

Una vez tenemos calculadas las diferencias revisadas para cada par de trabajos, estamos en condiciones

de proceder a realizar el Paso 3 del algoritmo, el cual consiste en determinar el conjunto de índices $l = 1, 2, 3, 4$, de manera que P_l nos indique el trabajo que se encuentra en la posición l en la solución titular. En este caso, tomando como solución titular la que habíamos obtenido de manera aleatoria ($\{4, 2, 1, 3\}$), tenemos que

$$P_1 = 4, \quad P_2 = 2, \quad P_3 = 1, \quad P_4 = 3.$$

Fijamos $a = 1$ y $b = 4$ (Paso 4). Tomamos el máximo valor de las diferencias revisadas entre el trabajo de la posición $a = 1$ respecto al resto de trabajos, siguiendo (3.2). Lo denotamos por X, es decir,

$$X = \max_{l\in(1,4]} d^R_{P_1,P_l} \overset{P_1=4}{=} \max_{l\in(1,4]} d^R_{4,P_l} = \max\{d^R_{4,P_2}, d^R_{4,P_3}, d^R_{4,P_4}\} = \max\{4'5, 2'6, 5'8\} = 5'8 = d^R_{4,3} \overset{P_4=3}{=} d^R_{4,P_4}$$

Sea u la posición del trabajo con mayor valor de diferencia revisada, tomamos $u = 4$.

De manera análoga, tomamos y denotamos por Y el mínimo valor de las diferencias revisadas del trabajo en la posición $b = 4$ respecto al resto de trabajos, según (3.3), es decir,

$$Y = \min_{l\in[1,4)} d^R_{P_l,P_4} \overset{P_4=3}{=} \min_{l\in[1,4)} d^R_{P_l,3} = \min\{d^R_{P_1,3}, d^R_{P_2,3}, d^R_{P_3,3}\} = \min\{5'8, 3'7, 4'6\} = 3'7 = d^R_{2,3} \overset{P_2=2}{=} d^R_{P_2,3}$$

De esta forma, si denotamos por v la posición del trabajo con menor valor de diferencia revisada, tomaremos $v = 2$.
En definitiva, de los Pasos 5 y 6 del algoritmo concluimos lo siguiente:

$$X = 5'8, \quad u = 4$$
$$Y = 3'7, \quad v = 2$$

A continuación, procedemos a comprobar las condicionales de los Pasos 7, 8 y 9 para elegir el camino oportuno. En este caso, se cumple la condición del Paso 9, puesto que tanto X como Y son valores positivos y se cumple que $|X| > |Y|$ ($|5'8| > |3'7|$). Por este motivo, pasamos al Paso 10 del algoritmo. De esta manera, tomamos $a = a + 1$, es decir, $a = 2$, e intercambiamos en la solución titular los trabajos situados en las posiciones $a = 2$ y $u = 4$. En definitiva, debemos intercambiar los trabajos P_2 y P_4, que son el trabajo 2 y el trabajo 3.

Recordemos que la solución titular venía dada por $\{4, 2, 1, 3\}$ con un C_{max} de 44. Al intercambiar los trabajos 2 y 3, la solución actualizada vendrá dada por $\{4, 3, 1, 2\}$.

En el Paso 12, calculamos el *makespan* de la nueva secuencia, que resulta ser de 42. Como el valor de la función objetivo ha mejorado levemente de 44 a 42, la nueva secuencia ($\{4, 2, 1, 3\}$) pasa a ser la secuencia titular a partir de este momento. Además, como se cumple que $b = a + 2$, ya que $b = 4$ y $a = 2$, debemos finalizar el algoritmo puesto que hemos llegado al criterio de parada establecido en el Paso 13.

3.2.1. Implementación del método

La implementación del algoritmo se ha desarrollado mediante RStudio. En el Anexo C podemos consultar el código que será utilizado posteriormente para obtener las simulaciones numéricas pertinentes.

3.3. Algoritmo 2

El algoritmo heurístico que desarrollaremos a continuación ha sido extraído de [4]. El algoritmo propone una mejora a la solución obtenida mediante el algoritmo NEH en cuanto a tiempo de flujo (*Flow Time*) se refiere. La idea principal del algoritmo es tratar de intercambiar los trabajos con mayor diferencia de tiempos de proceso con su consecutivo respectivamente.

1. Obtenemos una primera solución inicial mediante el algoritmo heurístico NEH, cuyo alcance abarca la minimización del tiempo de completación del último trabajo (*makespan*) pero desprecia otros aspectos a tener en cuenta. De esta manera, obtenemos una primera secuencia de trabajos que tomamos como base para aplicar los siguientes pasos del algoritmo.

2. Intercambiamos los trabajos en la posiciones i y $i+1$ de la secuencia inicial $\forall 1 \leq i \leq n-1$. De esta manera, obtenemos $n-1$ posibles secuencias diferentes, de las cuales nos quedamos con aquella que minimice el *makespan* asociado o, en caso de empate, con el que menor tiempo de flujo tenga. Denotamos por S esta secuencia con mínimo *makespan* de las $n-1$ obtenidas, la cual pasará a ser la secuencia semilla del algoritmo. Denotaremos por M y por F el *makespan* y el *flowtime* asociado a S respectivamente.

3. Para cada uno de los trabajos de la secuencia S y sus respectivos consecutivos, calculamos los siguientes valores que denotaremos por $D_i \; \forall 1 \leq i \leq n-1$ y que representa la diferencia de los tiempos de proceso totales de un trabajo y su consecutivo, es decir,

$$D_i = \sum_{j=1}^{m} p_{ij} - \sum_{j=1}^{m} p_{i+1,j}, \quad \forall 1 \leq i \leq n-1.$$

4. De manera análoga, obtenemos los siguientes valores, denotados por $D'_i \; \forall 1 \leq i \leq n-1$:

$$D'_i = \sum_{j=1}^{m} [(m-j+1)p_{ij}] - \sum_{j=1}^{m} [(m-j+1)p_{i+1,j}], \quad \forall 1 \leq i \leq n-1.$$

5. Tomamos los trabajos de la secuencia S cuyo valor calculado de D_i sea positivo, es decir,

$$JDP = \{S_i | D_i \geq 0\}$$

6. Si ningún trabajo del conjunto S tiene asociado un D_i positivo, entonces vamos directamente al Paso 12. En caso contrario, vamos al Paso 7.

$$\text{Si } JDP = \emptyset (D_i < 0 \, \forall 1 \leq i \leq n-1) \Rightarrow \text{Paso 12}$$
$$\text{Si } JDP \neq \emptyset (\exists \, i : D_i \geq 0) \Rightarrow \text{Paso 7}$$

7. Ordenamos el conjunto de trabajos JDP según valor decreciente de su D_i asociado. En caso de empate en el valor de D_i, se le asociará orden mayor al trabajo con mayor valor de D'_i.

8. Sea k la posición en S del primer trabajo de la secuencia JDP ordenada en el paso anterior. Intercambios en la secuencia semilla S los trabajos consecutivos k y $k+1$. Denotamos por S' la nueva secuencia resultante de aplicar a S este intercambio. Sean M' y F' el *makespan* y el *flowtime* de S' respectivamente. Notemos que a mayor valor positivo de D_i, mayores son las posiblilidades de reducción del tiempo de flujo (*Flow time*) cuando se intercambia dicho trabajo con el adyacente.

9. Calculamos el incremento relativo de *makespan* y de *Flow time* total entre a secuencia S y la secuencia S' y viceversa como sigue:

$$R_{S'} = \frac{M' - \text{mín}(M', M)}{\text{mín}(M', M)} + \frac{F' - \text{mín}(F', F)}{\text{mín}(F', F)}$$

$$R_S = \frac{M - \text{mín}(M', M)}{\text{mín}(M', M)} + \frac{F - \text{mín}(F', F)}{\text{mín}(F', F)}$$

10. Si $R_{S'} < R_S$ entonces S' pasa a ser la nueva secuencia semilla, es decir, se actualiza

$$S = S', \;\; F = F', \;\; M = M'$$

y volvemos al Paso 3.
En caso contrario, es decir, si $R_{S'} \geq R_S$, pasamos al Paso 11.

11. El trabajo que se encontraba en la posición k en la secuencia S es eliminado de la lista de trabajos ordenados JDP. Si siguen habiendo trabajos en la lista JDP actualizada, volvemos al Paso 8 y repetimos los cálculos pertinentes con la lista JDP actualizada. En caso contrario, procedemos al Paso 12.

12. La secuencia actual S, su *makespan* M y su *flowtime* total F constituyen la solución al problema obtenida por la heurística. Finalizamos el algoritmo.

3.3.1. Implementación del método

La implementación del algoritmo se ha desarrollado mediante RStudio. En el Anexo D podemos consultar el código que será utilizado posteriormente para obtener las simulaciones numéricas pertinentes.

3.4. Algoritmo GRASP

El algoritmo GRASP (*Greedy Randomized Adaptive Search Procedure*) [9] es una metaheurística de optimización combinatoria. La idea intuitiva del método consiste en una serie de iteraciones predeterminadas de antemano, de manera que en cada una de las cuales se realiza una primera fase de construcción seguida de una fase de mejora. Mediante la fase de construcción se obtiene una primera solución. Esta solución se obtiene de manera constructiva, es decir, se van añadiendo uno a uno los elementos en función de la mejora que aporten a la solución parcial. En la fase de mejora local, se inicializa con la solución obtenida en la fase de construcción para aplicar mejoras iterativas hasta que una solución localmente óptima es encontrada.

A continuación, desarrollamos el pseudocódigo de lo que sería el Algoritmo GRASP a rasgos muy generales para un problema de minimización. Sea x^* la mejor solución encontrada hasta el momento y f^* el valor de la función objetivo en esta solución, es decir, $f^* = f(x^*)$, tenemos:

1. $f^* = \infty$

2. Mientras (criterio de parada = FALSE)

a) Generar solución greedy aleatorizada x

b) Encontrar el óptimo local x_l con búsqueda local inicializada en x

c) Si $f(x_l) < f^*$, entonces

1) $f^* = f(x_l)$

2) $x^* = x_l$

En definitiva, mediante la fase de construcción se encuentra una solución factible cuya vecindad es investigada hasta llegar a un mínimo local o hasta que se cumpla el criterio de parada del algoritmo. El algoritmo retorna la mejor solución encontrada a lo largo de toda su desempeño.

3.4.1. Construcción Greedy

Como ya hemos comentado con anterioridad, la primera fase del algoritmo consiste en la construcción de una solución. Para ello, se utiliza un algoritmo *Greedy*. Decimos que un algoritmo es *Greedy* [11] o voraz cuando cuando encuentra soluciones globalmente óptimas a base de hacer elecciones localmente óptimas, es decir, el algoritmo siempre hace lo que considera que es mejor en cada momento y no reconsidera sus decisiones. En particular, en el caso del algoritmo GRASP el algoritmo se encarga de, en cada iteración, construir un conjunto C de elementos candidatos a añadirse como parte de la solución en dicha iteración. Con este objetivo, los componentes del conjunto C son evaluados mediante una función *greedy*, de manera que son clasificados en función de su valor asociado. Seguidamente, el elemento con mejor puntuación es añadido a la solución. Una vez actualizada la solución, debe de actualizarse también el conjunto C, de forma que el elemento introducido en la solución sea eliminado. Este procedimiento iterativo se repite hasta que el conjunto C esté vacío, es decir, $C = \emptyset$.

Lista de Candidatos Restringida (*RCL*)

Sea C el conjunto de candidatos a introducir en la solución parcial que se está construyendo de manera que sea factible. La elección del elemento a incorporar se determina mediante una evaluación *greedy*, es decir, una función que permite cuantificar el incremento en la función de coste a medida que se incluye algún elemento en la solución. De esta manera, se crea una lista de candidatos restringida, denotada por RCL, en la que se incluyen aquellos elementos que menos incrementen la función de coste (en el caso de los problemas de minimizar, aquellos que menos incrementen el valor de la función objetivo). La lista puede estar restringida en cuanto a la cantidad de elementos o en cuanto a la calidad de los elementos. De hecho, la calidad se restringe según la siguiente cota, en la cual se tiene en cuenta el valor máximo y mínimo de la función objetivo si son introducidos en la solución cada uno de los candidatos evaluados, con un cierto grado de aleatoriedad que es introducido mediante el parámetro α.

$$RCL = \{i \in C | fo_{min} \leq fo[i] \leq fo_{min} + \alpha(fo_{max} - fo_{min})\}$$

El parámetro $\alpha \in [0, 1]$ es el encargado de establecer un balance entre el costo computacional y la calidad de las soluciones. De hecho, nos permite elegir el grado de aleatoriedad que queremos introducir en cuanto a la construcción de la solución. Si $\alpha = 0$, el algoritmo es completamente *greedy*, mientras que si $\alpha = 1$ se trata de una estrategia completamente aleatoria. De la misma forma pasa con los valores intermedios, es decir, a medida

que α se acerca a 1 aumenta el grado de aleatoriedad (y viceversa).

Cabe destacar que, una vez se ha construído la RCL, el elemento que se incorpora a la solución es seleccionado de manera aleatoria. Seguidamente, se elimina dicho elemento de la lista de candidatos y se re-evaluan los costes incrementales.

A continuación, se presenta el pseudocódigo [10] que describe en rasgos generales la fase constructiva del algoritmo. $fo[i]$ representa el valor de la función objetivo al incluir el elemento i de la lista de candidatos en la solución.

1. Inicializamos el conjunto C de elementos candidatos.
2. Evaluamos el coste incremental de introducir el elemento $i \in C$ en la solución: $fo[i]\ \forall i \in C$.
3. Mientras $C \neq \emptyset$:
   a) $fo_{min} = \text{mín}\{fo[i] | i \in C\}$
   b) $fo_{max} = \text{máx}\{fo[i] | i \in C\}$
   c) $RCL = \{i \in C | fo[i] \leq fo_{min} + \alpha(fo_{max} - fo_{min})\}$
   d) Seleccionamos aleatoriamente un elemento t de RCL
   e) $solution = solution \cup \{t\}$
   f) Actualizar el conjunto de candidatos C: $C = C \backslash \{t\}$.
   g) Re-evaluar los costes incrementales $fo[i]\ \forall i \in C$.
4. *Return(solution)*

3.4.2. Búsqueda local

El algoritmo de búsqueda local es el encargado de, una vez se ha obtenido una primera solución, realizar estrategias de mejora (intercambios, inserciones, etc.) con el objetivo de encontrar óptimos locales en caso de que sea posible y lo permita el criterio de parada del algoritmo.

En este caso, la estrategia de búsqueda local a seguir puede ser de dos tipos: *best-improvement* y *first-improvement*. En el primer caso, se examinan todos los posibles movimientos en una vecindad y se selecciona el que produce la mayor mejora. En cambio, en la estrategia *first-improvement* se examina la vecindad de una solución y se selecciona el primer movimiento que produce una mejora.

3.4.3. Implementación del método

En el siguiente apartado se desarrollará con detalle la implementación en RStudio del método GRASP. En el Anexo E de este documento podemos consultar el código completo del algoritmo y que será utilizado posteriormente para obtener las simulaciones numéricas pertinentes. Sin embargo, resulta interesante comentar las particularidades introducidas tanto en la fase de construcción como en la búsqueda local en cuanto a técnicas de construcción de lista de candidatos, estrategias de búsqueda en el vecindario o criterios de parada. Para ello, comentaremos parte por parte el código implementado.

```r
# Funcion para generar la lista de candidatos (RCL)
generate_rcl <- function(current_solution, remaining_jobs, p, alpha) {
  rcldata <- data.frame(job=numeric(0), position=numeric(0), fo=numeric(0))
  for (i in 1:length(remaining_jobs)) {
    for (pos in 1:length(current_solution)){
    candidate_solution <- append(current_solution, remaining_jobs[i], after=pos
        -1)
    fo <- objectivefunction(candidate_solution, p)
    rcl <- c(remaining_jobs[i], pos, fo)
    rcldata[i,] <-rcl
    }
  }

  # Calcular el limite superior e inferior de la RCL
  min_fo <- min(rcldata$fo)
  max_fo <- max(rcldata$fo)
  lower_bound <- min_fo
  upper_bound <- min_fo + alpha * (max_fo - min_fo)

  # Filtrar los trabajos que estan dentro de la RCL
  listarcl <- rcldata[rcldata$fo >= lower_bound & rcldata$fo <= upper_bound,]

  # Ordenar la RCL en orden creciente de funcion objetivo
  listarcl <- listarcl[order(listarcl$fo), ]
  return(listarcl)
}
```

Figura 3.8: Función de R para obtener la Lista Restringida de Candidatos

En primer lugar, se deberían definir el cálculo de las funciones objetivo que se van a tener en cuenta. Recordemos que habíamos marcado como objetivo la minimización del *makespan*, *Flow Time* y *Idle Time* de forma simultánea. Más adelante se definirá la ponderación de la función objetivo a evaluar como combinación lineal de las medidas comentadas. El código de implementación lo encontramos en el Apéndice A.

A continuación, pasaríamos a implementar la función que, dada una solución parcial, una lista con los trabajos restantes a incluir, la matriz de tiempos y el valor del parámetros α, devuelve una lista con todas las posibilidades de inserción de los trabajos restantes en las posiciones posibles de la solución parcial, junto con el valor de la función objetivo en cada caso. Con este objetivo, se comprueba el coste incremental en la función objetivo definida al añadir cada uno de los trabajos restantes en cada una de las posiciones disponibles. Esta información se almacena en un *dataframe* con columnas "Trabajo", "Posición" y "Función objetivo". Una vez se han evaluado todas las casuísticas, se obtienen los valores máximo y mínimo de la función objetivo de entre todas las opciones (fo_{min} y fo_{max} respectivamente), a partir de los cuales se calculan los límites inferior y superior de la lista restringida de candidatos (RCL). Seguidamente, se realiza un filtrado del *dataframe* para eliminar aquellos casos que no se encuentran dentro de los límites establecidos. El código de R lo vemos en la figura 3.8.

Por otra parte, implementamos la función encargada de ejecutar la fase de construcción *greedy* del método. Las entradas de la función son únicamente la matriz de tiempos y el valor de α, que, como ya hemos comentado anteriormente, será determinante en el carácter aleatorio de la solución obtenida. Cabe comentar la estrategia tomada para la inicialización de solución parcial que se va a construir. En lugar de partir de una solución vacía, por motivos computacionales, se ha decidido inicializar la solución parcial incluyendo uno de los trabajos de manera aleatoria. De esta forma, mientras haya trabajos restantes a incluir en la solución parcial, se llama a la función de construcción de la lista de candidatos que hemos definido. Seguidamente, se selecciona aleatoriamente uno de los elementos de dicha lista, se incluye en la solución parcial y se elimina de la lista de trabajos restantes. Como ya hemos comentado, este procedimiento se ejecuta hasta que la lista de trabajos restante esté vacía, lo cual significaría que se ha completado la solución. Por tanto, la función devuelve como salida esta solución completa. El código en R lo podemos consultar en la figura 3.9.

Se ha decidido tomar el *swap* como estrategia de búsqueda local, mediante la cual se van realizando intercambios entre trabajos dentro de la propia solución para evaluar la variación de la función objetivo. En particular, la función que hemos implementado para ello se encarga de recorrer todos los trabajos posibles y evaluar la función objetivo si se intercambiara con todas los trabajos posibles. Por ejemplo, el primer trabajo de la solución dada se intercambia con el segundo, se evalúa el valor de la función objetivo en esta nueva solución y, si mejora respecto la actual, se actualiza. Así sucesivamente hasta haber comprobado o realizado todos los intercambios posibles. Sin embargo, para aumentar la aleatoriedad de esta búsqueda y evitar que la búsqueda converja a un único escenario se ha establecido una especie de criterio de parada que permite que, en caso de encontrar una mejora al hacer un intercambio, además de realizar el cambio, se inicialice la búsqueda desde el principio, comprobando de nuevo todos los trabajos en todas las posiciones. Esto permite explorar un vecindario más amplio y no acotar las posibilidades de búsqueda a medida que se toman decisiones. En la figura 3.10 podemos consultar el código en R.

Finalmente, se implementa una función encargada de llevar a cabo el algoritmo a completo, llamando a las funciones de construcción de la solución y de búsqueda local cuando sea pertinente. Para reducir los costes computacionales, se ha establecido un límite de tiempo temporal variable en función del tamaño de la instancia que se estaba ejecutando. De esta manera, mientras el tiempo de ejecución sea menor que el límite establecido, el algoritmo construye soluciones y las intenta mejorar de manera local, comprobando en cada iteración si la solución obtenida es mejor que la que ya se tenía para actualizarla en caso afirmativo. En particular, en nuestro caso hemos definido como límite temporal 60 segundos para evitar tiempos computaciones descomunales, es decir, el algoritmo se ejecuta entero varias veces hasta que el tiempo de computacional sea superior a 1 minuto. En la figura 3.11 podemos consultar el código en R.

En resumen, en el presente capítulo se presentan métodos heurísticos que tratan de mejorar las soluciones obtenidas mediante el Algoritmo NEH, uno de los métodos más utilizados para afrontar este problema de secuenciación y que ofrece soluciones bastante buenas. En particular, los métodos propuestos tienen en cuenta aspectos como la reducción de los huecos entre trabajos, que se alínea con nuestro objetivo de reducir tiempos de espera y bloqueo en las línea de producción. Además, se añade una semilla de aleatoriedad gracias al algoritmo GRASP, al cual le hemos introducido particularidades en cuanto a estrategias de construcción y mejora.

```
# Funci n para generar una soluci n inicial utilizando la RCL
greedyConstructionRCL <- function(p, alpha) {
  n <- nrow(p) # N mero de trabajos

  remaining_jobs <- 1:n # Trabajos a n no asignados a la soluci n
  firstjob <-sample(n, 1)
  current_solution <- c(firstjob) # Soluci n actual, comienza vac a
  remaining_jobs <- remaining_jobs[-firstjob]

  while (length(remaining_jobs) > 0) {
    listarcl <- generate_rcl(current_solution, remaining_jobs, p, alpha)
    if (length(listarcl) > 0) {
      # Elegir un trabajo aleatorio de la RCL
      indice_aleatorio <- sample(nrow(listarcl), 1)
      selected_job <- listarcl[indice_aleatorio, ]
      # Agregar el trabajo seleccionado a la soluci n actual en la posici n
      current_solution <- append(current_solution, selected_job$job, after=
          selected_job$position-1)
      # Eliminar el trabajo seleccionado de la lista de candidatos
      remaining_jobs <- setdiff(remaining_jobs, selected_job$job)
    } else {
      # Si la RCL est  vac a, tomar un trabajo aleatorio de los restantes
      selected_job <- sample(remaining_jobs, 1)

      # Agregar el trabajo seleccionado a la soluci n actual en una posici n
          random
      posrandom <-sample(current_solution, 1)
      current_solution <- append(current_solution, selected_job, after=posrandom
          -1)

      # Eliminar el trabajo seleccionado de la lista de candidatos
      remaining_jobs <- setdiff(remaining_jobs, selected_job$job)
    }
  }

  return(current_solution)
}
```

Figura 3.9: Función en R de la fase constructiva *Greedy*

```
#Fase b squeda local

localsearch<- function(solution, p, max_iter){
  best_solution <- solution
  n <- nrow(p)
  bestfo<- objectivefunction(solution,p)
  iter <-1
  while (iter <= max_iter){
    for (i in 1:n){
      for (k in 1:n){
        new_solution <- intercambiotrabajos(best_solution,i,k)
        newfo <- objectivefunction(new_solution,p)
        if (newfo < bestfo){
          best_solution<- new_solution
          bestfo <- newfo
          i<-1
          break
        }
      }
    }
    iter <- iter +1
  }
  return(best_solution)

}
```

Figura 3.10: Función en R de la búsqueda local

```
grasp <- function(p, maxiter, alpha, time_limit_seconds) {
  best_solution <- NULL  # Initialize the best solution to be empty
  start_time <- Sys.time()  # Record the start time

  while (difftime(Sys.time(), start_time, units = "secs") < time_limit_seconds)
      {
    solution <- greedyConstructionRCL(p, alpha)
    current_best <- localsearch(solution, p, maxiter)

    # Update the best solution if the current one is better
    if (is.null(best_solution) || current_best < best_solution) {
      best_solution <- current_best
    }
  }

  return(best_solution)
}
```

Figura 3.11: Función en R del algoritmo GRASP

Capítulo 4

Resultados numéricos

4.1. Función objetivo multicriterio

En primer lugar, es necesario definir y modelar matemáticamente la función objetivo del problema. Para ello, nos centramos en determinar cuál sería el propósito final de nuestro estudio. Notemos que la investigación realizada en este trabajo tiene como fin abordar un caso de estudio de una industria real. Por este motivo, son varios los medibles que quieren ser optimizados. las prioridades se centran en reducir el tiempo de producción para aumentar la productividad y evitar la inactividad de la maquinaria. Todo ello se traduce en minimizar el *makespan*, el tiempo de flujo y el tiempo de inactividad o *Idle Time*, conceptos revisados en el primer capítulo de este mismo trabajo.

Por una parte, el *makespan* es uno de los objetivos más estudiados en la literatura del *Flow Shop* y hace referencia al tiempo de completación del último trabajo procesado en la última máquina. Matemáticamente, se denota por C_{max} y se puede calcular mediante la siguiente expresión:

$$C_{max}(\pi) = \max_{i=1,\ldots,n} C_i(\pi),$$

donde C_i es el tiempo de completación del trabajo i y π representa una solución factible.

En segundo lugar, el tiempo de flujo asociado a un determinado trabajo viene dado por el tiempo total de procesado y el tiempo de indisponibilidad de la maquinaria.

$$F_i(\pi) = C_i(\pi) - r_i,$$

donde r_i representa la totalidad del tiempo que un trabajo ha permanecido en espera debido a la indisponibilidad de la maquinaria.

Otra forma de definir el tiempo de flujo de un determinado trabajo es mediante la suma de los tiempos de proceso y los tiempos de espera, es decir,

$$F_i(\pi) = p_i + W_i(\pi)$$

donde $W_i(\pi)$ es el tiempo de espera del trabajo i dada una solución factible π. En definitiva, W_i es el tiempo que el trabajo i permanece parado debido a que la próxima máquina siguiente sigue procesando el trabajo anterior.

Por tanto, el tiempo de flujo total corresponde a la suma de los tiempos de flujo de los n trabajos, es decir,

$$F(\pi) = \sum_{i=1}^{n} F_i$$

Análogamente,

$$F(\pi) = \sum_{i=1}^{n} (C_i(\pi) - r_i).$$

También,

$$F(\pi) = \sum_{i=1}^{n} p_i + W_i(\pi).$$

Por último, el tiempo de inactividad o *Idle Time* asociado a una máquina recoge todos aquellos tiempos muertos conocidos como tiempos de espera y de bloqueo. Aunque más adelante definiremos con más detalle este concepto propio de las líneas de producción, cabe comentar que los tiempos de inactividad afectan negativamente a la eficiencia y productividad de un proceso de fabricación, puesto que se trata de tiempos "muertos" en cuanto a productividad en que la maquinaria se encuentra sin material que procesar. En definitiva, el tiempo de inactividad de una máquina viene dada por:

$$I_j(\pi) = \max_{i=1,\dots,n} C_{ij}(\pi) - \sum_{i=1,\dots,n} p_{ij}$$

En definitiva, el tiempo de inactividad se calcula como la diferencia entre el tiempo de completación y el tiempo neto de producción (tiempos de proceso), puesto que esa discrepancia de tiempos recoge los tiempos de espera y bloqueo.

De la misma forma, el tiempo de inactividad total dada una solución factible π vendrá dado por:

$$I = \sum_{j=1}^{m} I_j$$

En capítulos anteriores se había presentado la posibilidad de enfocar el problema de *Flow Shop* como un problema multiobjetivo. Sin embargo, debido a la complejidad que esto supone, se ha decidido trabajar con un problema monoobjetivo con un carácter multicriterio. Esta decisión además viene fundamentada por el hecho de que los medibles que se van a optimizar no son mutuamente excluyentes, es decir, no entran en contradicción (el aumento de uno no supone la disminución de otro y viceversa). Además, notemos que se había planteado como objetivo final construir un algoritmo que minimizara los tiempos de espera y bloqueo debido a su significancia en las pérdidas de eficiencia. Todos los medibles definidos están teniendo en cuenta de alguna manera u otra este medible y en todos los casos el propósito es minimizar.

Por todo ello, se presenta la posibilidad de modelarlo mediante una única función objetivo en la que se incluyen las 3 características mediante ponderaciones. A partir de este momento, hablaremos de una función objetivo ponderada a minimizar.

El siguiente paso consiste en equilibrar los diferentes criterios. Notemos que los 3 objetivos son prácticamente igual de importantes, por lo que una posibilidad sería asignarles a los 3 la misma ponderación. Sin embargo, hemos optado por asignar importancias relativas a cada objetivo. El motivo de esta ponderación no neutral es porque de esta manera le hemos podido asignar un peso ligeramente superior al *makespan*, puesto que consideramos que dicha medida incluye en su significado las otras 2 y al minimizarla estaríamos minimizando también las otras. En cuanto a *Flow Time* y *Idle Time* se les ha asignado el mismo peso por simplicidad y neutralidad.

A continuación, presentamos la función objetivo que vamos a minimizar y que hemos implementado en los métodos planteados.

$$\min F.O.: \quad f(\pi) = 0{,}4 \cdot C_{max}(\pi) + 0{,}3 \cdot F(\pi) + 0{,}3 \cdot I(\pi).$$

4.2. Calibración de hiperparámetros

En el presente apartado se presenta el procedimiento llevado a cabo con el objetivo de calibrar los hiperparámetros de los algoritmos planteados, para que las soluciones obtenidas sean lo mejor posibles. En particular, procedemos a calibrar los hiperparámetros del algoritmo GRASP.

Llamamos hiperparámetros a todos aquellos parámetros de un modelo que se deben configurar por adelantado y que son proporcionados el autor antes de entrenar el algoritmo. Por este motivo, es tan importante el estudio a priori de qué valores favorecen el desempeño de algoritmo en vez de definirlos de manera arbitraria. De hecho, su significatividad se base en que el rendimiento de todo el modelo se fundamenta en los valores de los hiperparámetros especificados. En particular, los hiperparámetros del método a determinar son: en el caso de la fase de construcción greedy, el tamaño de la lista restringida de candidatos, y, en el caso de la fase de búsqueda local, el número máximo de iteraciones de esta fase iterativa. Para ello, se ha realizado un diseño de experimentos mediante el cual son estudiadas todas las posibles combinaciones de valores de α y número de iteraciones. En cada caso, es evaluado un conjunto de 20 instancias de distintos tamaños, con el objetivo de disponer de una muestra lo suficientemente representativa como para extraer conclusiones. De esta manera, para cada posible combinación de parámetros obtendremos una tabla con el valor de los estadísticos más utilizados (media, máximo, mínimo y desviación estándard) del valor de la función objetivo ponderada, así como del *makespan*, *flowtime* y *idletime*. Con todos estos resultados, obtendremos un resumen gráfico a partir del cual concluir cuál es la combinación de parámetros que mejor resultados nos ofrece.

En primer lugar, cabe recordar la influencia en el desempeño y resultados del método en función del valor de los parámetros introducidos.

Por una parte, el valor de α nos determina el nivel de aleatoriedad en la construcción de la lista de candidatos restringida (RCL), que recordemos que está formada por los elementos a incluir en la solución parcial

que menos incrementen la función de coste. Recordemos también que habíamos definido la restricción de la lista mediante la siguiente cota: $fo[i] \in [min(\{fo\}), min(\{fo\}) + \alpha \cdot (max(\{fo\}) - min(\{fo\}))]$, donde $fo[i]$ simboliza el valor de la función objetivo si se incluye el trabajo i en la solución parcial construída hasta el momento y $\{fo\}$ el conjunto de funciones objetivo de los posibles trabajos a incluir en la solución. El valor de α es el encargado de controlar el tamaño de la lista y de otorgarle el carácter aleatorio a la selección de candidatos, además de establecer un balance entre el coste computacional y la calidad de las soluciones. Toma valores entre 0 y 1 ($\alpha \in [0, 1]$). En particular, para $\alpha = 0$, se vuelve un algoritmo completamente greedy, mientras que para $\alpha = 1$ se vuelve una estrategia completamente aleatoria. En la literatura encontramos distintas formas de afrontar la elección del valor de α. Algunos optan por fijar un valor constante, otros por variar su valor dentro de un rango de valores según la calidad de las soluciones, favoreciendo con el tiempo aquellos que dan mejores resultados, mientras que otros seleccionan aleatoriamente dentro de cierto rango. En nuestro caso, hemos decidido por fijar un valor constante de α después de realizar las comprobaciones experimentales pertinentes de que rinde bien en cualquiera de los escenarios casuísticos. Por todo ello, se han considerado 5 valores equidistantes entre 0 y 1 para realizar las pruebas experimentales, incluyendo los valores extremos para evaluar el comportamiento del modelo en el caso de que el sea completamente aleatorio o completamente Greedy. En particular, se toman los siguientes valores de α: $\alpha \in \{0, 0{,}25, 0{,}5, 0{,}75, 1\}$.

Por otra parte, el número de iteraciones que queremos calibrar está referido al número de veces que la solución obtenida en la fase de construcción es mejorada con un procedimiento de búsqueda local. Notemos que se trata de buscar el equilibrio entre la calidad de la solución y el tiempo de ejecución. Por tanto, el número de iteraciones debe de ser lo suficientemente grande como para llegar a soluciones locales buenas, pero lo suficientemente pequeño como para no incrementar significativamente el coste computacional. De hecho, en algunos casos es posible que no sea necesario encontrar soluciones extremadamente precisas, sino más bien soluciones bastante precisas en tiempos razonables. Por todo ello, el objetivo que se busca mediante la calibración de este parámetro es encontrar el punto de inflexión de la curva entre el valor de la función objetivo y el número de iteraciones, es decir, el mínimo valor del número de iteraciones a partir del cual el valor de la función objetivo no mejora significativamente. En nuestro caso particular, vamos a realizar las pruebas tomando como valores $\{0, 50, 100\}$, donde $n \in \mathbb{Z}^+$ y $m \in \mathbb{Z}^+$ representan el número de trabajos y máquinas respectivamente. Cabe destacar que si tomamos como número de iteraciones el 0 estaríamos en el caso de que no se aplica la fase de búsqueda local. El hecho de realizar los experimentos considerando este valor nos permitirá más adelante comprobar si la fase de búsqueda local realmente aporta mejoras significativas a la calidad de la solución.

A continuación, definimos todas las posibles combinaciones de parámetros a computar. Hemos denotado con un identificador numérico O cada una de las posibles opciones/combinaciones de parámetros, α el tamaño de la lista restringida de candidatos (RCL) y n_{iter} el número máximo de iteraciones en la fase de búsqueda local. En la Tabla 4.1 vemos resumida esta información y la relación entre el identificador de opción y su significado para luego facilitar la interpretabilidad de los resultados numéricos y tablas.

En la Tabla 4.2 observamos un resumen del tamaño de cada una de las 20 instancias tomadas como muestra representativa en el experimento, donde I corresponde al número de instancia, n al número de trabajos y m al número de máquinas. En resumen, los tamaños comprenden desde 10 hasta 50 trabajos y el número de máquinas varía entre 5 y 20.

Opción	α	$\mathbf{n_{iter}}$
1	0	0
2	0.25	0
3	0.5	0
4	0.75	0
5	1	0
6	0	50
7	0.25	50
8	0.5	50
9	0.75	50
10	1	50
11	0	100
12	0.25	100
13	0.5	100
14	0.75	100
15	1	100

Cuadro 4.1: Posibles combinaciones de parámetros (α y n_{iter}) consideradas y el identificador asociado

I	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
n	10	10	10	10	20	20	20	20	30	30	30	30	40	40	40	40	50	50	50	50
m	5	10	15	20	5	10	15	20	5	10	15	20	5	10	15	20	5	10	15	20

Cuadro 4.2: Tabla resumen de tamaños de las instancias, donde n es el número de trabajos a secuenciar y m el número de máquinas

Una vez definido el tamaño de cada instancia y todas las posibles combinaciones de parámetros a evaluar, procedemos a realizar el experimento. En las tablas que encontramos en el Anexo H encontramos los resultados detallados obtenidos mediante la ejecuciíon del algoritmo para cada una de las posibles 15 combinaciones de parámetros y en cada una de las 20 instancias consideradas. En dichas tablas encontramos el valor de la función objetivo ponderada definida, el valor del *makespan*, el valor del *flow time* y el valor del *idle time* en cada caso.

Sin embargo, notemos que lo que realmente nos interesa es conocer cómo se comporta el algoritmo en promedio a medida que varía el valor de los parámetros, mas allá de lo que suceda a nivel particular para cada tipo de instancia. En la Tabla 4.3 se presenta el resumen clave de los experiments realizados. En ella, podemos observar el valor de los estadísticos que hemos considerado más importantes para tomar la decisión, es decir, para cada posible combinación de parámetros el valor medio de la función objetivo en las instancias evaluadas, así como el valor mínimo, máximo y la mediana. El análisis en profundidad de estas medidas nos ayudará a extraer conclusiones sobre cómo se comporta cada una de ellas a medida que varían los parámetros y, por tanto, en función de nuestros propósitos, elegir la opción adecuada.

Si analizamos en detalle los resultados mostrados en la Tabla 4.3 llegamos a la conclusión que la combi-

Opción	α	$\mathbf{n_{iter}}$	$\mathbf{mean(fo)}$	máx **fo**	mín **fo**
1	0	0	19521.22	47025.3	1730.6
2	0.25	0	20104.04	49199.4	1940.6
3	0.5	0	21268.26	53755	2005.7
4	0.75	0	21911.4	53548.5	2000.2
5	1	0	22195.75	55262.4	2050.4
6	0	50	18156.08	45068	1716.2
7	0.25	50	18183.92	45442.4	1729.7
8	0.5	50	18273.22	46001.9	1708.6
9	0.75	50	18337.42	46964.3	1713.4
10	1	50	18222.69	45765.9	1712.2
11	0	100	18149.49	45401.8	1706.5
12	0.25	100	18202.03	45236.7	1735.7
13	0.5	100	18222.94	45123	1712.3
14	0.75	100	18211.81	45156.7	1755.7
15	1	100	18287.72	45410.4	1719.4

Cuadro 4.3: Tabla resumen de los estadísticos de la función objetivo de los experimentos realizados para cada una de las posibles combinaciones de parámetros de GRASP

nación con menor valor de la función objetivo en promedio es la opción 11, que toma $\alpha = 0$ y 100 iteraciones en la búsqueda local. Sin embargo, respecto al máximo valor de la función objetivo encontrado en cada caso, notamos que la combinación más conveniente es la opción número 6, que toma los valores $\alpha = 0$ y n_{iter}.

Por una parte, los valores de la función objetivo promedio de las opciones 6 y 11 no varían excesivamente. De hecho, la función promedio en caso de la opción 6 es la segunda menor después de la función promedio de la opción 11. Sin embargo, es necesaria la evidencia estadística para justificar que puede despreciarse la diferencia existente entre ambas. Por este motivo, se ha realizado un Análisis de la varianza para comparar medias con el objetivo de analizar si el promedio de la función objetivo en el caso de tomar $\alpha = 0$ y $n_iter = 50$ es significativamente diferente a tomar $\alpha = 0$ y $n_{iter} = 100$. Tras comprobar las hipótesis de independencia, homocedasticidad y de normalidad de los datos, se ha obtenido la tabla resumen de ANOVA que encontramos en la Figura 4.10. Efectivamente, mirando el $p - value$ del análisis llegamos a la conclusión de que no existen diferencias estadísticas entre ambos promedios, por lo que elegir cualquiera de las opciones sería "equivalente".

```
              Df    Sum Sq   Mean Sq F value Pr(>F)
datos$Opcion   1 4.330e+02       433       0  0.999
Residuals     38 5.477e+09 144144263
```

Figura 4.1: Análisis de la varianza para comparar la media de la opción 6 con la media de la opción 11

Sin embargo, lo que no resultaría "equivalente" entre ambas opciones es el coste computacional del algoritmo. De hecho, el motivo por el que se está realizando esta comprobación y que ha despertado este debate, en lugar de considerar directamente la Opción 11, que era la que mejores resultados proporcionaba, es la

significativa diferencia en el tiempo necesario para ejecutar cada uno de los códigos. Es evidente que, computacionalmente hablando, no compensa realizar 100 iteraciones en lugar de 50 si la solución obtenida va a ser prácticamente equivalente en cuanto a calidad. Además, para reafirmar esta decisión, se ha representado el valor de la función objetivo promedio frente al número de iteraciones para $\alpha = 0$ (opciones 1, 6 y 11). En la Figura 4.2 vemos claramente como a partir de 50 iteraciones el valor de la función objetivo se estabiliza y no mejora significativamente. Al ser el mínimo valor de iteraciones a partir del cual este fenómeno sucede, es conveniente considerarlo como el número de iteraciones "óptimo". Como ya se había comentado al principio de la sección, es preferible encontrar soluciones bastante precisas en tiempos razonables a soluciones extremadamente buenas en tiempos de ejecución desmesurados.

Por otra parte, en cuanto al valor de α, es evidente que las mejores soluciones del algoritmo se han obtenido con el valor $\alpha = 0$. Cabe comentar que es un fenómeno bastante extraño, puesto que estaríamos despreciando la semilla aleatoria que caracteriza al algoritmo GRASP en la construcción de soluciones y estaríamos tomando un algoritmo completamente *greedy*. De hecho, tomando el valor $\alpha = 0$ estamos restringiendo la lista de candidatos a un único elemento, que resultará ser el elemento con menor incremento de la función objetivo al ser incluído en la solución parcial. En definitiva, se elimina la aleatoriedad y se restringe en cada iteración al elemento que mejora la calidad de la solución en mayor medida de entre todos los candidatos.

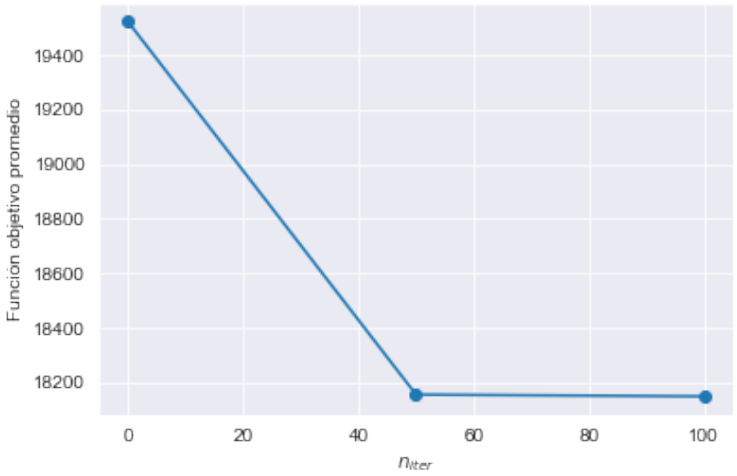


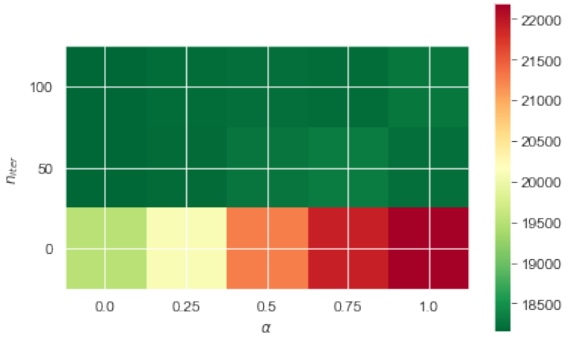
Figura 4.2: Valor promedio de la función objetivo frente el número de iteraciones para $\alpha = 0$

Para facilitar la interpretabilidad de los resultados y poder extraer conclusiones gráficas, se ha recurrido a una búsqueda de cuadrícula, también conocida como *Grid Search*. La búsqueda de cuadrícula [13] es una técnica de ajuste que intenta calcular los valores óptimos de los hiperparámetros. En definitiva, se trata de una búsqueda exhaustiva que se realiza sobre los valores de los parámetros específicos de un modelo.

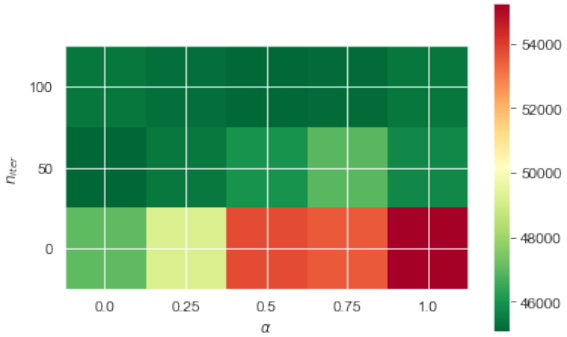
Con la ayuda de las gráficas de la Figura 4.3 podemos llegar a conclusiones interesantes sobre el comportamiento del método en función de los parámetros. Cabe comentar que los valores más rojizos corresponden a valores más elevados de la función objetivo. En cambio, los valores más amarillentos o verdes corresponden a valores más bajos de la función objetivo de manera gradual, es decir, cuanto más verde, mejor calidad de la

solución. Veamos qué podemos deducir analizando en detalle estos mapas de calor.

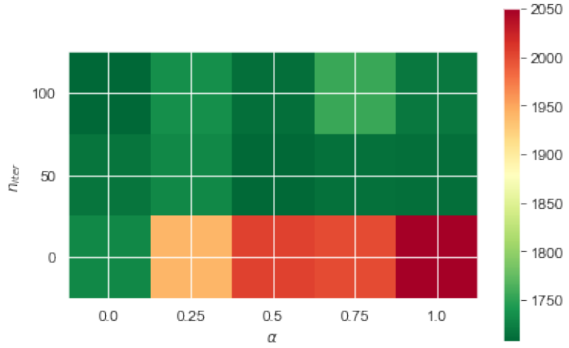
Por una parte, queda más que demostrada gráficamente la importancia de la fase de búsqueda local. De hecho, la calidad de los resultados mejora significativamente cuando pase de 0 iteraciones en la búsqueda local (no hay búsqueda local) a 50 o 100 iteraciones. Este hecho lo podemos ver gráficamente si nos fijamos en la cuadrícula inferior, que es de tonos más rojizos o anaranjados. Por otra parte, el valor de α no parece ser muy determinante en la calidad de las soluciones en el caso de un gran número de iteraciones. Sin embargo, para número de iteraciones 0 se ve claramente como a medida que aumenta el grado de aleatoriedad en la construcción de la solución aumenta el valor de la función objetivo. Esto justifica una vez más que se haya tomado como valor "óptimo" $\alpha = 0$, puesto que a medida que aumenta este valor disminuye la calidad de las soluciones obtenidas y encontramos tonos más rojizos.



(a) *Función objetivo promedio*



(b) Máximo de la función objetivo



(c) Mínimo de la función objetivo

Figura 4.3: *Grid Search* para la calibración de los hiperparámetros del algoritmo GRASP para $\alpha \in \{0, 0{,}25, 0{,}5, 0{,}75, 1\}$ y $n_{iter} \in \{0, 50, 100\}$

Los *scripts* de Python utilizados para graficar las Figuras 4.2 y 4.3 se pueden encontrar en el Anexo I.

En conclusión, tomaremos como hiperparámetros "óptimos" del algoritmo GRASP $\alpha = 0$ y n_{iter} al haberse demostrado por varios medios que daban resultados de buena calidad en tiempos razonables.

4.3. Comparación de algoritmos

En el capítulo anterior se han presentado 4 algoritmos heurísticos con diferentes alcances, características y enfoques. Sin embargo, es necesario realizar un estudio en detalle que nos ayude a decidir cuál de ellos se ajuste mejor al propósito del estudio y a los datos que se van a tratar. Con este propósito, en este capítulo se presentan y estudian en profundidad mediante técnicas estadísticas los resultados obtenidos al ejecutar cada uno de los algoritmos propuestos en instancias de diferentes magnitudes.

4.3.1. Diseño de experimentos: Generación de instancias

En primer lugar, debemos definir el alcance de nuestro experimento. Para evaluar el comportamiento de los algoritmos teniendo en cuenta el tamaño de los datos, es decir, el número de máquinas y trabajos a secuenciar, se han tomado 20 instancias de diferentes magnitudes. En particular, el número de trabajos a secuenciar varía entre 10 y 50, con saltos de 10; mientras que el número de máquinas varía entre 5 y 20 con saltos de 5. De esta manera, se han comprobado todas las posibles combinaciones de las variaciones. De hecho, en la Tabla 4.2 encontramos un resumen de las posibles variaciones y el identificados al que están asociadas. Cabe destacar que, aunque los tamaños de las instancias tomadas coincidan con los tamaños de las instancias utilizadas para calibrar los hiperparámetros del GRASP, se trata de instancias completamente diferentes, con datos numéricos distintos. Esta variación se realiza con el objetivo de que incorporar diversidad en los datos y evitar el sobreajuste de los hiperparámetros del modelo GRASP elegidos como "óptimos", permitiendo así evaluar los modelos en conjuntos de datos heterogéneos y verificar así su robustez y generalización.

Las instancias que se van a utilizar en este caso para realizar las simulaciones numéricas pueden ser consultadas en el Anexo J.

4.3.2. Resultados numéricos

En la siguiente subsección se presentarán los resultados numéricos obtenidos tras ejecutar los algoritmos introduciendo las 20 instancias tomadas. Cabe comentar que todos los algoritmos propuestos en el presente trabajo han sido programados, implementados y ejecutados en un ordenador con sistema operativo macOS Monterey versión 12.6.3 con procesador Intel Core i5 de doble núcleo y 1,8 GHz.

En las tablas 4.4, 4.5, 4.6 y 4.7 encontramos información detallada sobre el valor de la función objetivo, el *Makespan*, el *Flow Time* y el *Idle Time* para cada una de las 20 instancias tomadas al ser evaluadas con el Algoritmo NEH, el Algoritmo Heurístico 1, el Algoritmo Heurístico 2 y el Algoritmo GRASP respectivamente. Cabe comentar que el Algoritmo GRASP se ha implementado con los parámetros que habíamos demostrado ser "óptimos" en la calibración, es decir, $\alpha = 0$ y número máximo de iteraciones en la búsqueda local de 50.

Instancia	**Función objetivo**	***Makespan***	***Flow Time***	***Idle Time***
1	1981.4	875	4228	1210
2	4153	1294	7875	4243
3	7229.5	1630	11144	10781
4	9344.5	1897	13115	15504
5	6027.5	1577	16550	1439
6	9247	1981	23102	5080
7	12876.2	2273	29220	10670
8	16877.2	2749	34006	18586
9	9742.3	1891	28953	1000
10	16189	2554	44579	5979
11	20184.8	2789	50726	12838
12	29593.8	3660	70393	23373
13	16297.5	2610	49223	1622
14	25578.9	3213	74213	6766
15	32520.4	3718	88167	15277
16	38573.4	4035	99453	23745
17	26191	3244	81432	1546
18	32862.2	3605	97995	6739
19	45314.2	4420	129438	15716
20	52124.3	4595	142928	24693

Cuadro 4.4: Resultados numéricos obtenidos con el Algoritmo NEH

Instancia	**Función objetivo**	***Makespan***	***Flow Time***	***Idle Time***
1	2136.3	780	5187	894
2	4095.7	1201	8452	3599
3	7553.5	1564	12927	10166
4	9440.2	1837	14130	14888
5	6750.1	1546	19187	1252
6	9511.4	1823	25144	4130
7	13217.1	2115	31122	10115
8	17342.8	2551	36679	17729
9	10731.7	1894	32003	1244
10	16385.6	2390	46271	5161
11	20888.1	2727	54003	11988
12	29274.5	3374	70430	22653
13	18825.5	2453	58099	1382
14	25785.5	3008	75893	6048
15	32115.3	3522	89200	13155
16	37828.4	3719	101972	19164
17	27445.5	3132	85521	1788
18	33963.1	3391	103123	5566
19	45150.9	4170	130352	14591
20	54771.7	4627	151696	24707

Cuadro 4.5: Resultados numéricos obtenidos con el Algoritmo Heurístico 1

Instancia	**Función objetivo**	***Makespan***	***Flow Time***	***Idle Time***
1	1814	800	4023	957
2	3996.3	1236	7651	4022
3	7034	1589	10816	10512
4	9311.4	1842	12761	15821
5	5864.9	1544	16118	1373
6	8701.7	1865	21958	4561
7	11769.7	2113	27609	8806
8	16459.3	2641	33807	17536
9	9334.7	1841	27747	914
10	15510.1	2452	42772	5659
11	19408.2	2739	48977	12065
12	28178.7	3450	67820	21509
13	15443.4	2523	46646	1468
14	23441.4	3000	68037	6101
15	32161.9	3679	87414	14887
16	37619.8	3925	97934	22232
17	24702.5	3107	76750	1449
18	32413.7	3545	96806	6513
19	42672.1	4195	122490	14157
20	50990.6	4544	139915	23995

Cuadro 4.6: Resultados numéricos obtenidos con el Algoritmo Heurístico 2

Instancia	Función objetivo	*Makespan*	*Flow Time*	*Idle Time*
1	1788.7	739	4000	977
2	3900.8	1226	7415	3953
3	6674.5	1534	10611	9592
4	8862.4	1789	13384	13772
5	5422.5	1386	15126	1101
6	8397.3	1755	21593	4058
7	11284.3	2029	26911	7998
8	15917.9	2534	33811	15870
9	8725.3	1765	25984	747
10	14006.3	2300	39536	4085
11	17919.1	2566	46523	9786
12	25524.6	3093	63131	17827
13	14247.7	2323	43368	1027
14	21090.6	2796	62422	4152
15	27561.3	3204	77969	9630
16	33391.9	3616	89725	16760
17	22221.4	2932	69197	965
18	30276.9	3360	91228	5215
19	37677.9	3732	110850	9767
20	46213.2	4206	129896	18540

Cuadro 4.7: Resultados numéricos obtenidos con el Algoritmo GRASP

En primer lugar, cabe comprobar si el desempeño de los algoritmos cumple con las expectativas de alcance que se prometían en la definición del método. Un ejemplo claro es el del algoritmo heurístico 2, el cual se presentaba como una mejora en la solución respecto la solución obtenida con el Algoritmo NEH en cuanto a tiempo de flujo se refiere. De hecho, dicho algoritmo toma como solución inicial una solución obtenida del algoritmo NEH y realiza los intercambios de trabajos pertinentes, en función de la diferencia de tiempos de procesos con su consectuvio, puesto que estos intercambios aumentan las posibilidades de reducir el tiempo de flujo. Consecuentemente, a medida que disminuye el tiempo del flujo (*Flow Time*) disminuye el valor de la función objetivo por la manera en la que está definida. Sin embargo, en este caso consideramos oportuno comprobar que efectivamente entre las soluciones obtenidas con el Algoritmo NEH y el algoritmo heurístico 2 se observa una notable mejoría en cuanto a tiempo de flujo se refiere, además de analizar la magnitud de esta mejoría según el tamaño de las instancias donde se aplica. En la tabla 4.8 obtenemos un resumen del valor del tiempo de flujo con cada algoritmo y en cada instancia. Si bien es cierto que en instancias con menor número de trabajos la mejora entre uno y otro es prácticamente despreciable, a medida que aumenta el número de trabajos a secuenciar podemos afirmar que en promedio se obtienen menores valores de tiempo de flujo con el Algoritmo Heurístico 2 que con el Algortimo NEH. Por tanto, el Algoritmo Heurístico 2 cumple con las expectativas planteadas.

Sin embargo, el medible cuyo comportamiento nos interesa realmente a la hora de comparar la calidad de las soluciones entre algoritmos es el valor de la función objetivo, que recordemos que la hemos definido como una ponderación casi equilibrada de los 3 medibles a estudiar (*Makespan*, *Flow Time* y *Idle Time*. Por

Instancia	Algoritmo NEH	Algoritmo Heurístico 2
1	4228	4023
2	7875	7651
3	11144	10816
4	13115	12761
5	16550	16118
6	23102	21958
7	29220	27609
8	34006	33807
9	28953	27747
10	44579	42772
11	50726	48977
12	70393	67820
13	49223	46646
14	74213	68037
15	88167	87414
16	99453	97934
17	81432	76750
18	97995	96806
19	129438	122490
20	142928	139915
Promedio	54837	52902,55

Cuadro 4.8: Tabla comparativa del valor del tiempo de flujo (*Flow Time*) entre el Algoritmo NEH y el Algoritmo Heurístico 2

ello, en la Tabla 4.9 encontramos un resumen del valor obtenido de la función objetivo para cada instancia con cada algoritmo estudiado.

Además, en la tabla 4.10 encontramos esta misma información pero todavía más condensada. Se indica en cada caso el valor promedio de la función objetivo, el valor máximo y el valor mínimo en cada algoritmo. A partir de esta tabla ya podemos extraer un par de conclusiones. Por una parte, en promedio es claramente mejor en cuanto a calidad de soluciones el algoritmo GRASP. De la misma manera, el máximo valor de la función objetivo (de entre todas las instancias) es mínimo en el caso del Algoritmo GRASP de nuevo, también con una diferencia bastante abismal respecto los otros algoritmos. Sin embargo, aunque en el caso del valor mínimo de la función objetivo el más pequeño sigue siendo el del Algoritmo GRASP, en esta ocasión las diferencias entre unos y otros no resultan tan escandalosas. Este fenómeno podría indicar que en instancias más pequeñas (con menor número de trabajos o de máquinas) los algoritmos se comportan de forma bastante similiar y resulta casi indiferente utilizar uno u otro. Dicho de otra forma, el tamaño de la instancia resulta un factor significativo que debería ser tenido en cuenta en el estudio.

Sin embargo, todas estas afirmaciones se están realizando de manera meramente intuitiva a partir de la observación de los resultados, sin ningún tipo de fundamentación estadística. Por ello, necesitamos realizar

	Algoritmo NEH	Algoritmo Heurístico 1	Algoritmo Heurístico 2	Algoritmo GRASP
Instancia 1	1981,4	2136,3	1814	1788,7
Instancia 2	4153	4095,7	3996,3	3900,8
Instancia 3	7229,5	7553,5	7034	6674,5
Instancia 4	9344,5	9440,2	9311,4	8862,4
Instancia 5	6027,5	6750,1	5864,9	5422,5
Instancia 6	9247	9511,4	8701,7	8397,3
Instancia 7	12876,2	13217,1	11769,7	11284,3
Instancia 8	16877,2	17342,8	16459,3	15917,9
Instancia 9	9742,3	10731,7	9334,7	8725,3
Instancia 10	16189	16385,6	15510,1	14006,3
Instancia 11	20184,8	20888,1	19408,2	17919,1
Instancia 12	29593,8	29274,5	28178,7	25524,6
Instancia 13	16297,5	18825,5	15443,4	14247,7
Instancia 14	25578,9	25785,5	23441,4	21090,6
Instancia 15	32520,4	32115,3	32161,9	27561,3
Instancia 16	38573,4	37828,4	37619,8	33391,9
Instancia 17	26191	27445,5	24702,5	22221,4
Instancia 18	32862,2	33963,1	32413,7	30276,9
Instancia 19	45314,2	45150,9	42672,1	37677,9
Instancia 20	52124,3	54771,7	50990,6	46213,2

Cuadro 4.9: Tabla resumen de los valores de la función objetivo para cada instancia con cada uno de los algoritmos propuestos

análisis que nos den esta evidencia estadística. Por este motivo, vamos a recurrir al porcentaje de desviación relativa (RDP por sus siglas en inglés) entre los diferentes algoritmos para medir la calidad de las soluciones obtenidas. Sea $FO(Alg)$ el valor de la función objetivo del Algoritmo que estemos evaluando y $FO(best)$ el mejor valor de la función objetivo encontrada con todos los algoritmos, el porcentaje de desviación relativa se calcula como:

$$RDP = \frac{FO(Alg) - FO(best)}{FO(best)}. \tag{4.1}$$

Haciendo uso de la equación (4.1), calculamos el valor de RDP de cada instancia con cada algoritmo, esto es, para cada instancia y para cada algoritmo, la diferencia relativa entre el valor de la función objetivo y el mejor valor obtenido para esa instancia en concreto. Se tomaría la Tabla 4.9 como referencia de valores y obtendríamos la Tabla 4.11.

En la tabla 4.11 observamos el valor de RDP para cada instancia y cada algoritmo. Calculando los valores promedios podemos extraer diversas conclusiones. Por una parte, las soluciones obtenidas por el algoritmo heurístico 1 son un 16,87 % peores que las obtenidas por el algoritmo GRASP. Así mismo, las soluciones obtenidas por el algoritmo NEH y por el algoritmo heurístico 2 son un 12,84 % y un 8,06 % respectivamente peores que las obtenidas por el algoritmo GRASP. De hecho, el algoritmo GRASP tiene en promedio un 0 % de porcentaje de desviación relativa RDP o, dicho de otra forma, es en todas las instancias probadas el algoritmo que

Algoritmo	$\mathbf{mean(fo)}$	máx **fo**	mín **fo**
Algoritmo NEH	20645.405	52124.3	1981.4
Algoritmo Heurístico 1	21160.645	54771.7	2136.3
Algoritmo Heurístico 2	19841.42	50990.6	1814
Algoritmo GRASP	18055.23	46213.2	1788.7

Cuadro 4.10: Tabla resumen de los estadísticos de la función objetivo de los experimentos realizados aplicando los métodos heurísticos planteados

mejor resultados ofrece. Esto ya nos lleva a pensar una primera ordenación de algoritmos en cuanto a calidad de soluciones obtenidas, que de mejor a peor quedaría como sigue: Algoritmo GRASP, Algoritmo Heurístico 2, Algoritmo NEH y Algoritmo Heurístico 1. Cabe comentar también que para tamaños menores de número de trabajos (instancias de la 1 a la 8) estas diferencias relativas son en algunos casos ligeramente menos pronunciadas; mientras que para instancias con mayor número de trabajos (instancias de la 9 a la 20) estas diferencias relativas son incluso superiores al promedio. Este fenómeno propone que quizás el factor n correspondiente al número de trabajos secuenciar podría ser significativo en la calidad de las soluciones y debería ser estudiado.

4.3.3. Análisis de resultados y conclusiones

Con el objetivo de explorar analíticamente si existen diferencias estadísticamente significativas entre los resultados que proporciona cada uno de los algoritmos estudiados, vamos a hacer uso de la técnica de análisis de varianza, más conocida como ANOVA [12]. En particular, se realizará un ANOVA con los datos de la tabla 4.11 para analizar si las diferencias relativas RDP entre cada algoritmo con el mejor en cada caso son significativas. Además del factor algortimo, incluiremos otros factores que pensamos que podrían influir en los resultados, como puede ser el tamaño de la instancia.

La técnica de análisis de la varianza, ANOVA o análisis factorial fue desarrollada por Fisher en 1930. El método consiste en estudiar el efecto de uno o más factores sobre la media de una variable contínua. En definitiva, nos permite comparar las medias de dos o más grupos para concluir si son significativamente diferentes.

La hipótesis nula considerada es que la media de la variable estudiada es la misma en los diferentes grupos. De esta manera, la hipótesis alternativa es que existen al menos dos medias que difieren significativamente. En definitiva, ANOVA permite, a partir del estudio de las varianzas, comparar múltiples medias.

La idea intuitiva del método consiste en obtener la media de la variable estudiada en cada uno de los grupos para posteriormente comparar la varianza de estas medias (varianza explicada por la variable grupo, intervarianza) frente a la varianza promedio dentro de los grupos (varianza no explicada por la variable grupo intravarianza). Bajo la hipótesis nula de que todas las observaciones, aún formando parte de diferentes grupos, proceden de la misma población, la varianza ponderada entre grupos será la misma que la varianza promedio dentro de los grupos. Bajo la hipótesis alternativa de que las medias de los grupos están muy diferenciadas, la varianza entre medias se incrementará y dejará de ser igual a la varianza promedio dentro de los grupos.

Sea F_{ratio} el ratio entre la varianza de las medias de los grupos y el promedio de la varianza dentro de los grupos. Se trata del estadístico estudiado en ANOVA y que sigue una distribución F de $Fisher$. Como bajo

	Algoritmo NEH	Algoritmo Heurístico 1	Algoritmo Heurístico 2	Algoritmo GRASP
Instancia 1	10,7732 %	19,4331 %	1,4144 %	0 %
Instancia 2	6,4653 %	4,9964 %	2,4482 %	0 %
Instancia 3	8,3152 %	13,1695 %	5,3862 %	0 %
Instancia 4	5,4398 %	6,5197 %	5,0663 %	0 %
Instancia 5	11,1572 %	24,4832 %	8,1586 %	0 %
Instancia 6	10,1187 %	13,2674 %	3,6250 %	0 %
Instancia 7	14,1072 %	17,1282 %	4,3016 %	0 %
Instancia 8	6,0265 %	8,9516 %	3,4012 %	0 %
Instancia 9	11,6558 %	22,9952 %	6,9843 %	0 %
Instancia 10	15,5837 %	16,9874 %	10,7366 %	0 %
Instancia 11	12,6441 %	16,5689 %	8,3101 %	0 %
Instancia 12	15,9423 %	14,6913 %	10,3982 %	0 %
Instancia 13	14,3869 %	32,1301 %	8,3922 %	0 %
Instancia 14	21,2810 %	22,2606 %	11,1462 %	0 %
Instancia 15	17,9930 %	16,5232 %	16,6922 %	0 %
Instancia 16	15,5172 %	13,2862 %	12,6615 %	0 %
Instancia 17	17,8639 %	23,5093 %	11,1654 %	0 %
Instancia 18	8,5389 %	12,1750 %	7,0575 %	0 %
Instancia 19	20,2673 %	19,8339 %	13,2550 %	0 %
Instancia 20	12,7909 %	18,5196 %	10,3377 %	0 %
Promedio	**12, 84341 %**	**16,87148 %**	**8,056923 %**	**0 %**

Cuadro 4.11: Tabla resumen de los valores RDP en porcentaje de la función objetivo para cada instancia con cada uno de los algoritmos propuestos

la hipótesis nula hemos dicho que estas dos medidas son la misma, se cumple que $F_{ratio} \approx 1$.

Del mismo modo, a mayor diferencias entre las medias de los grupos, mayor será la diferencia de la varianza entre medias y el promedio de la varianza dentro de los grupos, por lo que F tomará valores superiores a 1.

Sea S_1^2 la varianza de una muestra de tamaño N_1 extraída de una población normal de varianza σ_1^2. Sea S_2^2 la varianza de una muestra de tamaño N_2 extraída de una población normal de varianza σ_2^2. Supongamos que ambas muestras son independientes entre sí. Entonces el cociente (4.2) se distribuye como una variable F de Snedecor con (N_1, N_2) grados de libertad.

$$F = \frac{S_1^2/\sigma_1^2}{S_2^2/\sigma_2^2} \tag{4.2}$$

Además, si suponemos la hipótesis de homocedasticidad que debe cumplirse para aplicar ANOVA, tendríamos $\sigma_1 = \sigma_2$, por lo que el cociente anterior quedaría como en (4.3)

$$F = \frac{S_1^2}{S_2^2}. \tag{4.3}$$

Existen diferentes tipos de ANOVA en función de si los datos son independientes, pareados, factores, etc. En particular, en nuestro caso aplicaremos un ANOVA de una vía para datos independientes puesto que nuestros datos no están pareados y el objetivo es analizar si existen diferencias significativas entre las medias de una variable aleatoria contínua en los diferentes niveles de otra variable cualitativa o factor. En este caso, la variable aleatoria contínua es el valor de la función objetivo ponderada, las observaciones son cada una de las instancias evaluadas en cada algoritmo comparado y los niveles de la variable cualitativa son cada uno de los algoritmos estudiados. En realidad, el ANOVA de una vía es una extensión de los $t - test$ independientes para más de dos grupos.

Por una parte, la hipótesis nula es que no existen diferencias entre las medias de los diferentes grupos. Por otra parte, la hipótesis alternativa es que al menos un par de medias son significativamente distintas. En definitiva, la hipótesis que vamos a contrastar es (4.4).

$$\begin{aligned} &H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 \\ &H_1 : \exists i, j : \mu_i \neq \mu_j \end{aligned} \tag{4.4}$$

Una manera alternativa de plantear el hipótesis sería la siguiente. Suponemos que μ es el valor esperado para una observación cualquiera de la población, es decir, la media de todas las observaciones sin tener en cuenta los diferentes niveles. Sea α_i el efecto introducido por el nivel i para $i = 1, 2, 3, 4$. La media del nivel i vendría dada por (4.5)

$$\mu_i = \mu + \alpha_i, \quad i = 1, 2, 3, 4. \tag{4.5}$$

Por tanto, el contraste de hipótesis quedaría como en (4.3.3)

$$\begin{aligned} &H_0 : \alpha_1 = \alpha_2 = \cdots = \alpha_k = 0 \\ &H_1 : \exists i : \alpha_i \neq 0 \end{aligned}$$

En nuestro caso particular, el contraste de hipótesis que vamos a plantear tiene como hipótesis nula que no existen diferencias significativas entre las medias de los valores de RDP de los diferentes Algoritmos. En definitiva, la variable respuesta cuantitativa del ANOVA será el valor del RDP en porcentaje y la variable cualitativa será el Algoritmo, que tiene 4 niveles (Algoritmo NEH, Algoritmo Heurístico 1, Algoritmo Heurístico 2 y Algoritmo GRASP).

En primer lugar, realizamos un estudio inmersivo de los datos para conocer el número de grupos, el número de observaciones por grupo y la distribución de las observaciones. Evidentemente, contamos con 4 grupos, uno por cada algoritmo estudiado y el número de observaciones en cada grupo es el mismo (20) porque así había sido diseñado el experimento.

En la figura 4.4 tenemos el cálculo de la media y la desviación de cada grupo.

Además, como el número de observaciones por grupo es el mismo, diremos que es un modelo equilibrado. Este hecho es relevante cuando se comprueben las hipótesis del modelo.

Por otra parte, consideramos oportuno realizar un Box-Plot para detectar posibles observaciones anómalas, asimetrías significativas o mucha diferencia de varianzas. En la figura 4.5 observamos el Box-Plot según

	Algoritmo	RDP
1	Alg1	16.871483
2	Alg2	8.046923
3	AlgNEH	12.843409
4	GRASP	0.000000

(a) Media

	Algoritmo	RDP
1	Alg1	6.476576
2	Alg2	4.034538
3	AlgNEH	4.607054
4	GRASP	0.000000

(b) Desviación típica

Figura 4.4: Media y desviación típia de cada grupo

algoritmos. A simple vista podemos ver como el tamaño de las cajas es bastante similar para todos niveles excepto para el caso del Algoritmo GRASP. Además, el Algortimo Heurístico 1 se detecta una observación extrema que corrseponde precisamente al caso en el que se tienen 40 trabajos a secuenciar en 5 máquinas, lo cual nos adelanta que puede que el número de trabajos sea significativo.

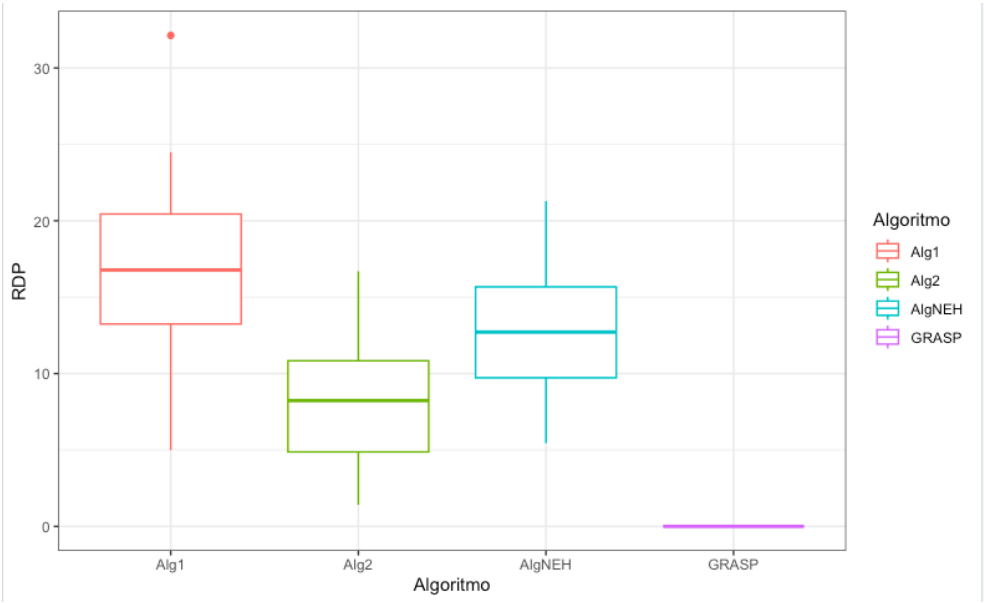


Figura 4.5: Box-Plot de RDP según algoritmo

Antes de realizar cualquier tipo de análisis de la varianza es necesario verificar que se cumplen las condiciones requeridas:

- **Independencia**: Las observaciones deben ser aleatorias, esto es, el tamaño total de la muestra perteneciente a cada grupo del factor debe ser menor al 10 $\%$ y los grupos deben ser independientes entre ellos.

  En nuestro caso se cumple efectivamente la hipótesis de independencia.

- **Normalidad**: La variable cuantitativa debe de distribuirse de forma normal en cada uno de los grupos.

La hipótesis de que la variable RDP se distribuye de forma normal en cada uno de los Algoritmos puede ser comprobada de forma gráfica o mediante test de hipótesis.

En la figura 4.7 vemos claramente como se cumple más o menos la hipótesis de normalidad gracias a la representación gráfica de los *qqplots*, ya que el valor de los cuantiles se sitúa sobre la diagonal.

Notemos que los valores de RDP claramente no siguen una distribución normal en el caso del Algoritmo GRASP, puesto que todos los valores son idénticos iguales a 0. Esto es debido a que el algoritmo GRASP es el que mejor solución ofrece para cualquiera de las instancias consideradas (por eso el porcentaje de mejora respecto el mejor resultado es nulo, porque es el mismo) . Sin embargo, vamos a despreciar este hecho, ya que sabemos que el ANOVA sigue siendo bastante robusto aun cuando existe falta de normalidad.

De igual forma, para corroborar la normalidad en el caso de los otros 3 algoritmos, vamos a realizar los test de hipótesis pertinentes. En particular, como tenemos menos de 50 eventos por grupo, podemos emplear el test de *Shapiro-Wilk*, el cual tiene como hipótesis que la muestra proviene de una población normalmente distribuída.

```
        Shapiro-Wilk normality test

data:  alg1$RDP
W = 0.97937, p-value = 0.9257
```

(a) Algoritmo Heurístico 1

```
        Shapiro-Wilk normality test

data:  alg2$RDP
W = 0.97503, p-value = 0.8554
```

(b) Algoritmo Heurístico 2

```
        Shapiro-Wilk normality test

data:  algneh$RDP
W = 0.9734, p-value = 0.8243
```

(c) Algoritmo NEH

Figura 4.6: Tests de normalidad de *Shapiro-Wilk*

En la figura 4.6 vemos como todos los tests tienen un *p-valor* bastante elevado y mayor que 0.05, por lo que no tenemos suficiente significatividad estadística para rechazar la hipótesis nula del test, es decir, no tenemos evidencia estadística de falta de normalidad.

- **Homocedasticidad**: La varianza dentro de los grupos debe de ser aproximadamente igual en todos ellos. La hipótesis nula del test para verificar la homocedasticidad considera que todas las observaciones proceden de la misma población (misma media y misma varianza).

Para comprobar la homocedasticidad vamos a emplear dos tests basados en la mediana: el test de *Levene* y el test de *Fligner-Killeen*.

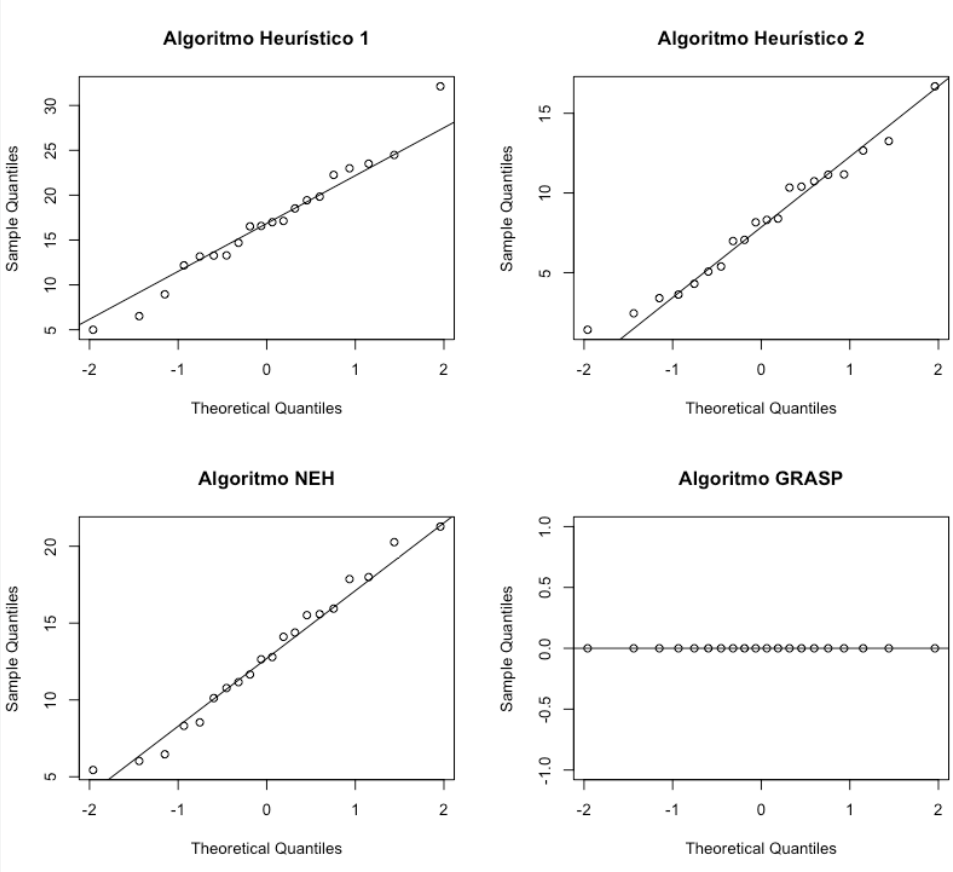


Figura 4.7: Estudio de normalidad gráfico mediante *qqplots* de la variable cuantitativa RDP en cada algoritmo

En las figuras 4.8 y 4.9 vemos el resultado de los tests de *Fligner-Killeen* y *Levene* respectivamente. En ambos casos el *p-value* es muy pequeño, por lo que rechazaríamos la hipótesis nula de homocedasticidad de varianzas. Sin embargo, cabe destacar que el ANOVA es bastante robusto a la falta de homocedasticidad si el diseño es equilibrado, que es nuestro caso, por lo que vamos a seguir con el análisis despreciando este inconveniente.

```
        Fligner-Killeen test of homogeneity of variances

data:  RDP by Algoritmo
Fligner-Killeen:med chi-squared = 33.5, df = 3, p-value = 2.526e-07
```

Figura 4.8: Test de homocedasticidad *Fligner-Killeen*

Una vez comprobadas las hipótesis de independencia, normalidad y homocedasticidad, estamos en condiciones de realizar el análisis ANOVA.

```
Levene's Test for Homogeneity of Variance (center = "median")
      Df F value    Pr(>F)
group  3  12.062 1.532e-06 ***
      76
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figura 4.9: Test de homocedasticidad *Levene*

En la figura 4.10 tenemos el *summary* del ANOVA generado en el que se ha tomado los valores de RDP como variable respuesta y los algoritmos como factores. Como el *p-value* es muy pequeño y menor que 0.05, rechazamos la hipótesis nula de que las medias de los valores de RDP entre los diferentes algoritmos son iguales. En definitiva, tenemos suficiente evidencia estadística para afirmar que existen diferencias significativas entre los valores de RDP, esto es, existe al menos un algoritmo cuyos valores de RDP son significativamente diferentes al resto. Dicho de otra forma, existe al menos un algoritmo que se comporta de manera diferente (en cuanto a calidad de soluciones se refiere) que el resto.

```
                Df Sum Sq Mean Sq F value Pr(>F)
datos$Algoritmo  3   3157  1052.4   52.99 <2e-16 ***
Residuals       76   1510    19.9
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figura 4.10: Resultados del modelo ANOVA

Aunque ya habíamos realizado el estudio de las hipótesis del modelo antes del cálculo del ANOVA (si no se cumplen no tiene sentido continuar), la forma más adecuada es analizar los residuos del modelo una vez generado el ANOVA. La representación gráfica de los residuos de la figura 4.11 muestra una ligera falta de homocedasticidad en el gráfico 1, la cual ya habíamos comentado que no era importante en este caso al tratarse de un conjunto de datos equilibrado. También vemos como los residuos se distribuyen muy cercanos a la línea de la normal (gráfico 2: Normal Q-Q).

En conclusión, hemos demostrado estadísticamente que existen diferencias significativas entre el valor de RDP entre los algoritmos y, por tanto, hay algoritmos que se comportan mejor que otros. Por este motivo, procedemos a realizar comparaciones múltiples entre ellos para detectar entre cuales algoritmos existen más diferencias. En particular, vamos a emplear Corrección de *Holm* y *Tukey HSD*.

Por una parte, el ajuste de Holm-Bonferroni nos permite realizar un *t-test* para todas las comparaciones y ordenarlas de menor a mayor *p-value*. El nivel de significancia para la primera comparación (menor *p-value*) se corrige diviendo α entre el número total de comparaciones. Si no resulta significativo se detiene el proceso, pero si sí que lo es se corrige el nivel de significancia de la siguiente comparación (segundo con menor *p-value*). El proceso se repite hasta detenerse cuando la comparación ya no sea significativa. En la figura 4.12 vemos el resultado de los *p-values* tras haber aplicado la corrección de *Holm*.

Por otra parte, el test HSD (*Honestly-significant-difference*) de *Tukey* es muy similiar a un *t-test*, excepto que corrige el *experiment wise error rate* mediante un estadístico que sigue una distribución llamada *studentized*

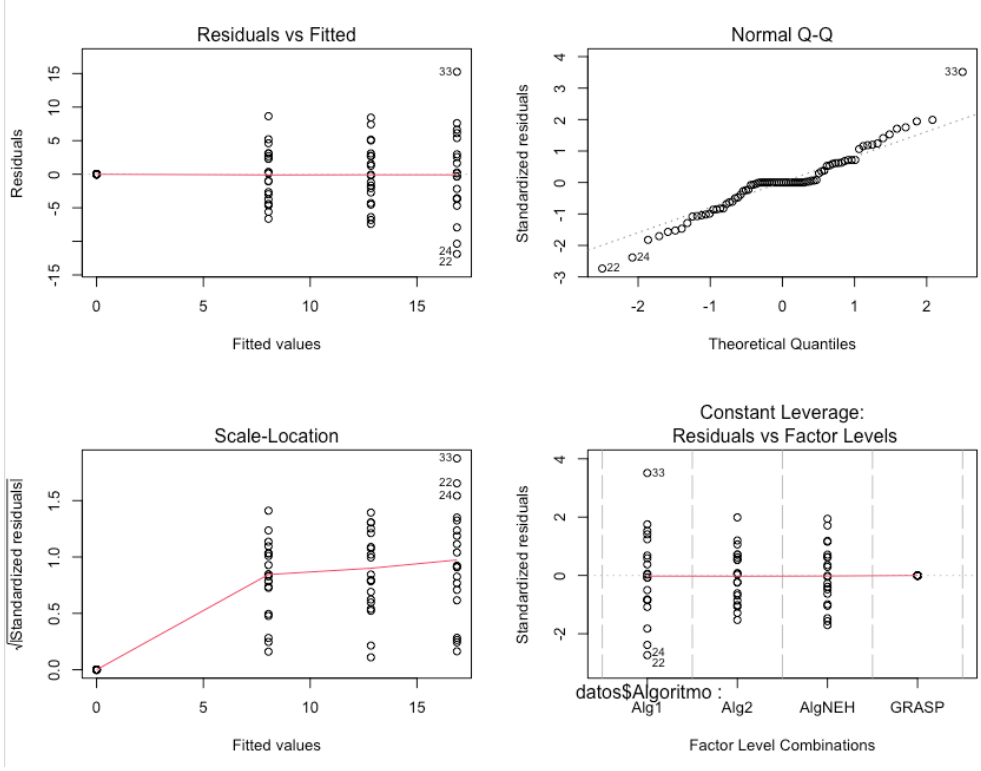


Figura 4.11: Representación gráfica de los residuos de ANOVA

```
        Pairwise comparisons using t tests with pooled SD

data:  datos$RDP and datos$Algoritmo

       Alg1    Alg2    AlgNEH
Alg2   8.3e-08 -       -
AlgNEH 0.0055  0.0021  -
GRASP  < 2e-16 6.2e-07 4.1e-13

P value adjustment method: holm
```

Figura 4.12: Matriz de resultados tras aplicar la corrección de *Holm*

range distribution y que se calcula como sigue:

$$q_{calculado} = \frac{\bar{x}_{max} - \bar{x}_{min}}{S\sqrt{2/n}},$$

donde $\bar{x}_{max}$ es la mayor de las medias de los dos grupos comparados, $\bar{x}_{min}$ la menor, S la pooled SD de ambos grupos y n el número total de observaciones en los dos grupos. Para cada par de grupos obtenemos el valor de

$q_{calculado}$ y se compara con el esperado acorde a una *studentized range distribution* con los correspondientes grados de libertad. Si la probabilidad es menor al nivel α establecido, se considera significativa la diferencia de medias. Además, es posible calcular intervalos HSD para estudiar el solapamiento entre grupos.

A continuación, veamos de qué manera aplicamos el test *Tukey* para comparar 2 a 2 los algoritmos. En la figura 4.13 vemos la tabla resultado de aplicar en RStudio la función predeterminada. La segunda y tercera columan nos indican los extremos inferior y superior respectivamente de los intervalos HSD; mientras que la última columna nos da el $p - value$ asociado al estadístico definido. Notemos que todos los *p-values* son menores que 0.05, por lo que a un nivel de significatividad del 5 % existe una diferencia significativa de medias entre todos los algoritmos. La única comparación que está cerca de no serlo sería entre el Algoritmo NEH y el algoritmo 1. De hecho, su intervalo HSD está muy cercano a incluir el 0.

Los intervalos HSD de todas las comparaciones 2 a 2 pueden ser analizados en detalle en la figura 4.14. En ella vemos como ninguno de los intervalos de diferencia de medias contienen el 0, lo cual muestra que existen diferencias significativas entre los pares de medias del valor RDP. El intervalo más cercano al 0 es el correspondiente al Algoritmo NEH y Algoritmo 1, que ya habíamos comentado que tenía el mayor *p-value* asociado. Otro fenómeno a destacar es que los intervalos que muestran las comparaciones de GRASP con el resto de algoritmos son precisamente los más alejados del 0 en sentido negativo, lo cual nos indica en este caso que los valores de RDP son significativamente menores para el GRASP que para el resto de algoritmos y que esta diferencia es mayor que en el caso del resto de comparaciones 2 a 2. Esto a su vez implica que la diferencia relativa del algoritmo GRASP respecto al algoritmo con mejor resultado en cada caso (definición del concepto de RDP) es la menor. De hecho, el valor de RDP en el caso de GRASP era 0 en todas las instancias, lo cual implica que el algoritmo GRASP es el que mejor rendimiento tiene y mejores soluciones proporciona para cualquier magnitud de las instancias. En conclusión, una vez se ha demostrado que existen diferencias significativas entre las soluciones obtenidas por cada algoritmo, estamos en condiciones de afirmar que el algoritmo GRASP, tomando $\alpha = 0$ y $n_{iter} = 50$, es el que mejor desempeño nos ofrece para una gran variedad de instancias de diversos tamaños.

En el anexo F encontramos el código de R utilizado para obtener los análisis y gráficos que hemos mostrado a lo largo del capítulo.

```
$`datos$Algoritmo`
                  diff        lwr        upr     p adj
Alg2-Alg1     -8.824560 -12.526584  -5.122536 0.0000001
AlgNEH-Alg1   -4.028073  -7.730098  -0.326049 0.0275736
GRASP-Alg1   -16.871483 -20.573507 -13.169458 0.0000000
AlgNEH-Alg2    4.796487   1.094462   8.498511 0.0057540
GRASP-Alg2    -8.046923 -11.748947  -4.344898 0.0000012
GRASP-AlgNEH -12.843409 -16.545434  -9.141385 0.0000000
```

Figura 4.13: Comparaciones múltiples de medias *Tukey* al 95 % de nivel de confianza

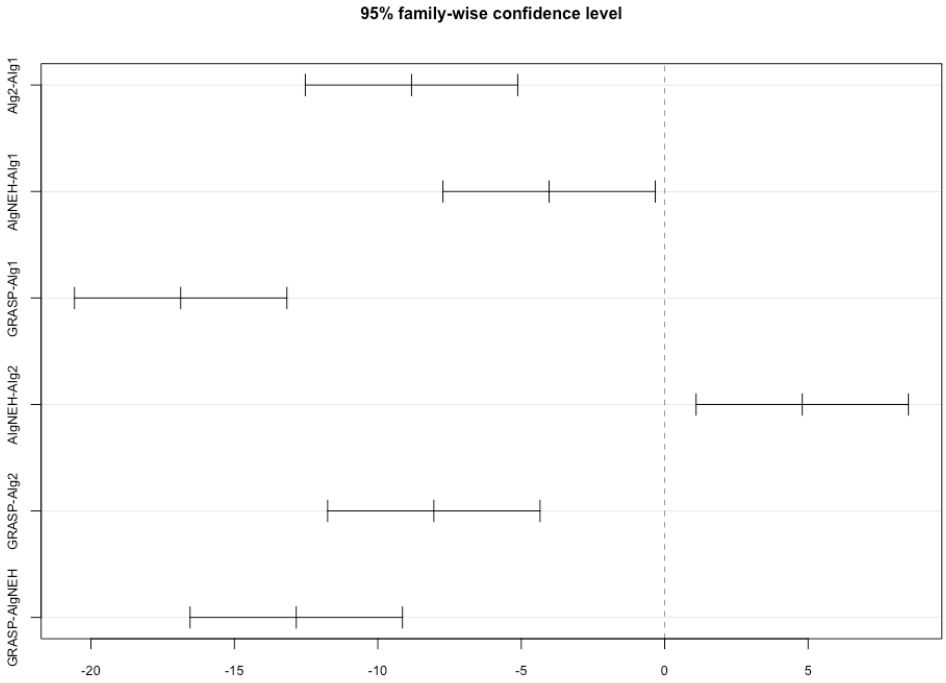


Figura 4.14: Gráfico de diferencias de medias al 95 % de nivel de confianza

Capítulo 5

Caso de estudio

En el presente capítulo abordaremos la modelización y optimización de una línea de producción real; en particular, una línea de soldadura de la planta de carrocerías de la Factoría de FORD Valencia situada en el polígono industrial Juan Carlos I de Almussafes. Para ello, a partir de datos reales de producción y con la ayuda de los algoritmos heurísticos propuestos y sus mejoras, obtendremos una serie de soluciones de secuenciación factibles. Finalmente, concluiremos qué algoritmo nos proporciona mejores soluciones en función de las necesidades productivas, del estatus de la línea en cada momento y, sobretodo, de cuáles serían los intereses prioritarios de la empresa.

5.1. Escenario industrial del estudio

A continuación, procedemos a describir el escenario industrial que vamos a modelizar y optimizar, contextualizando así el problema *Flow Shop* formulado y la respectiva obtención de los tiempos de proceso.

5.1.1. Línea de carrocerías y montaje

Sabemos que cuando hablamos de línea de producción nos referimos al conjunto de operaciones secuenciales establecidas en una factoría a través de las cuáles se obtiene un producto final. La pieza producto se desplaza a través de las estaciones y puestos de la línea, donde se le aplican las diferentes tareas de ensamblado, posicionamiento, soldadura, etc. En particular, una línea de carrocerías y montaje consiste en la configuración de estaciones de trabajo instaladas en línea o en serie a lo largo de una cinta transportadora o un transportador mecánico. De esta manera, las piezas a procesar son soldadas y acopladas de estación a estación. Además, en función de la tarea asignada a cada estación o de las funciones que se lleven a cabo, serán diferentes los elementos a intervenir en cada una de ellas. Por ejemplo, podemos encontrar estaciones configuradas únicamente por robots, totalmente automatizadas. En cambio, podemos tener estaciones configuradas para que únicamente trabajen operarios. Otra opción es la configuración mixta, donde parte de las tareas están automatizadas llevadas a cabo por robots pero es necesario el trabajo de operarios que faciliten el funcionamiento de las máquinas, como por ejemplo mediante la facilitación de piezas y recarga. Cabe destacar que los puestos de carga de operarios están cada vez más siendo sustituidos por robots o manipuladores.

La factoría de FORD Almussafes está compuesta principalmente por 3 plantas que siguen el flujo de

fabricación lógico del coche. En primer lugar, la planta de Prensas y Carrocerías (*Body Plant*) es la responsable de dar forma al esqueleto del vehículo. Despúes de que las prensas den la forma pertinente a cada una de las piezas o partes de la carrocería, a lo largo de la línea de carrocerías se realiza el posicionado y ensamblado mediante soldadura de las diferentes piezas metálicas que dan forma al coche. A continuación, la carrocería desnuda es llevada por unos túneles a la planta de pinturas, donde se le aplican una serie de procesos químicos como los baños de cataforesis y el posterior secado mediante hornos a altas temperaturas. Finalmente, en la planta de Montaje (*Assembly Plant*) los operarios se encargan de recubrir el interior del coche; montar los salpicaderos, los asientos o el maletero; incluir la configuración eléctrica, el motor, la batería y todos los complementos necesarios para el funcionamiento del coche. Posteriormente, al final de la propia línea de montaje se lleva a cabo el maridaje de la parte superior con el suelo del vehículo y se obtiene el producto final.

A partir de este momento vamos a centrarnos únicamente en la planta de carrocerías o *Body Plant*, es decir, el primer eslabón de la cadena de producción. Cabe destacar que esta planta integra tres plantas de grandes dimensiones, aunque actualmente únicamente se encuentra en funcionamiento una de ellas después de que algunos de sus modelos fueran retirados de la producción. Se trata de la planta más automatizada de la factoría, debido a que la mayoría de estaciones de trabajo están compuestas únicamente por celdas de robots encargados de realizar las tareas de soldadura o ensamblado y el papel de los operarios se reduce a la recarga de piezas, reparaciones y procedimientos de mantenimiento.

5.1.2. Estación de soldadura. Línea 8XY FORD Almussafes

La línea de soldadura en la que vamos a centrar nuestro estudio forma parte de la planta de carrocerías de FORD Almussafes. Se trata de una línea de soldadura compuesta por estaciones de trabajo donde se realizan los diferentes procesos de soldadura con robots trabajando en paralelo.

Figura 5.1: Linea de soldadura objeto de estudio, situada en la Factoría Ford de Almussafes (Valencia)

La línea de soldadura objeto de estudio se compone de 8 estaciones de trabajo independientes entre sí, donde cada una tiene unidades de soldadura trabajando en paralelo y a veces en serie. En particular, 3 de las estaciones de trabajo cuentan con 6 robots de soldadura, 4 tienen 4 robotos de soldadura y 1 tiene 1 único robot de soldadura. Además, la línea está organizada en 4 zonas diferenciadas, de manera que las dos primeras estaciones de trabajo forman parte de la Zona A, la tercera estación de trabajo compone la Zona B, la Zona C comprende las estaciones 4 y 5 y en la Zona D encontramos las estaciones 6, 7 y 8. Además, cada robot puede

hacer hasta 19 puntos de soldadura en una carrocería.

En la Figura 5.2 podemos observar el *layout* de la línea 8XY de la planta de carrocerías con sus respectivas zonas y estaciones de trabajo. Así mismo, en la Figura 5.3 vemos la distribución de las estaciones por zonas y la disposición de los robots en cada estación de trabajo (*Workstation*).

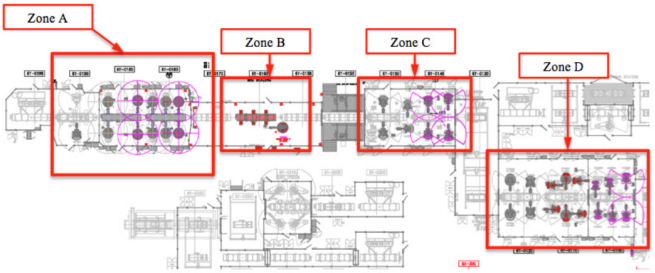
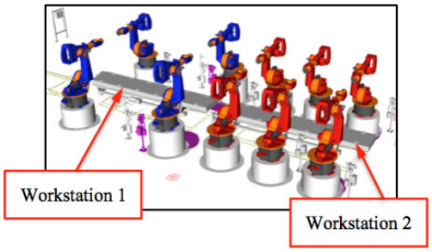
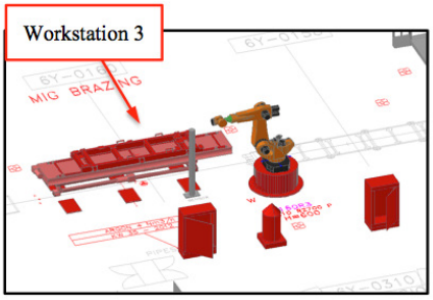


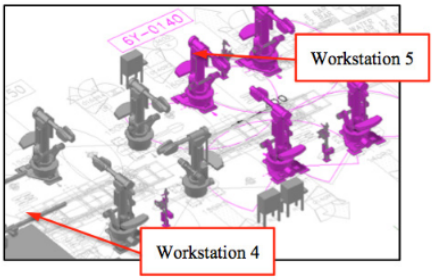
Figura 5.2: *Layout* de la línea de soldadura 8XY de la planta de carrocerías de FORD Almussafes



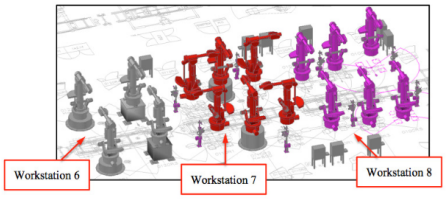
(a) Zona A



(b) Zona B



(c) Zona C



(d) Zona D

Figura 5.3: Zonas y estaciones de trabajo de la línea de soldadura 8XY

Por último, describimos el comportamiento genérico de una estación de soldadura. En primer lugar, cuando la pieza llega hasta la estación de trabajo correspondiente, el brazo del robot mueve la pinza de soldadura hasta el punto a soldar. A continuación, un cilindro neumático es el encargado de mover la pinza de soldadura.

Su movimiento se divide en 2 fases diferenciadas. En un primer momento, aproxima la pinza al punto objetivo para posteriormente aplicar la presión necesaria para soldar.

5.1.3. Sistema multimodelo

Una complejidad añadida a las líneas de producción es la variedad de productos a procesar. Las líneas de producción multimodelo suponen una complejidad añadida a las líneas de producción en el ámbito industrial. Estas líneas están diseñadas para fabricar múltiples variantes de productos en un solo flujo de producción, lo que requiere una gestión y coordinación precisa para garantizar un rendimiento eficiente.

Uno de los desafíos principales radica en la gestión de las distintas configuraciones y requisitos de cada variante. Cada producto puede tener características únicas que deben ser tenidas en cuenta durante el proceso de fabricación, como diferentes especificaciones de soldadura, puntos de unión y otros detalles específicos. Esto implica una planificación cuidadosa para cumplir con todos los requerimientos.

En resumen, las líneas de producción multimodelo presentan una complejidad significativa debido a las múltiples variantes de productos y configuraciones que deben ser fabricadas. La gestión de los requisitos específicos de cada variante, la programación y secuenciación de las operaciones de soldadura, la gestión del flujo de materiales y la capacitación del personal son algunos de los desafíos clave que deben abordarse para lograr un rendimiento eficiente y de calidad en este tipo de líneas de producción.

En particular, en la línea de soldadura objeto de estudio, 68 variantes diferentes atraviesan el proceso. Estas variantes representan productos o configuraciones distintas que requieren operaciones de soldadura en diversos puntos a lo largo de la línea. Cada variante puede tener requisitos o especificaciones específicas para el proceso de soldadura, como el número de puntos de soldadura necesarios. Esto implica, como veremos más adelante, que el tiempo de proceso en una estación será variable en función de la variante.

5.1.4. Tiempos de espera y bloqueo

Los tiempos de espera y bloqueo en las líneas de producción son problemáticas que pueden afectar negativamente la eficiencia y productividad de un proceso de fabricación.

Una estación de trabajo puede encontrarse en tres posibles estados: *working*, *starving* y *blocked*. De esta manera, cuando un trabajo en una determinada estación está en estado de *working* y termina comprueba el estado de la siguiente estación y, en función de ello, decide si proseguir su ciclo en la línea o esperar. En este caso, si la siguiente estación se encuentra en estado de *working*, la estación actual (que ya había terminado su trabajo) pasará a estado de *blocked*, puesto que no puede dejar pasar el trabajo a la próxima estación. La situación análoga la encontraríamos cuando una estación se encuentra libre para recibir una nueva pieza a procesar y comprueba la estación anterior. Si la estación anterior se encuenta en estado de *working*, la estación actual cambia a estado de *starving* hasta que la pieza acaba de ser procesada en la anterior y puede proseguir su flujo. Además, para simplificar la casuística, a partir de este momento vamos a suponer que todas las piezas se encuentran disponibles al inicio de la primera estación y que los trabajos terminados en la última estación pasan inmediatamente a una supuesta próxima línea. De esta manera, ni la primera estación tendrá asociado

el estado de *starving* ni la última el de *blocked*. En la Figura 5.4 podemos observar un pequeño esquema que resume los 3 posibles estados de una máquina y de qué manera se establece el flujo entre ellos.

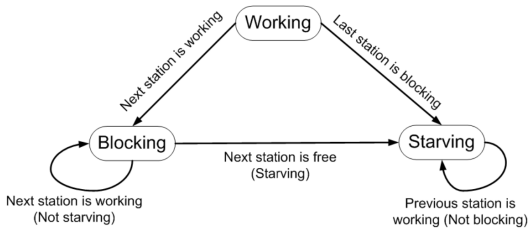


Figura 5.4: Posibles estados de una estación de trabajo

A continuación, veamos las principales problemáticas asociadas a la existencia de tiempos de espera y de bloqueo en una línea de producción:

- **Ineficiencia operativa**: Los tiempos de espera y bloqueo generan interrupciones en el flujo de trabajo, lo que resulta en una menor eficiencia operativa. En definitiva, se ralentiza la producción y se desperdician recursos.
- **Incremento de costos**: Los recursos, como la mano de obra y los equipos, pueden estar inactivos durante los períodos de espera, lo que implica un gasto de mantenimiento de la maquinaria sin una producción real.
- **Incumplimiento de plazos**: Resulta evidente la relación entre la existencia de tiempos de espera y de bloqueo y el aumento del tiempo de completación del producto final. De esta manera, la existencia de estos tiempos ”muertos”pueden llegar a provocar retrasos en la entrega del producto, lo cual influye negativamente en la satisfacción del cliente y la reputación de la empresa.
- **Impacto en la cadena de suministro**: Teniendo en cuenta el hecho de que la secuenciación de trabajos a través de la línea va ligada a la secuenciación de piezas asociadas, los tiempos de espera y bloqueo pueden provocar efecto dominó en la cadena de suministro. Si una etapa de la producción se retrasa o bloquea, puede afectar a las etapas siguientes, generando un desequilibrio que puede provocar un descuadre de recursos.

Para abordar estas problemáticas, es fundamental implementar estrategias eficientes de planificación y programación de la producción, así como gestionar adecuadamente la logística de los recursos. La identificación temprana de cuellos de botella y la adopción de medidas correctivas o alternativas pueden ayudar a minimizar los tiempos de espera y bloqueo, mejorando la eficiencia y el rendimiento general de las líneas de producción.

5.1.5. Impotancia de la secuenciación

La secuenciación desempeña un papel crucial en las líneas de producción multimodelo y tiene una gran importancia en varios aspectos:

- **Eficiencia operativa**: Una secuenciación adecuada permite optimizar el flujo de producción, minimizando los tiempos de configuración y cambio entre variantes de productos. Al programar las operaciones de soldadura de manera eficiente, se evitan cuellos de botella y se maximiza la productividad de la línea de producción.

- **Reducción de tiempos de espera**: Al secuenciar las operaciones de soldadura de manera inteligente, se minimiza el tiempo de espera entre las diferentes etapas del proceso. Esto ayuda a mantener un flujo continuo de trabajo y reduce los tiempos de inactividad, lo que a su vez aumenta la eficiencia global de la línea. **Gestión de recursos**: La secuenciación adecuada permite una gestión eficiente de los recursos disponibles, como materiales, equipos y mano de obra. Al asignar y programar los recursos de acuerdo con la secuencia de producción, se evita el desperdicio y se garantiza un uso óptimo de los recursos en función de las necesidades de cada variante de producto.

- **Calidad y cumplimiento de especificaciones**: La secuenciación en las líneas multimodelo también juega un papel clave en la garantía de la calidad y el cumplimiento de las especificaciones de cada variante de producto. Al programar las operaciones de soldadura en el orden correcto y siguiendo las secuencias adecuadas, se asegura que los productos se fabriquen según los estándares requeridos, evitando errores y defectos.

- **Flexibilidad y adaptabilidad**: La secuenciación permite una mayor flexibilidad y adaptabilidad en las líneas multimodelo. A medida que las demandas del mercado cambian y se introducen nuevas variantes de productos, la secuenciación eficiente permite una rápida adaptación y reconfiguración de la línea para satisfacer las necesidades cambiantes.

En resumen, la secuenciación desempeña un papel vital en las líneas de producción multimodelo, ya que contribuye a la eficiencia operativa, la reducción de tiempos de espera, la gestión de recursos, la calidad del producto y la adaptabilidad. Una secuenciación adecuada permite optimizar el rendimiento general de la línea, garantizando una producción eficiente y de alta calidad de las múltiples variantes de productos.

La programación y secuenciación de las operaciones de soldadura también es una tarea compleja en las líneas de producción multimodelo. Como ya hemos comentado, cada variante puede requerir diferentes acciones de soldadura, requisitos o especificaciones específicas, lo que implica una planificación minuciosa para garantizar la eficiencia y calidad del proceso y evitar la existencia de tiempos de espera y de bloqueo.

5.2. Minitérminos y tiempos de ciclo

Los minitérminos son subtiempos de un ciclo técnico, es decir, un minitérmino es el tiempo que tarda cualquier componente en realizar su tarea. En definitiva, por definición, un minitérmino es un parámetro de producción que nos indica el tiempo de un subciclo. Además, sólo tendría sentido utilizar este término en relación con la mejora de la producción. [8]

De alguna manera lo que se consigue con los minitérminos es dividir un proceso de producción amplio en unidades o etapas de trabajo más pequeñas, de manera que podamos tener una monitorización temporal más detallada de cada tarea realizada sobre el producto final. Esto permite además facilitar la planificación y el control de la línea, así como la eficiencia en el flujo de trabajo y la detección de anomalías para anticiparse a

fallos en la línea.

¿Qué relación tienen los minitérminos con el problema de secuenciación que estamos abordando? Recordemos que un dato imprescindible en la modelización del problema como un Problema de Secuenciación *Flow Shop* son los tiempos de proceso de cada trabajo en cada máquina. En nuestro caso, ya veremos más adelante con más detalle, que los trabajos son las piezas de cada uno de los modelos de coches a procesar y las máquinas son cada una de las estaciones de trabajo. Sin embargo, para obtener esta información sobre el tiempo que tarda cada tarea en ser realizada en cada estación necesitamos disponer de los subtiempos de cada componente de la estación.

Recordemos el comportamiento de una estación de soldadura. En primer lugar, el brazo del robot mueve la pinza de soldadura hasta el punto correspondiente de la pieza. Posteriormente, un cilindro neumático mueve la pinza de soldadura hasta colocarla en el punto objetivo. Finalmente, se aplica la presión necesaria. De esta manera, cada unidad de soldadura queda dividida en 3 mini-términos. El primer mini-término hace referencia al movimiento del brazo del robot. El segundo mini-término mide el tiempo necesario para realizar el movimiento de soldadura. Finalmente, el tercer mini-término recoge el tiempo de la propia acción de soldadura. Esta presión ejercida por la pinza de soldadura está controlada por un sistema de control. De esta manera, además del tiempo necesario por estos dispositivos para realizar su tarea correspondiente, se debe tener en cuenta como añadido el tiempo necesario por las componentes asociadas para realizar sus propias tareas. En la Figura 5.5 se muestra la configuración experimental para medir el tiempo de ciclo de cada mini-término en la estación de soldadura, donde el PLC y el PC son los encargados de medir dicho tiempo.

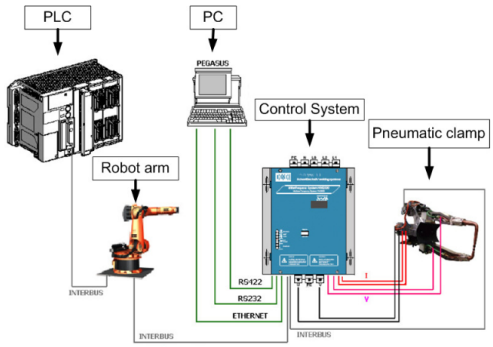


Figura 5.5: Configuración experimental de la estación de trabajo

De la misma forma, es importante definir bien el concepto de tiempo de ciclo, puesto que corresponde la base de nuestro estudio. Este concepto se utiliza para hacer referencia al tiempo necesario para que cada estación de trabajo, equipo o elemento realice una determinada tarea.

Mediante la suma ponderada de los mini-términos de cada uno de los puntos de soldadura realizados en cada estación para cada modelo, obtendríamos el tiempo de ciclo acumulado, es decir, el tiempo de proceso de ese determinado trabajo en dicha estación de trabajo.

Por último, consideramos interesante también de qué manera se mide la capacidad de la línea. Las unidades empleadas para ello son JPH (*Job Per Hour* y es la manera corporativa de cuantificar la producción y medir la eficiencia de la línea. Por otra parte, definimos ERC (*Engineering Running Capacity*) como la capacidad máxima de la línea. En nuestro caso, el ERC de la línea de soldadura objeto de estudio es de 60 JPH. Otro concepto importante es el GRR (*Get Ready Requirement*), también medido en JPH que nos indica los requisitos de mercado, es decir los pedidos de los clientes. Por último, definimos el concepto de ERR (*Engineering Running Rate*), un medible de capacidad dependiente de la disponibilidad y que está definido en 51 JPH. En definitiva, el objetivo ideal sería alcanzar una producción cercana a ERC. Sin embargo, en la práctica es extremadamente difícil lograr este valor máximo, por lo que se define esa nueva tasa de producción más realista: Engineering Running Rate (ERR). Estos conceptos se encuentran desarrollados con más detalle en [7].

5.2.1. Simulación de tiempos de ciclo

Los datos a partir de los cuáles vamos a trabajar se encuentran disponibles en [7].

Por una parte, en el Anexo 2 de [7] se muestra el modelado de la línea de soldadura, donde las filas muestran los minitérminos para cada robot en cada estación, el movimiento del robot, la soldadura de movimiento y la tarea de soldadura respectivamente, así como el desplazamiento. De esta manera, el movimiento del robot nos indica el tiempo necesario para que el robot realice el movimiento; el movimiento de soldadura hace referencia al número de puntos a realizar en cada caso; mientras que el offset recoge los segundos que debe esperar a otra unidad el robot de soldadura para realizar su trabajo.

Por otra parte, en el Anexo 4 de [7] disponemos de una tabla donde se recogen para cada uno de los 68 modelos en cada una de las estaciones de trabajo el valor temporal de los 3 mini-términos que configuran cada unidad de soldadura, así como el tiempo de off-set.

Como podemos observar, los puntos de soldadura a realizar, los tiempos de movimiento de robot y el tiempo de off-set varían en función del modelo fabricado. Este hecho justifica la influencia de la secuenciación de los trabajos en la eficiencia de la línea.

A partir de toda esta información, es posible obtener una estimación del Tiempo de Ciclo de cada modelo en cada estación de trabajo. Para ello, se realiza un sencillo cálculo que nos permite obtener el tiempo de ciclo global en cada estación según el modelo.
Sea p_{ij} el tiempo de ciclo acumulado del modelo i en la máquina j, tenemos que:

$$p_{ij} = t_{rob_{ij}} + t_{pinz_{ij}} + t_{sold_{ij}} \cdot n_{sold_{ij}}, \ \ \forall i = 1, \ldots, 68, \, j = 1, \ldots, 8,$$

donde $t_{rob_{ij}}$ es el tiempo total correspondiente a los movimientos de todos los robots de la estación j cuando procesa el modelo i; $t_{pinz_{ij}}$ es el tiempo total correspondiente al movimiento de las pinzas de los robots de la estación j cuando procesa el modelo i; $t_{sold_{ij}}$ es el tiempo medio de soldadura de 1 punto del modelo i en la estación j y $n_{sold_{ij}}$ es el número de puntos del modelo i que son soldados en la estación j.

Por tanto, esta suma, p_{ij}, nos ofrece información acerca del tiempo de ciclo de cada modelo en cada estación, es decir, el tiempo que tarda en procesarse cada trabajo en cada máquina, que es precisamente lo que necesitamos para modelar y resolver el problema como un *Flow Shop*.

5.3. Formulación matemática del problema

En nuestro caso particular, tomaremos los trabajos como modelos de coches a procesar por la línea de producción de carrocerías de la Factoría Ford de Almussafes (Valencia). De manera similar, tomaremos como máquinas cada una de las estaciones de trabajo. Un planteamiento más cercano a la realidad sería considerar como máquina cada uno de los robots de la línea. Sin embargo, esta visión no es posible desde el enfoque del problema que hemos decidido tomar, ya que los robots funcionan simultáneamente en paralelo sobre un mismo coche y se estaría incumpliendo la hipótesis del problema de Flow Shop que nos exige que cada trabajo (coche) únicamente puede ser procesado por una máquina (robot) al mismo tiempo.

Como ya hemos comentado, se trata de una línea multimodelo, donde son procesados 68 variantes de coche diferentes. Así mismo, cuenta con 8 estaciones de trabajo colocadas en serie. De esta manera, siguiendo la notación definida para el problema *Flow Shop*, consideramos $n = 68$ trabajos y $m = 8$ máquinas.

- Definimos $N = \{1, \ldots, 68\}$ como el conjunto de $n = 68$ coches (trabajos) que van a ser procesados por la línea de soldadura y cuyo orden quiere ser determinado.

- Definimos $M = \{1, \ldots, 8\}$ como el conjunto de estaciones de trabajo (máquinas) de la línea de soldadura 8XY.

A partir de este momento nos referiremos a coches y trabajos de manera indiferente. De forma análoga, cuando hablemos de estaciones de trabajo estaremos hablando de máquinas y viceversa.

En primer lugar, debemos asegurarnos de que disponemos de toda la información de los posibles 68 modelos de coche que pueden ser procesados por la línea de soldadura considerada y recopilarla de manera que siga la estructura exigida por las hipótesis del problema del Flow Shop.

En definitiva, hasta el momento tenemos una matriz de dimensión 8×68 de tiempos de ciclo acumulados de cada una de los 68 modelos en cada una de las 8 estaciones de trabajo. La información disponible hasta el momento la encontramos recogida en la tabla 5.1.

	Modelo 1	**Modelo 2**	$\cdots$	**Modelo 68**
WS 1	$p_{1,1}$	$p_{1,2}$	$\cdots$	$p_{1,68}$
WS 2	$p_{2,1}$	$p_{2,2}$	$\cdots$	$p_{2,68}$
WS 3	$p_{3,1}$	$p_{3,2}$	$\cdots$	$p_{3,68}$
WS 4	$p_{4,1}$	$p_{4,2}$	$\cdots$	$p_{4,68}$
WS 5	$p_{5,1}$	$p_{5,2}$	$\cdots$	$p_{5,68}$
WS 6	$p_{6,1}$	$p_{6,2}$	$\cdots$	$p_{6,68}$
WS 7	$p_{7,1}$	$p_{7,2}$	$\cdots$	$p_{7,68}$
WS 8	$p_{8,1}$	$p_{8,2}$	$\cdots$	$p_{8,68}$

Cuadro 5.1: Tabla de tiempos de ciclo acumulado de cada modelo en cada estación de trabajo de la línea de soldadura 8-XY

Notemos que esta tabla es en realidad la matriz de tiempos de proceso $\mathbb{P}$ que habíamos definido en la formulación del problema.

Ya tenemos definido cuál sería el conjunto de trabajos y de máquinas, así como la matriz de tiempos de proceso de cada trabajo en cada máquina.

Otro concepto comentado cuando definíamos el problema de *Flow Shop* era el del tiempo de *setup*. Sabemos que el tiempo de setup nos indican las unidades de tiempo necesarias para reajustar la máquina entre un trabajo y otro consecutivos sin que se procese ninguno entre medias. Sin embargo, notemos que en nuestro caso particular, el ajuste de la maquinaria de las estaciones de trabajo entre la soldadura de un coche y otro es un ajuste de software, por lo que pueden ser milisegundos y lo consideraremos despreciable en nuestro problema. Sin embargo, como ya hemos comentado anteriormente lo que se trata de conseguir es que el tiempo de espera y el tiempo de bloqueo de las estaciones de trabajo cuando la anterior o la siguiente siguen procesando trabajo respectivamente y la actual queda libre sea mínimo.

Por todo ello, en este caso las funciones objetivo a minimizar serán el tiempo de completación del último trabajo *makespan*, el tiempo de inactividad total por espera o bloqueo de las máquinas *idletime* y el tiempo de flujo total *Total Flow time*.
Sea π una solución de secuenciación factible. Sea $C_i(\pi)$ el tiempo de completación del trabajo i en la solución π para $i = 1, \ldots, 68$; $F_i(\pi)$ el tiempo de flujo del trabajo i en la solución π para $i = 1, \ldots, 68$; $W_i(\pi)$ el tiempo de espera total del trabajo i en la solución π para $i = 1, \ldots, 68$; r_i el tiempo de indisponibilidad del trabajo i para $i = 1, \ldots, 68$. Veamos como quedarían definidos formalmente cada uno de estos medibles en este caso particular.

- **Tiempo de completación del trabajo** i

$$C_i(\pi) = r_i + p_i + W_i(\pi), \quad \forall i = 1, \ldots, 68.$$

- ***Makespan***

$$C_{max} = \max_{i=1,\ldots,n} C_i,$$

- **Tiempo de flujo del trabajo** i

$$F_i(\pi) = p_i + W_i(\pi) = C_i(\pi) - r_i, \quad \forall i = 1, \ldots, 68$$

- **Tiempo total de flujo (*Total Flow Time*)**

$$F(\pi) = \sum_{i=1}^{68} F_i(\pi)$$

- **Tiempo de inactividad (*Idle time*) de la máquina** j

$$I_j = \max_{i=1,\ldots,68} C_{ij}(\pi) - \sum_{i=1,\ldots,68} p_{ij}, \quad j = 1, \ldots, 8$$

- **Tiempo total de inactividad de las máquinas**

$$I = \sum_{j=1}^{8} I_j$$

5.4. Resultados numéricos

En esta sección se presentan los resultados numéricos obtenidos tras aplicar los algoritmos heurísticos propuestos al caso de estudio planteado. Para ello, se tomará la matriz de tiempos de ciclo acumulado de cada modelo en cada estación que podemos consultar con más detalle en el Anexo K.

Aunque en el capítulo anterior se había concluído que el algoritmo GRASP era el que mejor resultados proporcionaba para una amplia variedad de instancias, se ha decidido obtener la secuenciación con los 4 algortimos propuestos para contrastar en este caso particular las diversas soluciones factibles que se presentan.

En la tabla 5.2 obtenemos un resumen detallado del valor de los objetivos estudiados y la función objetivo ponderada para cada algoritmo. Como era de esperar según el estudio realizado, el algoritmo que nos da el valor inferior de la función objetivo es el Algoritmo GRASP, mientras que la calidad de la solución ofrecida por el Algoritmo Heurístico 1 es considerablemente diferente a la del resto. Algo parecido pasa si desglosamos los medibles. En definitiva, se respeta la jerarquía de algoritmos que habíamos concluído en las pruebas experimentales.

	Función objetivo	***Makespan***	***Flow Time***	***Idle Time***
Algoritmo NEH	798014.5	79145.7	2428595	125925.9
Algoritmo Heurístico 1	952274.1	79023.43	2943159	125723
Algoritmo Heurístico 2	791051.4	78668.39	2408018	123929.3
Algoritmo GRASP	782555.9	78330.9	2381453	122625.6

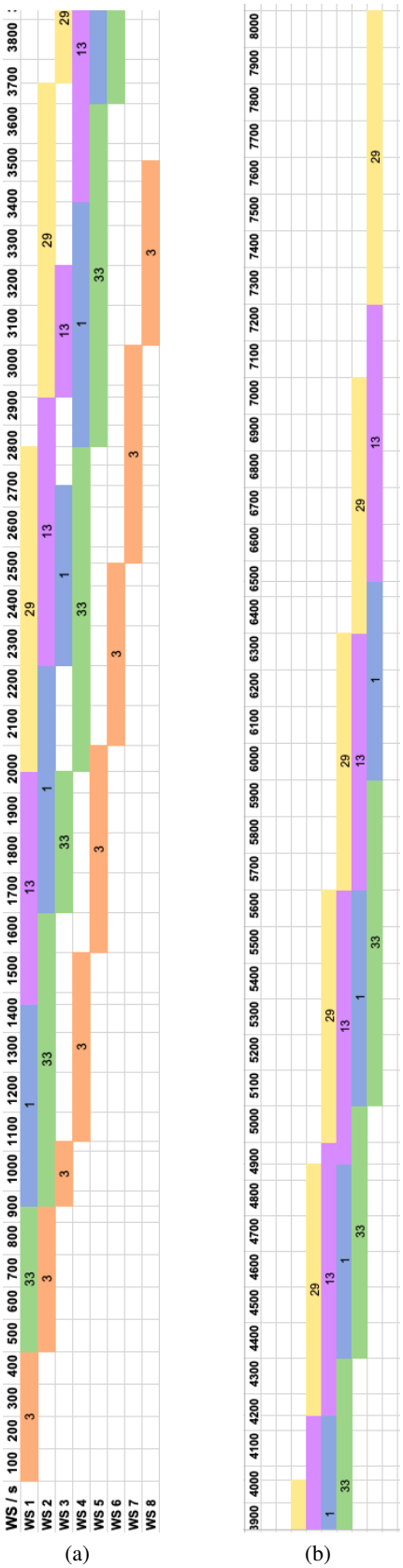
Cuadro 5.2: Valor (en segundos) de las funciones objetivo estudiadas obtenidas en el caso de estudio con cada algoritmo

En conclusión, tomaremos la secuenciación obtenida mediante el Algoritmo GRASP como la mejor solución factible encontrada para nuestro problema. En la figura **??** podemos observar el diagrama de Gantt de los 5 primeros trabajos secuenciados. Se ha representado únicamente los 5 primeros trabajos por simplicidad, aunque podrían ser representados los 68 que han sido secuenciados. Los primeros modelos secuenciados han sido, en este orden: 3, 33, 1, 13, 29. Cabe recalcar que los resultados de la tabla 5.2 corresponden a los 68 modelos.

En la el diagrama de Gantt de la figura 5.6 observamos como, a pesar de tratarse de la solución encontrada que reducía los huecos entre trabajos en mayor medida, siguen habiendo tiempos de inactividad de las máquinas. Un claro ejemplo lo vemos en la máquina 3 después de procesar el trabajo 3. En este caso, la estación de trabajo 3 entra en estado de espera hasta que el trabajo siguiente (33) acaba de ser procesado en la estación anterior (WS 2). Análogamente, la estación 3 entra en estado de bloqueo después de procesar el trabajo 1, puesto que la siguiente estación sigue procesando el trabajo anterior (33) y por tanto debe esperar para poder pasarle la pieza, lo que implica no poder recibir la nueva pieza (trabajo 13).

Sin embargo, en caso de que se deseará realizar una implementación del algoritmo en la producción, el tiempo de computación supone un factor bastante importante, ya que requeriría una actualización constante de datos que informaran sobre la situación de la línea y una aportación rápida de soluciones que se adapten

realidad exacta de la cadena en tiempos ínfimos. Además, debería de ser posible el reentrenamiento fugaz en caso de que cambiaran los objetivos o la demanda de producción. Por todo ello, sería preferible contar con un algoritmo rápido que otorgara soluciones bastante buenas que con otro con soluciones mejores pero con un elevado coste computacional. En este caso particular, notemos que la diferencia entre la calidad de la solución obtenida por el Algortimo Heurístico 2 y el Algoritmo GRASP podría resultar despreciable. Sin embargo, la diferencia en tiempos computacionales es abismal. Estaríamos hablando de unos pocos segundos en el caso del Algoritmo Heurístico 2 y un par de horas en el caso del GRASP. Por este motivo, si se planteara una posible implementación con reentrenamiento que se retroalimentara proactivamente de la situación real de la línea sería conveniente la utilización del Algoritmo Heurístico 2. Este enfoque se plantea como futuro trabajo.



(a) (b)

Figura 5.6: Diagrama de Gantt de los 5 primeros modelos secuenciados

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusiones

El presente Trabajo de Fin de Máster tenía como principal objetivo la revisión del *Flow Shop Scheduling Problem*, uno de los problemas más estudiados en el campo de la programación de la producción, con el propósito de implementar un algoritmo heurístico que nos ofreciera soluciones factibles para secuenciar en una línea de producción real. Para ello, se ha presentado en términos generales las peculiaridades y características de este problema, así como su modelización matemática y los diferentes objetivos que pueden ser optimizados. De la misma forma, se presentan una serie de algoritmos heurísticos con diferente alcances y se analiza numéricamente el rendimiento de cada uno de ellos en diferentes situaciones. Finalmente, se aplican los algoritmos propuestos a la resolución de un problema de secuenciación en una línea de soldadura multimodelo en la planta de carrocerías de una fábrica del sector del automóvil.

En primer lugar, se estudió el Algoritmo NEH, puesto que tras una indagación bibliográfica en profundidad resultó ser uno de los heurísticos más populares para resolver este problema y que servía como base de muchos otros algoritmos de mejora. Por este motivo, se planteó como reto proponer algún algoritmo que consiguiera igualar o mejorar las soluciones obtenidas mediante NEH. De hecho, al realizar las comparaciones numéricas de algoritmos, resultó ser el segundo que peor rendimiento tenía en cuanto a calidad de soluciones.

Por una parte, el primer algoritmo heurístico propuesto, que hemos denotado a lo largo del trabajo como Algoritmo Heurístico 1, se caracteriza por tomar como objetivo el tiempo de completación del último trabajo y el tiempo total de inactividad y se presentaba como un algoritmo que centraba su mejora en reducir los huecos entre trabajos sucesivos respecto otros algoritmos similares. Sin embargo, tras las simulaciones numéricas, ha resultado ser el algoritmo que peor calidad de soluciones ofrecía respecto a los otros algoritmos analizados.

Por otra parte, se propuso otro algoritmo heurístico que basaba su actuación en intercambiar los trabajos consecutivos cuyas diferencias de tiempo de proceso sean considerables y reducir así el tiempo de flujo, uno de los objetivos propuestos. Un rasgo a destacar de este algoritmo es que toma como solución inicial la solución ofrecida por el Algoritmo NEH y realiza mejoras iterativas hasta alcanzar el criterio de parada definido. De hecho, tras las pruebas experimentales, se comprobó que el tiempo de flujo era ligeramente inferior en las soluciones obtenidas mediante este algoritmo que las obtenidas mediante el algoritmo NEH, como era de espe-

rar. Además, esta reducción en el tiempo de flujo implicaba una mejora significativa en el valor de la función objetivo.

Finalmente, se propuso un Algoritmo GRASP que incorporaba fase de construcción *greedy* mediante la definición de una lista restringida de candidatos (RCL) y *swap* como estrategia de búsqueda local que se repite desde el principio en caso de mejora, aumentando así la aleatoriedad de la búsqueda y abriendo así el campo de posibilidades. Se realizó una calibración de los hiperparámetros del método mediante pruebas experimentales y se concluyó tomando el carácter más *greedy* para la fase de construcción y el menor número posible de iteraciones que aseguraran buenas soluciones en tiempos de computación razonables. Dicho en otras palabras, en nuestro caso particular no ha resultado importante la aleatorización en la construcción de soluciones. Además, se optó por preferir soluciones ligeramente peores (diferencias no significativas), pero que se llevaran a cabo en menores tiempos de cómputo.

En definitiva, las diferencias entre los métodos resultaron ser significativas tras realizar un Análisis de la Varianza, siendo el algoritmo GRASP el que mejores soluciones ofrecía, seguido del Algoritmo Heurístico 2 y el Algoritmo NEH, siendo el Algoritmo Heurístico 1 el que peor soluciones aportaba. Cabe comentar que se observó una ligera significancia en el número de trabajos de las instancias evaluadas, es decir, en función del tamaño de la instancia la diferencia de comportamientos entre algoritmos también era variante. Sin embargo, no se realizó un análisis en profundidad y se deja como trabajo futuro para próximas investigaciones.

Finalmente, se han tomado los tiempos de proceso referidos a los 68 modelos en las 8 estaciones de trabajo de la línea de soldadura de la planta de carrocerías de Ford y se han aplicado todos los algoritmos. Las conclusiones que se extraen son similares a las obtenidas con las instancias utilizadas en los experimentos numéricos y se obtienen en tiempos computacionales razonables.

6.2. Trabajo futuro

El presente trabajo es en resumen una primera toma de contacto con el Problema de *Flow Shop Scheduling Problem*. Se trata de un problema muy estudiado en el campo de programación de la producción y con múltiples vías de investigación abiertas. De hecho, ya hemos visto que este problema permite adaptarse a muchas casuísticas e incoporar variantes o restricciones. Por este motivo, consideramos interesante plantear futuras líneas de investigación que se nos han ocurrido a lo largo del desarrollo del trabajo y que quedarían pendientes en caso de proseguir con la realización del mismo.

Por una parte, en el primer capítulo de definición del problema se comentó la posiblilidad de incorporar más de una función objetivo a ser optimizada. En nuestro caso particular, los medibles a optimizar han resultado no ser contradictorios, lo cual ha permitido considerar una única función objetivo que los tuviera en cuenta todos y plantearlo como un problema multiobjetivo. Sin embargo, en la industria real son muchos los aspectos a tener en cuenta y pueden aparecer otros medibles que no necesariamente se comporten en la misma dirección. Estaríamos hablando entonces de un problema multiobjetivo que debería ser modelado y resuelto de otra manera totalmente diferente a la propuesta en el presente trabajo. Por este motivo, consideramos una buena propuesta de continuación acercarse todavía más a la realidad industrial e incorporar la posibilidad de considerar objetivos discrepantes.

Por otra parte, un inconveniente recurrente durante la realización del presente trabajo ha sido la duración extensa de cálculo computacional de los algoritmos, particularmente del Algoritmo GRASP. Recordemos que de la manera en la que se había definido incorporaba un criterio de parada temporal en el algoritmo general, antes de llamar a las funciones asociadas a cada una de las fases. Sin embargo, una vez entraba en la fase de búsqueda local, hasta que no se realizara el número de iteraciones introducido como parámetro, no se comprobaba de nuevo esta condición, lo que provocaba procesamientos prolongados y costosos. Por este motivo, se propone como trabajo futuro incorporar un criterio de parada dentro de la propia fase de búsqueda local, para evitar así los tiempos de cómputo extensos. Así mismo, sería interesante una revisión en profundidad de los códigos de implementación en R de los algortimos en búsqueda de posibles mejoras que hagan la codificación más eficiente y que supongan un menor coste computacional.

De igual manera, una futura investigación podría abarcar una combinación de los algoritmos propuestos, aprovechando las fortalezas de cada tipo y obteniendo así un método mejorado que tomara únicamente las virtudes de cada uno de ellos. En particular, una buena idea sería tomar como solución inicial de Algoritmo Heurístico 2 una solución obtenida mediante GRASP, puesto que son los 2 algoritmos que mejores soluciones nos ofrecían. Realmente sería equivalente a introducir una segunda fase de búsqueda local en el GRASP pero con una estrategia muy distinta a la ya definida. Además, convendría realizar un estudio más en profundidad de la significancia del número de trabajos en la diferencia entre algoritmos, puesto que todo apunta a que en función del tamaño de la instancia que vayamos a tratar sería conveniente utilizar un método u otro.

Notemos que en el caso de estudio hemos supuesto que se iban a producir los 68 modelos, es decir, un coche de cada tipo. Sin embargo, en una jornada de producción la situación es muy distinta. En particular, el número de coches a secuenciar cada vez y el modelo dependerá de la demanda de producción. Esto podría simularse a sabiendas de la probabilidad de producción de un determinado modelo, estimando una proporción del número de coches a producir de cada modelo. En caso de una implementación real, la proporción de coches a producir en cada momento sería conocido a priori.

Finalmente, la complejidad final reside en la puesta en producción de los algoritmos. Hasta hace pocos años, se tomaban medidas de tiempos de ciclo a mano con un simple cronómetro. Sin embargo, este procedimiento arcaico no es sostenible en la industria, ya que es un proceso lento con inexactitudes significativas debido a las limitaciones de percepción y tiempo de reacción humanos. Además, la falta de datos en tiempo real dificulta la toma de decisiones oportuna y la monitorización y optimización de los procesos. Sin embargo, gracias a los avances en conectividad, se nos permite tener información actualizada sobre tiempos de proceso, incidencias o paros en la línea. Esto supone una recogida de datos contínua que informe de la situación exacta de la línea en cada momento y permite realizar una programación lo más ajustada a la realidad posible, adaptándose así a las necesidades de la misma.

De la misma forma, antes de poner en marcha un algoritmo en producción debe de realizarse un análisis en profundidad de los beneficios económicos que supondría implantarlos en base a la calidad de las soluciones que se secuencian actualmente y el coste de implementación. Para ello, sería necesario conocer en profundidad de qué manera se realiza la secuenciación en la línea de carrocerías de Ford Almussafes, en base a qué criterios y qué restricciones se aplican. Una vez conociéramos la secuenciación que se aplica o incluso la ordenación que han seguido productos ya secuenciados, podríamos realizar experimentos reales y comparaciones múltiples

con las soluciones obtenidas mediante nuestros algoritmos para desarrollar si existe una mejora real respecto al sistema que está implementado a día de hoy y verificar que resulta rentable la implementación. Sin embargo, por motivos de confidencialidad de la empresa no disponemos de esta información y es muy complicado llegar a ella.

Bibliografía

[1] J.C. Yepes-Borrero, F. Perea, F. Villa, E. Vallada, *Flowshop with additional resources during setups: Mathematical models and a GRASP algorithm*, Computers and Operations Research (154), Elsevier, València (Spain), 2023.

[2] M.M. Yenisey, B. Yagmahan, *Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends*, Computers and Operations Research (154), Omega, Elsevier, Istanbul (Turkey), 2013.

[3] J.C. Ho, Y.L. Chang, *A new heuristic for the n-job, M-machine flow-shop problem*, European Journal of Operational Research 52, 194-202, North-Holland, 1991.

[4] C. Rajendran, *A heuristic for scheduling in flowshop and flowline-based manufacturing cell with multi-criteria*, International Journal of Production Research, 32:11, 2541-2558, 1994.

[5] R. Puka, A. Stawowy, J. Duda, *Input Sequence of Jobs on NEH Algorithm for Permutation Flowshop Scheduling Problem*, Management and Production Engineering Review, Volume 13, Number 1, 32:43, 2022

[6] C. Andrés, [Universitat Politècnica de València - UPV] (6 noviembre 2014). *Secuenciación en taller de flujo con la regla NEH*

[7] E. García, N. Montés, *Análisis de los sub-tiempos de ciclo técnico para la mejora del rendimiento de las línea de fabricación*, Tesis Doctoral, Universidad CEU Cardenal Herrera, Dep. de Ciencias Físicas, Matemáticas y de la Computación, Valencia 2016

[8] J. Llopis, A. Lacasa, E. García, N. Montés, L. Hilario, J. Vizcaíno, C. Vilar, J. Vilar, L. Sánchez, J.C. Latorre, *Manufacturing Maps, a Novel Tool for Smart Factory Management Based on Petri Nets and Big Data Mini-Terms*, MDPI, Mathematics 2022, 10, 2398

[9] Leonidas S. Pitsoulis, Mauricio G.C. Resende, *Greedy Randomized Adaptive Search Procedures*, AT&T Labs Research Technical Report, 2001

[10] *Greedy Randomized Adaptive Search (GRASP)*, `https://ccc.inaoep.mx/~emorales/Cursos/Busqueda/node66.html`

[11] *Algoritmos Voraces*, `https://aprende.olimpiada-informatica.org/algoritmia-voraz`

[12] *ANOVA análisis de varianza para comparar múltiples medias*, https://cienciadedatos.net/documentos/19_anova

[13] *¿Qué es Grid Search?*, https://ichi.pro/es/que-es-grid-search-211243277676298

Apéndice A

Implementación en RStudio de las funciones objetivo evaluadas

```
#FUNCIONES
#Funcion calculo makespan a partir de una secuencia de trabajos
makespan <- function(sec, p) {
  #N mero de trabajos en la secuencia
  n <- length(sec)
  #N mero de m quinas
  m <- ncol(p)
  tiempomaquina <- matrix(0, nrow = n, ncol = m)
  for (i in 1:n) {
    for (j in 1:m) {
      if (i == 1 & j==1) {
        tiempomaquina[i, j] <- p[sec[i], j]
      } else {
        tiempomaquina[i, j] <- max(tiempomaquina[i-1, j], tiempomaquina[i, j-1])
            + p[sec[i], j]
      }
    }
  }
  return(tiempomaquina[n, m])
}

# Funcion para calcular el flowtime total
flowtime <- function(sec, p) {
  #N mero de trabajos en la secuencia
  n <- length(sec)
  #N mero de m quinas
  m <- ncol(p)
  tiempomaquina <- matrix(0, nrow = n, ncol = m)
  for (i in 1:n) {
    for (j in 1:m) {
      if (i == 1 & j==1) {
        tiempomaquina[i, j] <- p[sec[i], j]
```

```
      } else {
        tiempomaquina[i, j] <- max(tiempomaquina[i-1, j], tiempomaquina[i, j-1])
            + p[sec[i], j]
      }
    }
  }
  flowtime_total <- sum(tiempomaquina[,m])
  return(flowtime_total)
}
# Funcion para calcular el Idle Time

idletime <- function(sec,p){
  n<- length(sec)
  m<- ncol(p)
  #tiempo maquina
  tiempomaquina <- matrix(0, nrow = n, ncol = m)
  for (i in 1:n) {
    for (j in 1:m) {
      if (i == 1 & j==1) {
        tiempomaquina[i, j] <- p[sec[i], j]
      } else {
        tiempomaquina[i, j] <- max(tiempomaquina[i-1, j], tiempomaquina[i, j-1])
            + p[sec[i], j]
      }
    }
  }
  #tiempo maquina
  #Ij
  tp <- rep(0,m)
  I <- rep(0,m)
  for (j in 1:m){
    tp[j]<-0
    I[j] <-0
    for (i in 1:n){
      tp[j] <- tp[j]+p[i,j]
    }
    I[j] <- tiempomaquina[n,j] - tp[j]
  }
  #Ij
  idlt <- sum(I)
  return(idlt)
}
```

```
 #Funci n objetivo
objectivefunction <- function(sec,p){
  fo<- 0.4*makespan(sec,p) + 0.3*flowtime(sec,p) + 0.3*idletime(sec,p)
  return(fo)
}
```

Apéndice B

Implementación en RStudio Algoritmo NEH

```
#Funci n  heur stico NEH

heuristicoNEH <- function(p) {
  #N mero de trabajos
  n <- nrow(p)
  #N mero de m quinas
  m <- ncol(p)

  #1) Calcular tiempo de proceso total de cada trabajo en todas las m quinas
  Ttrab <- rowSums(p)
  #2) Ordenar los trabajos de mayor a menor valor de tiempo de proceso total
  orden <- order(Ttrab, decreasing = TRUE)
  #3) Elegir los 2 primeros trabajos de la lista ordenada y construir las 2
     posibles secuencias iniciales
  sec1 <- c(orden[1], orden[2])
  sec2<- c(orden[2],orden[1])
  #4) Nos quedamos con la secuencia parcial con menor makespan
  makespan1<- makespan(sec1,p)
  makespan2 <- makespan(sec2,p)
  if(makespan1<=makespan2){
    sec <- sec1
  }
  if(makespan2<makespan1){
    sec <- sec2
  }

  #Secuencia parcial inicial
  sec
  #Makespan de la secuencia parcial inicial
  makespan(sec,p)

  #5) Insertar los trabajos restantes de mayor a menor tiempo de proceso en la
     posici n que minimice makespan
  for (i in 3:n) {
```

```
33    mp <- rep(0,i)
34    secTmp <- matrix(0, nrow = i, ncol = i)
35    for (j in 1:i) {
36      #Insertar el trabajo con orden i en las i posibles posiciones
37      secTmp[j,j]<- orden[i]
38      #Rellenar las i posibles secuencias con los i-1 trabajos ya secuenciados
           en la sec parcial
39      for (k in 1:i){
40        s<-0
41        for(l in 1:i){
42          if(k!=l){
43          s<-s+1
44          secTmp[k,l]<-sec[s]
45            }
46          if(k==l){
47            secTmp[k,l] = secTmp[k,l]
48          }
49          }
50
51      }
52      mp[j] <-makespan(secTmp[j,],p)
53    }
54    #secTmp: i posibles secuencias de i trabajos
55    #mp: makespan de las i posibles secuencias de i trabajos
56    minmp <- which.min(mp[j])
57    sec <- secTmp[minmp,]
58    }
59  return(sec)
60    }
```

Apéndice C

Implementación en RStudio Algoritmo 1

```
#Funcion para determinar la maquina con mayor valor de diferencias revisadas
maxdifrev <- function(dR, solucion, a,b){
  maxdR <- -Inf
  maxmachine <- 0
  for (l in a:b){
    value <- dR[solucion[a],solucion[l]]
    if(value > maxdR){
      maxdR <- value
      maxmachine <- l
    }
  }
  return(maxmachine)
}

#Funcion para determinar la maquina con menor valor de diferencias revisadas
mindifrev <- function(dR, solucion, a, b){
  mindR <- Inf
  minmachine <- 0
  for (l in a:b){
    value <- dR[solucion[l],solucion[b]]
    if(value < mindR){
      mindR <- value
      minmachine <- l
    }
  }
  return(minmachine)
}


#Funcion intercambio de trabajos
intercambiotrabajos <- function(solucionini, i, k){
  solucion <- solucionini
  solucion[i]<- solucionini[k]
  solucion[k] <- solucionini[i]
```

```
35   return(solucion)
36 }
37
38 ######
39
40
41 #######
42 #ALGORITMO HEURISTICO
43 heuristico <- function(p) {
44
45 #Numero de trabajos
46 n<-nrow(p)
47 n
48 #Numero de maquinas
49 m<-ncol(p)
50 m
51
52 #Diferencias de tiempo de procesos
53 dist <- list(0,(m-1))
54 for (j in 1:(m-1)){
55   dist[[j]]<- matrix(0, nrow = n, ncol = n)
56   for (i in 1:n){
57     for (k in 1:n){
58       if(i!=k){
59         dist[[j]][i,k] = p[i,j+1] - p[k,j]
60       }
61     }
62   }
63 }
64
65
66
67 #Factor
68 factor<-rep(0,m-1)
69 for (j in 1:(m-1)){
70   factor[j] <- ((1-0.1)/(m-2))*(m-j-1)+0.1
71 }
72
73
74 desc <- list(0,(m-1))
75 for (j in 1:(m-1)){
76   desc[[j]]<- matrix(0, nrow = n, ncol = n)
77   for (i in 1:n){
78     for (k in 1:n){
79       if(i!=k){
80         if (dist[[j]][i,k]<0){
81           desc[[j]][i,k] <- factor[j]
82         } else{
83           desc[[j]][i,k]<-1
```

```
84          }
85        }
86      }
87    }
88  }
89
90
91  #Diferencias revisadas
92  dR <- matrix(0, nrow = n, ncol = n)
93  for (i in 1:n){
94    for (k in 1:n){
95      if(i!=k){
96        for (j in 1:(m-1)){
97          dR[i,k] <- dR[i,k] + dist[[j]][i,k]*desc[[j]][i,k]
98        }
99      }
100   }
101 }
102
103
104 #PASO 1: Solucion inicial (solucion titular): Permutacion aleatoria de los n
      trabajos
105 sol_inicial <- sample(n)
106 bestsolution <- sol_inicial
107 bestfo <- makespan(bestsolution,p)
108 #PASO 2: C lculo de los valores de las diferencias revisadas
109 dR
110 #PASO 4: Fijar a=1 y b=n
111 a<- 1
112 b<- n
113 #PASO 13: Criterio de parada (b <=a+2)
114   while (b > a + 2) {
115     # PASO 5: Encontrar la m quina con el mayor valor de diferencias revisadas
116     u <- maxdifrev(dR, bestsolution, a, b)
117     X <- dR[bestsolution[a], bestsolution[u]]
118
119     # PASO 6: Encontrar la m quina con el menor valor de diferencias revisadas
120     v <- mindifrev(dR, bestsolution, a, b)
121     Y <- dR[bestsolution[v], bestsolution[b]]
122
123     #PASO 7, 8, 9: Condicionales seg n valores de X, Y
124     if (X < 0 && Y > 0 && abs(X) <= abs(Y)) {
125       # PASO 10: Intercambiar trabajos en las posiciones a y u
126       a <- a + 1
127       solution <- intercambiotrabajos(bestsolution, a, u)
128     } else if (X < 0 && Y > 0 && abs(X) > abs(Y)) {
129       # PASO 11: Intercambiar trabajos en las posiciones b y v
130       b <- b - 1
131       solution <- intercambiotrabajos(bestsolution, b, v)
```

```
132     } else if (abs(X) > abs(Y)) {
133       # PASO 10: Intercambiar trabajos en las posiciones a y u
134       a <- a + 1
135       solution <- intercambiotrabajos(bestsolution, a, u)
136     } else {
137       # PASO 11: Intercambiar trabajos en las posiciones b y v
138       b <- b - 1
139       solution <- intercambiotrabajos(bestsolution, b, v)
140 
141     }
142 
143     # PASO 12: Evaluar si la solucion encontrada mejora la funcion objetivo
144     valor_fo <- makespan(solution,p)
145     if (valor_fo < bestfo) {
146       bestsolution <- solution
147       bestfo <- valor_fo
148     }
149   }
150 
151   # Retornar la mejor solucion encontrada
152   return(bestsolution)
153 }
154 
155 
156 #CPU: TIEMPO DE COMPUTACION
157 timing <- system.time({
158   sol<- heuristico(p)
159 })
160 print(timing)
```

Apéndice D

Implementación en RStudio Algoritmo 2

```
# Funcion ALGORITMO HEURISTICO 2
heuristico2 <- function(p) {
  n <- nrow(p)
  m <- ncol(p)

  # Paso 1: Solucion inicial mediante heuristica NEH
  secuencia_inicial <- heuristicoNEH(p)
  #Inicializamos variables
  S <- secuencia_inicial
  F <- flowtime(S, p)
  M <- makespan(S, p)

  paso <- 2

  repeat {
    # Paso 2: Intercambiamos los trabajos en las posiciones i y i+1 de la
       secuencia inicial
    for (i in 1:(n-1)) {
      # Intercambiar trabajos
      sec_int <- S
      sec_int[i] <- S[i+1]
      sec_int[i+1] <- S[i]

      # Calcula makespan y flowtime de la nueva secuencia
      M_int <- makespan(sec_int, p)
      F_int <- flowtime(sec_int, p)

      # Compara con la secuencia actual y actualiza si es mejor
      if (M_int < M || (M_int == M && F_int < F)) {
        S <- sec_int
        M <- M_int
        F <- F_int
      }
    }
```

```
34
35    paso <- paso + 1
36
37    # Paso 3: Calcula valores Di para cada trabajo de la secuencia S
38    Di <- rep(0,n-1)
39    for (i in 1:(n-1)) {
40      Di[i] <- sum(p[S[i], ] - p[S[i+1], ])
41    }
42
43    #Paso 4: Calcula valores Di' para cada trabajo de la secuencia S
44    Di2 <- rep(0,n-1)
45    for (i in 1:(n-1)) {
46      for (j in 1:m){
47      Di2[i] <- Di2[i] + (m-j+1)*p[S[i],j] - (m-j+1)*p[S[i+1],j]
48      }
49    }
50
51    # Paso 5: Toma los trabajos con Di positivo
52    # ndice  en S de los trabajos con Di positivo
53    JDP <- which(Di >= 0)
54    #Conjunto de trabajos con Di positivo
55    S[JDP]
56
57    # Paso 6: Encuentra la posici n k del primer trabajo en la secuencia JDP
          ordenada
58    #k <- JDP[JDP_ordenados][1]
59
60    #Paso 6: Comprobamos si el conjunto JDP est  vac o
61    if (length(JDP) == 0) {
62      # Paso 12: Criterio de parada del algoritmo
63      break
64    } else {
65      # Paso 7: Ordenamos el conjunto de trabajos JDP seg n valor decreciente
            de su Di asociado.
66      #En caso de empate en el valor de Di se le asociar  orden mayor al
            trabajo con mayor valor de Di2
67      JDP_ordenados <- order(-Di[JDP],-Di2[JDP])
68      # Paso 8: Intercambia los trabajos consecutivos k y k+1 en la secuencia S
69      k<- JDP[JDP_ordenados][1] #posici n en S del primer trabajo de la
            secuencia JDP ordenada
70      S2 <- S
71      S2[k] <- S[k+1]
72      S2[k+1] <- S[k]
73
74      # Calcula makespan y flowtime de la nueva secuencia
75      M2 <- makespan(S2, p)
76      F2 <- flowtime(S2, p)
77      #Paso 9: Calcular incremento relativo de makespan y flowtime
78      RS2 <- (M2 - min(M2, M))/(min(M2,M)) + (F2 - min(F2, F))/(min(F2,F))
```

```
79        RS <- (M - min(M2, M))/(min(M2,M)) + (F - min(F2, F))/(min(F2,F))
80
81        if (RS2 < RS) {
82          # Paso 10: Actualiza secuencia, makespan y flowtime
83          S <- S2
84          M <- M2
85          F <- F2
86
87          #Paso3
88        } else {
89          # Paso 11: Elimina trabajo en posici n k de JDP
90          JDP <- JDP[-which(JDP == k)]
91
92          if (length(JDP) == 0) {
93            # Paso 12: Finaliza el algoritmo si no hay trabajos en JDP actualizado
94            break
95          }
96
97        }
98      }
99      break
100   }
101   # Retorna la secuencia, makespan y flowtime total obtenidos
102   return(S)
103 }
```

Apéndice E

Implementación en RStudio Algoritmo GRASP

```
  #Fase constructiva GREEDY
# Funci n para generar la lista de candidatos (RCL)
generate_rcl <- function(current_solution, remaining_jobs, p, alpha) {
  rcldata <- data.frame(job=numeric(0), position=numeric(0), fo=numeric(0))
  for (i in 1:length(remaining_jobs)) {
    for (pos in 1:length(current_solution)){
    candidate_solution <- append(current_solution, remaining_jobs[i], after=pos
        -1)
    fo <- objectivefunction(candidate_solution, p)
    rcl <- c(remaining_jobs[i], pos, fo)
    rcldata[i,] <-rcl
    }
  }

  # Calcular el l mite superior e inferior de la RCL
  min_fo <- min(rcldata$fo)
  max_fo <- max(rcldata$fo)
  lower_bound <- min_fo
  upper_bound <- min_fo + alpha * (max_fo - min_fo)

  # Filtrar los trabajos que est n dentro de la RCL
  listarcl <- rcldata[rcldata$fo >= lower_bound & rcldata$fo <= upper_bound,]

  # Ordenar la RCL en orden creciente de funci n objetivo
  listarcl <- listarcl[order(listarcl$fo), ]
  return(listarcl)
}

# Funci n para generar una soluci n inicial utilizando la RCL
greedyConstructionRCL <- function(p, alpha) {
  n <- nrow(p) # N mero de trabajos

```

```
32   remaining_jobs <- 1:n # Trabajos a n no asignados a la soluci n
33   firstjob <-sample(n, 1)
34   current_solution <- c(firstjob) # Soluci n actual, comienza vac a
35   remaining_jobs <- remaining_jobs[-firstjob]
36
37   while (length(remaining_jobs) > 0) {
38     listarcl <- generate_rcl(current_solution, remaining_jobs, p, alpha)
39     if (length(listarcl) > 0) {
40       # Elegir un trabajo aleatorio de la RCL
41       indice_aleatorio <- sample(nrow(listarcl), 1)
42       selected_job <- listarcl[indice_aleatorio, ]
43       # Agregar el trabajo seleccionado a la soluci n actual en la posici n
44       current_solution <- append(current_solution, selected_job$job, after=
             selected_job$position-1)
45       # Eliminar el trabajo seleccionado de la lista de candidatos
46       remaining_jobs <- setdiff(remaining_jobs, selected_job$job)
47     } else {
48       # Si la RCL est  vac a, tomar un trabajo aleatorio de los restantes
49       selected_job <- sample(remaining_jobs, 1)
50
51       # Agregar el trabajo seleccionado a la soluci n actual en una posici n
             random
52       posrandom <-sample(current_solution, 1)
53       current_solution <- append(current_solution, selected_job, after=posrandom
             -1)
54
55       # Eliminar el trabajo seleccionado de la lista de candidatos
56       remaining_jobs <- setdiff(remaining_jobs, selected_job$job)
57     }
58   }
59
60   return(current_solution)
61 }
62
63
64
65 ########
66 #Fase b squeda local
67
68 localsearch<- function(solution, p, max_iter){
69   best_solution <- solution
70   n <- nrow(p)
71   bestfo<- objectivefunction(solution,p)
72   iter <-1
73   while (iter <= max_iter){
74     for (i in 1:n){
75       for (k in 1:n){
76         new_solution <- intercambiotrabajos(best_solution,i,k)
77         newfo <- objectivefunction(new_solution,p)
```

```
78          if (newfo < bestfo){
79            best_solution<- new_solution
80            bestfo <- newfo
81            i<-1
82            break
83          }
84        }
85      }
86      iter <- iter +1
87    }
88    return(best_solution)
89  }
90
91  #FUNCI N ALGORITMO GRASP
92  grasp <- function(p, maxiter, alpha, time_limit_seconds) {
93    best_solution <- NULL  # Initialize the best solution to be empty
94    start_time <- Sys.time()  # Record the start time
95
96    while (difftime(Sys.time(), start_time, units = "secs") < time_limit_seconds)
          {
97      solution <- greedyConstructionRCL(p, alpha)
98      current_best <- localsearch(solution, p, maxiter)
99
100     # Update the best solution if the current one is better
101     if (is.null(best_solution) || current_best < best_solution) {
102       best_solution <- current_best
103     }
104   }
105
106   return(best_solution)
107 }
```

Apéndice F

ANOVA comparación de algortimos en RStudio

```
library(openxlsx)
datos <- read.xlsx("anovardp.xlsx")

#N mero de grupos y cantidad de observaciones por cada grupo para determinar si
     es un modelo equilibrado
table(datos$Alg)
#Calculamos media y desviaci n t pica de cada grupo
aggregate(RDP ~ Algoritmo, data = datos, FUN = mean)
aggregate(RDP ~ Algoritmo, data = datos, FUN = sd)

#Box-Plot: Identificar asimetr as, datos at picos o diferencias de varianzas
library(ggplot2)
ggplot(data = datos, aes(x = Algoritmo, y = RDP, color = Algoritmo)) +
  geom_boxplot() +
  theme_bw()

#Verificar condiciones para un ANOVA
#Independencia
#Distribuci n normal de las observaciones: Variable cuantitativa normal en cada
     uno de los grupos
par(mfrow = c(2,2))
qqnorm(datos[datos$Algoritmo == "Alg1","RDP"], main = "Algoritmo Heur stico 1")
qqline(datos[datos$Algoritmo == "Alg1","RDP"])
qqnorm(datos[datos$Algoritmo == "Alg2","RDP"], main = "Algoritmo Heur stico 2")
qqline(datos[datos$Algoritmo == "Alg2","RDP"])
qqnorm(datos[datos$Algoritmo == "AlgNEH","RDP"], main = "Algoritmo NEH")
qqline(datos[datos$Algoritmo == "AlgNEH","RDP"])
qqnorm(datos[datos$Algoritmo == "GRASP","RDP"], main = "Algoritmo GRASP")
qqline(datos[datos$Algoritmo == "GRASP","RDP"])

#Como el grupo tienen menos de 50 eventos emplear amos el test Shapiro-Wilk
#Shapiro-Wilk
```

```
alg1 <- datos[datos$Algoritmo=="Alg1",]
alg2 <- datos[datos$Algoritmo=="Alg2",]
algneh <- datos[datos$Algoritmo=="AlgNEH",]
alggrasp <- datos[datos$Algoritmo=="GRASP",]
shapiro.test(alg1$RDP)
shapiro.test(alg2$RDP)
shapiro.test(algneh$RDP)
shapiro.test(alggrasp)

#Homocedasticidad: Varianza constante entre grupos
#Test de Fligner-Killeen
fligner.test(RDP ~ Algoritmo,datos)

#Test de Levene
library(car)
leveneTest(RDP ~ Algoritmo,datos,center = "median")


#An lisis de la varianza ANOVA
anova <- aov(datos$RDP ~ datos$Algoritmo + datos$njobs)
summary(anova)
plot(anova)

#Tama o del efecto de un ANOVA
#Valores del summary del ANOVA
eta_cuadrado <- 0.0076/(0.0076 + 0.4080) #n meros del summary de anova
eta_cuadrado

#Comparaciones m ltiples
#Si el ANOVA resulta significativo --> comparaciones 2 a 2
#M todos de comparaciones m ltiples y correcciones: Correcci n de Holm y
    TukeyHSD
#Correcci n de Holm
pairwise.t.test(x = datos$RDP, g = datos$Algoritmo, p.adjust.method = "holm",
                pool.sd = TRUE, paired = FALSE, alternative = "two.sided")
#TukeyHSD
TukeyHSD(anova)
par(cex.axis = 0.8, cex.main = 1, cex.lab = 0.8)
plot(TukeyHSD(anova))
```

Apéndice G

Instancias calibración de hiperparámetros

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	45	31	54	54	64
$\mathbf{J_2}$	44	7	52	66	57
$\mathbf{J_3}$	26	19	65	34	27
$\mathbf{J_4}$	74	83	94	76	60
$\mathbf{J_5}$	19	41	31	50	33
$\mathbf{J_6}$	20	28	42	63	29
$\mathbf{J_7}$	23	30	10	27	26
$\mathbf{J_8}$	46	19	11	5	36
$\mathbf{J_9}$	76	76	34	6	91
$\mathbf{J_{10}}$	57	31	33	8	19

Cuadro G.1: **Instancia 1**: $n = 10$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	8	92	39	92	59	40	96	6	12	41
$\mathbf{J_2}$	78	49	42	85	48	81	20	63	52	30
$\mathbf{J_3}$	90	34	7	95	70	28	93	52	24	91
$\mathbf{J_4}$	33	94	50	49	94	82	98	49	48	97
$\mathbf{J_5}$	94	2	79	4	37	42	15	79	2	66
$\mathbf{J_6}$	35	65	67	32	68	36	43	7	94	9
$\mathbf{J_7}$	64	26	4	24	87	78	81	78	2	29
$\mathbf{J_8}$	25	89	31	67	8	16	7	19	42	74
$\mathbf{J_9}$	25	64	24	26	21	60	32	5	24	60
$\mathbf{J_{10}}$	37	36	74	50	49	58	26	86	73	31

Cuadro G.2: **Instancia 2**: $n = 10$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	28	78	77	96	79	99	95	50	5	10	32	66	80	9	38
$\mathbf{J_2}$	82	31	56	15	39	52	28	62	29	26	84	24	64	90	52
$\mathbf{J_3}$	96	17	81	49	43	20	69	94	24	31	53	73	46	11	77
$\mathbf{J_4}$	70	92	68	94	78	72	54	85	38	16	20	9	83	91	82
$\mathbf{J_5}$	44	90	41	41	43	48	36	50	17	8	11	40	8	77	32
$\mathbf{J_6}$	43	38	66	1	65	26	88	1	3	9	1	37	6	23	17
$\mathbf{J_7}$	38	42	20	25	97	90	34	2	9	85	8	9	93	98	56
$\mathbf{J_8}$	7	91	23	37	11	36	76	23	8	3	47	25	65	73	58
$\mathbf{J_9}$	10	37	12	18	85	29	25	91	78	80	32	4	11	82	79
$\mathbf{J_{10}}$	77	47	31	69	50	83	87	90	5	89	99	80	29	20	1

Cuadro G.3: **Instancia 3**: $n = 10$ trabajos, $m = 15$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$	$\mathbf{M_{16}}$	$\mathbf{M_{17}}$	$\mathbf{M_{18}}$	$\mathbf{M_{19}}$	$\mathbf{M_{20}}$
$\mathbf{J_1}$	84	38	69	12	42	86	44	54	4	13	6	51	93	65	87	22	95	47	63	20
$\mathbf{J_2}$	77	9	94	8	50	80	26	1	11	41	26	10	90	82	85	53	7	38	90	92
$\mathbf{J_3}$	92	47	55	25	53	98	82	38	66	76	72	96	39	87	14	36	81	48	14	61
$\mathbf{J_4}$	44	68	60	99	61	53	20	32	74	8	13	16	27	24	23	78	67	73	31	25
$\mathbf{J_5}$	95	45	79	5	40	1	6	31	44	34	9	44	7	50	28	49	15	16	46	97
$\mathbf{J_6}$	39	40	64	72	93	74	88	96	93	64	37	83	33	78	58	96	90	89	13	86
$\mathbf{J_7}$	32	1	67	11	49	33	94	78	79	10	87	60	78	22	22	47	46	45	31	70
$\mathbf{J_8}$	28	39	78	99	39	74	34	86	90	39	2	25	6	33	44	32	62	57	3	88
$\mathbf{J_9}$	99	41	20	85	63	88	93	54	1	6	35	98	6	62	12	40	90	2	83	44
$\mathbf{J_{10}}$	28	82	66	4	87	35	97	86	76	59	73	63	33	89	71	97	51	18	10	44

Cuadro G.4: **Instancia 4**: $n = 10$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	52	85	39	44	78
$\mathbf{J_2}$	74	77	26	84	63
$\mathbf{J_3}$	50	63	74	72	54
$\mathbf{J_4}$	61	36	34	89	11
$\mathbf{J_5}$	57	73	48	85	54
$\mathbf{J_6}$	21	46	45	70	22
$\mathbf{J_7}$	32	3	56	29	21
$\mathbf{J_8}$	79	52	4	67	65
$\mathbf{J_9}$	22	59	56	50	96
$\mathbf{J_{10}}$	21	68	86	59	92
$\mathbf{J_{11}}$	78	20	56	41	64
$\mathbf{J_{12}}$	79	14	57	45	12
$\mathbf{J_{13}}$	15	35	81	31	41
$\mathbf{J_{14}}$	85	98	72	8	90
$\mathbf{J_{15}}$	21	11	30	39	23
$\mathbf{J_{16}}$	68	3	15	43	8
$\mathbf{J_{17}}$	87	28	10	14	35
$\mathbf{J_{18}}$	8	68	90	79	97
$\mathbf{J_{19}}$	31	28	99	24	68
$\mathbf{J_{20}}$	18	22	15	17	28

Cuadro G.5: **Instancia 5**: $n = 20$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	72	15	68	96	3	54	3	80	73	80
$\mathbf{J_2}$	70	39	39	6	63	37	63	40	10	76
$\mathbf{J_3}$	38	36	93	66	80	77	3	51	1	43
$\mathbf{J_4}$	29	75	75	84	62	98	77	98	31	2
$\mathbf{J_5}$	42	84	6	42	81	22	58	39	65	4
$\mathbf{J_6}$	36	8	35	10	12	75	16	9	1	36
$\mathbf{J_7}$	48	22	63	38	58	91	39	6	24	24
$\mathbf{J_8}$	67	43	11	17	6	62	1	83	48	50
$\mathbf{J_9}$	17	24	83	50	23	26	79	27	60	80
$\mathbf{J_{10}}$	62	52	40	46	29	15	7	2	16	42
$\mathbf{J_{11}}$	44	73	4	6	81	17	48	30	33	96
$\mathbf{J_{12}}$	54	24	70	97	28	26	6	44	85	11
$\mathbf{J_{13}}$	55	37	1	7	88	15	22	21	89	94
$\mathbf{J_{14}}$	80	80	95	76	78	19	86	86	32	74
$\mathbf{J_{15}}$	19	10	96	89	59	37	19	12	46	14
$\mathbf{J_{16}}$	66	46	76	88	49	4	15	69	32	50
$\mathbf{J_{17}}$	56	65	91	48	96	52	76	39	90	90
$\mathbf{J_{18}}$	74	30	9	6	87	53	3	67	71	33
$\mathbf{J_{19}}$	65	75	4	74	30	8	62	53	47	90
$\mathbf{J_{20}}$	46	72	42	21	75	21	19	9	26	48

Cuadro G.6: **Instancia 6**: $n = 20$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	38	26	88	81	12	95	85	88	60	18	55	89	68	4	53
$\mathbf{J_2}$	64	29	40	76	92	84	26	56	12	88	45	58	5	15	43
$\mathbf{J_3}$	59	82	39	72	1	52	50	19	48	92	94	39	93	76	47
$\mathbf{J_4}$	63	85	99	46	42	77	26	72	39	22	50	27	7	57	24
$\mathbf{J_5}$	49	92	62	21	88	58	71	20	8	20	86	33	85	28	17
$\mathbf{J_6}$	19	48	61	35	45	55	89	37	96	2	31	39	40	20	25
$\mathbf{J_7}$	52	94	5	72	14	76	79	29	64	95	78	51	10	39	23
$\mathbf{J_8}$	32	1	76	9	72	13	50	36	68	24	51	41	34	9	84
$\mathbf{J_9}$	23	63	99	87	71	13	96	46	30	62	9	11	57	76	65
$\mathbf{J_{10}}$	37	77	82	32	79	20	29	21	9	94	99	77	55	61	89
$\mathbf{J_{11}}$	74	83	67	69	69	43	88	99	80	69	83	70	97	17	41
$\mathbf{J_{12}}$	64	40	50	92	90	15	29	20	22	50	59	65	35	94	96
$\mathbf{J_{13}}$	35	45	18	30	62	70	47	71	48	77	27	34	13	95	88
$\mathbf{J_{14}}$	86	26	93	60	94	91	53	65	96	2	1	78	86	6	60
$\mathbf{J_{15}}$	41	83	36	59	58	15	92	79	31	53	91	48	83	72	15
$\mathbf{J_{16}}$	16	1	12	14	49	89	81	71	49	14	23	10	69	75	21
$\mathbf{J_{17}}$	4	93	44	42	39	94	56	22	19	92	34	41	16	72	87
$\mathbf{J_{18}}$	61	15	41	20	7	6	24	34	20	7	40	6	19	38	93
$\mathbf{J_{19}}$	97	36	70	85	95	21	22	25	11	32	80	87	82	56	64
$\mathbf{J_{20}}$	68	68	23	62	12	96	2	3	20	35	11	50	68	99	67

Cuadro G.7: **Instancia 7**: $n = 20$ trabajos, $m = 15$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_1	64	38	94	96	80	93	8	77	43	43	8	7	33	20	94	44	86	79	45	52
J_2	99	5	40	19	1	41	22	99	83	25	32	91	41	10	37	23	34	10	52	22
J_3	8	16	96	99	24	50	67	60	69	59	8	16	69	76	91	81	15	87	69	72
J_4	72	88	35	22	46	26	25	2	3	38	33	18	90	65	18	90	99	73	52	65
J_5	51	14	8	7	15	35	90	66	88	89	82	33	86	14	39	97	32	84	44	8
J_6	36	60	43	74	20	76	81	71	4	49	36	31	76	97	15	23	15	5	51	94
J_7	17	62	32	73	73	81	26	79	82	51	3	13	90	96	73	76	8	47	69	85
J_8	40	27	49	54	23	32	22	64	55	50	87	64	79	38	60	36	40	57	26	54
J_9	17	54	28	92	30	7	9	85	50	91	32	26	68	44	70	60	98	4	42	31
J_{10}	28	76	7	50	37	60	78	19	50	1	15	33	85	84	68	25	26	88	28	39
J_{11}	25	36	91	39	6	29	23	99	47	60	9	27	34	27	11	3	34	89	68	66
J_{12}	25	58	13	55	72	98	90	88	60	80	8	52	92	21	39	24	29	95	73	50
J_{13}	42	15	75	55	83	82	99	52	98	96	81	77	44	65	84	87	94	17	83	62
J_{14}	20	58	18	16	95	1	39	28	70	51	36	16	93	46	14	23	92	64	64	1
J_{15}	39	70	41	3	46	46	42	23	1	24	57	28	1	38	25	97	61	46	23	53
J_{16}	22	89	68	28	41	93	21	36	86	34	94	30	48	11	40	44	26	15	42	78
J_{17}	72	66	41	69	6	74	90	43	99	39	93	51	33	92	75	42	57	63	8	5
J_{18}	96	89	75	98	3	34	41	54	36	74	84	1	27	59	82	71	46	61	71	71
J_{19}	1	77	24	27	10	34	7	43	50	26	12	39	62	69	84	39	60	68	78	36
J_{20}	86	17	18	61	9	76	52	19	41	32	19	80	40	71	58	34	19	75	55	41

Cuadro G.8: **Instancia 8**: $n = 20$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	36	45	94	17	71
$\mathbf{J_2}$	79	82	97	28	27
$\mathbf{J_3}$	21	19	77	81	64
$\mathbf{J_4}$	16	82	74	6	60
$\mathbf{J_5}$	23	57	66	30	25
$\mathbf{J_6}$	31	73	16	94	74
$\mathbf{J_7}$	29	90	65	49	6
$\mathbf{J_8}$	95	3	9	16	4
$\mathbf{J_9}$	45	30	35	75	46
$\mathbf{J_{10}}$	50	93	15	71	66
$\mathbf{J_{11}}$	54	86	55	35	39
$\mathbf{J_{12}}$	54	20	92	74	58
$\mathbf{J_{13}}$	17	63	77	33	33
$\mathbf{J_{14}}$	40	96	26	84	42
$\mathbf{J_{15}}$	59	96	85	1	30
$\mathbf{J_{16}}$	78	62	19	53	40
$\mathbf{J_{17}}$	44	95	32	83	75
$\mathbf{J_{18}}$	32	87	69	99	46
$\mathbf{J_{19}}$	81	47	24	63	10
$\mathbf{J_{20}}$	29	81	62	85	48
$\mathbf{J_{21}}$	11	97	85	80	58
$\mathbf{J_{22}}$	22	16	18	56	84
$\mathbf{J_{23}}$	25	34	18	70	75
$\mathbf{J_{24}}$	74	90	16	40	48
$\mathbf{J_{25}}$	21	16	97	27	34
$\mathbf{J_{26}}$	98	34	71	47	40
$\mathbf{J_{27}}$	78	21	49	43	26
$\mathbf{J_{28}}$	48	19	6	36	45
$\mathbf{J_{29}}$	81	48	97	83	1
$\mathbf{J_{30}}$	78	5	97	76	6

Cuadro G.9: **Instancia 9**: $n = 30$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	45	38	55	70	20	57	21	91	56	81
$\mathbf{J_2}$	61	21	1	10	18	8	56	11	54	2
$\mathbf{J_3}$	36	66	81	76	13	40	31	63	4	64
$\mathbf{J_4}$	78	22	22	2	32	88	76	50	3	12
$\mathbf{J_5}$	22	73	37	44	75	83	92	50	52	8
$\mathbf{J_6}$	55	64	91	22	57	94	76	25	31	63
$\mathbf{J_7}$	58	87	9	20	55	1	55	86	76	12
$\mathbf{J_8}$	96	1	83	22	76	82	8	69	52	29
$\mathbf{J_9}$	75	41	25	15	11	82	41	62	12	92
$\mathbf{J_{10}}$	41	97	40	18	16	8	41	4	9	71
$\mathbf{J_{11}}$	5	51	78	30	35	83	94	3	72	31
$\mathbf{J_{12}}$	51	94	5	90	98	17	44	5	50	43
$\mathbf{J_{13}}$	71	82	1	63	69	41	65	98	90	51
$\mathbf{J_{14}}$	27	12	33	22	51	2	34	33	11	80
$\mathbf{J_{15}}$	84	27	16	76	33	5	62	92	36	89
$\mathbf{J_{16}}$	24	31	68	56	55	64	6	38	1	90
$\mathbf{J_{17}}$	27	53	99	33	24	22	46	54	2	67
$\mathbf{J_{18}}$	42	99	96	11	53	88	65	5	56	19
$\mathbf{J_{19}}$	18	81	31	55	82	96	48	14	45	85
$\mathbf{J_{20}}$	38	63	89	89	65	16	30	94	19	56
$\mathbf{J_{21}}$	26	4	11	49	2	30	44	28	24	80
$\mathbf{J_{22}}$	68	34	10	73	67	77	56	4	26	30
$\mathbf{J_{23}}$	26	55	75	80	31	11	13	3	89	30
$\mathbf{J_{24}}$	51	11	54	78	39	29	30	31	47	74
$\mathbf{J_{25}}$	43	92	41	32	23	1	53	89	96	1
$\mathbf{J_{26}}$	38	41	23	12	81	87	18	38	86	90
$\mathbf{J_{27}}$	34	62	55	53	2	48	50	46	52	51
$\mathbf{J_{28}}$	97	89	71	73	67	65	55	47	37	20
$\mathbf{J_{29}}$	73	29	32	82	51	21	7	13	30	63
$\mathbf{J_{30}}$	78	36	79	70	65	52	46	92	69	15

Cuadro G.10: **Instancia 10**: $n = 30$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	68	15	4	64	96	96	50	46	80	68	21	91	44	19	63
$\mathbf{J_2}$	28	52	68	56	13	62	35	5	38	71	53	18	75	51	49
$\mathbf{J_3}$	35	59	96	78	95	47	39	74	38	6	89	70	97	1	20
$\mathbf{J_4}$	95	36	5	99	50	99	35	38	14	20	3	7	45	87	21
$\mathbf{J_5}$	36	71	64	7	19	81	17	83	55	61	75	53	27	53	75
$\mathbf{J_6}$	11	90	11	19	10	14	81	53	2	10	56	76	13	27	44
$\mathbf{J_7}$	12	34	19	86	92	27	91	94	71	74	91	83	36	43	3
$\mathbf{J_8}$	57	94	13	51	29	65	8	48	10	81	30	6	98	1	76
$\mathbf{J_9}$	12	39	69	96	51	43	99	10	13	87	38	6	13	39	45
$\mathbf{J_{10}}$	88	17	67	22	31	61	32	33	67	92	74	29	22	72	49
$\mathbf{J_{11}}$	67	61	30	63	67	26	48	10	69	56	16	23	74	45	92
$\mathbf{J_{12}}$	95	80	58	58	16	52	23	44	25	78	68	76	12	15	61
$\mathbf{J_{13}}$	66	86	5	90	42	95	38	68	91	75	85	73	75	61	11
$\mathbf{J_{14}}$	42	22	91	67	82	5	78	5	78	87	32	63	96	68	54
$\mathbf{J_{15}}$	34	16	61	58	90	46	92	60	38	8	26	51	32	76	6
$\mathbf{J_{16}}$	55	59	31	6	35	88	45	23	78	28	20	42	97	48	46
$\mathbf{J_{17}}$	77	29	60	34	41	83	57	68	20	28	55	56	58	65	97
$\mathbf{J_{18}}$	88	58	56	3	28	60	5	72	91	68	81	94	4	25	88
$\mathbf{J_{19}}$	94	19	50	58	41	49	8	51	84	54	49	12	99	62	63
$\mathbf{J_{20}}$	69	70	40	26	9	6	31	20	49	4	22	87	2	5	18
$\mathbf{J_{21}}$	14	88	69	24	72	58	3	76	74	19	7	59	94	72	11
$\mathbf{J_{22}}$	5	75	32	41	70	37	52	5	23	30	88	17	96	13	95
$\mathbf{J_{23}}$	85	82	52	4	37	80	69	44	47	15	70	18	9	97	59
$\mathbf{J_{24}}$	16	53	97	96	47	1	25	63	22	18	67	51	13	96	32
$\mathbf{J_{25}}$	68	66	90	54	50	60	65	12	5	23	53	44	39	48	29
$\mathbf{J_{26}}$	39	36	92	19	84	90	23	43	57	12	50	35	13	58	81
$\mathbf{J_{27}}$	81	3	66	63	99	17	76	5	23	52	81	97	77	28	35
$\mathbf{J_{28}}$	78	53	14	8	98	26	45	87	92	63	32	31	62	26	89
$\mathbf{J_{29}}$	39	25	59	42	7	80	18	92	69	4	81	43	43	68	49
$\mathbf{J_{30}}$	15	29	50	23	7	44	76	18	89	79	8	56	17	9	80

Cuadro G.11: **Instancia 11**: $n = 30$ trabajos, $m = 15$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_1	50	59	41	37	25	7	5	3	39	52	72	40	55	38	44	31	74	14	89	65
J_2	5	68	89	49	82	9	34	32	10	94	70	54	91	11	98	45	99	55	43	6
J_3	6	48	61	66	33	10	64	99	55	58	51	33	11	18	96	88	27	46	38	40
J_4	74	24	44	71	78	50	68	75	85	55	22	27	33	14	16	13	87	20	53	12
J_5	52	63	77	56	43	11	3	9	95	90	89	48	58	22	34	40	59	47	83	32
J_6	86	62	5	23	91	37	64	62	15	85	90	97	98	23	32	83	27	18	40	99
J_7	21	31	22	59	54	94	61	94	99	72	85	56	1	8	39	63	23	58	93	84
J_8	55	52	85	49	45	34	84	3	85	82	20	64	1	9	26	62	2	75	76	36
J_9	86	96	22	29	22	1	53	32	36	68	48	41	97	90	83	25	65	94	12	38
J_{10}	33	27	38	17	8	14	48	37	85	24	66	28	28	95	51	71	41	25	61	30
J_{11}	13	10	31	79	71	98	60	44	50	4	99	55	84	50	73	90	82	92	95	52
J_{12}	36	27	48	98	50	44	37	43	41	43	56	69	5	76	1	39	56	2	25	92
J_{13}	13	63	41	59	45	26	88	30	38	39	33	65	26	95	72	40	3	19	51	42
J_{14}	4	92	17	75	33	17	76	45	73	49	21	19	86	75	38	91	19	6	26	92
J_{15}	54	1	81	22	78	69	24	9	52	98	73	7	4	88	4	90	90	27	34	15
J_{16}	91	20	24	37	24	45	98	35	3	18	14	52	83	74	21	59	57	16	19	24
J_{17}	87	62	92	13	12	21	61	69	39	56	6	53	24	18	39	24	1	77	40	11
J_{18}	91	67	62	95	49	12	88	1	73	25	7	71	18	93	4	76	93	59	14	81
J_{19}	11	5	83	54	30	72	10	21	8	80	87	79	73	34	63	24	44	14	80	60
J_{20}	83	18	69	97	72	74	10	52	12	47	72	9	10	27	17	2	12	42	9	33
J_{21}	62	63	71	39	32	75	92	52	31	34	48	41	90	38	22	16	1	88	23	34
J_{22}	22	6	28	49	9	24	13	67	20	10	43	30	97	35	34	29	47	27	9	11
J_{23}	43	76	4	35	39	59	13	3	71	25	62	55	76	31	55	14	51	53	36	13
J_{24}	48	86	50	45	68	10	22	27	93	98	52	41	84	97	15	23	23	79	21	98
J_{25}	76	69	54	87	22	98	13	28	51	22	9	88	37	10	4	75	50	99	91	49
J_{26}	40	78	48	2	30	5	10	74	79	24	61	8	25	12	33	41	2	59	39	30
J_{27}	92	86	11	34	71	41	8	83	38	21	7	50	53	29	3	63	22	58	71	50
J_{28}	90	44	79	99	48	43	42	67	54	38	59	75	30	59	82	78	82	72	1	95
J_{29}	49	89	38	26	93	91	5	12	77	69	41	95	85	18	54	6	95	56	32	54
J_{30}	12	41	77	30	89	2	18	39	45	19	51	99	19	24	65	10	16	82	28	46

Cuadro G.12: **Instancia 12**: $n = 30$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	77	66	16	85	30
$\mathbf{J_2}$	94	38	93	13	15
$\mathbf{J_3}$	31	71	34	75	19
$\mathbf{J_4}$	18	22	41	78	17
$\mathbf{J_5}$	84	41	50	67	84
$\mathbf{J_6}$	54	94	38	84	30
$\mathbf{J_7}$	63	97	25	24	80
$\mathbf{J_8}$	73	57	79	40	21
$\mathbf{J_9}$	56	99	76	29	66
$\mathbf{J_{10}}$	92	56	79	54	74
$\mathbf{J_{11}}$	10	34	90	64	90
$\mathbf{J_{12}}$	29	65	16	29	93
$\mathbf{J_{13}}$	18	58	29	23	39
$\mathbf{J_{14}}$	46	15	64	67	42
$\mathbf{J_{15}}$	36	12	95	80	74
$\mathbf{J_{16}}$	15	53	81	21	35
$\mathbf{J_{17}}$	31	25	19	12	72
$\mathbf{J_{18}}$	89	77	33	39	26
$\mathbf{J_{19}}$	18	3	25	25	2
$\mathbf{J_{20}}$	59	80	78	40	1
$\mathbf{J_{21}}$	69	61	4	94	70
$\mathbf{J_{22}}$	88	70	90	98	98
$\mathbf{J_{23}}$	84	75	4	95	12
$\mathbf{J_{24}}$	22	64	37	10	25
$\mathbf{J_{25}}$	11	47	65	27	16
$\mathbf{J_{26}}$	7	50	87	46	32
$\mathbf{J_{27}}$	97	30	79	56	16
$\mathbf{J_{28}}$	77	2	69	79	71
$\mathbf{J_{29}}$	66	89	63	22	59
$\mathbf{J_{30}}$	54	16	35	14	69
$\mathbf{J_{31}}$	85	23	69	56	48
$\mathbf{J_{32}}$	84	55	68	89	17
$\mathbf{J_{33}}$	62	81	50	98	21
$\mathbf{J_{34}}$	80	44	43	94	83
$\mathbf{J_{35}}$	88	25	73	88	18
$\mathbf{J_{36}}$	20	17	33	77	25
$\mathbf{J_{37}}$	71	88	96	25	83
$\mathbf{J_{38}}$	46	13	41	65	64
$\mathbf{J_{39}}$	78	80	89	67	79
$\mathbf{J_{40}}$	55	25	31	51	52

Cuadro G.13: **Instancia 13**: $n = 40$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	60	32	10	32	5	65	16	96	78	65
$\mathbf{J_2}$	44	8	30	1	60	25	39	28	97	20
$\mathbf{J_3}$	91	31	41	71	76	80	37	67	71	30
$\mathbf{J_4}$	32	99	13	20	31	76	56	71	7	8
$\mathbf{J_5}$	84	80	29	23	21	52	63	42	39	33
$\mathbf{J_6}$	80	99	94	24	76	80	15	96	76	49
$\mathbf{J_7}$	10	31	89	56	9	53	30	50	28	51
$\mathbf{J_8}$	28	54	18	70	83	57	54	49	50	51
$\mathbf{J_9}$	66	26	46	1	22	30	27	23	9	46
$\mathbf{J_{10}}$	69	39	72	4	54	93	23	43	45	72
$\mathbf{J_{11}}$	3	51	94	58	77	65	40	9	5	1
$\mathbf{J_{12}}$	51	46	81	43	61	86	78	8	75	52
$\mathbf{J_{13}}$	16	58	52	52	31	45	99	14	49	1
$\mathbf{J_{14}}$	19	1	56	93	8	57	31	44	87	63
$\mathbf{J_{15}}$	91	41	33	86	63	22	97	59	63	93
$\mathbf{J_{16}}$	88	36	65	16	18	94	4	79	31	81
$\mathbf{J_{17}}$	4	39	4	31	61	77	48	1	17	40
$\mathbf{J_{18}}$	75	96	81	34	89	92	14	1	37	60
$\mathbf{J_{19}}$	71	98	15	58	56	49	19	62	52	56
$\mathbf{J_{20}}$	86	27	21	71	59	2	85	79	83	53
$\mathbf{J_{21}}$	46	35	8	72	81	67	52	53	53	19
$\mathbf{J_{22}}$	66	70	92	12	86	14	58	44	64	39
$\mathbf{J_{23}}$	85	53	38	44	57	15	53	85	92	50
$\mathbf{J_{24}}$	61	92	96	66	48	46	50	56	84	18
$\mathbf{J_{25}}$	21	20	84	10	14	1	18	14	11	36
$\mathbf{J_{26}}$	21	97	55	72	89	88	88	86	61	97
$\mathbf{J_{27}}$	39	19	11	12	18	20	17	58	86	42
$\mathbf{J_{28}}$	79	3	79	45	93	36	15	5	6	82
$\mathbf{J_{29}}$	23	13	13	99	60	44	77	84	10	5
$\mathbf{J_{30}}$	66	90	79	90	48	18	32	33	94	42
$\mathbf{J_{31}}$	37	91	67	84	10	41	7	48	72	43
$\mathbf{J_{32}}$	32	58	76	14	25	97	45	25	71	17
$\mathbf{J_{33}}$	40	92	58	58	62	48	68	31	28	96
$\mathbf{J_{34}}$	16	17	93	2	9	8	14	53	44	81
$\mathbf{J_{35}}$	58	12	58	94	21	29	95	31	97	97
$\mathbf{J_{36}}$	74	35	15	70	37	30	32	49	52	84
$\mathbf{J_{37}}$	67	52	76	94	92	93	97	77	14	31
$\mathbf{J_{38}}$	32	13	38	81	30	57	78	1	25	30
$\mathbf{J_{39}}$	54	72	21	54	90	62	10	77	29	28
$\mathbf{J_{40}}$	61	29	80	91	29	72	16	13	60	67

Cuadro G.14: **Instancia 14**: $n = 40$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	61	29	2	68	52	9	6	26	6	46	32	41	16	59	13
$\mathbf{J_2}$	47	12	91	76	37	64	9	15	27	26	10	82	8	13	27
$\mathbf{J_3}$	99	93	49	56	34	87	30	34	83	27	59	47	87	84	79
$\mathbf{J_4}$	61	28	33	3	81	16	96	3	96	58	53	33	60	59	59
$\mathbf{J_5}$	69	14	96	5	10	68	79	28	71	72	88	49	56	54	25
$\mathbf{J_6}$	89	89	11	52	97	62	78	95	2	99	14	19	43	61	54
$\mathbf{J_7}$	20	72	47	63	88	22	45	56	36	61	70	71	87	16	50
$\mathbf{J_8}$	27	80	15	86	91	35	73	86	46	75	41	29	14	14	72
$\mathbf{J_9}$	35	97	37	99	41	41	51	7	94	9	76	39	30	45	98
$\mathbf{J_{10}}$	8	85	70	13	55	72	5	25	99	9	44	91	83	78	25
$\mathbf{J_{11}}$	13	77	68	44	11	25	13	37	61	27	37	7	51	99	71
$\mathbf{J_{12}}$	82	93	65	94	35	12	35	14	74	75	23	4	28	58	82
$\mathbf{J_{13}}$	49	43	36	51	62	91	67	65	85	57	94	58	91	29	84
$\mathbf{J_{14}}$	62	29	39	92	83	98	16	90	15	42	44	40	99	83	47
$\mathbf{J_{15}}$	63	73	58	35	57	43	8	81	57	71	47	32	31	84	14
$\mathbf{J_{16}}$	48	81	28	50	42	69	18	9	53	90	84	54	76	72	90
$\mathbf{J_{17}}$	1	49	74	75	34	39	1	63	85	3	61	26	51	8	52
$\mathbf{J_{18}}$	98	99	33	21	81	96	89	50	80	25	33	78	68	7	55
$\mathbf{J_{19}}$	31	67	44	73	34	89	23	62	41	77	64	89	85	45	12
$\mathbf{J_{20}}$	56	40	99	85	47	58	91	44	49	11	42	34	86	49	25
$\mathbf{J_{21}}$	25	84	72	10	44	57	61	98	81	82	6	10	3	21	60
$\mathbf{J_{22}}$	77	12	64	78	94	46	49	27	9	84	58	78	31	17	91
$\mathbf{J_{23}}$	22	43	94	89	57	2	70	31	52	72	81	48	75	42	48
$\mathbf{J_{24}}$	20	46	3	93	42	53	98	93	93	36	65	20	26	34	32
$\mathbf{J_{25}}$	38	68	45	46	69	14	47	39	31	24	78	57	10	78	78
$\mathbf{J_{26}}$	14	37	57	27	38	66	21	51	35	81	30	70	78	69	31
$\mathbf{J_{27}}$	9	89	54	85	90	89	62	24	4	23	71	56	90	88	67
$\mathbf{J_{28}}$	61	97	23	94	47	16	28	4	49	61	34	99	3	93	47
$\mathbf{J_{29}}$	65	22	84	5	10	54	60	61	54	5	92	63	79	46	12
$\mathbf{J_{30}}$	80	67	56	67	25	36	50	51	52	40	98	73	35	80	54
$\mathbf{J_{31}}$	43	52	55	19	96	38	40	33	80	83	89	61	79	77	78
$\mathbf{J_{32}}$	7	47	30	38	93	71	33	64	72	97	26	78	60	79	84
$\mathbf{J_{33}}$	92	18	36	58	96	12	30	5	67	26	4	10	74	20	31
$\mathbf{J_{34}}$	90	33	47	73	21	24	72	47	19	71	63	46	92	81	44
$\mathbf{J_{35}}$	73	36	45	25	63	14	67	59	23	81	4	79	11	27	82
$\mathbf{J_{36}}$	28	18	36	48	69	46	15	78	62	82	10	40	20	16	30
$\mathbf{J_{37}}$	80	82	48	51	84	3	92	48	53	92	64	99	51	89	8
$\mathbf{J_{38}}$	54	47	50	10	55	63	73	97	58	75	5	18	93	42	82
$\mathbf{J_{39}}$	51	51	65	26	3	2	99	72	70	75	59	13	94	50	99
$\mathbf{J_{40}}$	93	27	44	70	74	79	96	62	53	35	82	96	10	44	14

Cuadro G.15: **Instancia 15**: $n = 40$ trabajos, $m = 15$ máquinas

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18	M19	M20
J1	22	43	16	36	87	64	99	18	94	25	17	37	28	54	66	77	43	36	47	81
J2	61	21	66	33	17	25	8	39	82	64	67	66	59	87	2	3	23	7	35	61
J3	51	38	29	50	86	90	7	58	95	78	85	49	42	48	65	49	70	31	69	83
J4	79	66	31	91	31	67	92	17	14	16	42	67	72	7	2	84	52	79	60	62
J5	44	8	99	17	10	45	68	82	26	27	80	95	89	91	80	97	57	70	63	42
J6	15	51	59	22	18	24	35	30	16	55	36	20	77	59	5	39	76	54	38	46
J7	89	27	58	34	36	34	25	19	46	30	15	74	45	93	27	17	87	76	6	61
J8	81	22	93	19	74	13	40	5	18	44	43	63	5	43	89	37	53	59	26	47
J9	96	24	77	44	86	43	88	52	22	47	68	32	65	54	44	5	4	93	71	33
J10	74	1	51	60	56	79	44	28	89	1	96	32	95	76	5	11	85	22	88	37
J11	98	16	53	83	98	97	52	81	20	74	92	77	48	29	32	76	65	39	63	11
J12	10	66	99	82	19	56	78	70	54	72	72	86	34	56	66	36	29	68	33	12
J13	70	71	24	25	8	56	57	56	94	18	69	74	91	29	51	63	33	75	7	50
J14	66	55	86	25	85	37	91	93	14	51	51	15	90	45	11	93	85	31	1	52
J15	45	8	57	89	27	36	19	97	91	96	23	11	61	53	80	83	49	24	89	45
J16	28	3	25	67	2	4	26	35	76	19	36	24	57	95	2	9	21	4	21	67
J17	39	22	90	45	42	68	4	2	57	74	94	59	59	48	26	4	20	99	69	34
J18	42	8	40	40	9	64	50	70	92	18	27	11	6	3	94	78	76	24	10	27
J19	84	45	22	88	9	86	72	19	45	30	67	8	6	7	89	78	18	90	32	93
J20	77	37	60	25	16	33	51	70	93	64	51	14	44	86	61	70	93	82	39	62
J21	69	37	70	6	55	43	64	21	43	49	47	78	23	85	93	65	77	55	1	12
J22	91	51	83	25	86	44	76	22	43	75	48	42	72	11	60	13	43	67	45	59
J23	78	7	77	12	90	64	52	43	91	55	10	28	27	68	62	32	3	58	13	44
J24	1	71	77	26	24	84	41	43	85	46	15	94	61	28	77	61	56	38	13	29
J25	48	62	73	75	25	22	47	13	23	66	88	22	26	48	54	36	81	36	45	45
J26	63	70	57	33	65	71	81	75	11	47	3	50	23	55	12	44	71	35	96	53
J27	17	1	99	23	89	56	55	57	92	90	32	2	20	48	57	90	3	92	96	89
J28	58	97	2	94	83	54	71	56	19	52	72	48	90	86	19	95	77	12	13	26
J29	94	17	86	25	65	12	3	89	28	39	10	20	17	71	41	5	85	28	26	69
J30	82	38	38	99	91	46	94	72	92	5	9	42	12	31	20	64	70	9	82	60
J31	30	58	82	64	18	15	5	81	74	88	76	69	42	80	56	16	7	98	53	70
J32	46	94	76	61	60	92	71	97	73	12	85	42	42	88	54	5	10	49	26	85
J33	85	92	48	79	41	82	95	16	60	56	95	85	68	11	1	94	41	7	7	38
J34	73	10	5	82	32	65	41	6	51	62	90	33	39	45	96	52	87	71	22	25
J35	17	28	2	1	1	53	19	15	49	25	16	31	36	16	33	34	9	22	34	16
J36	18	16	36	98	50	93	99	76	52	92	4	74	78	15	27	78	1	90	18	51
J37	66	43	18	55	90	60	66	58	20	81	1	63	96	39	52	17	18	32	65	69
J38	72	14	14	8	89	19	2	65	39	65	39	88	61	3	57	22	29	95	98	27
J39	53	68	88	63	91	33	79	76	33	99	25	85	47	21	23	72	30	32	35	92
J40	98	87	73	72	95	89	6	73	9	93	31	17	34	96	18	11	20	69	10	26

Cuadro G.16: **Instancia 16**: $n = 40$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	97	12	29	75	94
$\mathbf{J_2}$	34	25	56	57	94
$\mathbf{J_3}$	22	48	19	12	80
$\mathbf{J_4}$	52	89	98	45	88
$\mathbf{J_5}$	98	60	19	97	46
$\mathbf{J_6}$	44	53	35	81	37
$\mathbf{J_7}$	41	79	69	78	22
$\mathbf{J_8}$	15	9	39	42	57
$\mathbf{J_9}$	95	55	21	42	71
$\mathbf{J_{10}}$	52	16	92	96	83
$\mathbf{J_{11}}$	97	33	10	63	66
$\mathbf{J_{12}}$	11	96	73	72	58
$\mathbf{J_{13}}$	72	72	29	91	30
$\mathbf{J_{14}}$	49	30	31	37	72
$\mathbf{J_{15}}$	94	97	64	54	25
$\mathbf{J_{16}}$	15	83	18	36	24
$\mathbf{J_{17}}$	75	33	90	76	21
$\mathbf{J_{18}}$	71	55	89	79	89
$\mathbf{J_{19}}$	57	57	73	34	16
$\mathbf{J_{20}}$	53	25	37	65	84
$\mathbf{J_{21}}$	25	50	99	62	33
$\mathbf{J_{22}}$	19	5	12	17	7
$\mathbf{J_{23}}$	62	33	4	22	40
$\mathbf{J_{24}}$	86	97	99	92	61
$\mathbf{J_{25}}$	69	55	54	55	41
$\mathbf{J_{26}}$	89	83	68	13	28
$\mathbf{J_{27}}$	96	96	8	34	6
$\mathbf{J_{28}}$	71	32	96	35	84
$\mathbf{J_{29}}$	71	11	90	91	73
$\mathbf{J_{30}}$	95	11	84	94	8
$\mathbf{J_{31}}$	14	86	6	52	13
$\mathbf{J_{32}}$	83	70	4	92	24
$\mathbf{J_{33}}$	68	79	41	43	9
$\mathbf{J_{34}}$	32	85	76	59	54
$\mathbf{J_{35}}$	88	93	87	17	93
$\mathbf{J_{36}}$	52	89	71	24	53
$\mathbf{J_{37}}$	56	98	79	57	90
$\mathbf{J_{38}}$	83	84	81	96	37
$\mathbf{J_{39}}$	80	90	78	18	68
$\mathbf{J_{40}}$	59	41	95	37	61
$\mathbf{J_{41}}$	73	80	43	8	45
$\mathbf{J_{42}}$	8	22	52	73	36
$\mathbf{J_{43}}$	49	31	14	25	77
$\mathbf{J_{44}}$	1	92	71	20	95
$\mathbf{J_{45}}$	92	45	59	20	47
$\mathbf{J_{46}}$	38	99	38	8	23
$\mathbf{J_{47}}$	87	43	97	82	75
$\mathbf{J_{48}}$	3	61	60	2	11
$\mathbf{J_{49}}$	57	46	8	15	88
$\mathbf{J_{50}}$	79	21	75	41	59

Cuadro G.17: **Instancia 17**: $n = 50$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	9	64	47	84	43	11	7	73	20	73
$\mathbf{J_2}$	83	2	38	47	6	57	75	65	90	38
$\mathbf{J_3}$	36	68	57	10	37	89	21	96	1	41
$\mathbf{J_4}$	14	26	3	35	18	56	9	54	33	30
$\mathbf{J_5}$	75	70	95	46	22	74	33	48	85	58
$\mathbf{J_6}$	18	54	14	75	48	1	70	52	84	57
$\mathbf{J_7}$	47	49	96	32	10	73	27	93	20	4
$\mathbf{J_8}$	51	64	19	76	7	61	54	39	18	93
$\mathbf{J_9}$	93	84	24	46	52	1	19	3	22	16
$\mathbf{J_{10}}$	26	85	99	19	84	22	34	84	27	62
$\mathbf{J_{11}}$	35	29	50	7	31	86	34	24	64	72
$\mathbf{J_{12}}$	55	54	48	69	79	43	39	76	58	48
$\mathbf{J_{13}}$	90	73	53	2	92	19	48	37	18	65
$\mathbf{J_{14}}$	51	8	24	70	68	44	83	43	23	46
$\mathbf{J_{15}}$	43	26	41	1	35	36	28	59	78	2
$\mathbf{J_{16}}$	7	28	69	65	97	56	26	34	60	26
$\mathbf{J_{17}}$	42	11	7	59	70	39	98	19	35	33
$\mathbf{J_{18}}$	43	71	50	71	74	12	79	98	90	89
$\mathbf{J_{19}}$	86	49	57	89	35	75	21	37	64	73
$\mathbf{J_{20}}$	93	45	27	10	80	29	74	16	82	35
$\mathbf{J_{21}}$	36	86	92	76	5	2	54	70	48	71
$\mathbf{J_{22}}$	67	14	92	59	52	40	58	34	8	59
$\mathbf{J_{23}}$	35	63	23	58	33	48	32	12	48	52
$\mathbf{J_{24}}$	16	52	24	82	97	70	92	29	82	72
$\mathbf{J_{25}}$	43	61	32	91	44	43	88	44	95	15
$\mathbf{J_{26}}$	71	76	30	52	27	39	57	6	1	86
$\mathbf{J_{27}}$	19	69	44	87	25	44	8	96	62	87
$\mathbf{J_{28}}$	60	17	10	71	36	69	97	82	12	22
$\mathbf{J_{29}}$	92	39	50	70	6	9	84	66	38	58
$\mathbf{J_{30}}$	1	14	27	60	65	18	56	38	1	29
$\mathbf{J_{31}}$	27	29	93	14	28	98	8	72	27	31
$\mathbf{J_{32}}$	21	53	86	71	11	59	82	20	73	28
$\mathbf{J_{33}}$	52	75	23	37	28	86	28	76	22	78
$\mathbf{J_{34}}$	80	76	65	29	87	80	20	38	51	12
$\mathbf{J_{35}}$	19	9	65	77	2	24	46	91	45	96
$\mathbf{J_{36}}$	76	20	16	24	36	81	90	81	33	55
$\mathbf{J_{37}}$	17	23	78	79	17	79	49	40	46	76
$\mathbf{J_{38}}$	48	79	75	11	47	87	51	53	73	49
$\mathbf{J_{39}}$	41	66	77	46	82	64	83	56	97	47
$\mathbf{J_{40}}$	34	2	25	76	50	12	87	96	87	26
$\mathbf{J_{41}}$	5	99	54	51	11	99	20	38	39	59
$\mathbf{J_{42}}$	40	48	82	79	60	17	46	55	86	55
$\mathbf{J_{43}}$	46	58	33	24	18	88	81	79	47	13
$\mathbf{J_{44}}$	15	77	36	85	6	85	54	9	87	40
$\mathbf{J_{45}}$	55	90	17	35	46	5	11	52	41	46
$\mathbf{J_{46}}$	42	51	48	56	86	87	57	53	81	21
$\mathbf{J_{47}}$	29	85	40	43	15	10	13	24	49	41
$\mathbf{J_{48}}$	7	60	82	39	3	90	73	33	19	77
$\mathbf{J_{49}}$	38	86	25	3	29	57	2	13	92	10
$\mathbf{J_{50}}$	87	10	30	88	79	78	60	42	62	88

Cuadro G.18: **Instancia 18**: $n = 50$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	74	79	9	85	14	74	26	87	85	84	82	30	17	36	22
$\mathbf{J_2}$	48	4	1	23	20	14	18	8	53	33	77	44	70	56	58
$\mathbf{J_3}$	43	61	7	54	67	98	19	24	89	6	50	52	76	28	90
$\mathbf{J_4}$	5	81	72	69	21	63	84	11	67	85	89	68	47	45	21
$\mathbf{J_5}$	31	58	38	12	6	11	70	24	3	73	63	23	76	4	7
$\mathbf{J_6}$	50	7	74	76	34	24	5	4	45	5	64	63	91	91	5
$\mathbf{J_7}$	38	92	28	25	44	1	94	97	50	49	94	99	19	38	30
$\mathbf{J_8}$	19	46	97	58	43	89	85	19	45	94	4	87	21	72	59
$\mathbf{J_9}$	74	47	93	77	80	65	81	57	2	3	31	39	4	93	97
$\mathbf{J_{10}}$	97	39	63	13	99	69	75	25	80	14	74	85	38	38	57
$\mathbf{J_{11}}$	16	39	45	59	27	88	56	49	86	10	66	52	33	34	26
$\mathbf{J_{12}}$	60	13	2	85	90	69	10	99	97	6	76	75	63	19	3
$\mathbf{J_{13}}$	15	12	64	32	93	16	99	69	54	11	19	90	33	12	73
$\mathbf{J_{14}}$	53	35	19	47	99	12	46	25	8	38	1	75	30	77	23
$\mathbf{J_{15}}$	34	28	62	35	62	89	62	81	18	89	6	48	57	34	56
$\mathbf{J_{16}}$	86	63	89	91	2	52	94	16	63	34	1	10	41	98	99
$\mathbf{J_{17}}$	3	2	65	22	4	92	22	14	56	9	33	43	76	24	3
$\mathbf{J_{18}}$	74	96	20	9	15	36	57	52	40	55	63	94	24	70	38
$\mathbf{J_{19}}$	60	47	41	88	41	68	49	63	27	60	36	39	41	53	64
$\mathbf{J_{20}}$	65	73	47	24	94	12	76	95	45	41	42	45	59	52	36
$\mathbf{J_{21}}$	9	74	65	33	90	11	86	29	89	41	31	35	20	4	85
$\mathbf{J_{22}}$	67	99	61	10	90	14	34	70	98	75	42	82	58	58	50
$\mathbf{J_{23}}$	96	25	48	97	39	22	58	34	22	40	98	32	58	96	84
$\mathbf{J_{24}}$	22	54	62	46	68	72	44	51	96	76	81	68	36	11	95
$\mathbf{J_{25}}$	24	27	62	5	42	20	27	78	55	65	62	16	32	58	3
$\mathbf{J_{26}}$	53	64	5	78	39	1	4	41	48	30	85	19	2	45	32
$\mathbf{J_{27}}$	29	15	31	67	31	14	78	17	93	70	6	49	18	47	32
$\mathbf{J_{28}}$	56	31	65	51	48	38	26	90	50	19	47	49	37	80	14
$\mathbf{J_{29}}$	62	46	15	27	25	39	27	19	77	28	27	65	37	67	94
$\mathbf{J_{30}}$	1	18	73	75	48	94	40	55	77	78	55	25	83	18	32
$\mathbf{J_{31}}$	39	47	59	10	11	97	11	98	23	15	93	57	33	33	86
$\mathbf{J_{32}}$	45	94	20	96	74	42	62	4	56	98	75	59	29	3	4
$\mathbf{J_{33}}$	51	85	69	10	58	70	33	39	50	91	34	20	12	72	38
$\mathbf{J_{34}}$	45	99	42	13	25	56	36	8	26	43	63	83	27	60	46
$\mathbf{J_{35}}$	13	54	72	55	40	84	62	85	68	19	61	54	23	81	51
$\mathbf{J_{36}}$	6	89	74	29	61	43	12	67	52	17	5	36	16	69	68
$\mathbf{J_{37}}$	41	47	38	85	15	36	16	31	59	31	72	55	22	10	96
$\mathbf{J_{38}}$	70	63	52	54	96	34	40	70	55	84	26	87	19	64	55
$\mathbf{J_{39}}$	59	59	96	33	80	46	17	62	56	85	20	92	70	17	68
$\mathbf{J_{40}}$	70	7	88	34	73	21	87	84	60	78	4	42	78	4	37
$\mathbf{J_{41}}$	66	15	5	88	51	72	63	88	25	8	17	62	95	72	60
$\mathbf{J_{42}}$	26	91	38	52	60	78	92	97	24	56	38	45	83	60	44
$\mathbf{J_{43}}$	25	82	39	18	1	68	64	96	31	66	95	84	48	50	23
$\mathbf{J_{44}}$	87	10	53	43	6	74	38	48	96	12	85	32	30	43	98
$\mathbf{J_{45}}$	42	69	76	27	38	72	87	42	27	98	45	35	90	45	95
$\mathbf{J_{46}}$	16	20	67	54	25	32	51	63	2	26	40	13	53	5	34
$\mathbf{J_{47}}$	40	28	71	18	56	97	74	80	52	75	64	40	90	51	81
$\mathbf{J_{48}}$	79	99	1	88	56	3	16	24	46	51	57	27	38	68	10
$\mathbf{J_{49}}$	73	85	2	28	16	64	33	69	19	29	24	63	33	43	70
$\mathbf{J_{50}}$	72	45	19	97	26	13	11	32	54	43	46	18	35	1	71

Cuadro G.19: **Instancia 19**: $n = 50$ trabajos, $m = 15$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_1	29	94	73	32	89	18	18	33	67	19	49	7	87	42	86	62	50	30	34	92
J_2	61	81	72	69	12	12	3	25	30	41	73	97	4	85	98	3	62	44	57	36
J_3	29	91	33	56	21	27	30	73	35	44	79	49	53	10	15	93	60	76	85	4
J_4	45	60	37	90	43	63	78	67	28	26	12	26	75	22	91	2	40	55	46	10
J_5	27	1	82	9	51	3	36	15	13	40	65	83	35	1	77	22	29	27	51	5
J_6	4	34	84	1	29	17	89	48	98	75	8	14	77	48	18	40	29	16	64	59
J_7	37	31	18	33	81	70	68	40	33	22	79	21	72	63	78	65	54	74	65	79
J_8	61	4	37	86	28	78	89	42	26	30	28	62	41	71	97	4	83	45	33	48
J_9	16	38	54	58	76	28	98	32	50	92	35	64	9	28	8	86	9	31	26	49
J_{10}	43	43	58	63	68	79	34	32	53	73	45	10	27	13	94	91	27	59	51	22
J_{11}	59	60	91	55	35	5	46	31	39	88	82	26	48	73	87	14	7	69	32	34
J_{12}	16	72	31	44	6	27	42	27	96	14	44	63	61	65	5	85	47	63	29	91
J_{13}	27	80	43	71	70	61	51	71	80	50	74	12	12	30	71	30	79	7	69	6
J_{14}	23	65	88	91	74	89	20	72	69	2	85	3	69	81	14	18	5	65	82	98
J_{15}	51	27	21	48	96	6	92	4	66	24	72	1	8	8	94	53	56	9	59	15
J_{16}	36	87	96	53	53	11	33	28	9	65	19	71	22	97	7	9	36	27	78	58
J_{17}	70	96	74	16	16	56	75	97	6	95	23	95	97	60	93	90	54	33	59	98
J_{18}	7	63	1	69	35	59	45	42	15	93	60	46	87	86	27	91	76	48	46	86
J_{19}	56	19	85	81	19	29	60	20	37	58	84	61	83	77	67	80	70	99	24	81
J_{20}	27	75	24	74	2	2	6	97	85	37	21	85	84	40	99	19	79	73	31	6

Cuadro G.20: **Instancia 20 (parte 1)**: $n = 50$ trabajos, $m = 20$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_{21}	54	42	21	77	90	50	59	2	48	7	9	40	85	12	4	13	85	95	14	39
J_{22}	97	99	20	71	4	18	93	39	69	44	94	67	49	85	12	43	42	22	38	40
J_{23}	26	59	41	17	13	12	87	84	3	66	81	51	37	32	60	26	68	6	73	40
J_{24}	96	60	87	85	53	35	24	86	18	35	98	1	3	36	41	90	48	57	15	35
J_{25}	3	58	65	6	88	51	63	41	69	53	20	68	81	13	99	95	88	81	53	45
J_{26}	60	30	36	10	15	6	2	25	45	96	75	18	28	59	49	4	2	93	51	67
J_{27}	39	4	91	52	59	61	73	91	63	62	21	81	76	50	83	92	38	37	96	78
J_{28}	68	53	13	19	6	5	40	80	90	18	48	56	47	96	24	72	77	39	1	77
J_{29}	10	30	8	75	85	82	28	91	68	1	47	21	19	61	27	28	4	96	72	44
J_{30}	31	75	46	27	88	17	15	66	18	75	75	45	74	70	4	4	62	69	52	43
J_{31}	38	34	78	86	23	87	77	80	68	73	66	86	63	2	17	23	80	14	57	42
J_{32}	1	1	82	48	60	69	42	2	1	26	93	62	43	31	47	63	87	55	28	22
J_{33}	21	8	44	28	95	54	37	63	83	24	38	51	52	73	64	31	88	71	41	9
J_{34}	86	6	22	27	63	28	29	64	72	4	69	50	37	16	72	45	4	45	60	51
J_{35}	9	30	55	21	97	64	72	94	41	15	68	91	34	31	24	19	86	3	95	82
J_{36}	22	88	66	14	49	3	66	44	94	53	59	94	71	53	3	56	48	63	31	86
J_{37}	53	22	34	42	94	84	10	18	10	69	12	84	61	78	50	14	51	62	84	54
J_{38}	11	19	29	99	95	55	68	39	95	69	31	11	21	8	95	79	50	16	8	9
J_{39}	35	82	39	15	2	57	23	3	89	78	29	33	24	47	83	74	15	33	37	25
J_{40}	16	97	37	19	97	75	52	48	14	37	67	75	74	38	35	94	33	21	60	83
J_{41}	16	97	37	19	97	75	52	48	14	37	67	75	74	38	35	94	33	21	60	83
J_{42}	72	24	51	64	42	8	56	32	69	27	88	50	68	78	81	15	93	16	53	22
J_{43}	36	24	58	4	46	67	51	21	31	82	4	13	64	58	44	58	3	34	4	82
J_{44}	7	13	95	24	92	54	68	57	77	11	89	98	94	18	70	82	78	43	34	67
J_{45}	53	65	68	97	87	34	49	51	79	24	1	39	82	9	30	35	13	82	71	52
J_{46}	76	30	98	3	6	22	16	1	4	75	27	85	39	10	38	95	24	88	50	8
J_{47}	76	39	42	19	60	69	57	46	36	35	36	59	74	32	64	87	14	7	23	49
J_{48}	67	16	67	3	14	69	66	18	10	53	77	18	31	5	44	38	12	88	36	97
J_{49}	53	96	76	89	14	65	14	30	97	5	44	21	18	52	89	21	80	11	79	25
J_{50}	43	43	8	97	14	12	43	15	16	2	24	7	34	66	77	6	94	67	21	35
J_{51}	70	11	51	10	14	13	21	22	31	17	72	31	75	66	62	89	28	12	94	68

Cuadro G.21: **Instancia 20 (parte 2)**: $n = 50$ trabajos, $m = 20$ máquinas

Apéndice H

Resultados numéricos calibración de parámetros Algoritmo GRASP

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1730.6	758	3931	827
2	4328.5	1309	8479	4204
3	6606.1	1600	10736	9151
4	11350.9	2218	15486	19393
5	5170.6	1438	14078	1240
6	9127.9	1942	21587	6250
7	12703	2350	28081	11129
8	17400.9	2742	34907	19440
9	10718.2	2131	31045	1841
10	14081.9	2237	38553	5404
11	20852.3	2906	52188	13445
12	25989.1	3382	59394	22727
13	16744.4	2702	50493	1719
14	22156.5	2895	64930	5065
15	31468.9	3655	85798	14225
16	37334.7	4119	92904	26053
17	26652.2	3500	81716	2458
18	30261.2	3521	90989	5187
19	38721.2	4103	110122	13478
20	47025.3	4389	129741	21158

Cuadro H.1: **Opción 1**: $\alpha = 0$, $n_{iter} = 0$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1940.6	755	4453	1009
2	4717.8	1368	9279	4623
3	6947.5	1636	11249	9728
4	11380.8	2214	15655	19329
5	5335.1	1433	14578	1295
6	9204.4	1942	22159	5933
7	13500.2	2357	30458	11400
8	17154.5	2687	36017	17582
9	10774.4	2072	31680	1472
10	14491.8	2337	39333	5857
11	21318.1	2932	54235	12916
12	24936.1	3220	59553	19274
13	16837.6	2665	50964	1608
14	23660.1	3198	67595	7008
15	31173.8	3593	86310	12812
16	38591.4	4023	97772	25502
17	28236.7	3610	86489	2820
18	32981.7	3759	97040	7887
19	39698.8	3997	113600	13400
20	49199.4	4548	135042	22892

Cuadro H.2: **Opción 2**: $\alpha = 0{,}25$, $n_{iter} = 0$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	2005.7	746	4766	925
2	4143.9	1242	8226	3931
3	6659.6	1577	11092	9004
4	11059.6	2134	15605	18415
5	5630.9	1394	15778	1133
6	9078.7	1792	22678	5195
7	14165.9	2396	32661	11364
8	17421.4	2587	37384	17238
9	11799.3	2214	34442	1937
10	15604	2464	42223	6505
11	23031.7	3097	57663	14980
12	27322.4	3395	63817	22731
13	18772.3	2695	57013	1968
14	25585.7	3281	74349	6562
15	33138.9	3840	90365	14978
16	40656.8	4220	101987	27909
17	28704	3618	87753	3103
18	34048.5	3654	102444	6179
19	42780.9	4272	121684	15223
20	53755	4765	146603	26227

Cuadro H.3: **Opción 3**: $\alpha = 0{,}5$, $n_{iter} = 0$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	2000.2	757	4655	1003
2	4243.1	1271	8518	3931
3	7596.1	1702	13398	9653
4	11051.7	2199	15473	18434
5	5961.9	1473	16692	1217
6	9621.3	1815	24441	5210
7	14168	2456	32077	11875
8	18776.8	2827	39543	19277
9	11602.1	2000	34598	1409
10	16729	2503	45846	6580
11	22710.3	3018	58851	12826
12	30269.2	3544	70096	26076
13	19508.3	2621	59866	1667
14	25892.6	3116	74664	7490
15	32636.5	3853	90078	13573
16	40605.3	4014	105207	24792
17	30797.7	3564	95516	2391
18	37408.9	3673	112099	7700
19	43100.5	4018	124289	14022
20	53548.5	4725	148795	23400

Cuadro H.4: **Opción 4**: $\alpha = 0{,}75$, $n_{iter} = 0$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	2050.4	758	4814	1010
2	4354.7	1175	9243	3706
3	7678.6	1753	13442	9816
4	11001.1	2083	16310	17583
5	5977.9	1450	16820	1173
6	9752.1	1842	25053	4998
7	14195.1	2376	33452	10697
8	19380.1	2842	40875	19936
9	12337.4	2216	36126	2044
10	17339.5	2422	47812	6757
11	23316.1	3124	60355	13200
12	26870.8	3211	65324	19964
13	19370.2	2785	58788	2066
14	26851.7	3203	78557	6678
15	34384.7	3809	95946	13591
16	41135.5	4144	106716	24877
17	30431.5	3463	94507	2314
18	37332.4	3715	112452	7036
19	44892.8	4241	128517	15471
20	55262.4	4845	152887	24861

Cuadro H.5: **Opción 5**: $\alpha = 1$, $n_{iter} = 0$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1716.2	707	3983	795
2	3995.1	1170	8385	3372
3	6275.5	1444	10518	8475
4	9784.8	1914	15340	14724
5	4922.7	1290	14013	676
6	8349.9	1764	20682	4799
7	11984.5	2164	28127	8936
8	15503.4	2472	33342	15040
9	9733.1	1832	29159	842
10	13804.3	2143	38673	4484
11	19083.2	2591	50023	10133
12	23460.6	2892	57250	17096
13	15528.8	2396	47421	1147
14	20762.1	2799	61283	4192
15	28540.9	3253	80785	10014
16	34528.5	3540	90814	19561
17	24290.4	3117	75362	1450
18	29091.5	3278	88796	3805
19	36698	3542	108273	9331
20	45068	4130	126998	17722

Cuadro H.6: **Opción 6**: $\alpha = 0$, $n_{iter} = 50$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1729.7	758	3928	827
2	3937.3	1144	8163	3436
3	6260.4	1464	10722	8194
4	9742.1	1952	14335	15536
5	4809.1	1273	13616	717
6	8330.3	1760	20634	4787
7	11809.2	2091	27906	8670
8	15558.9	2496	34301	14234
9	9958.1	1853	29781	942
10	13649.6	2183	38156	4432
11	18865.9	2620	49745	9648
12	23230.6	2920	57535	16007
13	15799	2431	48236	1186
14	20748.7	2704	61275	4282
15	28832.5	3289	81561	10162
16	35308.7	3647	92501	20332
17	24347	3119	75567	1431
18	28841.1	3219	87978	3867
19	36477.9	3669	107055	9646
20	45442.4	4049	129383	16693

Cuadro H.7: **Opción 7**: $\alpha = 0{,}25$, $n_{iter} = 50$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1708.6	745	3770	932
2	4007.7	1146	8378	3453
3	6224.2	1405	10837	8037
4	9742.1	1952	14335	15536
5	4871	1271	13812	730
6	8428.8	1794	20919	4785
7	11955.1	2080	28557	8520
8	15807.1	2455	34058	15359
9	9693	1875	28856	954
10	13756.4	2141	38391	4609
11	19212.6	2589	50864	9726
12	23797.5	2937	58883	16526
13	15561.5	2396	47530	1147
14	20482.7	2717	60242	4411
15	28943.9	3296	82094	9991
16	35003	3569	92721	19197
17	24127.7	3119	74837	1430
18	29604.8	3218	90381	4011
19	36534.9	3657	107274	9633
20	46001.9	4097	130054	17823

Cuadro H.8: **Opción 8**: $\alpha = 0{,}5$, $n_{iter} = 50$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1713.4	745	3786	932
2	3960.7	1186	8061	3560
3	6267.3	1464	10612	8327
4	9814.7	1943	15919	14206
5	4796	1286	13554	718
6	8314.6	1771	20461	4893
7	11848	2098	27395	9301
8	16036.7	2489	35112	15025
9	9719.7	1851	29059	872
10	13550.1	2184	37876	4379
11	19461.3	2586	51832	9591
12	23644.2	3015	57613	17181
13	15449.3	2420	47107	1164
14	20931.7	2782	61788	4275
15	28552	3295	81392	9388
16	35918.2	3595	95386	19548
17	24155.3	3101	75040	1343
18	29287.7	3254	89319	3968
19	36363.3	3582	106904	9531
20	46964.3	4217	132528	18397

Cuadro H.9: **Opción 9**: $\alpha = 0{,}75$, $n_{iter} = 50$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1712.2	745	3782	932
2	3922.8	1128	8278	3294
3	6273.4	1525	10780	8098
4	9800.5	1918	15274	14837
5	4737.3	1272	13574	521
6	8210.2	1651	20824	4342
7	11993.9	2108	28350	8819
8	15924.6	2496	34149	15605
9	9885.1	1843	29545	948
10	13475.5	2134	37987	4086
11	19046.7	2703	50277	9608
12	23116.6	2887	57029	16177
13	15529.4	2396	47423	1147
14	20873.2	2806	61840	3996
15	28265.3	3218	80435	9492
16	35156.7	3585	93420	18989
17	24391.1	3101	75826	1343
18	29355.2	3236	89533	4003
19	37018.2	3591	109041	9565
20	45765.9	4128	130155	16894

Cuadro H.10: **Opción 10**: $\alpha = 1$, $n_{iter} = 50$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1706.5	745	3763	932
2	4033.6	1159	8259	3641
3	6275.5	1444	10518	8475
4	9885.8	1943	15383	14979
5	4793.5	1279	13572	701
6	8230.4	1751	20303	4797
7	11689.6	2095	27407	8765
8	15423.1	2419	33540	14645
9	10221.5	1904	30451	1082
10	13542.4	2101	37989	4351
11	19018.8	2619	49876	10028
12	23117.7	2964	57212	15895
13	15662.3	2420	47817	1164
14	20828.4	2790	60980	4728
15	28841.8	3343	81908	9774
16	34774.9	3499	92133	19118
17	24206.8	3073	75121	1471
18	29068.7	3257	88694	3859
19	36266.8	3631	106210	9838
20	45401.8	3997	128932	17078

Cuadro H.11: **Opción 11**: $\alpha = 0$, $n_{iter} = 100$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1735.7	743	3871	924
2	4027.8	1194	8299	3535
3	6233.3	1493	10293	8494
4	9741.1	1951	15278	14591
5	4828.4	1265	13701	707
6	8256.7	1699	20809	4448
7	11671.7	2099	27864	8243
8	15720	2457	34060	15064
9	9875.3	1880	29371	1040
10	13544.6	2126	38152	4162
11	19140.3	2589	50925	9424
12	23104.2	2904	57184	15958
13	15582.5	2396	47600	1147
14	20934.2	2690	62028	4166
15	29185	3340	82767	10063
16	35262	3585	93143	19617
17	24272.5	3085	75312	1483
18	29013.7	3253	88447	3928
19	36674.9	3626	107738	9677
20	45236.7	4062	128311	17062

Cuadro H.12: **Opción 12**: $\alpha = 0{,}25$, $n_{iter} = 100$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1712.3	758	3870	827
2	4095.8	1238	8227	3775
3	6098.5	1450	10310	8085
4	9833.9	2033	14595	15474
5	4861.2	1311	13693	763
6	8192.9	1691	20546	4509
7	11816	2117	27418	9146
8	15627.4	2419	34054	14812
9	9920.5	1828	29810	821
10	13606.9	2044	38470	4161
11	19455.5	2678	51546	9735
12	23673.5	2987	58559	16370
13	15854.7	2421	48455	1166
14	20855.4	2799	61063	4723
15	29505.6	3330	81572	12340
16	34343.2	3400	89241	20703
17	24170.6	2999	74890	1680
18	28195	3040	85900	4030
19	37516.8	3525	110789	9567
20	45123	4050	127890	17120

Cuadro H.13: **Opción 13**: $\alpha = 0{,}5$, $n_{iter} = 100$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1755.7	730	3989	890
2	4040.4	1245	8340	3468
3	6428.4	1398	11004	8560
4	9824.4	2040	15350	14678
5	4523.6	1349	12590	690
6	8524	1720	21560	4560
7	11204	2300	25980	8300
8	15736.7	2450	33989	15200
9	9810	1785	29340	980
10	13933	2230	39240	4230
11	19148.6	2456	51204	9350
12	23118.2	3002	57049	16009
13	15888	2430	48500	1220
14	20860.3	2560	61956	4165
15	29243.9	3245	83103	10050
16	35157	3450	93050	19540
17	24241.8	3090	75256	1430
18	28918	3289	88005	4003
19	36723.6	3564	108120	9540
20	45156.7	3976	128234	16987

Cuadro H.14: **Opción 14**: $\alpha = 0{,}75$, $n_{iter} = 100$

Instancia	Función objetivo	Makespan	Flow Time	Idle Time
1	1719.4	745	3806	932
2	4037.8	1192	8413	3457
3	6118.4	1505	10243	8145
4	10096.9	1996	15014	15981
5	4791.3	1296	13492	751
6	8348.8	1681	21127	4461
7	11866.9	2095	28267	8496
8	16011.4	2455	34735	15363
9	9910.7	1883	29416	1109
10	13747.8	2067	38996	4074
11	19589	2555	52259	9631
12	23472.4	2962	58526	15766
13	15620.4	2418	47684	1160
14	20811.9	2691	61923	3862
15	28863.1	3214	82547	9378
16	35646.7	3631	94277	19704
17	24263.4	3126	75229	1481
18	29152.6	3217	89111	3775
19	36275.1	3546	106984	9205
20	45410.4	4047	129048	16924

Cuadro H.15: **Opción 15**: $\alpha = 1$, $n_{iter} = 100$

Apéndice I

Código Python *Grid Search*

```
#GRID SEARCH
# Cargar los datos desde el archivo Excel
data = pd.read_excel('resultados.xlsx')

# Extraer los valores de las columnas
iteraciones = data['niter']
alfa = data['alfa']
func_obj = data['min']

# Crear un gr fico de cuadros de colores
fig, ax = plt.subplots()
im = ax.imshow(func_obj.values.reshape(len(np.unique(iteraciones)), len(np.
    unique(alfa))), cmap='RdYlGn_r', vmin=min(func_obj), vmax=max(func_obj))

# A adir una barra de color
fig.colorbar(im)

# Etiquetar los ejes
ax.set_xticks(range(len(np.unique(alfa))))
ax.set_yticks(range(len(np.unique(iteraciones))))
ax.set_xticklabels(np.unique(alfa))
ax.set_yticklabels(np.unique(iteraciones))
ax.set_xlabel('$\\alpha$') # Cambiar el nombre del eje x
ax.set_ylabel('$n_{iter}$') # Cambiar el nombre del eje y

# Invertir el eje y
ax.invert_yaxis()

# Mostrar el gr fico
plt.show()
```

```
import matplotlib.pyplot as plt

# Cargar los datos desde el archivo Excel
```

```
4 data = pd.read_excel('limite.xlsx')
5
6 # Extraer los valores de las columnas
7 iteraciones = data['niter']
8 func_obj = data['fo']
9
10 # Crear un gr fico de puntos unidos por una l nea
11 fig, ax = plt.subplots()
12 ax.plot(iteraciones, func_obj, '-o')
13
14 # Etiquetar los ejes
15 ax.set_xlabel('$n_{iter}$')
16 ax.set_ylabel('Funci n objetivo promedio')
17
18 # Mostrar el gr fico
19 plt.show()
```

Apéndice J

Instancias comparación de Algoritmos

Claro, aquí está la tabla actualizada con los números que proporcionaste:

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	79	67	10	48	52
$\mathbf{J_2}$	40	40	57	21	54
$\mathbf{J_3}$	48	93	49	11	79
$\mathbf{J_4}$	16	23	19	2	38
$\mathbf{J_5}$	38	90	57	73	3
$\mathbf{J_6}$	76	13	99	98	55
$\mathbf{J_7}$	73	85	40	20	85
$\mathbf{J_8}$	34	6	27	53	21
$\mathbf{J_9}$	38	6	35	28	44
$\mathbf{J_{10}}$	32	11	11	34	27

Cuadro J.1: **Instancia 1**: $n = 10$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	33	70	23	76	56	74	2	39	8	81
$\mathbf{J_2}$	95	68	2	64	57	79	73	39	2	79
$\mathbf{J_3}$	9	5	85	37	96	48	23	2	99	10
$\mathbf{J_4}$	17	57	44	43	38	76	50	6	89	3
$\mathbf{J_5}$	76	64	17	82	29	9	64	20	77	94
$\mathbf{J_6}$	95	29	65	69	13	9	12	52	45	57
$\mathbf{J_7}$	46	63	2	5	14	37	4	42	50	46
$\mathbf{J_8}$	40	57	10	38	5	11	5	57	25	25
$\mathbf{J_9}$	69	91	79	56	81	85	58	71	88	98
$\mathbf{J_{10}}$	39	64	21	92	46	65	4	99	45	42

Cuadro J.2: **Instancia 2**: $n = 10$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	27	7	23	26	38	28	8	2	18	98	4	37	19	5	35
$\mathbf{J_2}$	53	69	49	91	5	84	28	35	46	66	47	58	10	18	19
$\mathbf{J_3}$	46	91	31	1	13	92	18	8	98	67	11	78	42	66	30
$\mathbf{J_4}$	83	80	96	4	20	32	18	4	5	67	44	4	92	72	39
$\mathbf{J_5}$	92	56	50	10	91	11	40	41	69	77	27	29	41	11	69
$\mathbf{J_6}$	91	91	35	90	86	74	74	93	50	77	54	91	32	32	21
$\mathbf{J_7}$	80	10	32	10	80	94	83	45	34	52	90	69	31	22	66
$\mathbf{J_8}$	19	94	97	4	99	55	3	61	41	4	70	44	15	82	17
$\mathbf{J_9}$	86	52	63	68	53	1	34	76	38	1	68	35	13	77	72
$\mathbf{J_{10}}$	80	51	66	21	42	50	30	32	73	97	35	33	77	77	87

Cuadro J.3: **Instancia 3**: $n = 10$ trabajos, $m = 15$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$	$\mathbf{M_{16}}$	$\mathbf{M_{17}}$	$\mathbf{M_{18}}$	$\mathbf{M_{19}}$	$\mathbf{M_{20}}$
$\mathbf{J_1}$	59	52	86	79	15	29	24	55	6	8	72	99	3	47	64	40	77	32	63	88
$\mathbf{J_2}$	92	96	92	85	26	17	73	41	44	52	51	50	2	82	38	53	7	59	99	66
$\mathbf{J_3}$	16	77	74	13	16	33	91	11	26	23	27	12	24	6	29	95	62	19	83	57
$\mathbf{J_4}$	62	37	5	55	73	4	84	89	36	76	19	43	14	70	5	5	45	84	77	37
$\mathbf{J_5}$	57	43	20	89	39	7	79	45	45	4	57	57	8	11	75	23	58	77	19	89
$\mathbf{J_6}$	93	69	15	95	30	77	96	89	45	12	68	85	85	99	59	35	95	21	38	43
$\mathbf{J_7}$	9	80	29	41	75	90	43	95	49	5	31	31	50	42	92	83	54	75	32	9
$\mathbf{J_8}$	4	92	29	21	31	36	54	39	33	83	69	51	45	46	74	39	94	60	10	15
$\mathbf{J_9}$	47	24	2	67	64	11	26	18	31	34	36	27	40	66	47	13	23	36	12	68
$\mathbf{J_{10}}$	73	45	8	81	4	88	20	5	28	65	95	25	59	14	11	29	72	9	62	62

Cuadro J.4: **Instancia 4**: $n = 10$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	11	95	26	46	20
$\mathbf{J_2}$	47	40	98	98	16
$\mathbf{J_3}$	35	84	40	52	98
$\mathbf{J_4}$	94	28	20	31	99
$\mathbf{J_5}$	59	57	88	89	9
$\mathbf{J_6}$	91	6	13	48	73
$\mathbf{J_7}$	11	55	14	13	50
$\mathbf{J_8}$	94	87	33	90	14
$\mathbf{J_9}$	40	65	62	62	41
$\mathbf{J_{10}}$	70	67	91	49	37
$\mathbf{J_{11}}$	69	27	16	8	17
$\mathbf{J_{12}}$	20	89	32	95	9
$\mathbf{J_{13}}$	93	40	99	95	2
$\mathbf{J_{14}}$	91	53	91	59	38
$\mathbf{J_{15}}$	49	3	3	18	64
$\mathbf{J_{16}}$	53	92	47	81	79
$\mathbf{J_{17}}$	48	5	18	30	56
$\mathbf{J_{18}}$	97	46	52	88	75
$\mathbf{J_{19}}$	57	62	56	20	99
$\mathbf{J_{20}}$	21	87	27	95	71

Cuadro J.5: **Instancia 5**: $n = 20$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	88	49	50	61	60	19	82	83	90	67
$\mathbf{J_2}$	33	39	58	44	58	67	24	40	72	19
$\mathbf{J_3}$	5	68	92	97	92	77	41	90	72	53
$\mathbf{J_4}$	7	97	89	5	2	25	89	30	58	99
$\mathbf{J_5}$	3	16	56	18	69	21	79	41	11	27
$\mathbf{J_6}$	27	92	12	77	48	22	51	36	63	54
$\mathbf{J_7}$	97	20	90	14	76	98	91	9	51	81
$\mathbf{J_8}$	15	24	88	97	84	29	95	66	74	25
$\mathbf{J_9}$	54	22	65	82	83	4	64	30	6	60
$\mathbf{J_{10}}$	49	14	9	17	52	29	14	91	8	85
$\mathbf{J_{11}}$	2	90	28	79	53	83	67	67	7	55
$\mathbf{J_{12}}$	3	59	42	99	42	95	36	66	25	65
$\mathbf{J_{13}}$	60	12	91	74	8	94	31	79	62	20
$\mathbf{J_{14}}$	20	43	68	27	34	80	51	74	91	93
$\mathbf{J_{15}}$	89	64	34	22	54	9	92	51	55	35
$\mathbf{J_{16}}$	66	60	65	77	17	2	17	7	74	37
$\mathbf{J_{17}}$	75	95	80	83	46	12	26	28	78	95
$\mathbf{J_{18}}$	72	32	53	71	45	32	71	76	37	83
$\mathbf{J_{19}}$	98	32	86	36	17	70	82	27	80	6
$\mathbf{J_{20}}$	38	17	4	10	25	23	3	18	54	13

Cuadro J.6: **Instancia 6**: $n = 20$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	95	35	46	65	38	7	76	90	74	2	47	22	44	4	60
$\mathbf{J_2}$	16	78	67	29	48	10	72	67	48	79	13	22	77	91	55
$\mathbf{J_3}$	54	68	47	49	84	61	55	79	47	85	7	86	8	92	68
$\mathbf{J_4}$	17	62	34	17	45	97	29	27	8	4	78	88	30	45	21
$\mathbf{J_5}$	9	34	49	21	29	63	69	53	61	54	10	17	21	63	48
$\mathbf{J_6}$	86	17	91	74	97	26	96	31	8	46	78	38	96	31	66
$\mathbf{J_7}$	25	84	33	56	6	85	13	71	70	30	90	39	82	14	46
$\mathbf{J_8}$	19	94	68	1	21	41	54	31	87	46	14	92	47	73	57
$\mathbf{J_9}$	64	51	9	48	41	46	25	12	27	61	5	8	10	4	3
$\mathbf{J_{10}}$	6	20	80	68	79	14	8	72	3	17	73	6	2	25	43
$\mathbf{J_{11}}$	34	17	56	57	41	64	93	39	80	99	60	7	33	28	18
$\mathbf{J_{12}}$	81	30	25	89	13	83	38	32	73	52	94	24	97	19	51
$\mathbf{J_{13}}$	33	56	64	33	81	94	58	66	33	87	14	99	57	72	61
$\mathbf{J_{14}}$	61	48	71	44	5	7	21	53	37	15	49	34	32	52	94
$\mathbf{J_{15}}$	40	69	41	42	27	63	22	13	96	59	84	56	39	15	21
$\mathbf{J_{16}}$	10	41	2	64	36	43	79	33	48	83	49	4	82	40	68
$\mathbf{J_{17}}$	57	86	68	90	6	20	9	69	67	99	85	29	57	23	40
$\mathbf{J_{18}}$	2	12	61	68	71	64	2	48	85	39	78	76	20	23	35
$\mathbf{J_{19}}$	63	10	55	83	47	74	15	73	13	13	17	82	47	36	41
$\mathbf{J_{20}}$	83	69	65	50	58	87	20	79	58	98	60	99	34	29	74

Cuadro J.7: **Instancia 7**: $n = 20$ trabajos, $m = 15$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_1	16	89	28	27	99	72	26	63	74	8	59	25	35	44	1	1	17	51	53	59
J_2	10	47	16	77	5	66	17	13	1	30	19	93	30	44	1	53	4	39	87	50
J_3	67	13	82	34	20	56	12	62	46	66	1	72	51	70	31	55	17	25	48	9
J_4	97	69	30	37	35	22	19	48	66	76	89	56	39	26	12	83	78	11	98	40
J_5	45	91	94	98	95	74	11	16	40	45	18	53	38	86	69	9	89	97	91	50
J_6	19	29	83	24	42	92	9	78	79	38	59	12	47	79	37	50	36	21	87	11
J_7	76	26	59	3	27	7	94	71	52	54	69	93	68	21	34	67	78	17	28	72
J_8	19	5	55	78	87	91	74	34	45	51	10	64	80	23	94	54	99	84	23	12
J_9	82	80	39	8	1	60	21	69	4	57	10	59	18	7	15	58	3	97	88	51
J_{10}	68	89	37	68	57	53	35	54	16	10	38	33	44	75	31	17	75	40	37	83
J_{11}	24	35	86	84	26	77	27	92	29	73	2	98	31	43	46	89	63	88	81	91
J_{12}	62	6	89	61	99	45	9	42	29	32	88	35	96	26	77	33	18	77	47	41
J_{13}	8	5	74	70	34	4	25	60	7	58	23	1	53	26	86	41	15	2	79	98
J_{14}	77	93	98	17	31	63	92	39	95	5	93	85	79	83	7	53	53	43	78	84
J_{15}	50	51	38	63	28	58	6	39	55	67	87	95	9	65	3	93	87	40	28	80
J_{16}	86	87	49	26	85	10	39	32	43	18	23	39	3	15	94	5	53	23	87	42
J_{17}	91	72	16	39	92	69	48	2	80	6	73	80	47	88	60	83	70	92	48	82
J_{18}	33	67	89	67	29	51	49	96	22	73	34	55	91	52	87	32	69	10	1	45
J_{19}	72	68	3	8	57	1	39	99	81	47	52	81	31	12	25	86	43	22	49	54
J_{20}	54	95	15	7	27	52	65	69	62	75	38	1	79	93	83	77	26	53	58	37

Cuadro J.8: **Instancia 8**: $n = 20$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	70	13	34	19	88
$\mathbf{J_2}$	66	79	95	22	31
$\mathbf{J_3}$	94	25	65	27	89
$\mathbf{J_4}$	11	45	77	21	89
$\mathbf{J_5}$	59	48	80	61	18
$\mathbf{J_6}$	5	40	22	58	22
$\mathbf{J_7}$	80	98	11	50	52
$\mathbf{J_8}$	47	22	1	39	57
$\mathbf{J_9}$	90	40	24	82	55
$\mathbf{J_{10}}$	82	81	93	29	19
$\mathbf{J_{11}}$	42	98	97	37	9
$\mathbf{J_{12}}$	14	9	9	18	53
$\mathbf{J_{13}}$	13	54	64	73	49
$\mathbf{J_{14}}$	5	55	10	82	29
$\mathbf{J_{15}}$	45	62	26	95	67
$\mathbf{J_{16}}$	42	12	76	76	73
$\mathbf{J_{17}}$	2	24	53	15	36
$\mathbf{J_{18}}$	77	43	48	69	69
$\mathbf{J_{19}}$	33	43	41	54	75
$\mathbf{J_{20}}$	4	88	65	33	42
$\mathbf{J_{21}}$	79	80	87	48	39
$\mathbf{J_{22}}$	51	36	48	1	22
$\mathbf{J_{23}}$	80	66	78	50	76
$\mathbf{J_{24}}$	30	84	3	92	95
$\mathbf{J_{25}}$	99	82	3	72	85
$\mathbf{J_{26}}$	15	13	77	4	7
$\mathbf{J_{27}}$	88	48	72	44	46
$\mathbf{J_{28}}$	93	72	94	6	67
$\mathbf{J_{29}}$	87	39	38	82	50
$\mathbf{J_{30}}$	37	46	12	4	6

Cuadro J.9: **Instancia 9**: $n = 30$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	82	18	49	78	58	99	83	71	66	70
$\mathbf{J_2}$	38	61	63	1	13	65	72	74	40	60
$\mathbf{J_3}$	50	7	31	83	1	15	7	88	92	81
$\mathbf{J_4}$	93	23	43	93	58	93	80	21	6	35
$\mathbf{J_5}$	94	82	50	4	35	22	66	58	20	61
$\mathbf{J_6}$	45	1	72	38	28	84	77	29	97	24
$\mathbf{J_7}$	39	8	47	51	96	6	41	10	88	66
$\mathbf{J_8}$	57	75	17	9	77	57	21	44	37	5
$\mathbf{J_9}$	76	13	82	28	79	37	85	12	65	93
$\mathbf{J_{10}}$	8	74	1	69	14	12	22	47	94	7
$\mathbf{J_{11}}$	22	28	29	19	3	95	93	77	49	80
$\mathbf{J_{12}}$	31	27	92	69	87	14	68	79	20	21
$\mathbf{J_{13}}$	80	79	22	66	32	42	3	31	90	56
$\mathbf{J_{14}}$	46	98	85	15	8	2	65	86	51	7
$\mathbf{J_{15}}$	56	15	8	5	7	17	25	64	2	59
$\mathbf{J_{16}}$	45	76	80	50	6	4	7	69	73	92
$\mathbf{J_{17}}$	61	15	45	32	47	58	86	64	53	19
$\mathbf{J_{18}}$	74	67	97	4	76	76	49	84	84	16
$\mathbf{J_{19}}$	4	94	78	50	57	67	99	78	10	65
$\mathbf{J_{20}}$	35	39	82	51	71	85	41	9	57	58
$\mathbf{J_{21}}$	56	26	69	64	40	69	99	38	36	49
$\mathbf{J_{22}}$	81	40	65	20	38	20	47	87	19	62
$\mathbf{J_{23}}$	40	17	51	70	90	39	29	71	62	39
$\mathbf{J_{24}}$	13	41	3	36	16	88	65	10	90	44
$\mathbf{J_{25}}$	74	31	61	39	40	92	43	73	21	60
$\mathbf{J_{26}}$	58	10	12	76	36	45	12	96	50	34
$\mathbf{J_{27}}$	72	39	51	79	60	73	69	10	68	87
$\mathbf{J_{28}}$	14	6	61	96	91	50	45	88	16	5
$\mathbf{J_{29}}$	90	33	88	59	52	72	54	77	62	89
$\mathbf{J_{30}}$	92	51	96	22	68	46	34	58	64	83

Cuadro J.10: **Instancia 10**: $n = 30$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	58	50	75	98	47	87	87	92	61	81	89	67	37	19	63
$\mathbf{J_2}$	39	84	51	67	93	73	15	90	78	93	91	84	58	19	6
$\mathbf{J_3}$	11	2	53	11	7	10	39	16	56	92	55	37	5	56	45
$\mathbf{J_4}$	91	64	70	31	67	3	8	25	25	8	89	39	25	61	79
$\mathbf{J_5}$	43	87	76	94	38	21	84	49	33	76	68	29	45	24	69
$\mathbf{J_6}$	70	66	52	33	2	64	31	71	45	3	99	30	15	8	56
$\mathbf{J_7}$	18	45	20	82	26	36	91	9	39	98	31	11	24	41	12
$\mathbf{J_8}$	94	39	28	74	12	17	2	9	11	92	46	92	8	85	36
$\mathbf{J_9}$	8	53	76	6	85	1	11	12	36	30	31	6	54	63	30
$\mathbf{J_{10}}$	7	34	80	53	60	15	34	20	27	37	96	20	15	1	20
$\mathbf{J_{11}}$	13	78	38	58	43	24	45	8	37	99	41	65	56	80	45
$\mathbf{J_{12}}$	80	4	44	87	88	34	13	3	33	3	22	63	12	39	19
$\mathbf{J_{13}}$	54	7	54	3	96	92	5	77	73	20	72	64	21	23	41
$\mathbf{J_{14}}$	98	64	9	68	9	39	1	85	44	24	76	85	23	52	99
$\mathbf{J_{15}}$	70	89	1	41	65	10	73	51	32	58	18	2	38	61	35
$\mathbf{J_{16}}$	38	46	99	94	96	55	95	55	78	93	29	35	80	45	53
$\mathbf{J_{17}}$	8	34	64	23	7	98	5	39	58	10	80	72	93	74	16
$\mathbf{J_{18}}$	32	87	70	65	84	39	95	5	41	94	58	31	55	62	17
$\mathbf{J_{19}}$	17	26	84	15	5	80	60	55	41	70	32	16	84	39	86
$\mathbf{J_{20}}$	7	45	10	28	9	23	23	20	57	62	16	13	51	48	24
$\mathbf{J_{21}}$	2	5	93	43	79	67	11	84	69	87	50	22	72	97	64
$\mathbf{J_{22}}$	92	55	22	23	17	53	67	79	43	80	34	85	35	59	49
$\mathbf{J_{23}}$	81	52	22	18	27	95	36	44	31	24	62	76	17	50	52
$\mathbf{J_{24}}$	56	22	80	26	91	4	63	3	85	25	43	84	82	35	49
$\mathbf{J_{25}}$	80	18	50	69	58	79	24	64	41	25	49	51	88	2	33
$\mathbf{J_{26}}$	51	41	72	43	65	95	43	98	49	63	33	85	79	15	58
$\mathbf{J_{27}}$	82	49	19	98	55	19	19	66	51	17	30	15	80	82	17
$\mathbf{J_{28}}$	46	97	60	3	3	93	23	33	50	80	62	66	2	17	77
$\mathbf{J_{29}}$	77	20	6	65	8	13	80	59	83	6	33	14	80	46	74
$\mathbf{J_{30}}$	97	35	93	17	52	94	77	25	43	76	43	74	81	87	13

Cuadro J.11: **Instancia 11**: $n = 30$ trabajos, $m = 15$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_1	88	21	97	31	21	84	4	8	90	66	79	86	9	38	86	70	66	78	99	86
J_2	33	96	22	49	46	15	43	50	53	36	72	9	96	5	84	99	48	35	59	49
J_3	38	71	10	69	20	93	28	71	83	92	88	80	2	75	87	28	69	79	15	11
J_4	75	88	87	45	5	3	85	33	88	91	65	39	16	35	54	44	90	33	43	6
J_5	6	34	29	70	57	23	82	90	16	54	30	57	39	36	92	45	92	88	3	36
J_6	2	43	1	60	65	44	89	77	85	77	76	84	25	44	56	21	59	30	4	17
J_7	12	68	72	89	86	76	27	93	12	19	54	82	99	35	34	38	51	19	54	54
J_8	75	57	10	12	6	44	19	66	21	48	40	29	14	83	89	29	63	20	69	66
J_9	93	32	71	19	44	71	21	43	67	14	53	82	28	8	68	89	92	41	80	13
J_{10}	6	83	66	96	3	94	90	17	14	80	36	39	45	49	1	60	52	14	7	91
J_{11}	34	14	53	87	17	75	12	16	7	21	19	35	31	4	49	33	86	30	51	17
J_{12}	85	82	85	81	49	75	1	91	88	81	27	46	96	39	87	17	13	12	42	79
J_{13}	44	55	27	79	61	30	41	97	99	48	10	9	28	59	11	56	65	96	45	69
J_{14}	71	52	6	13	29	40	50	39	91	77	99	70	77	97	9	63	17	44	87	95
J_{15}	5	43	70	97	90	48	47	24	31	85	22	20	70	60	55	45	30	1	4	40
J_{16}	30	42	62	70	56	85	84	71	68	85	20	76	86	40	99	4	27	96	68	40
J_{17}	58	92	34	41	8	99	78	40	26	40	69	29	5	79	49	56	9	37	88	62
J_{18}	37	51	60	36	61	99	28	72	88	60	87	54	38	54	95	98	54	21	66	40
J_{19}	68	8	38	34	95	75	16	1	36	54	68	61	96	94	19	16	70	12	52	67
J_{20}	31	2	55	13	36	89	37	30	35	32	98	35	37	86	59	4	82	22	60	74
J_{21}	18	23	86	68	71	30	95	99	79	1	30	70	12	92	63	57	94	98	4	43
J_{22}	26	99	45	57	38	91	52	7	49	16	17	39	11	10	19	48	38	27	78	20
J_{23}	82	25	84	86	88	9	40	23	33	66	52	17	55	47	63	85	3	71	95	33
J_{24}	79	67	59	54	47	66	56	42	26	12	46	54	2	73	5	41	21	39	55	42
J_{25}	48	60	56	97	99	57	1	45	90	9	33	92	58	90	3	75	36	25	1	56
J_{26}	14	73	66	23	82	68	41	78	70	76	17	42	12	29	99	5	90	70	94	6
J_{27}	86	17	75	14	72	9	97	50	64	2	72	69	32	53	92	93	99	66	10	62
J_{28}	99	24	71	44	72	5	85	28	11	83	69	81	3	43	93	48	11	52	37	25
J_{29}	92	96	3	32	65	53	72	79	53	56	89	80	73	88	6	66	55	5	26	26
J_{30}	93	51	40	32	77	67	24	66	64	59	50	39	60	91	38	63	46	33	38	38

Cuadro J.12: **Instancia 12**: $n = 30$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	89	75	38	93	9
$\mathbf{J_2}$	92	36	90	30	33
$\mathbf{J_3}$	67	16	68	83	14
$\mathbf{J_4}$	54	58	30	9	27
$\mathbf{J_5}$	73	80	8	85	84
$\mathbf{J_6}$	20	37	93	27	75
$\mathbf{J_7}$	74	5	56	45	93
$\mathbf{J_8}$	78	6	32	50	9
$\mathbf{J_9}$	82	61	10	44	60
$\mathbf{J_{10}}$	98	32	37	62	87
$\mathbf{J_{11}}$	81	36	28	57	3
$\mathbf{J_{12}}$	19	26	97	2	39
$\mathbf{J_{13}}$	45	92	17	2	76
$\mathbf{J_{14}}$	38	51	29	72	13
$\mathbf{J_{15}}$	42	61	40	72	41
$\mathbf{J_{16}}$	96	17	75	62	91
$\mathbf{J_{17}}$	20	66	16	50	73
$\mathbf{J_{18}}$	71	26	81	61	96
$\mathbf{J_{19}}$	10	10	58	17	59
$\mathbf{J_{20}}$	62	21	95	61	69
$\mathbf{J_{21}}$	62	91	34	48	37
$\mathbf{J_{22}}$	23	84	56	97	4
$\mathbf{J_{23}}$	91	49	15	74	92
$\mathbf{J_{24}}$	55	62	84	18	69
$\mathbf{J_{25}}$	14	13	85	25	2
$\mathbf{J_{26}}$	15	84	87	36	42
$\mathbf{J_{27}}$	82	93	81	84	38
$\mathbf{J_{28}}$	14	34	3	20	29
$\mathbf{J_{29}}$	65	81	47	33	10
$\mathbf{J_{30}}$	3	6	4	3	16
$\mathbf{J_{31}}$	11	26	30	77	22
$\mathbf{J_{32}}$	27	13	64	16	75
$\mathbf{J_{33}}$	47	86	90	63	85
$\mathbf{J_{34}}$	63	53	48	18	34
$\mathbf{J_{35}}$	97	40	84	15	96
$\mathbf{J_{36}}$	63	40	25	61	21
$\mathbf{J_{37}}$	52	40	25	38	87
$\mathbf{J_{38}}$	54	71	73	33	50
$\mathbf{J_{39}}$	13	18	53	18	76
$\mathbf{J_{40}}$	15	85	58	70	86

Cuadro J.13: **Instancia 13**: $n = 40$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	92	85	87	39	7	52	81	10	23	21
$\mathbf{J_2}$	56	34	67	84	64	48	22	57	45	5
$\mathbf{J_3}$	68	34	54	8	93	93	78	7	93	7
$\mathbf{J_4}$	29	10	45	7	54	90	55	8	32	89
$\mathbf{J_5}$	98	9	23	51	6	4	13	36	59	78
$\mathbf{J_6}$	52	1	12	80	83	48	38	80	16	70
$\mathbf{J_7}$	76	71	78	41	29	60	76	50	24	68
$\mathbf{J_8}$	16	63	51	24	80	56	40	23	73	62
$\mathbf{J_9}$	32	98	59	28	63	13	66	70	63	71
$\mathbf{J_{10}}$	34	63	40	8	26	27	1	16	26	73
$\mathbf{J_{11}}$	29	1	96	31	26	91	78	78	69	31
$\mathbf{J_{12}}$	31	24	66	32	79	35	95	82	90	3
$\mathbf{J_{13}}$	62	10	94	85	17	67	54	69	4	83
$\mathbf{J_{14}}$	4	48	51	89	40	5	11	27	41	36
$\mathbf{J_{15}}$	9	87	18	77	75	40	50	99	45	58
$\mathbf{J_{16}}$	46	86	66	37	38	18	54	68	98	28
$\mathbf{J_{17}}$	73	23	36	97	32	41	10	99	29	45
$\mathbf{J_{18}}$	70	25	87	21	20	35	64	27	36	37
$\mathbf{J_{19}}$	13	31	26	65	43	87	78	25	13	1
$\mathbf{J_{20}}$	45	19	33	37	77	73	80	79	75	43
$\mathbf{J_{21}}$	44	52	47	28	46	32	72	68	25	45
$\mathbf{J_{22}}$	30	89	63	14	23	96	3	93	7	54
$\mathbf{J_{23}}$	65	25	30	17	93	72	7	64	69	53
$\mathbf{J_{24}}$	6	66	62	53	24	7	43	91	65	7
$\mathbf{J_{25}}$	39	88	49	38	26	36	31	39	91	80
$\mathbf{J_{26}}$	93	36	2	69	43	56	56	25	30	88
$\mathbf{J_{27}}$	32	28	70	12	23	46	35	11	48	39
$\mathbf{J_{28}}$	37	35	77	1	39	92	92	50	48	65
$\mathbf{J_{29}}$	97	69	83	83	24	37	25	26	13	33
$\mathbf{J_{30}}$	81	99	69	90	42	28	9	26	87	67
$\mathbf{J_{31}}$	45	84	79	99	99	66	70	97	21	29
$\mathbf{J_{32}}$	71	51	47	77	97	92	49	97	67	75
$\mathbf{J_{33}}$	42	81	20	82	46	49	95	71	40	81
$\mathbf{J_{34}}$	65	88	34	11	36	1	55	20	41	29
$\mathbf{J_{35}}$	32	63	2	83	4	17	34	42	8	92
$\mathbf{J_{36}}$	15	11	12	99	32	55	90	30	5	6
$\mathbf{J_{37}}$	76	98	32	71	28	68	2	1	82	37
$\mathbf{J_{38}}$	75	95	10	68	38	54	71	95	68	92
$\mathbf{J_{39}}$	45	34	61	95	80	62	30	68	92	81
$\mathbf{J_{40}}$	47	5	16	55	25	91	52	79	21	17

Cuadro J.14: **Instancia 14**: $n = 40$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	23	82	22	8	38	17	54	1	67	7	95	74	76	11	29
$\mathbf{J_2}$	74	87	62	34	47	56	39	88	73	43	88	29	96	41	41
$\mathbf{J_3}$	51	40	58	63	52	37	58	60	45	20	16	7	3	52	48
$\mathbf{J_4}$	76	3	86	12	99	88	83	88	55	70	36	99	3	26	71
$\mathbf{J_5}$	10	89	64	31	24	88	87	52	83	65	47	99	17	71	70
$\mathbf{J_6}$	16	93	15	38	96	93	50	96	12	93	92	60	15	20	55
$\mathbf{J_7}$	46	21	37	46	13	54	58	98	27	75	98	96	73	53	9
$\mathbf{J_8}$	36	65	40	67	29	39	64	12	33	34	47	29	82	89	1
$\mathbf{J_9}$	51	32	12	88	23	53	90	25	25	36	74	64	84	98	23
$\mathbf{J_{10}}$	3	2	88	12	86	67	49	42	11	62	66	83	41	49	28
$\mathbf{J_{11}}$	70	2	16	73	1	5	14	78	27	22	3	48	40	16	17
$\mathbf{J_{12}}$	87	75	87	75	68	92	7	37	31	17	22	60	70	37	87
$\mathbf{J_{13}}$	3	13	1	38	3	67	49	91	1	16	81	87	95	79	46
$\mathbf{J_{14}}$	67	68	59	93	61	81	65	49	40	66	58	65	47	44	68
$\mathbf{J_{15}}$	91	17	34	93	59	43	74	56	53	57	21	87	49	29	25
$\mathbf{J_{16}}$	77	19	94	5	85	24	89	41	23	78	17	88	40	24	1
$\mathbf{J_{17}}$	70	83	56	99	81	6	9	67	60	32	63	3	56	42	57
$\mathbf{J_{18}}$	17	33	93	93	14	88	15	94	53	44	33	38	55	84	12
$\mathbf{J_{19}}$	60	60	16	42	43	46	81	61	94	72	61	41	91	15	34
$\mathbf{J_{20}}$	52	63	86	82	78	14	77	77	71	18	47	53	41	41	63
$\mathbf{J_{21}}$	78	38	75	68	80	69	76	4	63	47	43	35	51	77	4
$\mathbf{J_{22}}$	70	27	20	61	38	57	84	32	78	24	57	86	79	99	46
$\mathbf{J_{23}}$	60	37	41	8	4	32	81	95	83	60	52	30	66	47	71
$\mathbf{J_{24}}$	90	27	36	53	72	75	34	59	66	5	50	76	49	78	75
$\mathbf{J_{25}}$	86	94	24	61	87	35	10	20	56	49	5	42	69	49	58
$\mathbf{J_{26}}$	14	33	77	44	67	76	63	64	1	14	78	1	63	77	4
$\mathbf{J_{27}}$	53	69	85	87	21	13	80	63	94	66	65	63	84	16	31
$\mathbf{J_{28}}$	23	93	52	20	91	96	7	5	74	19	30	63	80	93	27
$\mathbf{J_{29}}$	67	49	54	15	39	75	94	53	24	61	89	70	87	70	77
$\mathbf{J_{30}}$	86	48	46	32	97	84	33	84	89	92	12	61	77	72	30
$\mathbf{J_{31}}$	66	91	78	95	7	74	61	62	50	37	70	26	60	94	21
$\mathbf{J_{32}}$	6	43	95	27	89	16	95	51	56	63	37	38	11	50	62
$\mathbf{J_{33}}$	26	87	8	46	41	26	19	29	35	2	47	5	17	76	4
$\mathbf{J_{34}}$	64	87	5	51	29	70	99	70	8	12	68	49	61	87	30
$\mathbf{J_{35}}$	72	68	30	16	18	17	79	1	82	71	14	2	5	87	7
$\mathbf{J_{36}}$	64	51	71	25	6	78	7	74	2	2	10	31	24	13	62
$\mathbf{J_{37}}$	29	56	51	43	8	77	1	70	15	58	70	9	23	19	99
$\mathbf{J_{38}}$	7	25	62	66	69	95	48	11	91	33	95	1	91	43	32
$\mathbf{J_{39}}$	87	30	98	5	41	97	6	39	70	13	60	11	40	63	55
$\mathbf{J_{40}}$	62	65	70	7	73	44	77	78	72	40	81	34	59	41	55

Cuadro J.15: **Instancia 15**: $n = 40$ trabajos, $m = 15$ máquinas

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18	M19	M20
J1	89	81	54	76	94	63	78	75	43	12	67	80	38	6	70	81	85	23	12	60
J2	14	49	13	65	55	74	46	24	33	21	29	52	25	60	93	39	8	23	88	10
J3	2	6	7	5	58	36	44	15	81	36	26	67	64	10	67	59	63	34	45	63
J4	82	31	98	79	9	67	91	89	18	75	1	29	71	16	7	14	46	36	16	63
J5	74	60	28	7	97	41	31	75	45	62	89	13	99	46	53	51	10	29	16	47
J6	65	7	81	60	38	49	32	29	52	42	97	75	51	18	74	38	62	65	50	23
J7	79	40	10	11	14	10	56	71	83	15	58	66	53	24	3	43	23	5	27	56
J8	39	58	53	59	87	77	46	56	68	24	68	17	83	39	13	80	71	39	22	2
J9	76	43	39	32	50	54	34	21	87	80	12	83	85	61	99	80	49	54	51	89
J10	91	77	34	82	14	30	81	25	32	18	18	61	24	74	5	63	86	44	49	47
J11	34	68	82	82	81	97	3	62	90	7	24	97	69	11	95	50	64	76	76	32
J12	69	5	61	27	49	16	28	13	60	61	23	81	2	55	14	6	35	81	81	87
J13	55	41	25	74	7	24	5	59	71	39	33	71	71	24	22	35	64	20	8	63
J14	30	92	32	92	69	23	53	90	64	1	7	92	75	6	43	45	69	41	11	57
J15	8	72	65	95	40	73	21	77	72	84	71	34	21	47	53	57	55	49	29	55
J16	5	13	1	75	72	95	92	68	88	6	69	98	85	27	63	41	90	20	30	6
J17	65	52	69	74	21	53	66	58	17	9	44	3	2	43	35	91	58	54	14	36
J18	59	80	35	47	57	48	92	27	54	99	74	28	22	12	82	66	80	58	51	87
J19	49	67	76	17	20	91	37	52	31	64	73	72	41	50	64	73	13	77	95	53
J20	73	23	30	11	12	96	16	27	19	33	91	45	88	91	82	66	34	24	80	28
J21	80	59	66	35	58	18	15	55	82	51	92	5	8	64	24	77	93	85	1	4
J22	15	65	71	38	10	39	96	2	34	78	96	47	84	15	12	13	96	8	48	79
J23	36	1	78	47	28	75	34	11	14	4	97	79	12	66	75	82	24	63	26	55
J24	24	64	62	94	9	52	59	30	9	66	23	39	79	53	26	32	21	72	91	50
J25	10	20	90	67	21	64	90	99	86	16	57	45	23	60	11	16	11	40	79	41
J26	10	57	4	37	60	32	84	91	87	45	79	93	6	48	92	15	12	10	7	22
J27	86	40	52	68	63	16	24	12	34	90	74	35	79	66	55	83	65	52	79	40
J28	41	53	30	95	18	50	41	1	23	74	13	86	55	76	50	7	91	49	36	40
J29	14	61	71	32	60	99	42	94	19	30	13	76	11	25	29	71	49	31	2	39
J30	55	48	2	41	62	61	66	27	66	50	34	20	72	56	22	87	62	82	34	96
J31	85	70	62	81	39	61	13	89	92	41	65	74	71	43	65	39	18	25	92	73
J32	64	79	54	25	23	22	7	88	54	62	97	22	68	79	85	19	81	32	69	80
J33	28	77	51	84	3	88	99	43	67	93	98	79	72	67	85	88	65	53	32	53
J34	74	4	5	13	10	54	13	35	10	77	39	30	29	80	3	82	40	39	2	23
J35	98	89	27	55	94	14	69	2	91	89	31	34	22	95	28	16	90	90	52	61
J36	8	9	14	83	53	37	92	10	42	64	68	55	37	29	2	98	47	90	97	12
J37	79	16	64	19	61	44	41	89	31	76	89	7	74	7	35	43	50	79	18	45
J38	28	59	98	27	49	24	20	87	60	39	15	25	98	60	16	23	19	77	19	98
J39	92	97	85	69	23	23	98	67	58	85	3	96	83	34	95	31	37	81	57	82
J40	79	69	15	7	6	98	99	97	88	44	39	13	71	34	79	25	80	78	50	76

Cuadro J.16: **Instancia 16**: $n = 40$ trabajos, $m = 20$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$
$\mathbf{J_1}$	66	65	98	70	44
$\mathbf{J_2}$	6	50	99	85	9
$\mathbf{J_3}$	54	78	92	46	36
$\mathbf{J_4}$	15	93	77	28	31
$\mathbf{J_5}$	72	51	26	59	44
$\mathbf{J_6}$	69	64	46	54	58
$\mathbf{J_7}$	94	50	94	91	67
$\mathbf{J_8}$	77	41	69	49	81
$\mathbf{J_9}$	68	47	26	19	97
$\mathbf{J_{10}}$	65	85	41	86	26
$\mathbf{J_{11}}$	60	67	51	20	85
$\mathbf{J_{12}}$	66	92	97	10	36
$\mathbf{J_{13}}$	75	30	40	69	51
$\mathbf{J_{14}}$	19	44	31	59	41
$\mathbf{J_{15}}$	3	9	27	39	12
$\mathbf{J_{16}}$	79	52	90	75	28
$\mathbf{J_{17}}$	52	20	18	13	96
$\mathbf{J_{18}}$	45	32	40	69	52
$\mathbf{J_{19}}$	28	15	31	55	58
$\mathbf{J_{20}}$	73	56	86	15	15
$\mathbf{J_{21}}$	87	10	43	75	32
$\mathbf{J_{22}}$	68	21	42	63	97
$\mathbf{J_{23}}$	78	67	47	75	67
$\mathbf{J_{24}}$	42	71	77	11	57
$\mathbf{J_{25}}$	53	67	84	35	76
$\mathbf{J_{26}}$	49	12	15	50	37
$\mathbf{J_{27}}$	76	46	14	41	83
$\mathbf{J_{28}}$	94	22	70	20	82
$\mathbf{J_{29}}$	84	91	96	18	47
$\mathbf{J_{30}}$	10	84	68	34	46
$\mathbf{J_{31}}$	54	12	93	31	98
$\mathbf{J_{32}}$	38	13	53	95	8
$\mathbf{J_{33}}$	95	14	35	84	18
$\mathbf{J_{34}}$	76	24	89	8	33
$\mathbf{J_{35}}$	34	60	14	2	68
$\mathbf{J_{36}}$	1	48	43	52	46
$\mathbf{J_{37}}$	38	56	75	10	11
$\mathbf{J_{38}}$	80	49	57	86	36
$\mathbf{J_{39}}$	38	31	72	61	1
$\mathbf{J_{40}}$	27	75	35	42	98
$\mathbf{J_{41}}$	70	94	45	29	94
$\mathbf{J_{42}}$	83	62	2	67	94
$\mathbf{J_{43}}$	61	66	33	89	72
$\mathbf{J_{44}}$	71	95	6	72	13
$\mathbf{J_{45}}$	87	91	99	26	83
$\mathbf{J_{46}}$	27	93	12	58	12
$\mathbf{J_{47}}$	3	74	58	47	95
$\mathbf{J_{48}}$	28	17	21	81	71
$\mathbf{J_{49}}$	41	88	11	62	62
$\mathbf{J_{50}}$	6	23	31	9	48

Cuadro J.17: **Instancia 17**: $n = 50$ trabajos, $m = 5$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$
$\mathbf{J_1}$	44	40	60	3	57	71	14	72	3	56
$\mathbf{J_2}$	66	29	30	12	86	6	45	12	22	59
$\mathbf{J_3}$	26	11	60	39	88	19	4	68	21	86
$\mathbf{J_4}$	47	2	74	57	11	27	8	81	82	29
$\mathbf{J_5}$	12	14	62	32	46	26	6	86	87	10
$\mathbf{J_6}$	17	71	65	70	49	22	22	99	86	13
$\mathbf{J_7}$	35	9	38	45	88	55	62	28	79	77
$\mathbf{J_8}$	93	45	88	39	49	51	60	38	38	37
$\mathbf{J_9}$	76	42	95	61	2	38	15	37	48	21
$\mathbf{J_{10}}$	47	6	84	58	7	71	83	3	86	42
$\mathbf{J_{11}}$	90	37	66	17	44	77	95	23	2	75
$\mathbf{J_{12}}$	50	71	27	72	17	8	59	28	54	56
$\mathbf{J_{13}}$	74	4	20	3	80	72	69	49	44	10
$\mathbf{J_{14}}$	78	38	21	56	26	46	30	72	49	69
$\mathbf{J_{15}}$	11	32	58	64	43	11	63	20	28	71
$\mathbf{J_{16}}$	67	63	45	60	14	41	45	55	93	6
$\mathbf{J_{17}}$	43	33	60	18	19	52	95	44	36	60
$\mathbf{J_{18}}$	34	94	44	76	32	47	20	12	47	41
$\mathbf{J_{19}}$	60	21	54	47	46	3	16	40	18	84
$\mathbf{J_{20}}$	1	26	45	76	82	15	93	64	3	28
$\mathbf{J_{21}}$	91	73	48	84	71	43	91	8	38	74
$\mathbf{J_{22}}$	6	47	73	50	16	5	1	34	89	3
$\mathbf{J_{23}}$	49	7	59	54	1	86	22	97	18	49
$\mathbf{J_{24}}$	73	25	63	40	68	8	76	40	9	93
$\mathbf{J_{25}}$	69	46	3	47	48	52	56	40	70	56
$\mathbf{J_{26}}$	73	16	5	38	93	89	52	57	78	66
$\mathbf{J_{27}}$	32	86	38	68	59	82	48	17	56	44
$\mathbf{J_{28}}$	7	9	75	16	30	82	30	39	80	67
$\mathbf{J_{29}}$	57	65	84	46	65	67	76	90	28	26
$\mathbf{J_{30}}$	83	53	37	73	27	1	26	20	17	47
$\mathbf{J_{31}}$	25	3	60	17	6	20	5	28	28	46
$\mathbf{J_{32}}$	95	7	3	50	16	46	26	51	36	75
$\mathbf{J_{33}}$	49	76	92	24	98	29	73	70	48	66
$\mathbf{J_{34}}$	80	16	7	34	96	36	97	40	70	90
$\mathbf{J_{35}}$	15	27	9	87	85	99	36	49	92	18
$\mathbf{J_{36}}$	47	16	24	38	86	77	49	93	10	36
$\mathbf{J_{37}}$	49	29	22	24	42	26	93	47	80	2
$\mathbf{J_{38}}$	98	9	30	56	31	44	25	93	91	73
$\mathbf{J_{39}}$	88	65	40	46	48	84	74	46	81	27
$\mathbf{J_{40}}$	73	61	94	14	99	54	85	37	80	83
$\mathbf{J_{41}}$	71	78	84	58	72	28	73	22	79	10
$\mathbf{J_{42}}$	4	17	50	62	94	54	9	36	92	92
$\mathbf{J_{43}}$	54	8	99	20	85	21	50	73	6	95
$\mathbf{J_{44}}$	39	41	75	86	53	42	10	58	48	72
$\mathbf{J_{45}}$	45	21	72	64	27	48	43	88	82	57
$\mathbf{J_{46}}$	3	14	90	37	29	93	63	43	85	30
$\mathbf{J_{47}}$	83	2	2	69	87	88	54	99	69	82
$\mathbf{J_{48}}$	69	56	58	93	74	78	83	32	69	54
$\mathbf{J_{49}}$	32	58	91	41	77	35	1	69	9	72
$\mathbf{J_{50}}$	93	55	17	24	54	79	91	93	64	2

Cuadro J.18: **Instancia 18**: $n = 50$ trabajos, $m = 10$ máquinas

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$	$\mathbf{M_5}$	$\mathbf{M_6}$	$\mathbf{M_7}$	$\mathbf{M_8}$	$\mathbf{M_9}$	$\mathbf{M_{10}}$	$\mathbf{M_{11}}$	$\mathbf{M_{12}}$	$\mathbf{M_{13}}$	$\mathbf{M_{14}}$	$\mathbf{M_{15}}$
$\mathbf{J_1}$	66	20	82	61	35	26	25	58	91	76	32	91	63	2	54
$\mathbf{J_2}$	17	43	93	97	50	13	29	38	51	36	75	52	97	60	84
$\mathbf{J_3}$	52	78	35	53	23	97	61	30	95	64	9	74	12	23	75
$\mathbf{J_4}$	22	54	33	73	89	51	17	69	32	90	30	8	48	82	12
$\mathbf{J_5}$	1	71	69	42	55	9	72	96	83	20	14	74	6	37	62
$\mathbf{J_6}$	54	40	66	89	84	58	32	39	39	24	50	26	65	65	40
$\mathbf{J_7}$	48	25	35	12	60	98	61	87	33	3	6	48	23	49	63
$\mathbf{J_8}$	11	4	32	77	75	45	28	64	1	40	52	53	98	6	1
$\mathbf{J_9}$	2	48	26	54	32	83	99	62	6	94	13	96	25	42	50
$\mathbf{J_{10}}$	61	33	93	16	91	53	15	37	82	80	75	43	36	28	67
$\mathbf{J_{11}}$	73	88	77	21	49	5	6	90	96	9	62	36	29	1	36
$\mathbf{J_{12}}$	6	5	96	39	35	38	58	79	75	15	34	16	38	43	4
$\mathbf{J_{13}}$	13	25	55	98	26	60	9	69	13	64	91	79	25	18	6
$\mathbf{J_{14}}$	74	26	22	81	56	72	38	63	46	11	78	60	30	99	71
$\mathbf{J_{15}}$	45	11	69	16	59	16	51	18	54	34	59	61	57	81	8
$\mathbf{J_{16}}$	51	75	18	79	46	53	56	97	54	46	57	49	68	88	83
$\mathbf{J_{17}}$	81	60	59	52	56	60	79	35	25	97	92	68	92	64	99
$\mathbf{J_{18}}$	82	46	46	78	67	76	14	80	26	47	89	96	84	42	17
$\mathbf{J_{19}}$	47	69	31	91	4	59	23	56	89	67	91	56	88	35	9
$\mathbf{J_{20}}$	85	60	17	93	13	62	3	91	35	4	7	25	99	84	7
$\mathbf{J_{21}}$	82	23	17	29	37	51	7	5	18	78	47	40	65	98	64
$\mathbf{J_{22}}$	84	47	61	9	81	27	67	84	40	93	96	46	58	6	92
$\mathbf{J_{23}}$	35	95	6	12	12	67	12	31	4	99	84	75	92	91	30
$\mathbf{J_{24}}$	83	1	56	93	54	52	39	41	23	6	33	49	64	15	51
$\mathbf{J_{25}}$	33	27	33	11	96	35	77	53	85	21	35	93	59	76	48
$\mathbf{J_{26}}$	74	40	23	37	78	76	55	26	14	52	97	1	78	18	77
$\mathbf{J_{27}}$	1	69	22	15	21	98	82	71	92	18	8	14	52	76	76
$\mathbf{J_{28}}$	83	22	57	20	77	67	70	74	81	2	96	95	10	46	9
$\mathbf{J_{29}}$	56	52	74	15	58	18	97	97	37	25	70	60	52	3	13
$\mathbf{J_{30}}$	76	64	78	1	21	65	60	46	36	6	74	82	41	29	78
$\mathbf{J_{31}}$	30	97	15	83	91	98	95	50	23	34	79	28	54	39	9
$\mathbf{J_{32}}$	85	84	59	47	17	57	61	42	37	85	73	39	4	4	64
$\mathbf{J_{33}}$	27	26	47	56	41	41	91	50	60	77	10	22	72	37	22
$\mathbf{J_{34}}$	80	89	57	39	33	43	32	45	87	85	23	55	67	85	68
$\mathbf{J_{35}}$	36	49	89	75	99	35	76	70	93	11	49	17	33	66	27
$\mathbf{J_{36}}$	39	83	72	28	18	58	43	66	35	3	24	2	86	81	89
$\mathbf{J_{37}}$	2	88	78	41	32	50	99	25	57	96	2	36	27	37	25
$\mathbf{J_{38}}$	61	96	43	82	69	75	67	33	73	29	12	49	9	66	47
$\mathbf{J_{39}}$	50	6	90	14	68	49	93	78	63	54	17	18	3	95	32
$\mathbf{J_{40}}$	59	72	89	81	87	94	87	5	8	59	47	33	25	87	92
$\mathbf{J_{41}}$	28	76	4	66	99	22	71	50	31	85	19	90	82	12	65
$\mathbf{J_{42}}$	73	69	54	23	21	28	37	29	8	10	42	42	27	81	16
$\mathbf{J_{43}}$	4	42	2	96	43	13	22	14	87	79	23	3	98	42	78
$\mathbf{J_{44}}$	30	43	68	17	46	81	18	27	82	27	76	5	76	94	15
$\mathbf{J_{45}}$	69	97	66	37	75	81	27	75	95	19	51	41	68	59	44
$\mathbf{J_{46}}$	55	22	3	68	90	23	9	10	29	13	40	62	1	22	18
$\mathbf{J_{47}}$	37	51	87	66	26	71	8	57	71	72	5	93	27	2	86
$\mathbf{J_{48}}$	88	7	14	26	37	61	67	4	70	30	93	67	54	43	56
$\mathbf{J_{49}}$	16	72	98	17	22	88	18	53	26	2	51	14	77	19	13
$\mathbf{J_{50}}$	89	16	25	42	43	2	14	22	38	49	8	18	39	17	95

Cuadro J.19: **Instancia 19**: $n = 50$ trabajos, $m = 15$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_1	10	25	56	30	52	32	25	41	94	76	83	80	44	9	54	75	67	71	64	34
J_2	16	38	44	81	48	49	93	61	84	91	63	13	39	88	18	63	19	13	77	91
J_3	87	29	48	51	35	36	83	66	83	52	58	31	42	76	20	78	85	17	65	99
J_4	5	63	60	32	79	50	86	43	44	37	92	76	68	97	83	45	66	2	72	84
J_5	95	26	80	95	28	30	62	47	92	84	75	14	24	31	52	69	64	23	16	68
J_6	13	64	58	71	6	61	72	12	76	25	6	99	43	25	81	16	99	66	92	41
J_7	18	26	78	17	93	96	33	46	88	7	4	64	42	21	81	54	73	37	93	76
J_8	82	41	42	33	23	70	57	5	16	17	56	12	16	54	26	98	52	77	71	34
J_9	2	94	91	84	83	91	90	6	58	83	19	44	87	96	7	59	87	73	46	3
J_{10}	67	75	15	16	50	20	63	24	16	2	4	57	13	66	69	7	18	32	35	5
J_{11}	49	94	18	92	76	54	99	3	52	34	36	79	79	3	53	9	89	15	31	38
J_{12}	87	15	38	92	88	90	70	38	61	55	83	35	48	8	54	78	17	90	86	40
J_{13}	85	40	58	27	8	22	75	50	15	68	65	58	50	74	53	66	53	26	72	10
J_{14}	63	44	26	79	61	96	85	35	59	94	7	98	16	69	88	65	44	20	5	97
J_{15}	11	49	98	80	70	53	95	58	86	20	2	80	6	42	70	24	4	74	68	70
J_{16}	36	71	67	59	30	90	49	43	11	48	87	15	71	64	57	78	99	36	85	98
J_{17}	31	9	27	37	1	57	48	56	98	93	50	70	65	39	49	52	70	57	8	34
J_{18}	62	22	88	62	98	62	64	99	69	24	12	69	83	18	62	89	96	63	9	64
J_{19}	77	45	77	98	45	30	99	61	38	16	75	37	88	21	92	7	4	29	37	71
J_{20}	94	32	32	4	98	62	68	77	73	95	24	39	78	60	89	92	57	89	39	21

Cuadro J.20: **Instancia 20 (parte 1)**: $n = 50$ trabajos, $m = 20$ máquinas

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}
J_{21}	94	99	12	78	10	89	6	13	3	88	88	94	74	87	77	86	90	44	50	65
J_{22}	44	55	82	71	47	9	22	67	65	79	56	16	55	32	19	14	80	36	54	48
J_{23}	21	6	27	48	89	33	47	55	29	58	60	45	52	8	96	56	38	98	7	17
J_{24}	18	73	11	83	95	26	46	21	87	30	83	82	91	84	55	87	4	26	12	98
J_{25}	45	85	93	35	77	81	98	85	15	78	69	8	45	60	35	15	48	81	85	8
J_{26}	50	9	60	86	56	65	72	44	62	36	11	55	23	95	90	79	31	68	39	59
J_{27}	45	34	31	85	60	43	28	48	4	89	78	6	21	71	73	31	63	30	73	27
J_{28}	97	48	55	18	9	50	40	50	57	26	90	72	8	92	9	72	58	38	60	77
J_{29}	10	30	8	75	85	82	28	91	68	1	47	21	19	61	27	28	4	96	72	44
J_{30}	31	75	46	27	88	17	15	66	18	75	75	45	74	70	4	4	62	69	52	43
J_{31}	38	34	78	86	23	87	77	80	68	73	66	86	63	2	17	23	80	14	57	42
J_{32}	1	1	82	48	60	69	42	2	1	26	93	62	43	31	47	63	87	55	28	22
J_{33}	21	8	44	28	95	54	37	63	83	24	38	51	52	73	64	31	88	71	41	9
J_{34}	86	6	22	27	63	28	29	64	72	4	69	50	37	16	72	45	4	45	60	51
J_{35}	9	30	55	21	97	64	72	94	41	15	68	91	34	31	24	19	86	3	95	82
J_{36}	22	88	66	14	49	3	66	44	94	53	59	94	71	53	3	56	48	63	31	86
J_{37}	53	22	34	42	94	84	10	18	10	69	12	84	61	78	50	14	51	62	84	54
J_{38}	11	19	29	99	95	55	68	39	95	69	31	11	21	8	95	79	50	16	8	9
J_{39}	35	82	39	15	2	57	23	3	89	78	29	33	24	47	83	74	15	33	37	25
J_{40}	16	97	37	19	97	75	52	48	14	37	67	75	74	38	35	94	33	21	60	83
J_{41}	16	97	37	19	97	75	52	48	14	37	67	75	74	38	35	94	33	21	60	83
J_{42}	72	24	51	64	42	8	56	32	69	27	88	50	68	78	81	15	93	16	53	22
J_{43}	36	24	58	4	46	67	51	21	31	82	4	13	64	58	44	58	3	34	4	82
J_{44}	7	13	95	24	92	54	68	57	77	11	89	98	94	18	70	82	78	43	34	67
J_{45}	53	65	68	97	87	34	49	51	79	24	1	39	82	9	30	35	13	82	71	52
J_{46}	76	30	98	3	6	22	16	1	4	75	27	85	39	10	38	95	24	88	50	8
J_{47}	76	39	42	19	60	69	57	46	36	35	36	59	74	32	64	87	14	7	23	49
J_{48}	67	16	67	3	14	69	66	18	10	53	77	18	31	5	44	38	12	88	36	97
J_{49}	53	96	76	89	14	65	14	30	97	5	44	21	18	52	89	21	80	11	79	25
J_{50}	43	43	8	97	14	12	43	15	16	2	24	7	34	66	77	6	94	67	21	35
J_{51}	70	11	51	10	14	13	21	22	31	17	72	31	75	66	62	89	28	12	94	68

Cuadro J.21: **Instancia 20 (parte 2)**: $n = 50$ trabajos, $m = 20$ máquinas

Apéndice K

Tiempos de ciclo línea de soldadura

Modelo	WS 1	WS 2	WS 3	WS 4	WS 5	WS 6	WS 7	WS 8
1	535.6025	615.82425	479.4905	516.0355	546.2635	560.71925	585.6215	555.775
2	564.94775	707.20175	387.07375	534.26725	446.41825	708.46775	706.70925	675.0915
3	395.03225	450.941	172.62475	469.39425	432.2625	432.48025	413.934	450.9235
4	631.99425	700.39725	357.82225	737.84	715.119	670.71325	626.69125	719.17375
5	588.57575	736.05525	341.3745	755.67325	700.62375	710.70175	681.10375	741.5255
6	597.73625	730.3035	342.87375	747.89575	878.1395	732.873	662.74925	726.062
7	488.531	693.82275	346.73725	769.24925	912.136	728.04925	682.589	729.46375
8	547.7945	724.95425	339.927	749.4655	720.7425	724.89975	698.5005	738.439
9	478.5715	704.9095	332.39625	780.09075	715.90425	715.5495	696.7535	737.893
10	634.1715	736.75025	313.22875	780.42	732.8205	713.62525	686.60375	738.93725
11	584.68425	733.832	330.44925	779.68725	721.09175	719.513	693.32725	737.037
12	651.80725	741.75825	301.039	757.5555	718.862	721.52	700.15025	744.48125
13	568.22875	685.557	314.995	750.286	714.8495	719.0735	692.478	738.234
14	634.2595	684.37775	309.253	776.659	724.06425	720.57925	695.894	743.34675
15	543.524	733.72225	334.861	755.7675	724.836	720.27125	677.22175	739.14925
16	533.31425	737.23875	316.48925	767.958	899.74	718.39225	692.7965	732.431
17	485.384	736.0815	340.142	770.30725	913.40975	719.99875	693.87825	745.638
18	524.51975	731.91525	343.70175	767.9025	911.943	716.4515	686.759	739.1945
19	535.89775	738.13575	330.02325	756.574	907.85125	713.3055	691.51375	736.64
20	657.42025	742.89775	334.8885	765.8225	892.6545	715.10025	692.6695	735.108
21	560.39325	741.66525	356.5115	767.9545	909.5045	713.35875	698.33725	733.828
22	664.07525	746.06325	335.12525	770.499	910.6235	716.81525	693.179	735.23675
23	553.69375	686.7595	331.48125	766.1185	909.4915	719.001	694.24975	742.25875
24	650.9805	692.378	315.79825	780.57175	725.2995	715.9205	689.42525	732.5295
25	573.96275	689.8385	348.72675	757.11275	721.4735	711.7705	696.07125	732.525
26	665.4205	702.3315	334.65325	780.25675	726.35525	716.8195	688.911	734.83725
27	573.89875	737.1375	344.841	767.877	722.5055	714.89775	692.58675	740.44025
28	661.952	733.9015	326.52775	759.88175	721.8555	711.753	686.97125	737.62175
29	896.344	760.143	319.19025	755.5435	717.4975	736.45048	702.436	738.489
30	1077.43175	789.794	341.544	684.8195	908.057	717.272	685.144	737.2007
31	571.50275	775.50325	342.83325	683.92725	908.7455	720.097	698.74075	746.2985
32	610.7855	777.64675	340.26	710.615	909.6605	718.9445	682.06075	734.418
33	466.36075	757.779	323.266	685.467	910.60775	706.5875	718.60525	737.05075
34	517.74625	756.53175	313.9735	727.5705	908.24275	722.2115	691.6895	740.5845

Cuadro K.1: Tiempos de ciclo acumulados de cada modelo en cada estación de trabajo (*Workstation*) de la línea de soldadura (modelos 1-34)

Modelo	WS 1	WS 2	WS 3	WS 4	WS 5	WS 6	WS 7	WS 8
35	452.9255	755.42975	349.1655	692.7365	909.444	718.55675	703.29925	736.866
36	506.539	760.257	323.2165	675.261	902.1425	715.347	694.7385	738.965
37	447.335	787.869	317.4155	683.99925	909.60525	714.76425	700.6985	737.2575
38	494.39925	789.58925	329.3465	726.39075	899.2345	720.269	704.9425	743.969
39	540.6935	786.537	333.5095	696.7415	934.58675	711.7995	715.4395	743.86275
40	603.50775	744.36875	340.4235	677.24	928.51025	720.906	659.0255	739.593
41	555.26425	741.16275	325.4985	696.86325	925.467	713.84725	703.162	736.30125
42	623.39325	744.978	322.4495	677.0035	922.685	718.81525	701.467	738.13
43	566.1295	736.85375	340.0185	677.73325	929.244	711.04925	722.458	737.54675
44	621.96775	1254.82325	321.88375	701.14325	927.286	716.875	713.835	737.905
45	560.65675	1243.124	328.201	799.65525	1672.002	818.9255	719.5725	1305.312
46	559.1215	1253.08875	325.3135	822.9185	1649.45025	843.99375	710.71425	1276.973
47	558.64025	1236.477	321.4375	826.3535	1661.42325	845.65875	711.91275	1296.287
48	560.591	1229.05925	312.70275	827.335	1641.98075	956.9335	712.4865	1297.0853
49	564.144	1232.79175	312.483	819.75825	1624.39075	960.285	721.1045	1294.095
50	565.17275	1235.88125	314.88675	826.64475	1611.4725	980.7425	725.14525	1296.522
51	569.93	1209.76525	319.22125	824.704	1613.29675	983.64975	725.38425	1301.865
52	568.5745	1211.956	321.59725	811.179	1598.598	979.63775	731.283	1300.789
53	574.0345	1221.8515	312.77975	812.26175	1596.1775	981.94025	723.1725	1298.575
54	574.04725	1220.096	321.3625	809.1765	1602.7035	981.963	729.74175	1301.9034
55	574.8245	1225.9035	313.82425	812.24525	1602.02	982.144	726.4895	1297.636
56	394.922	747.044	172.55275	506.06925	895.174	654.449	432.3685	783.99075
57	569.98475	1238.28475	314.5765	836.18525	1596.78425	980.9915	731.8015	1298.711
58	575.131	1227.14825	320.305	834.95125	1597.64275	980.9355	724.4965	1306.639
59	568.68	1223.101	322.7775	851.189	1575.35125	981.42625	723.48925	1309.634
60	568.94925	1224.27575	325.92925	846.9595	1569.08875	981.91025	724.03825	1310.712
61	575.12375	1226.20275	318.235	846.04825	1574.27325	979.7975	728.827	1309.706
62	592.5645	1228.57275	317.07475	853.557	1569.1195	982.07025	730.69275	1309.591
63	588.38275	1255.61475	316.35275	834.9085	1551.8355	980.823	734.43125	1311.714
64	593.646	1248.25725	317.03975	833.932	1551.82125	976.919	736.48725	1311.648
65	588.688	1253.32925	314.02325	827.66125	1553.953	980.704	736.90275	1308.927
66	593.67975	1253.2615	314.24225	830.6005	1549.92725	981.6595	733.622	1308.324
67	587.6035	1250.37125	314.094	830.73725	1550.21075	988.6735	737.37425	1306.501
68	592.40225	1250.349	319.304	832.6795	1554.827	982.65175	738.60025	1314.199

Cuadro K.2: Tiempos de ciclo acumulados de cada modelo en cada estación de trabajo (*Workstation*) de la línea de soldadura (modelos 35-68)

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ANEXO I. RELACIÓN DEL TRABAJO CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE DE LA AGENDA 2030

Anexo al Trabajo de Fin de Grado y Trabajo de Fin de Máster: Relación del trabajo con los Objetivos de Desarrollo Sostenible de la agenda 2030.

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**				X
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**			X	
ODS 8. **Trabajo decente y crecimiento económico.**			X	
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades**.				X
ODS 11. **Ciudades y comunidades sostenibles**.				X
ODS 12. **Producción y consumo responsables.**		X		
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**			X	

Descripción de la alineación del TFG/TFM con los ODS con un grado de relación más alto.

***Utilice tantas páginas como sea necesario.

El presente trabajo de fin de máster es una herramienta importante para la gestión eficiente de la producción en la industria, y puede ser aplicada para contribuir a varios Objetivos de Desarrollo Sostenible (ODS).

En primer lugar, tiene una ligera influencia en el ODS 7, correspondiente a Energía asequible y no contaminante. Esto es porque la secuenciación permite reducir tiempos de espera y bloqueo en la maquinaria y, por tanto, reducir en ineficiencia energética de los robots.

Editor: Secretaria General / UPV · D.L.: V-5092-2006 · ISSN: 1887-2298 · www.upv.es/secgen

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Anexo al Trabajo de Fin de Grado y Trabajo de Fin de Máster: Relación del trabajo con los Objetivos de Desarrollo Sostenible de la agenda 2030. (Numere la página)

De la misma forma, la reducción en ineficiencia de la maquinaria supone una reducción en costes de mantenimiento, lo cual implicaría en una reestructuración de la división de tareas de los operarios de la línea de producción, pudiendo asignarles labores más comprometidas con su bienestar como trabajadores al requerir menos esfuerzos físicos o mentales. Esto justifica una ligera influencia en el ODS 8 (Trabajo decente y crecimiento económico)

Por otra parte, se trata claramente de un proyecto de innovación, en este caso aplicado a la industria, puesto que busca la mejora de procesos a través de ideas creativas que tratan de reinventar el funcionamiento clásico de una línea de producción. Además, como ya hemos comentado, la optimización de la secuencia de producción implica mejora de la eficiencia y reducción de la huella de carbono. Por este motivo, aplica en gran medida al ODS 9 (Industria, Innovación e Infraestructuras)

Por último, la secuencia inteligente de la producción mejora la planificación y el control de la cadena de suministro, lo que puede contribuir al ODS 12 (Producción y consumo responsables) al reducir el desperdicio de recursos y promover la sostenibilidad.

En resumen, aplicar la secuenciación de la producción de manera inteligente puede tener un impacto positivo en la eficiencia de la producción, la sostenibilidad y la responsabilidad social, lo que contribuye a varios ODS.

Editor: Secretaria General / UPV · D.L.: V-5092-2006 · ISSN: 1887-2298 · www.upv.es/secgen