UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Industrial

Desarrollo de un software para la replanificación y programación de la producción de una empresa textil de la comunidad valenciana

Trabajo Fin de Grado

Grado en Ingeniería de Organización Industrial

AUTOR/A: Arocas Fernández, Alejandro

Tutor/a: Andrés Romano, Carlos

CURSO ACADÉMICO: 2023/2024

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

TRABAJO FINAL DE GRADO EN INGENIERIA DE ORGANIZACIÓN INDUSTRIAL.

DESARROLLO DE UN SOFTWARE PARA LA REPLANIFICACIÓN Y PROGRAMACIÓN DE LA PRODUCCIÓN DE UNA EMPRESA TEXTIL DE LA COMUNIDAD VALENCIANA.

Autor: Alejandro Arocas Fernández
Tutor: Carlos Andrés Romano

Curso: 2023-202

Contenido

GLOSARIO

Estación: Conjunto de máquinas que realizan un proceso dado. En este caso la empresa cuenta con siete estaciones: Emburrado, lavado, rame, estampación, estampación digital, calandra, presentación.

Máquina: Lugar físico en el que se procesa un trabajo.

Trabajo: Producto individual demandado por un cliente concreto.

Ruta: Secuencia ordenada y constante de estaciones que recorre un trabajo

1. INTRODUCCIÓN Y MOTIVACIONES

En este trabajo de fin de grado se presentará un problema real al que se enfrenta una empresa valenciana del sector textil. Se estudiará y describirá su situación actual, y se desarrollará una solución personalizada para ayudar a lidiar con el problema dado. Posteriormente, se analizarán los resultados obtenidos para mejorar las posibles soluciones.

Las motivaciones personales que me han llevado como alumno a la realización de este proyecto en cuestión son varias, pero pueden resumirse en tres:

El querer trabajar y enfrentarme a un problema real de una empresa real antes de finalizar mis estudios de grado. Hasta ahora se ha trabajado con casos, muchas veces ideales, y en entornos controlados. No quería que mi último trabajo en la universidad (al menos de momento) tuviera la misma sinfonía irreal.

En segundo lugar, el querer explotar de forma seria y práctica mis capacidades para aprender una materia, como en este caso es la programación. En el grado hemos trabajado con numerosos lenguajes de programación, pero evidentemente no se ha podido realizar una inmersión completa en ninguno de ellos por una cuestión de agenda. En este trabajo deseaba retarme, comprobar hasta qué punto podía profesionalizar el uso de un lenguaje y llevar el nivel más allá a lo aprendido durante la carrera.

Finalmente, el tercer punto, la vocación. En esta carrera se ha trabajado sobre numerosas perspectivas, pero ha habido dos que me han cautivado, el programar/ optimizar y el solucionar problemas. En este proyecto tengo la posibilidad de ayudar a solucionar un problema real, mediante programación, mientras aprendo. Por ello, cuando mi tutor Carlos Andrés me propuso el caso, no dude en decirle que sí.

1.1. RESUMEN DEL TRABAJO

Actualmente, la empresa objeto de este TFG no cuenta con un sistema informático que le permita calcular las fechas de los trabajos en curso y sufre desestabilizaciones por la llegada de pedidos urgentes o cambios de las programaciones por los clientes. Así mismo tampoco posee una política de secuenciación de trabajos, la cual se base meramente en un FIFO (first in fisrt out) con ajustes manuales en momentos puntuales.

En consecuencia, se estiman aproximadamente las fechas de entrega a los clientes gracias a la experiencia de los responsables del área de operaciones de la empresa, que pueden predecir con cierta holgura cuando acabarán y/o empezarán los trabajos planificados. Sin embargo, la llegada de trabajos urgentes por parte de clientes principales en muchas ocasiones genera desestabilizaciones en el Plan y en el Programa de Producción que, invalida las aproximaciones anteriormente mencionadas. Las cuales no son corregidas ni informadas a los clientes en la mayoría de los casos.

Por eso, en este trabajo se plantea desarrollar una herramienta informática que (teniendo en cuenta los criterios y experiencia de los responsables de la empresa) permita desarrollar planes y programas de producción de una manera rápida y automatizada. Esta herramienta permitirá evaluar y comparar diferentes alternativas a los mismos. Cuantificará en que medida se solucionan los problemas, se estimará el coste de un implantación y se aportará la forma de continuar con la mejora de esta herramienta.

1.2. EMPRESA

La empresa sobre la cual se va a realizar este proyecto es una empresa textil valenciana, situada en la localidad de Agullent, al sur de Valencia y fundada en 1986.

En sus orígenes, fue una empresa de estampación de tejidos, suscrita a otro grupo empresarial, pero no fue sino hasta el año 2001 cuando empezó a formar parte del accionado de un grupo importante en el sector textil. El grupo adquirió en el inicio el 70 % de las acciones, para luego adquirir en 2004 el 100 % de la empresa. En este periodo de tres años la empresa se consolidó en su actividad actual marcando un punto de inflexión entre lo que venía haciendo y lo que se haría pasado ese momento.

Actualmente, la empresa que nos acontece no fabrica textiles, sino que los trata para posteriormente ser vendidos a otros grupos y empresas, algunos líderes de mercados nacionales e internacionales. La compra de unidades productivas en la que está inmersa la empresa desde noviembre de 2023 y la ambición de añadir la parte de fabricación de textiles (tejedoras) por parte de la dirección apunta a que el rumbo de la empresa la hará virar hacia una expansión vertical en el sector en un periodo menor a diez años.

La empresa está organizada jerárquicamente de forma orgánica, posee una dirección general ante la que responden cinco responsables: producción, calidad, administración, proyectos y compras-RRHH, puestos de trabajo formados por personal polivalente que puede ocupar cualquier posición si la situación lo requiere.

Una de las partes más importantes de esta empresa son sus trabajadores, que son los encargados de desarrollar la actividad productiva. Esta empresa aplica uno de los principios fundamentales del ‘Kaizen’, el cual dice que, “cuando contratas unas manos te regalan un cerebro”, lo que se ve reflejado en el proceso de selección de la empresa, donde uno de los pilares fundamentales que se busca es la proactividad por parte del trabajador, así como el apoyo y la aportación de ideas. Realidad que fortalece organizacionalmente la empresa mediante formaciones continuas y mediante valores basados en el trabajo en equipo, la implicación, la participación y, como se ha visto reflejado en la dirección, la polivalencia.

La empresa tiene una gran responsabilidad ética en el trabajo y el medioambiente, vía prevención de riesgos laborales como señales en planta y dotando a sus cuadrillas de equipos de seguridad y depurando siempre el agua y los químicos usados por la planta.

Como cualquier empresa, ha sufrido durante estos años por la situación mundial, lo que ha dejado clara evidencia de la resiliencia de la dirección. En la crisis del 2008 la empresa redujo gastos y consiguió seguir operando con su actividad. Haciendo ajustes de horarios en 2011, consiguió no reducir drásticamente la plantilla y, sobre todo, no cerrar. Años después, cuando ya se encontraba prácticamente recuperada, llegaron dos situaciones más. La primera de ellas fue la pandemia y la segunda el conflicto de Rusia. De nuevo, la empresa explicitó la polivalencia y adaptabilidad

caracterizada, redujo el trabajo al justo y necesario, tomó medidas basadas en ajustes económicos y consiguió superar ambas situaciones.

1.3. SECTOR

La empresa trabaja en el sector textil, específicamente y cuando se realiza este TFG, en el sector del tratamiento de textiles, no en su fabricación.

Es un sector caracterizado por su fuerte competencia, sobre todo por la presencia de los países asiáticos, los cuales lideran el mercado en costes con mucha ventaja. Estos países son líderes del mercado a cambio de comportamientos contra ecológicos como un tratamiento escaso o nulo de residuos, vertidos de agua en zonas comunes... lo que les permite ahorrarse grandes cantidades de capital.

Otra característica de este sector se da en las pocas fluctuaciones de personal de las empresas. La mayoría de estas entidades están en villas o zonas de exterior, nutriéndose de la mano de obra de las personas de los pueblos cercanos, junto a convenios que normalizan los salarios, provoca que muchas personas no cambien de empresa en toda su vida profesional. No hay cambios de personal tan agresivos como los que podrían darse en sectores como el transporte donde son comunes los movimientos de los camioneros entre empresas.

Por otra parte, los accidentes laborales no son comunes. La construcción de las plantas se hace bajo ciertos convenios y políticas de seguridad y prevención de riesgos se llevan concienzudamente a cabo, y la gestión de la actividad profesional esta rigurosamente asegurada.

A nivel social, según los datos de EURATEX en el año 2021 el 70% de los trabajadores del sector son mujeres, se habla de una industria donde la brecha de género ha sido superada ya desde hace años.

A nivel económico, España se encuentra el tercero de Europa, igualado con Francia, en exportaciones de textiles. Así mismo, se observa en Europa un valle en el desarrollo económico. Como se ha nombrado anteriormente, los problemas relacionados con las guerras, la inflación, la crisis energética... no sólo afectaron a esta empresa, sino también al sector en general, cerrando el año 2022 con un gasto medio por europeo de 490€ en ropa, en comparación con los 600€/persona que se gastaron de media en el año 2020, un decremento de más del 20%.

2. SITUACIÓN ACTUAL

Actualmente la empresa trabaja de lunes a viernes a tres turnos, sin embargo, no todas las máquinas trabajan todos los días. Dependiendo de la semana y de los productos en curso, hay máquinas que pueden encontrarse ociosas durante un periodo, y ser cuello de botella en el siguiente. A continuación, se dará una descripción de la situación actual de la empresa: que vende, como lo produce, que máquinas intervienen y como gestiona actualmente la planta.

Pese a que dependiendo de la semana cualquier máquina puede ser cuello de botella, usualmente el centro de la actividad de concentra en las estaciones de tintura y rame, que son aquellas que aportan proporcionalmente mayo valor al producto, es decir, aquellas en las que se producen los cambios más significativos.

2.1. PRODUCTOS

Los productos que trata la empresa usualmente son sábanas de gramaje 110g/m2 y lonetas de menos de 150g/m2, como se ha dicho anteriormente, la empresa no fabrica estos textiles, si no que los limpia, tinta, trata... a través de diferentes procesos que serán nombrados posteriormente.

Hay tres condicionantes principales a la hora de tratar las telas: el color, el material y la anchura.

La empresa trabaja con todo tipo de colores y a través de estampaciones y teñidos es capaz de producir diferentes siluetas y tonos, sin embargo, la presencia de colores en las máquinas afecta al desarrollo de la actividad, se deben respetar órdenes "jerárquicos" de colores para evitar limpiezas intermedias, teniendo el orden de prioridad de claro a oscuro, y de esta forma poder aumentar la producción.

Por otra parte, los materiales con los que trabaja esta empresa son el algodón, el poliéster y mezclas de los dos anteriores. A diferencia de lo que ocurre con los colores, algunas máquinas sí que tienen limitaciones con determinados materiales (véase apartado "procesos y máquinas"). Al igual que ocurre con los colores, que se busca una jerarquía en función de la oscuridad del color, con los materiales se intenta consolidar los grupos y agrupar los tipos para evitar ajustes intermedios.

A las jerarquías anteriores se añade una tercera variable, la anchura. A la hora de tratar las telas se busca que las anchuras sean parecidas para evitar, una vez más, ajustes. Las tolerancias de las máquinas toleran fluctuaciones de 20cm para el poliéster y de 40cm en algodón y mezclas de poliéster-algodón.

Con todo esto se tienen tres variables técnicas (sin contra plazos de entrega, pedidos urgentes...) que influyen a la hora de ordenar las órdenes en las máquinas: el color, el material y la anchura. Es decir, lo óptimo en cada máquina sería que pasaran de claro a oscuro cada material separado y que, además, tuvieran una anchura semejante.

Una cuarta variable afecta a los productos, la forma en la que llegan. Las lonas y sábanas pueden llegar de dos formas distintas, emburradas (enrolladas sobre si mismas) o sin emburrar.

Un ejemplo de producto emburrado se aprecia en la ilustración 1:

Ilustración 1: Ejemplos de producto emburrado

En caso de que el tejido llegue sin emburrar, normalmente en palés, lo primero que se realiza es el proceso de enrollarlo para posteriormente guardarlo en el almacén.

De esta forma, se considera como producto primario de la empresa aquel textil que, independientemente del color, anchura y tejido, está emburrado y listo para ser procesado. Debido a que, una vez emburrado, el tejido pasa directamente al almacén.

2.2. PROCESOS Y MÁQUINAS

Para poder realizar una buena solución es fundamental entender bien el proceso y que máquinas intervienen en el mismo. Por ello, se realizará una explicación detenida del flujo de trabajo de la nave.

El proceso se clasifica en 7 fases: emburrado, lavado, tintura, rame, estampación, calandra y presentación. No todos los productos pasan por todas las fases anteriores, dependiendo del material, estado, forma de llegada... la ruta puede variar. Sin embargo, hay una constante, el flujo sigue siempre el orden anteriormente nombrado (pudiendo saltar fases, pero nunca retrocediéndolas) a excepción de determinados casos donde *el flujo puede ir de estampación a rame.*

EMBURRADO:

Este proceso no es común a todos los productos, solamente se realiza cuando la tela ha llegado sin embalar, usualmente en forma de palés.

Consta de dos máquinas

- Emburradora 1: Balas, palets y rollos superiores a 700m
- Emburradora 3: Rollos inferiores a 700m

El orden usual de planificación es el FIFO, pero dependiendo de las necesidades de las siguientes máquinas o de la presencia de pedidos urgentes puede variar.

Esta sección normaliza la materia prima que llega, como tal, no añade ningún valor al producto, tan sólo lo prepara para que el resto de las máquinas puedan trabajarlo. Una vez emburrado, el producto se dirige usualmente al almacén de materia prima, donde es posteriormente lavado.

LAVADO:

Al igual que el proceso anterior, esta sección normaliza el producto. La empresa tiene organizada su planta para teñir sobre determinado tono y de determinada forma. El lavado y el emburrado se encargan de llevar todo el producto en crudo a estas condiciones iniciales estandarizadas para posteriormente trabajar sobre ellas.

Consta de una única máquina, la lavadora, la cual se pone en marcha con cantidades superiores a los 100.000 m por optimización económica.

- Primero se descruda el producto, de oscuros a claros, para aprovechar el agua (mínimo 10.000m)
- Si el producto tiñe el agua se vacía antes del blanqueo, si no tiñe, se aprovecha el agua.

A continuación, se pasa a la fase de blanqueo, que se realiza por fórmulas y artículos.

- Primero: Algodón con chamuscado y fórmula fuerte (1)
- Segundo: Sin chamuscado y fórmula fuerte (1)
- Tercero: El resto de fórmula 1, siguiendo orden de anchura y de viejo a nuevo (FIFO) **con** chamuscado
- Cuarto: El resto de fórmula 1, siguiendo orden de anchura y de viejo a nuevo (FIFO) **sin** chamuscado

Después se realiza el mismo proceso en el mismo orden con formula 10. La principal característica de la lavadora es que los productos del mismo tipo deben lavarse a la vez. Todo el algodón junto, todo el poliéster junto... Por lo que una vez encendida la lavadora, primero se lavan una serie de productos, se extraen, se lava otro tipo... no se realiza el lavado de los 100.000 m de forma simultánea, sino que se hace por tandas.

TINTURA

La sección de tintura, como su propio nombre indica, es la sección en la que se pintan los diferentes textiles. En las anteriores secciones, la planificación se llevaba a cabo desde el propio proceso de planificación, en cambio, en este caso, la planificación puede realizarse desde las propias máquinas, dependiendo de las necesidades del día.

Diferenciamos tres partes: jiggers alta, jiggers baja y autoclaves. De cada tipo de máquina hay dos unidades las cuales se planifican discretizando entre ellas, no por sección, además, en el caso de las autoclaves, hay una máquina auxiliar llamada PAU. Cada máquina tiene una capacidad media de 1500m cada 8h

En el caso de los jiggers, se separan las órdenes que corresponden los programas de baja, de los programas de alta. Posteriormente se ordenan las órdenes por colores y se realizan los tintes.

En el caso de las autoclaves, se programa la PAU, se separan las órdenes por colores y se programan por anchos dentro de cada máquina.

En esta sección encontramos la primera orden de priorización en el proceso productivo. En el caso de los jiggers, se sigue un orden por color al tener que diferenciar entre ordenes altas y bajas En caso de las autoclaves, la separación por colores se hace de entrada, para diferenciar dentro de cada máquina, como segunda condición, la anchura. Es decir, la prioridad del color es superior a la anchura.

Una vez han sido teñidos los productos, se dejan reposar dando vueltas hasta que terminan de secarse.

RAME

Esta sección es el centro de distribución de la planta. En la que se estiran, y se terminan de secar los productos.

Recapitulando, las dos primeras secciones no añaden valor al producto, tan solo lo dejan listo. La sección de tinte es la primera que realiza un cambio (en sentido valorativo), sin embargo, es en la sección de RAME donde concentra la actividad.

Por ello mismo, a esta sección pueden llegar productos de todas las secciones anteriores directamente, junto a productos de la fase de estampación.

En esta sección se encuentran tres máquinas: Rame-1, Rame-2, Rame-3; las cuales se planifican, como en la anterior sección, por separado y atendiendo a cada máquina.

Se dan diferentes condicionantes en esta sección.

- A nivel capacitivo, R1 solamente puede trabajar con lonas inferiores a los 280cm mientras que en R2 y R3 el máximo es de 320cm
- Conectado con la sección anterior, nunca pueden haber más de 6 torpedos y 5 bancos en reposo (esperando a ser tratados en RAME)
- Determinados colores como el blanco deben ramearse en horarios donde el laboratorio esté operativo
- Agrupación máxima de colores y anchos

Una vez rameados, los productos pueden ir a estampación, calandra, presentación y/o en caso de necesitar modificarlos, a tintura de nuevo.

ESTAMPACIÓN

En esta máquina/sección, sobre los tejidos se imprimen diferentes formas y siluetas mediante estampado con rodillos, realizándose una planificación según la necesidad de plazos de entrega.

La capacidad es de 9-10k metros en 12h

CALANDRA

Esta máquina es única y se programa desde planificación.

Se ordena según fecha de llegada o urgencia.

PRESENTACIÓN

Última sección de la planta, el producto ya está terminado solo queda prepararlo para su expedición.

Esta máquina la programa el responsable de presentación siguiendo las indicaciones de prioridades desde planificación y el máximo aprovechamiento de las máquinas

Hay 3 tipos de empaquetados:

- Al ancho, con 2 máquinas: una con empaquetadora automática (EN4) y otra sin empaquetadora (EN2)
- Al lomo, solo una máquina llamada ED2. No empaqueta, solo prepara
- Al libro, hay una máquina: la PL2. Se empaqueta mientras se enrolla.

Además, hay dos máquinas auxiliares:

- Cortadora: Hay una maquina con cuchilla. Auxiliar según pedido. No se programa.
- Empaquetadora: Se empaqueta según se rolla y dependiendo de urgencias. No se programa.

En conclusión, el flujo de los productos puede ser variado y depende de diversos factores como el material o la logística de llegada. De todos los procesos que se realizan, los más importantes son tintura, rame y estampación, ya que son los únicos que le aportan valor al producto, entendiendo como procesos que aportan valor los que cambian la forma, el color o la disposición del producto de forma voluntaria, y el cliente está dispuesto a pagar más por dichos cambios.

También se observa que la mayoría de las secciones siguen una política FIFO en ausencia de pedidos urgentes, y se gestiona su programación desde la propia máquina.

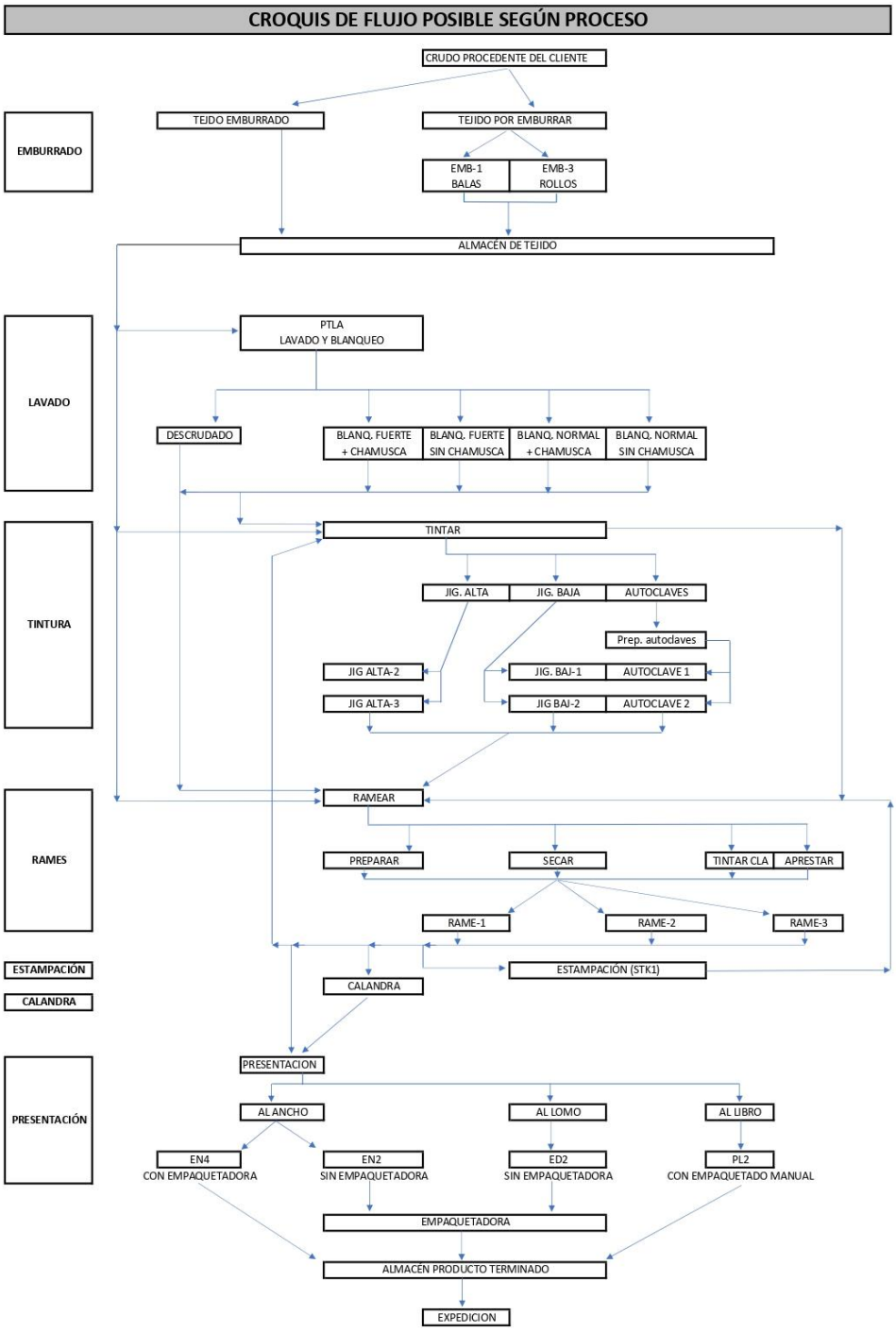

Ilustración 2: Resumen del flujo de los procesos de la plant. Fuente: Empresa

2.3. SISTEMA ACTUAL DE PLANIFICACIÓN Y GESTIÓN DE DATOS

Como se ha nombrado anteriormente en el apartado de procesos y máquinas, la planificación se puede realizar desde las propias máquinas para solucionar problemas puntuales o venir desde los responsables.

La planta cuenta con el apoyo de un ERP, el cual se puede programar por operaciones o por carga de máquinas, es decir, por ejemplo, en el caso de RAME, se podría programar una HDR (hoja de ruta) por la sección, o adjudicándola a una máquina en cuestión.

A su vez, cada día, se realiza una reunión de 5-10 minutos de duración para establecer que se va a fabricar, revisar priorizaciones y gestionar pedidos urgentes en caso de haberlos.

El uso del ERP, debido a sus limitaciones, tiende al apoyo más que a la gestión.

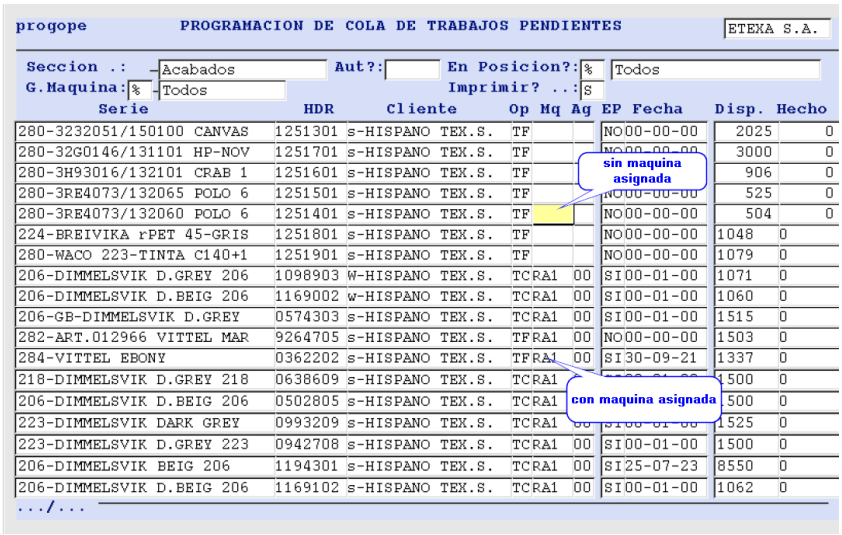

Ilustración 3: ERP de la empresa. Fuente: Elaboración propia a partir de la información de la empresa.

Como se observa, el ERP da información de clientes, hojas de ruta… pero no indica cuando estarán listos los pedidos, ni asigna de forma automática las órdenes. Ayuda a tener una visión global, pero las decisiones, la adjudicación y los tiempos son realizados y estimados por el personal de la planta.

Debido a la situación actual de la empresa, algunos de los problemas a los que se ha enfrentado son las roturas de stock, las averías y el aumento del coste energético, y entre ellos, uno de los más importantes, la incertidumbre. La empresa puede saber la mayoría de las ocasiones cuando entran las órdenes, pero no es capaz de determinar con exactitud cuando serán terminadas, tampoco cuenta con un sistema de priorización más allá de lo explicado anteriormente respecto a seguir patrones de colores, anchura y material, por una parte, y por otra limitarse a seguir el FIFO.

3. SOLUCIÓN PROPUESTA

El objetivo de este trabajo final de grado será abordar y solucionar el problema relacionado con la incertidumbre nombrado anteriormente. La solución se llevará a cabo mediante la programación de un software que no sólo podrá calcular las fechas de fin de una serie de trabajos dados, sino que también será capaz de aportar soluciones a la hora de secuenciar dichos trabajos (Chen, 2009)

La razón para solucionar este problema mediante esta vía y no a través de la compra de un software de distribución es debido a las numerosas desventajas que tienen este tipo de softwares. Como la estandarización de sus herramientas o el uso de motores genéricos para resolver problemas, los cuales no permiten saber de qué manera se están solucionando.

Por ello, el siguiente punto comenzará con una comparación de un software como el que se elaborará frente a uno comercial. Posteriormente se definirá el problema que se aborda desde el marco teórico, se definirá el campo de aplicación del software y se justificará el uso de Python y no de otro lenguaje de programación.

3.1. COMPARACIÓN CON OTRO SOFTWARE DEL MERCADO

Datamon es una empresa catalana especializada en el desarrollo de sistemas de gestión de información el sector textil, con más de 35 años de experiencia, cuenta más de 100 instalaciones a lo largo del mundo.

Su software está especializado en la gestión de datos, no sólo de lo que ocurre en planta sino también en el ámbito inter-empresa. Es capaz de gestionar compras, facturación, trazabilidad... en resumen es un software de empresa, no solamente de planta.

En su gestión de la producción, Datamon es capaz de generar hojas de ruta, captar información a tiempo real y controlar máquinas y operarios entre otras funciones.

A todos los efectos, evidentemente, un software profesional es más completo e inmersivo. Sin embargo, las ventajas que posee el software objeto de este TFG respecto a uno profesional son principalmente tres.

Por una parte, la precisión; este software está diseñado por y para la empresa de la localidad de Agullent, todas sus características han sido adaptadas a la situación de esta. Los problemas que atiende son estrictamente los requeridos por la empresa y los soluciona según lo esperado. Un problema común en los softwares comerciales es que su objetivo es que sean adquiridos por el mayor número de empresas, por lo que suelen estar bastante estandarizados y no permiten grandes ajustes o adaptaciones. Este software, sin embargo, no requiere de adaptaciones, debido a que está elaborado alrededor de la empresa y siguiendo directamente las demandas de la misma.

La segunda razón es la usabilidad, un software profesional del calibre de Datamon está pensado para grandes empresas que quieren centralizar toda su información, lo que hace que entre otras cosas se requiera de un manual para aprender a usarlo. Este software es sencillo, y en menos de 15 minutos cualquier persona es capaz de aprender a usarlo. Además, este software está pensado como programa de apoyo al jefe de producción para mejorar la situación de la planta, a diferencia del software profesional que está pensado para una empresa en su conjunto. De esta forma esta herramienta se centra en el problema de la empresa y se orienta a solucionar ese problema, no se han destinado funciones ni energías a referencias que no hayan sido demandadas por la empresa. De modo que el 100% del software es usable para la empresa y no hace que el operario sea abrumado por la cantidad de información.

Finalmente, el precio. Este software no ha sido pensado para su distribución, ni está a la venta, pero tanto por razones de uso, como de prestaciones, sería más barato su desarrollo e implantación que el de uno profesional, sobre todo cuando los problemas que se pretenden solucionar son los dados en este caso.

Para el caso de la empresa en cuestión, no sería rentable una inversión que tarda años en ser recuperada y que además debido a su complejidad y al no ser intuitiva para el usuario, no sería 100% amortizada desde el punto de vista de uso. Tampoco una que sea de código cerrado y que no permita, entre otras cosas, evaluar diferentes soluciones o definir nuevas formas de buscarlas, es decir, que se centre en soluciones genéricas.

En conclusión, un software comercial para el caso dado, por una parte, sería insuficiente en la función de mejora de solucione y, por otra parte, aportaría información excesivamente compleja e innecesaria sobre asuntos que no se requiere. Sería una solución cara e imprecisa.

3.2. DEFINICIÓN DEL PROBLEMA

En el marco real, se tiene un problema de incertidumbre sobre las fechas de fin de trabajos que pasan en orden sobre una serie de máquinas, en el marco teórico, este tipo de problema debido a las características de la planta se puede definir como un problema de taller de flujo a rasgos generales.

Se definen los problemas de secuenciación de taller de flujo con máquinas paralelas (parallel machines flowshop scheduling problems) como un conjunto de trabajos que pasan en el mismo orden por un mismo conjunto de máquinas y que, independientemente del orden, el tiempo de procesamiento del trabajo en la máquina no varía.

En este caso debido a la configuración de la empresa no se puede hablar de un taller de flujo simple, debido a varias razones:

1. No todos los trabajos pasan por las mismas estaciones.
2. La secuencia afecta a los tiempos en cada estación (debido a diferencias de materiales, máquina en la que se procesan dentro de la misma estación...)

Por ello, el problema tiende al denominado “problema de secuenciación de taller de flujo híbrido” (Ruíz y Vázquez-Rodríguez, 2009). Debido a que cumple las tres condiciones necesarias:

1. Hay al menos dos estaciones de procesamiento.
2. Las estaciones poseen k>=1 máquinas en paralelo.
3. Todos los trabajos pertenecientes a una misma ruta recorren las estaciones en el mismo orden

De esta forma, desde el marco teórico, el problema se aproxima más a un flowshop híbrido que a uno simple.

La discrepancia del problema genérico respecto a este objeto es el hecho de que el orden afecta al tiempo de procesamiento de los trabajos en las estaciones. Ello se debe a que las máquinas no son estrictamente paralelas ya que cada una trabaja a una velocidad diferente. Esto supone que dependiendo de la máquina en la que se procese un trabajo dentro de una misma sección, el tiempo de procesamiento del trabajo será uno u otro.

Los tiempos dependen de la secuencia debido a que, además de las discrepancias en velocidades nombradas anteriormente, la política de distribución de trabajos en máquinas en la empresa sigue la política de máquina vacía. Es decir, los trabajos se procesarán en la máquina que antes quede liberada.

Por lo que el mismo trabajo que recorre, por ejemplo, la estación de tintura (6 máquinas) podrá tardar más o menos tiempo en función de en la máquina en la que se procese. Si se coloca el trabajo 1 primero en la secuencia, se procesará en la máquina uno a una velocidad de 1 m/s, mientras que, si el trabajo 1 se coloca el segundo en la secuencia se procesará en la máquina 2 (la máquina 1 ya está procesando un trabajo) y se realizará a 1,25 m/s.

Corolario, la misma cantidad de metros y/o el mismo trabajo, se procesarán antes en la máquina dos que, en la uno, por lo que los tiempos de procesamiento serán diferentes y, por tanto, los tiempos dependen de la secuencia como se ha demostrado.

Está clase de problemas se encuentran contenidos en el conjunto de problemas NP-Hard, aquellos problemas de computación más complejos y cuyo corolario principal es que no se pueden resolver mediante un número N de iteraciones dentro de un tiempo humano (tiempo polinómico), es decir, que para resolver mediante fuerza (probar todas las posibilidades) se necesitaría un tiempo de iteración que supera en muchos casos los millones de años. (Sipser 2013)

Estos problemas se resuelven mediante la búsqueda de soluciones suficientemente buenas o, en muchos casos, no especialmente malas, lo que supone en sí mismo la definición de heurística (Fink, Voss, 2003)

3.3. SELECCIÓN DEL ÁREA DE APLICACIÓN

El principio (o la regla) de Pareto establece que en la mayoría de los escenarios el 80% de las consecuencias están sujetas al 20% de las causas. En una empresa, en muchos casos la mayoría de la facturación está sujeta a un reducido número de productos.

En el caso de esta empresa, los productos recorren la planta a través de las estaciones siguiendo un orden determinado, este orden es llamado "ruta". Debido a la gran cantidad de productos que posee la empresa, ésta cuenta con aproximadamente 30 rutas de productos.

Al realizar una agrupación por referencias y rutas, se observó que el 25% de la facturación estaba sujeta a dos rutas. Siendo que el 6% de las rutas suponen el 25% de la facturación, cumpliéndose Pareto.

Referencia producto	Tejido					
	Liso o Chenilla	Gramaje	Pol	Pol-Alg	Alg	Ancho
DIMMELSVIK D.GREY 206	liso	268		x		2,06
DIMMELSVIK D.GREY 223	liso	268		x		2,23
DIMMELSVIK D.BEIG 206 JIGGER	liso	268		x		2,06
DIMMELSVIK D.BEIG 206	liso	268		x		2,06
DIMMELSVIK D.BEIG 218	liso	268		x		2,18
DIMM. D.BEIG 206	liso	268		x		2,06
DIMMELSVIK D.GREY 218	liso	268		x		2,18
DIMMELSVIK DARK GREY 223						
DIMMELSVIK D.BEIG-218 JIGG	liso	261		x		2,18
DIMM D.BEIG 218						
DIMMELSVIK BEIG 206	liso	268		x		2,06
DIMMELSVIK WHITE 230	liso	268		x		2,3
DIMMELSVIK WHITE 218	liso	268		x		2,18
DIMMELS.WHITE 208	liso	268		x		2,08
DIMMELSVIK WHITE 206	liso	268		x		2,06

Ilustración 4: Datos de facturación. Fuente: Elaboración propia a partir de la información de la empresa.

Fases Proceso							
EMBURRADO	LAVADORA	RAME 1	ESTAMPADO	TINTURA	RAME 2	CALANDRA	ENROLLADO
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1			2	3		4
	1				2		3
	1				2		3
	1				2		3
	1				2		3
	1				2		3

Ilustración 5: Datos de facturación. Fuente: Elaboración propia a partir de la información de la empresa.

KPIs Facturados		
Metros lineales	€	Precio Venta
319.719	264.827	0,83
224.328	185.314	0,83
152.243	122.797	0,81
143.974	116.201	0,81
91.959	74.353	0,81
83.413	67.357	0,81
68.515	56.587	0,83
34.630	28.743	0,83
28.296	22.738	0,80
20.482	16.521	0,81
188.484	107.917	0,57
173.461	82.741	0,48
71.414	34.065	0,48
54.949	25.862	0,47
34.008	16.222	0,48
1.689.876	**1.222.243**	
	25,87%	

Ilustración 6: Porcentajes de facturación. Fuente: Elaboración propia a partir de la información de la empresa.

Como ya ha sido comentado, las estaciones que aportan valor directo al producto de la empresa son tintura, rame, estampado y calandra. Emburrado y lavado establecen las condiciones iniciales, y enrollado es la forma de presentación. A la hora de elaborar estas tablas no se tuvo en cuenta la estación de emburrado, ya que el producto que se emburra va al almacén de materia prima para ser lavado posteriormente. La estación de emburrado no es tratada como un proceso directo del producto ya que se realiza o no en función de como se haya entregado, por eso no se pueden incluir las referencias ni el proceso en las rutas, debido a que es un proceso ocasional.

Lo que supone que, en estas tablas, aunque no aparezca reflejado, se encuentran contenidos los productos que además de por esas estaciones, previamente han sido emburrados.

El software desarrollado a lo largo de este TFG estará orientado a trabajar con estas dos rutas incluyendo en ambas la sección de emburrado situándose de este modo en la peor situación posible, en la que en una secuencia el 100% de los trabajos deben ser emburrados.

Por lo que, la ruta 1 estará formada por: Emburrado – Lavado – Tintura – Rame – Enrollado

Así mismo, la ruta 2 estará formada por: Emburrado – Lavado – Rame – Enrollado

3.4. LENGUAJE PYTHON

El lenguaje que se usará para la elaboración de este software será el lenguaje Python.

La decisión de usar este lenguaje de programación se debe a diversos factores, por una parte, Python es un lenguaje altamente utilizado en todo tipo de campos lo que supone que tiene una gran cantidad de bibliografía, foros, blogs y formaciones donde buscar información, por otra parte, permite mostrar gráficos y tiene una gran cantidad de librerías que simplifican el proceso de programación (El Amri, 2024).

A lo largo del desarrollo del programa ha habido numerosas funcionalidades que han sido simplificadas gracias al uso de librerías de Python. Por ejemplo, el estudio muestral realizado en el análisis de resultados (ver punto "Análisis de resultados") se implementó con relativa facilidad gracias a las librerías de Python, o el proceso de leer datos de aplicaciones como Excel se simplificó mucho con la librería Pandas.

Por otra parte, en una función de la herramienta se implementó combinatoria de funciones, con la librería Itertools, las combinaciones se realizaban de forma automática y no fue necesario programar manualmente el proceso de combinatoria.

En la elaboración de este software se han usado las siguientes librerías:

- Numpy: Permite trabajar con matrices y vectores de forma intuitiva y orgánica.
- Matplotlib: Permite generar gráficos, entre ellos, diagramas de Gantt y diagramas de líneas
- Itertools: Es una librería de funciones estadísticas, en este caso se utiliza para implementar la combinatoria en una función.
- Pandas: Usado para extraer datos de otras aplicaciones, en este caso, de Excel.

Otra razón para usar Python fue su sintaxis. La forma de programar a través de Python es muy intuitiva cuando es fusionada con librerías, por ejemplo, en C++ para ver cual es el valor máximo de un vector, fue enseñado en la carrera que se tiene que recorrer el vector completo comparando resultados, en caso de Python y con el uso de Numpy, con tan solo escribir "np.max(nombre_vector)", Python es capaz de devolver el valor máximo, lo mismo con la media de un vector, la cual se puede obtener con "np.mean(nombre_vector)". De esta manera, numerosas funcionalidades que se querían aplicar, simplemente con intuición y sin la necesidad de buscar información se podían llevar a cabo. Así mismo, el código de Python es mucho más legible al agrupar las funciones en lugar de con paréntesis, con sangrías.

Por todas estas razones entre otras como la portabilidad o la integración con otros lenguajes, se decidió usar Python para el desarrollo de la herramienta (Schurrmann, 2024).

4. PROGRAMA

Una vez explicado el tipo de problema dado, la configuración de la empresa y el lenguaje que se usará para el desarrollo del software, así como el campo de actividad sobre el cual se trabajará, se explicará el desarrollo de la herramienta. En las siguientes páginas se explicará todo sobre el programa desarrollado. Tanto de las funciones implementadas, la lógica que sigue el software para realizar las acciones y los resultados que son entregados.

El programa está organizado por funciones, en las que cada una de ellas es paralela e independiente a las otras, de esta forma, el programa acepta las modificaciones y mejoras de mejor forma, al tener que crear una nueva función en caso de implementar nuevas opciones, o tan solo modificar la variable que devuelve la función en caso de modificaciones.

Primero se explicarán los datos de partida con los que funcionará el programa y posteriormente las funciones que son usadas para el proceso de cálculo. El programa está dividido en dos bloques principales, por una parte, el bloque de cálculo que será tratado en este punto, y por otra parte el bloque de soluciones, el cual se tratará en el punto 5.

El bloque de cálculo se compone de las funciones que son usadas únicamente para el cálculo de fechas de fin, mientras que el bloque de soluciones es el que, sirviéndose del bloque de cálculo, mejora las secuencias.

4.1. DATOS INICIALES

Para que el programa funcione es necesario enviarle una serie de datos iniciales desde Excel. Debido a que Excel es un programa muy extendido y usado no solo por la empresa, en pro de aumentar el utilitarismo de este programa se decidió que los datos de partida fueran enviados desde este programa. Además, Excel permite la gestión de tablas de forma sencilla, lo cual combinado con la librería Numpy facilita la gestión de matrices.

Los datos en cuestión están formados por una matriz de N filas (N= número de trabajos) por 4 columnas.

Trabajo	Ruta	Metros a procesa	Tipo de tejido
1	1	500	1
2	1	880	1
3	1	600	1
4	1	1000	1
5	1	600	2
6	2	380	2
7	1	650	2
8	2	2000	3
9	1	350	3
10	2	960	3
11	1	2000	1
12	1	650	3
13	2	300	1
14	2	700	1
15	1	1500	1
16	1	2780	3
17	1	1200	2

Ilustración 7: Matriz de ejemplo de datos iniciales. Fuente: Elaboración propia.

La primera columna indica el número del trabajo, la segunda la ruta que sigue (ver Glosario), la tercera columna la cantidad de metros y la cuarta columna el tipo de tejido. Este programa ignora los colores de los tejidos debido a que en estas rutas suelen ser similares.

En el caso de la segunda columna los valores están limitados a dos opciones, el 1 y el 2. Debido a que no hay más rutas. En caso de que el valor sea otro, el programa ignorará esa fila, es decir, ese trabajo no será tratado por el programa.

Por otra parte, la última columna solo acepta tres tipos de valores, el 1, 2 y 3. El programa no necesita que se especifique el tipo de tejido, debido a que, a efectos prácticos, lo que importa es que se diferencie cuáles son diferentes y se agrupen los iguales. Esto es necesario para la sección de lavado principalmente como ha sido explicado en la parte de “Procesos y Máquinas”

En la empresa, se deben lavar los productos que estén hechos del mismo material a la vez y, por ello es importante que se indique cuáles son iguales y cuales diferentes, debido a que para que los resultados sean reales, no puede empezar a lavarse producto tipo 2, hasta que el último producto de tipo 1 haya salido de la lavadora y de esta forma para todas las combinaciones posibles.

Al igual que con la columna 2, en caso de que el valor sea uno diferente a 1, 2 o 3, el programa ignorará esa fila, es decir, ese trabajo no será tratado por el programa.

El hecho de que se ignoren los trabajos que no cumplen los requisitos de ruta o de tipo de producto, se debe a que, de esta forma, la hoja de Excel de datos iniciales puede ser una genérica de la empresa. Así no se requiere de preparación previa de la hoja para la gestión de estas dos rutas mediante el programa

4.2. FUNCIONES

Las funciones son elementos programados que reciben una serie de variables (input), las operan y devuelven una serie de valores (output).

Cuando se programa por funciones en cualquier lenguaje de programación, es común referirse al uso de estas como "llamada a la función". La acción de llamar a la función consiste en la acción de enviarle los datos de entrada mediante la sintaxis adecuada, la cual depende del lenguaje de programación.

La forma de programar por funciones es una forma de optimizar, por una parte, el espacio usado y por otra, la cantidad de información que se trata. Al utilizar funciones para realizar labores sencillas como leer una tabla, o extraer determinados valores, se permite que esas mismas funciones se puedan usar múltiples veces llamando a la función. De lo contrario, cada vez que se quisiera realizar una acción, se debería programar a mano.

En el caso de Python, la llamada a la función se realiza escribiendo en una línea de texto, el nombre de las variables de salida (output), posteriormente un igual y finalmente el nombre de la función poniendo entre paréntesis el nombre de las variables de entrada (input). Como se observa en la siguiente ilustración.

```
Wr1=calcWR(works, Njobs)
Wm1=calcWM(works, Njobs)
Wtej1=calcWTEJ(works, Njobs)
VectorRuta=calcvector(works, Njobs)
```

Ilustración 8: Llamada a diferentes funciones. Fuente: Elaboración propia a través del programa.

El funcionamiento de este programa se basa en un flujo de intercambio de datos entre las diferentes funciones, en las cuales unas crean los datos que son posteriormente enviados a otras que los modifican. De esta forma cada función tiene, su objetivo.

La información es extraída de Excel, esta información es gestionada por una serie de funciones, y pasada esa instancia, las funciones envían y reciben entre ellas los distintos datos hasta lograr el cálculo de las fechas de fin.

Una función se encarga de leer los datos de Excel, otras cuatro gestionan el tipo de datos. Esta información procesada se envía a una función que construye las rutas y distribuye los trabajos en las máquinas, posteriormente, se envía esa información a otra que transforma las unidades de distancia (metros de cada trabajo) en unidades de tiempo en función de las diferentes velocidades de las máquinas. Finalmente, una función calcula las fechas finales con la información de tiempo.

El esquema siguiente muestra el flujo de información que sigue el programa para la elaboración de la matriz de fechas finales. En él se incluyen todos los valores que son creados, enviados y recibidos entre las diferentes funciones, es decir, muestra al completo el flujo de información del programa.

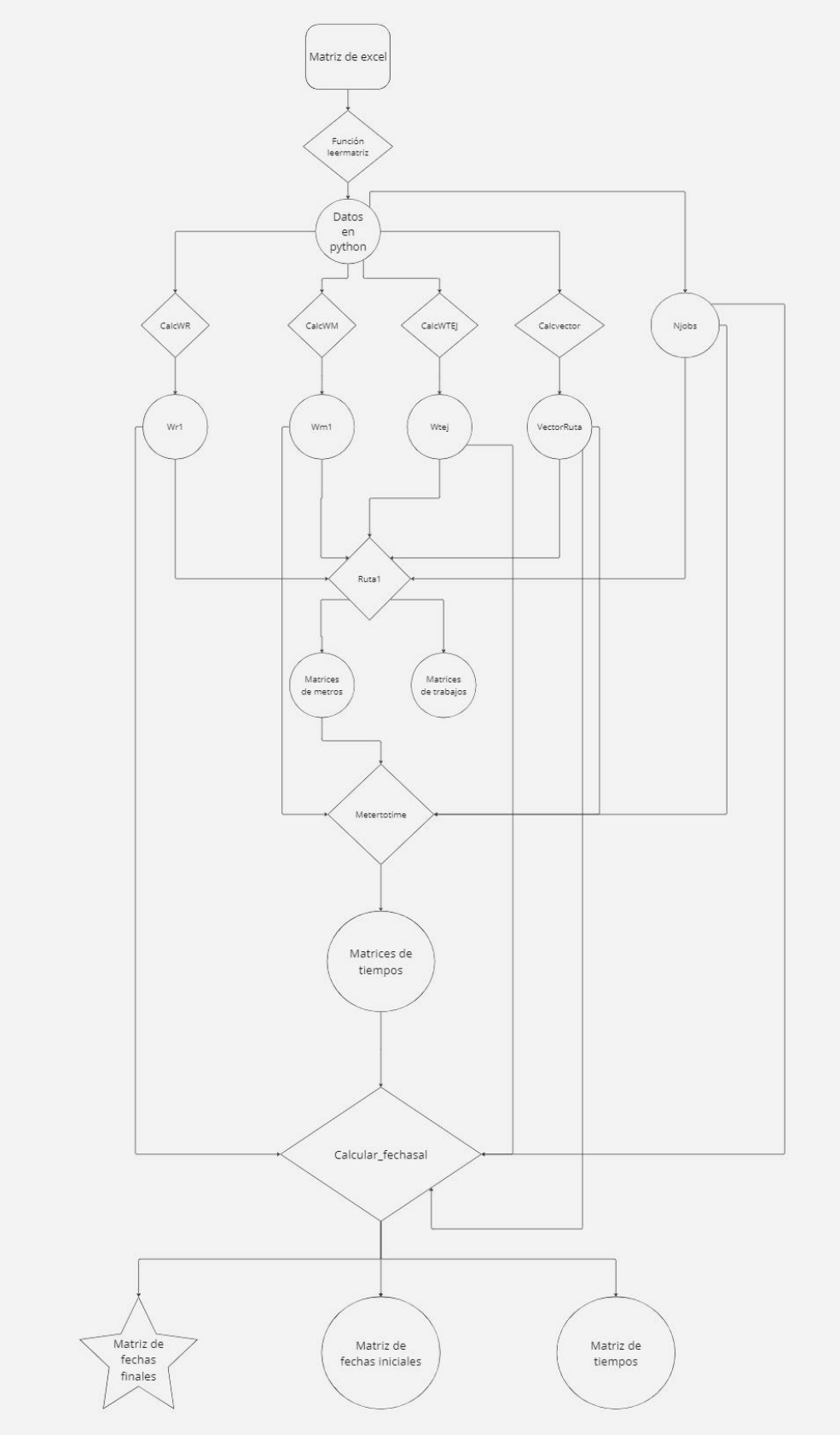

Ilustración 9: Bloque de cálculo y flujo de información. Fuente: elaboración propia con la app "Miró"

En la imagen se observan con rombos las funciones, con círculos los datos generados por funciones y en el cuadrilátero los datos iniciales de Excel. Finalmente, y en forma de estrella se observan los datos solución.

A continuación, se explicarán todas las funciones usadas en el proceso de cálculo que se observa en la ilustración 9 y que componen el "Bloque de cálculo" nombrado anteriormente.

Función "leermatriz"

En el punto anterior se ha hablado de los datos iniciales, los cuales parten de Excel. Para poder transferir la información de Excel a Python es necesario que Python acceda a esa hoja de Excel y transfiera a Python los datos de esa hoja. Esta función recibe el nombre del fichero de Excel que se desea leer como dato de entrada (para que el programa pueda ser usado con diferentes ficheros de diferentes usuarios), y extrae la matriz de datos iniciales.

Además, también permite especificar el nombre de la hoja del fichero. De este modo, se puede usar la herramienta con un mismo fichero formado por numerosas hojas, y trabajar con cada hoja de forma independiente.

Funciones de apoyo "calcWR, calcWM, calcWTEJ, calcvector"

Todas estas funciones han sido agrupadas debido a que son usadas, como su nombre indica, como funciones de apoyo. Todas ellas reciben los mismos datos, la matriz de datos iniciales y el número de trabajos "Njobs" y devuelven un vector de N espacios, en el que cada espacio corresponde a un trabajo. Es decir, la posición uno, corresponde al trabajo 1, la dos al 2...

Cada función se encarga de agrupar una serie de datos los cuales serán posteriormente usados por otras funciones del programa.

La función "calcWR" devuelve el vector con el número de cada trabajo. Este vector recibe el nombre de "Wr1"

La función "calcWM" devuelve el vector con los metros que deben ser procesados de cada trabajo. Este vector recibe el nombre de "Wm1"

La función "calcWTEJ" devuelve el vector con el tipo de tejido de cada trabajo. Este vector recibe el nombre de "Wtej"

Finalmente, la función "calcvector" devuelve el vector con el tipo de ruta de cada trabajo. Este vector recibe el nombre de "VectorRuta"

Si los datos iniciales fueran:

Trabajo	Ruta	Metros a procesa	Tipo de tejido
1	1	500	1
2	1	880	1
3	1	600	1
4	1	1000	1
5	1	600	2
6	2	380	2
7	1	650	2
8	2	2000	3
9	1	350	3
10	2	960	3

Ilustración 10: Datos iniciales de ejemplo Fuente: Elaboración propia.

Los valores que devolverían las funciones calcWR, calcWM, calcWTEJ, calcvector respectivamente serian:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[500, 880, 600, 1000, 600, 380, 650, 2000, 350, 960]

[1, 1, 1, 1, 2, 2, 2, 3, 3, 3]

[1 1 1 1 1 2 1 2 1 2]
```

Ilustración 11: Valores que tomarían los diferentes vectores. Fuente: Elaboración propia a través del programa.

El tratar los datos de esta manera permite trabajar con vectores pequeños en lugar de con una gran matriz, y permite tratar la información de manera más precisa lo que reduce la complejidad y la probabilidad de errores.

También permite un acceso más orgánico a la información, al enviar tan solo la que estrictamente necesita la función. Como se puede observar en la ilustración 9, hay funciones como “metertotime” que sólo acceden a un vector de los cuatro anteriormente nombrados, al enviarse únicamente el vector requerido, el flujo de información es más eficiente lo que se traduce en un menor tiempo de procesamiento y en un mejor flujo de información.

Funciones activas “ruta1, metertotime, calcular_fechasal”

Al igual que las primeras secciones de la empresa sirven para preparar la materia prima previamente a modificarla, las primeras funciones de este programa se encargan única y exclusivamente de la

gestión de la información para posteriormente usarla en los cálculos pertinentes (funciones de apoyo).

Las siguientes funciones se podría decir que son las funciones activas, las que aportan valor al programa, al igual que las secciones de tintura, rame y estampación lo hacen.

La primera función activa que usa el programa es la función "ruta1" esta función recibe los 4 vectores devueltos por las funciones de apoyo además del número de trabajos (Njobs). Esta función se encarga de repartir los trabajos en las diferentes secciones según la política de máquina vacía, es decir, introduce los trabajos en aquellas máquinas que, o bien no tienen ningún trabajo adjudicado o en aquellas que tienen el menor número de trabajos.

Para ello la función crea 5 matrices diferentes: emburrado, lavado, tintura, rame y presanch; una para cada sección. Estas matrices tienen un tamaño de M filas, siendo M el número de máquinas que tiene la sección (ver apartado "Procesos y Máquinas) y por N columnas, siendo N el número de trabajos.

Así mismo, también crea otras 5 matrices extra, pero esta vez en lugar de indicar el trabajo que es, indica el número de metros de cada trabajo.

Estás dos funciones son llenadas en el orden que marcan los vectores y siguiendo la política antes nombrada. Manteniendo los datos iniciales del punto anterior (ilustración 10), los valores devueltos por la función serian de la siguiente forma para todas las funciones:

```
Emburrado
 [[ 1.  3.  5.  7.  9.  0.  0.  0.  0.  0.]
 [ 2.  4.  6.  8. 10.  0.  0.  0.  0.  0.]]


Metros que se van a procesar en emburrado
 [[ 500  600  600  650  350    0    0    0    0    0]
 [ 880 1000  380 2000  960    0    0    0    0    0]]
```

Ilustración 12: Valor de la sección de emburrado devuelto por la función "Ruta1". Fuente: Elaboración propia a través del programa.

.

Tomando el ejemplo de emburrado, esta sección posee dos máquinas, por ello, la matriz tiene dos filas; viendo los valores de la matriz superior, se observa que los trabajos 1, 3, 5, 7 y 9 se procesarán en la primera máquina mientras que el resto en la segunda, ver que la sucesión de números de números se de esta forma también permite verificar que la matriz se ha llenado según la política de máquina vacía.

Por otra parte, en la matriz inferior se observa que el número del trabajo ha sido sustituido por los metros del trabajo.

La segunda función activa que usa el programa es la función "metertotime", esta función recibe el vector "Wm1", el "VectorRuta", "Njobs" y todas las matrices de metros devueltas por la función anterior (equivalentes a las mostradas en la ilustración 12).

Como bien indica el nombre de la función, ésta se encarga de hacer la conversión de unidades de distancia a unidades de tiempo, en este caso, de metros a segundo. Esto lo realiza mediante el uso de las velocidades de procesamientos de las diferentes máquinas, que como se indicó en "Definición del problema", son diferentes para cada máquina.

De forma que, en el caso de emburrado, los trabajos 1, 3, 5, 7 y 9, al ser procesados en la máquina uno, se realizarán a un tiempo aproximado de 1 s/m mientras que el resto, al ser procesados en la siguiente máquina, se realizarán el doble de lentos a 2 s/m. En esta instancia, se observa de forma explícita que cada trabajo tendrá una velocidad diferente dependiendo de la secuencia, y se observa por qué no se puede hablar íntegramente de máquinas paralelas.

Manteniendo los datos iniciales del primer punto y del anterior, las cinco matrices devueltas por la función "metertotime" serán equivalentes a la siguiente:

```
Emburrado
 [[ 500.  600.  600.  650.  350.    0.    0.    0.    0.    0.]
 [1760. 2000.  760. 4000. 1920.    0.    0.    0.    0.    0.]]
```

Ilustración 13: Tiempos de procesamiento de los 10 trabajos en la sección de emburrado. Fuente: Elaboración propia a través del programa.

Como se observa, los datos de la primera fila (máquina 1) no han sido modificados, ya que la velocidad es de 1 m/s, mientras que los de la segunda fila (máquina 2) son el doble que el valor en metros, debido a que la velocidad es de 0,5 m/s.

Recapitulando, el programa ha leído información de un Excel, ha gestionado los datos, ha localizado trabajos en máquinas, ha transferido la capacidad en metros que recae en cada máquina y ha calculado cuanto tiempo de procesamiento se requiere para cada trabajo con la secuencia dada.

La tercera y última función activa es la función "calcular_fechasal", como se ha dicho a lo largo de este trabajo, la problemática de la empresa reside en que no es capaz de conocer las fechas a las que terminan los trabajos de forma precisa, y por tanto no es capaz de dar fechas exactas a los clientes. En este momento, el programa ya es capaz de realizar esos cálculos.

La función "calcular_fechasal" recibe las cinco matrices (una por cada sección) devueltas por la función "metertotime", también recibe: Njobs, Wr1, Wtej1, VectorRuta.

Esta es la función más compleja del programa debido a que tiene numerosos limitantes que se deben tener en cuenta, por una parte la suma de tiempos no se puede realizar de forma directa debido a que las matrices no son iguales (p.ej emburrado 2xN, tintura 6xN) por otra parte, se debe de tener en cuenta el tipo de tejido, ya que en la sección de lavado, pueden procesarse simultáneamente los productos del mismo tipo de tejido, pero hasta que no termina **por completo** un tipo, no puede procesarse el siguiente.

Sumado a lo anterior, en los casos de flowshop estándar (aquellos flowshops en los que tan solo hay una máquina por sección), el cálculo de fechas de fin se realiza mediante un proceso lógico que consiste en lo siguiente: Siendo i el número del trabajo y j el número de la máquina.

1. Fecha de fin del trabajo anterior en la máquina actual (posición [i-1,j]) = A
2. Fecha de fin del trabajo actual en la máquina anterior (posición [i,j-1]) = B
3. Fecha de fin del trabajo actual = Valor máximo entre A y B + Tiempo que tarda en procesarse el trabajo actual en la máquina actual

Computacionalmente:

```
for j in range(1,Njobs):
    for i in range(1,Nmaq):
        Fechas_fin[j,i]=max(Fechas_fin[i-1,j],Fechas_fin[i,j-1])+tiempos_proceso[secuencia[i],j]
```

Ilustración 14: Programación del proceso lógico. Fuente: Elaboración propia a través del programa.

Ese proceso lógico se hace la pregunta, "¿Que me va a limitar? ¿El tiempo en que se libera la máquina en la quiero procesar (A)? ¿O lo que tarda el trabajo que tengo que procesar en llegarme (B)?" Una vez determinado el limitante (máximo entre A y B) se suma el tiempo que tarda en procesarse el trabajo nombrado en la máquina en la que se quiere procesar, y de esta forma se habría calculado su fecha de fin.

Computacionalmente, el proceso lógico anterior para un conjunto de trabajos en un taller de flujo resulta de la forma:

```
#Calculo las fechas de fin del primer trabajo en la primera máquina
Fechas_fin[0,0]=tiempos_proceso[secuencia[0],0]
#Calculo las fechas de fin del primer trabajo en las siguientes máquinas
for i in range(Nmaq):
    Fechas_fin[0,i]=Fechas_fin[0,i-1]+tiempos_proceso[secuencia[0],i]
#Calculo las fechas de fin del resto de trabajos en la primera máquina
for j in range(1,Njobs):
    Fechas_fin[j,0]=Fechas_fin[j-1,0]+tiempos_proceso[secuencia[j],0]
#Calculo las fechas de fin del resto de trabajos en las máquinas a partir de la segunda
for j in range(1,Njobs):
    for i in range(1,Nmaq):
        Fechas_fin[j,i]=max(Fechas_fin[j-1,i],Fechas_fin[j,i-1])+tiempos_proceso[secuencia[j],i]
for j in range(Njobs):
    for i in range(Nmaq):
        Fechas_ini[j,i]=Fechas_fin[j,i]-tiempos_proceso[secuencia[j],i]
```

Ilustración 15: Proceso de cálculo de fechas fin en un conjunto de trabajos en un flowshop simple. Fuente: elaboración propia de ejemplo.

El cual se basa en comparar las posiciones anteriores (de máquina [i,j-1] y de trabajo [i-1,j]) de una matriz de fechas finales "fecha_fin" con la posición actual (de trabajo y máquina) fecha_fin[i,j]. En resumen, comprobar qué valor es superior, A o B.

En el caso del Flowshop hibrido, la lógica seguida es la misma, pero el cálculo de B se dificulta al haber más de una máquina por sección, debido a que no es un cálculo inmediato como si es en el

caso del flowshop simple. Cálculo simple que se puede observar en la ilustración 15, en el que B=fecha_fin[i,j-1].

Esto se debe a que la matriz de fechas finales referencia esas fechas **en la sección**, no en la máquina. En el caso del flowshop simple **siempre** hay una máquina por sección, por eso B siempre es "fecha_fin[i,j-1]".

Dada la matriz de fechas finales siguiente:

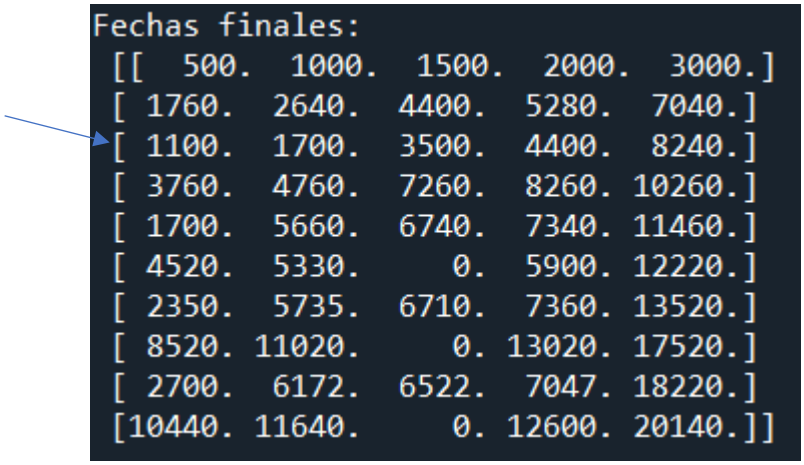

Ilustración 16: Matriz de fechas finales. Fuente: elaboración propia a través del programa.

Esta matriz de fechas finales representa a la planta. Donde cada línea es un trabajo y cada columna una sección, siendo la primera columna la sección de emburrado, la segunda lavado... y siendo la primera línea el trabajo 1, la segunda el trabajo 2...

En el caso de realizar el cálculo de estas fechas finales siguiendo el modelo del flowshop simple, el cálculo del valor marcado (1100 en la primera columna), sería de la forma: Valor de A = 0 (debido a que es la primera sección y anteriormente no hay nada) valor de B = fecha_fin[i, j-1]. Es decir, valor de B = fecha_fin [2,1]=1760.

Al ser B>A, al valor de B se le sumaria el tiempo de procesamiento de ese trabajo en esa máquina, que como se observa en la ilustración 13, es 600. Sin embargo, al hacer el cálculo de esa manera, se asume que tan solo hay una máquina por sección.

Como se observa en la ilustración 13, emburrado posee dos máquinas, y este trabajo, el trabajo 3, se procesa en la máquina 1. Mientras que el trabajo 2, se procesa en la máquina dos (ver ilustración 13, posición de la matriz [2,1]= 1760). Por lo que realizar el cálculo mediante B = fecha_fin [i, j-1], sería erróneo, ya que realmente el trabajo 2 y el 3 se procesan en diferentes máquinas, y por tanto B no depende de la posición anterior, la cual refleja al trabajo 2.

Al procesarse el trabajo 3 en la primera máquina de la sección, el trabajo 3 depende del trabajo que se procesa antes que él en la primera máquina, que en este caso es 1. De esta forma, B = fecha_fin [i, j-2]= 500 (como se observa en la ilustración 13). Ya que el trabajo 1 y 3 se procesan en la misma máquina.

En el flowshop simple, al sólo haber una máquina por sección, siempre se comparan las posiciones anteriores, pero en los modelos híbridos al haber más máquinas, es erróneo realizar los cálculos de esta manera como se ha demostrado.

En un modelo híbrido la fecha de fin del trabajo actual en la máquina/sección anterior, el cálculo de B, no tiene por qué ser fecha_fin[i,j-1]. Ya que en un modelo híbrido puede haber más de una máquina por sección. Por ello, el cálculo de B depende del número de máquinas que tenga la sección, en caso de 3 máquinas sería fecha_fin [i,j-3], en caso 4 máquinas sería fecha_fin [i,j-4]... Lo que supone que el cálculo de las fechas de fin no se puede realizar en un único bucle sencillo como en el caso de flowshop simple.

Por ello, la forma de solucionar este problema se basó en dividir los cálculos. Por una parte, se realizó una matriz Z que calculaba y almacenaba las fechas de fin de cada trabajo **dentro de la sección** (el valor que se utiliza para el cálculo de B) y por otra parte se realizó un bucle simple que en lugar de trabajar con fecha_fin[i,j-M] (siendo M el número de máquinas), como trabaja el flowshop simple, usaba las posiciones de la matriz Z. Es decir, se sustituyó fecha_fin[i,j-M], por matriz_Z [i,j].

De esta forma en un bucle simple se podían comparar A y B, A mediante fecha_fin[i-1,j] y B mediante matriz_Z[i,j], sin que hubiera errores en el cálculo de B.

En un flowshop simple, el modelo computacional sería de la forma:

```
for j in range(1,Njobs):
    for i in range(1,Nmaq):
        Fechas_fin[j,i]=max(Fechas_fin[i-1,j],Fechas_fin[i,j-1])+tiempos_proceso[secuencia[i],j]
```

Ilustración 17: Flowshop simple. Fuente: elaboración propia a través del programa.

Es decir, la comparación de las posiciones anteriores de máquina y trabajo.

Mientras que en un modelo híbrido seria de la forma:

```
fechafinR[i,j]= max(matriz_Z[i, j], fechafinR[i-1, j]+ times1[i,j])
```

Ilustración 18: Flowshp híbrido. Fuente: elaboración propia a través del programa.

Donde la matriz Z ha debido ser previamente calculada, a diferencia del modelo simple, que no requiere de cálculos previos.

De esta forma la función calcula las fechas de fin sin errores. La función “calcular_fechasal” devuelve el cálculo de las fechas de fin de todos los trabajos. Que para el caso de los datos iniciales de la ilustración 10 resulta ser:

```
Fechas finales:
 [[  500.  1000.  1500.  2000.  3000.]
 [ 1760.  2640.  4400.  5280.  7040.]
 [ 1100.  1700.  3500.  4400.  8240.]
 [ 3760.  4760.  7260.  8260. 10260.]
 [ 1700.  5660.  6740.  7340. 11460.]
 [ 4520.  5330.     0.  5900. 12220.]
 [ 2350.  5735.  6710.  7360. 13520.]
 [ 8520. 11020.     0. 13020. 17520.]
 [ 2700.  6172.  6522.  7047. 18220.]
 [10440. 11640.     0. 12600. 20140.]]
```

Ilustración 19: Matriz solución, fechas finales Fuente: elaboración propia a través del programa.

Esta función debía de respetar otra restricción, el trabajar con varias rutas. El hecho de usar una sola función y matriz, en pro de unificar la información, para trabajar con varias rutas supone que a la hora de realizar los cálculos se debe de tener en cuenta que no todos los trabajos pasan por todas las secciones. Por eso, en la tercera columna, hay varios espacios con ceros, si se observa la ilustración 10 debido a que esos trabajos corresponden a la ruta 2, y la columna 3 corresponde a la sección de tintura, y como ha sido dicho, la ruta 2 no pasa por tintura.

Así mismo la función compila todos los tiempos de cada trabajo en cada sección para unificarlos en una matriz de tiempos del mismo tamaño que la matriz de fechas. Esta matriz es la unificación de todos los valores devueltos por “metertotime” en un conjunto único:

```
Tiempos:
 [[ 500.  500.  500.  500. 1000.]
 [1760.  880. 1760.  880. 1760.]
 [ 600.  600. 1800.  900. 1200.]
 [2000. 1000. 2500. 1000. 2000.]
 [ 600.  900. 1080.  600. 1200.]
 [ 760.  570.    0.  570.  760.]
 [ 650.  975.  975.  650. 1300.]
 [4000. 2500.    0. 2000. 4000.]
 [ 350.  437.  350.  525.  700.]
 [1920. 1200.    0.  960. 1920.]]
```

Ilustración 20: Matriz de tiempos unificada. Fuente: elaboración propia a través del programa.

.

Función gráfica “dibujar_gantt”

Para que la observación de los resultados fuera mejor se añadió una graficación mediante un Gantt.

El diagrama de Gantt es un gráfico o herramienta de gestión, que muestra un cronograma de actividades. Se basa en una serie de barras horizontales que, en este caso, representan cada uno de los trabajos. Cada barra incluye separaciones por colores (un color para cada sección) según las fechas de inicio y fin de cada trabajo en cada sección. Todo ello acompañado de una línea de tiempo que permite visualizar las nombradas fechas. Este tipo de gráficos ayudan a visualizar y controlar, así como a mejorar y optimizar secuencias (Viana y Sousa, 2000)

La función “dibujar_gantt” recibe las fechas de fin y de inicio, y la matriz de tiempos devueltas por la función “calcular_fechasal”, también recibe “Njobs” y “Wr1”, que a todos los efectos es la secuencia dada.

EL Gantt resultante es el siguiente:

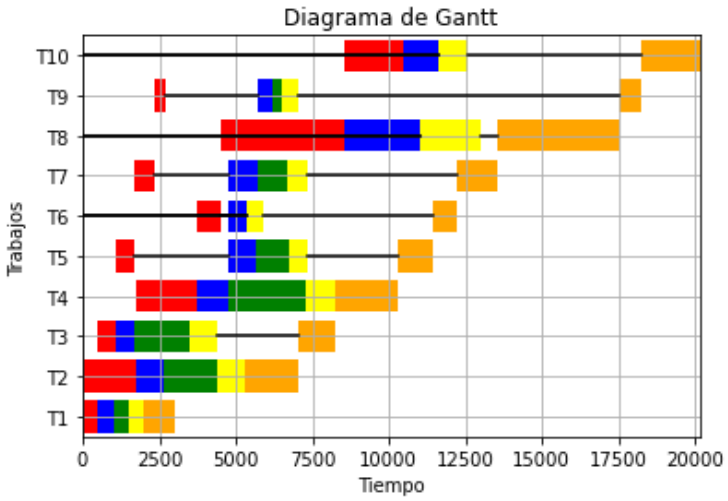

Ilustración 21: Ejemplo de diagrama de Gantt. Fuente: elaboración propia a través del programa.

Como se observa, el Gantt muestra también el número del trabajo.

Y como cabía esperar, los trabajos 6, 8 y 10, no poseen línea verde, debido a que esta corresponde a la sección de tintura.

5. MEJORA DE LAS SECUENCIAS

Este TFG orbita alrededor del problema de la empresa sobre las fechas de fin, sin embargo, además de implementar la propia solución a la situación de las fechas, una vez realizada esa función, se implementaron diferentes métodos que mejoran las soluciones.

De esta forma, la empresa no solo puede conocer las fechas de fin de sus trabajos en curso, si no que puede comparar manualmente los resultados de las diferentes secuencias propuestas o puede usar algunas de las soluciones aportadas por el programa.

La mejora de las soluciones se realiza utilizando dos formas distintas, por una parte, se usa la regla NEH y por otra se usa una iteración gravitatoria, así mismo, las mejoras se realizan atendiendo a dos parámetros diferentes, el Cmed y Cmax.

De esta forma el usuario puede elegir que quiere mejorar según el criterio elegido (Cmax o Cmed), y obtiene resultados de dos formas distintas.

Por otra parte, el programa está diseñado de forma que acepta fácilmente la incorporación de nuevas funciones, en el último punto de este apartado se explicará el proceso a seguir para poder ampliar las formas de mejorar los resultados mediante nuevos algoritmos.

5.1. DEFINICIÓN DE PARÁMETROS

A la hora de mejorar un resultado, éste debe de ser mejorado respecto a un parámetro concreto.

En economía se habla de la riqueza de un país en función de su producto interior bruto, y se dice, entre otras cosas, que un país es más rico que otro en función de este parámetro.

En el caso de las secuencias, es necesario fijar respecto a que parámetros se desea mejorar, debido a que, si y solo si, se podrá decir que una solución es mejor que otra, cuando ambas están siendo comparadas bajo la misma perspectiva.

Por ello, una secuencia no es mejor a otra por sí misma, es mejor en cuanto que mejora en tanto más que la otra un parámetro determinado.

En este caso los dos parámetros comparativos o modelos para la mejora de las secuencias han sido el Cmax y el Cmed.

CMAX

El Cmax se puede definir como la fecha final en la que termina el último trabajo.

Las razones para mejorar este parámetro son dos principales, por una parte, la mejora del Cmax resulta en una mejora de toda la secuencia en sí misma, si se reduce la fecha de fin del último trabajo, es decir, la fecha mayor, necesariamente el resto de las fechas se verán reducidas, lo que supone una mejora en los tiempos de entrega de cada trabajo, acabando todos ellos en menor tiempo respecto a una secuencia aleatoria.

La segunda razón para elegir el Cmax como uno de los parámetros es debido a su gran uso y presencia en la literatura. Aproximadamente el 60% de los parámetros elegidos a lo largo de la literatura fijan el Cmax como función objetivo a mejorar (Hybrid flow-shop scheduling problem, Ruíz y Vázques-Rodríguez, 2009), esto quiere decir que la gran mayoría de los autores y en general, ingenieros, profesores... utilizan el Cmax como parámetro de mejora debido a los buenos resultados que aporta su minimización. En pro de dar perspectiva, el siguiente parámetro más usado como función objetivo es el C/F y es usado el 11% de las ocasiones.

CMED

El cmax a pesar de sus numerosas ventajas, también tiene sus desventajas. Su principal inconveniente es que se calcula tan solo con un valor de la secuencia, por lo que el resto de los valores se encuentran arrastrados a lo que marque el Cmax, además de que no se ven reflejados. Por esto mismo, el sumatorio de tiempos ociosos es mayor, debido a que no se tiene en cuenta lo que ocurre en el resto de los trabajos y por tanto no se pueden mejorar de forma directa.

El Cmed se puede definir como la media de finalización de todos los trabajos, y se calcula mediante una media aritmética de los tiempos de finalización de todos los trabajos de la secuencia.

El Cmed cubre perfectamente las desventajas del Cmax, al ser un mejor reflejo de cómo se comportan todos los trabajos en la secuencia al mostrar la variabilidad y la consistencia de la secuencia. Por lo que una minimización del Cmed tiende a alisar la variabilidad de las fechas de

finalización, y esa es una de las razones principales de su uso (Tavakkoli-Moghaddam, Rahimi-Vahed, Mirzaei, 2007) (Shore, Warden, 2007).

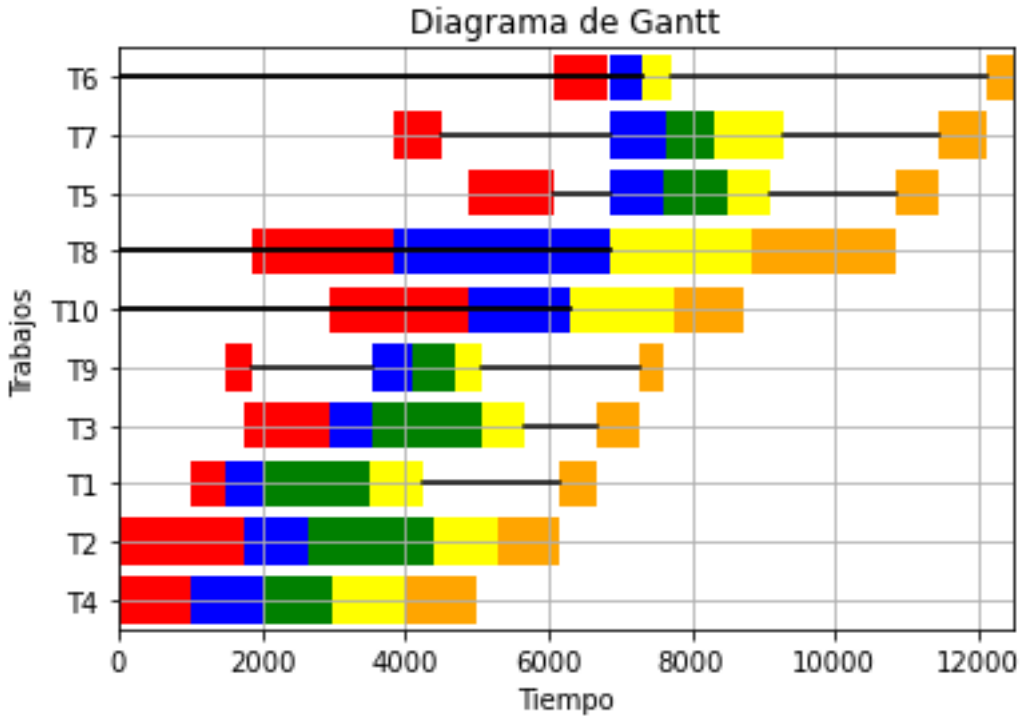

Ilustración 22: Secuencia mejorada con CMax. Fuente: elaboración propia a través del programa.

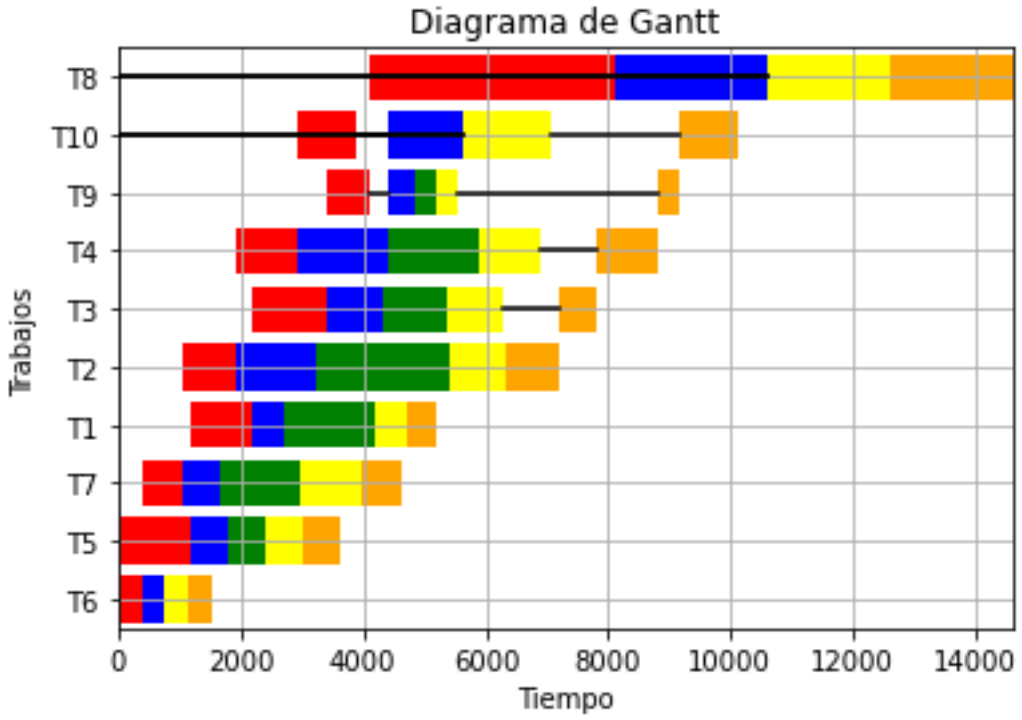

Ilustración 23: Secuencia mejorada con CMed. Fuente: elaboración propia a través del programa.

Como se observa en las imágenes, la diferencia en tiempos y secuencias de usar Cmax a usar CMed es notoria.

Por una parte, la secuencia mejorada con Cmax acabará estrictamente antes, es decir, el último trabajo de la secuencia acaba en el segundo 12480, a las 3h y media aproximadamente. Mientras que la secuencia mejorada con Cmed acabará después de las 4h.

Sin embargo, en la secuencia mejorada con Cmax se observa que los tiempos ociosos son mayores. Definiendo tiempos ociosos como tiempo que pasan los trabajos, desde que entran en la ruta hasta que salen, sin procesarse. Ello se puede observar en aquellas líneas situadas entre dos estaciones (entre colores), cuanta más cantidad de esas líneas y más largas sean, mayor será el sumatorio total de tiempos ociosos. Fijándose en la ilustración 22, se comprueba visualmente que la cantidad y longitud de líneas no coloreadas entre estaciones es considerablemente superior respecto a la ilustración 23.

Se realizó un estudio muestral para comprobar en qué medida el Cmed ayudaba a reducir los tiempos ociosos de las secuencias. Para ello se generaron 4000 matrices aleatorias de datos iniciales, de entre 10 y 20 trabajos, y entre 500 a 2500 m a procesar por trabajo. Así mismo se elegia de manera aleatoria tanto la ruta como el tipo de producto. La única modificación que se hizo a las matrices una vez generadas fue la agrupación de trabajos por tipo de tejidos para plantearlas en un origen de datos real, ya que esa agrupación por tipo de tejido es llevada a cabo siempre por la empresa.

Cada una de estas 4000 matrices aleatorias fue mejorada atendiendo a ambos parámetros, por una parte, se mejoraba el Cmax, se calculaban los tiempos ociosos y se guardaba el resultado, a continuación, esa misma matriz aleatoria se mejoraba atendiendo al Cmed y se guardaba el resultado de la suma de tiempos ociosos. De forma que se generaron 8000 soluciones en total.

Los datos obtenidos fueron los siguientes:

De las 4000 matrices generadas (el equivalente a más de 10 años de trabajo de la empresa), el Cmed daba menor tiempo ocioso en 2736 de ellas, es decir, el 68,4% de las veces, el Cmed lográ reducir los tiempos ociosos respecto al Cmax. Lo que supone que el Cmed reduce la presencia de tiempos ociosos con el doble de efectividad.

Así mismo, de esas 2736 matrices en las que el Cmed daba mejores resultados en la reducción de tiempos ociosos, la media de mejora fue del 38,78%. El Cmed lograba reducir en esa medida la presencia de tiempos ociosos en las secuencias.

Este proceso se repitió más de diez veces, habiendo revisado de esta forma más de 80.000 posibilidades, y todos los resultados daban la misma información con desviaciones menores al 10% en cada caso.

Por lo que se puede decir, que, en este problema en cuestión, el Cmed aporta aproximadamente un 40% de mejores resultados, en la reducción de tiempos ociosos, el doble de las veces.

Finalmente, a excepción del último trabajo de la secuencia (T6 en il. 22 y T8 en il. 23), todos los trabajos ordenados por Cmed acaban antes que los ordenados según Cmax, lo que supone que esas máquinas se quedarán vacías antes para aceptar nuevos trabajos.

En este aspecto también se realizó un estudio muestral de iguales magnitudes que el caso anterior, 4000 matrices aleatorias de las mismas características, más de 10 tandas generadas, sólo que en

este caso se comparaban las fechas de fin de todos los trabajos de la secuencia a excepción del último. Lo que se deseaba medir era en que medida el Cmed ayuda a reducir las fechas de fin de los trabajos de la secuencia a excepción del último trabajo (el cual siempre será menor al ser mejorado con Cmax por definición)

Se obtuvieron los siguientes resultados en la primera tanda de 4000 matrices:

De las primeras 4000 matrices, se analizaron 55.692 trabajos generados de forma aleatoria (al ser la cantidad de trabajos un número aleatorio entre 10-20 en cada una de las matrices, la cantidad de trabajos totales varia en cada batería). El 79,59% de esos trabajos tenían menor fecha de finalización al ser mejorada la secuencia con Cmed, que al ser la secuencia mejorada con Cmax. De la misma forma que se ha explicado anteriormente la evaluación consistía en la comparación de las posiciones de la secuencia.

Por ejemplo, dada la primera matriz aleatoria, se mejoraba su secuencia atendiendo a Cmax y a continuación atendiendo a Cmed, ambos resultados se guardaban. Acto seguido se comparaban las fechas de fin de los trabajos de la secuencia de la forma:

- "¿Qué valor es mayor, la fecha de fin del primer trabajo de la secuencia mejorada con Cmed o la fecha del primer trabajo de la secuencia evaluada con Cmax?
- En caso de ser el de Cmed se sumaba 1 en una variable, y en caso de ser Cmax se sumaba 1 en otra variable
- Finalmente se sacaban los porcentajes.

En la segunda tanda de 4000 matrices, se analizaron 55.524 trabajos. En el 80,05% de los cuales, la fecha de finalización era menor al mejorarse con Cmed. Lo que aportaba los mismos resultados que el caso anterior prácticamente.

Una vez más se repitió este proceso en más de 10 ocasiones, y los resultados aportados diferían en menos de un 2% entre ellos. De forma que, puede decirse que, en este problema concreto, los trabajos de la secuencia mejorada con Cmed (a excepción del último trabajo), acabarán el 80% de las veces antes.

En esta instancia se ve reflejada la importancia de usar varios parámetros. En función de lo que le interese a la empresa podrá usar uno u otro. En caso de necesidad de acabar lo antes posible la secuencia para cerrar, usará Cmax, por otra parte, si le interesa que la utilización de la máquina sea máxima, y que por tanto los tiempos ociosos mínimos, usará Cmed. También usará Cmed en caso de querer rotar rápidamente las máquinas.

5.2. REGLA NEH

La regla NEH, Nawaz, Enscore y Ham (1983), es un algoritmo heurístico que se usa para resolver problemas de secuenciación de trabajos. Esta regla a pesar de su sencillez aporta buenos resultados en la minimización (no sólo pero especialmente) del Cmax en comparación con su tiempo de ejecución

Hay variaciones de la propia regla y diferentes versiones, en este caso se ha usado la regla NEH original, que se basa en diferentes pasos.

El primer paso consiste en la ordenación de trabajos por suma de tiempos de procesamiento, se suman los tiempos de cada trabajo a través de todas las máquinas y se ordenan de forma descendiente.

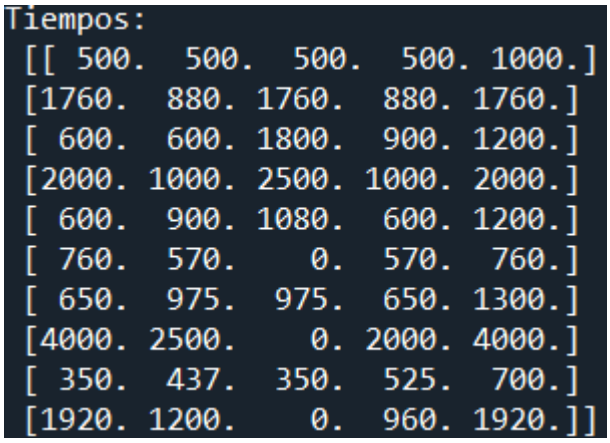

Ilustración 24: Tiempos de procesamiento de trabajos en máquinas. Fuente: elaboración propia a través del programa.

El NEH sumaría las filas de esta matriz, y posteriormente las ordenaría de mayor a menor.

La aplicación de la regla NEH en este caso posee un hándicap debido a que las máquinas no son paralelas. Como ya se ha dicho en puntos anteriores, la secuencia afecta a los tiempos de procesamiento debido al hecho de que cada máquina trabaja a un ritmo diferente, debido a esto, la lista ordenada que resulta del cálculo del sumatorio previamente nombrado se puede ver afectada.

La consecuencia que resulta de esto es que, dependiendo de la ordenación inicial de los trabajos, el NEH dará un resultado u otro. Todos buenos pero algunos mejores que otros.

Por ejemplo, tomando la secuencia inicial de [1,2,3,4,5,6,7,8,9,10]. El resultado de mejorar Cmax mediante el NEH será Cmax=13500 segundos = 3,75h

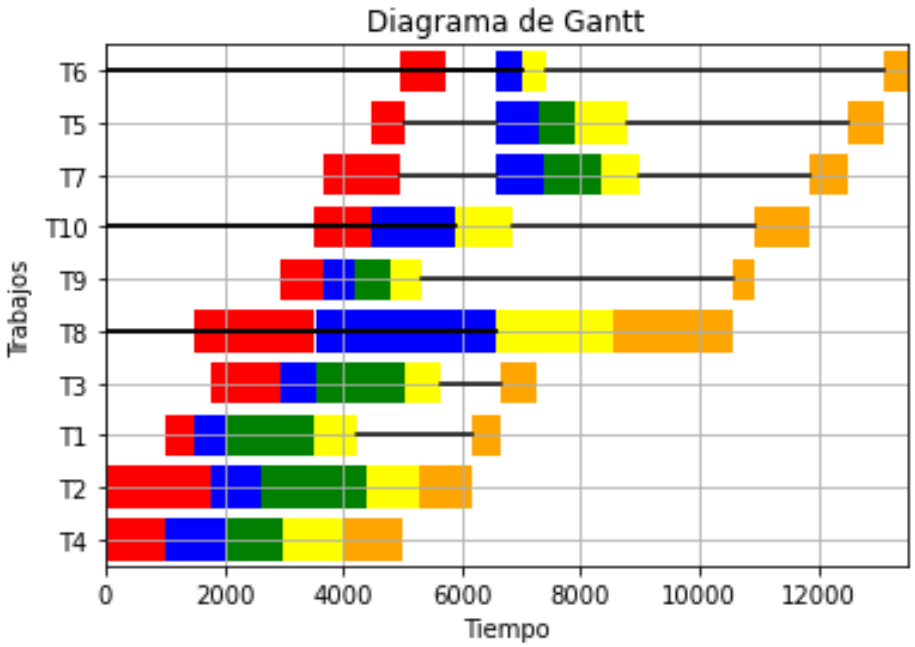

Ilustración 25: NEH mejorando Cmax con la secuencia 1-10. Fuente: elaboración propia a través del programa.

Sin embargo, si la secuencia inicial es [4, 2, 3, 1, 8, 9, 10, 7, 5, 6] el resultado de mejorar Cmax mediante el NEH será Cmax=11970 segundos =3,325h

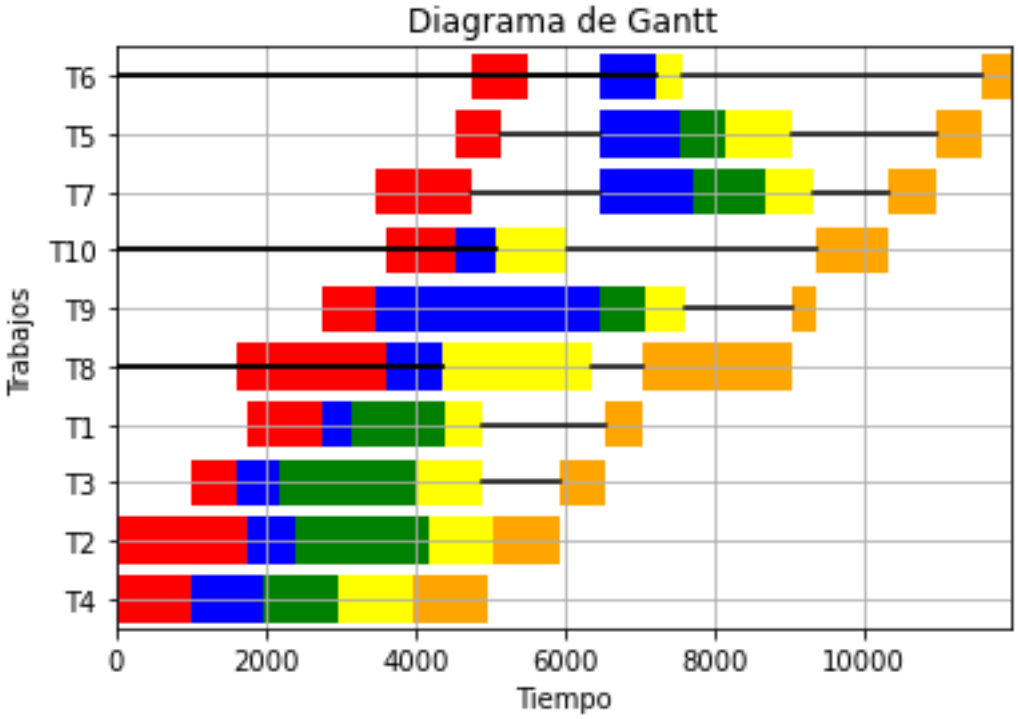

Ilustración 26: NEH mejorando Cmax con la secuencia [4, 2, 3, 1, 8, 9, 10, 7, 5, 6]. Fuente: elaboración propia a través del programa.

Se ha comprobado mediante miles de iteraciones, que la máxima diferencia de resultados del NEH mejorando Cmax dependiendo de la secuencia inicial, es del 12%, y es la diferencia mostrada en las pasadas imágenes.

Mientras que en el caso del NEH mejorando Cmed, la diferencia de resultados en función de la secuencia es menos del 5%, con lo que una vez más se comprueba, que a todos los efectos el Cmed es menos variable que el Cmax.

Continuando con los pasos del NEH, la segunda fase una vez ordenada la secuencia, consiste en la elaboración de la secuencia inicial. Se selecciona el primer trabajo de la lista ordenada, y se coloca en la primera posición de la secuencia del NEH, posteriormente, se selecciona el segundo trabajo de la lista ordenada y se inserta en todas las posiciones posibles de la secuencia calculando el valor del parámetro. Finalmente se elige la secuencia que más mejora el parámetro.

A partir de este momento, se construye mediante un proceso iterativo la secuencia siguiendo el paso dos. Se selecciona el tercer trabajo de la lista ordenada, se inserta en todas las posiciones, se comprueba el parámetro y se escoge la secuencia que mejore el parámetro.

Dada una secuencia de 4 trabajos = [0,1,2,3]. El NEH realizaría lo siguiente:

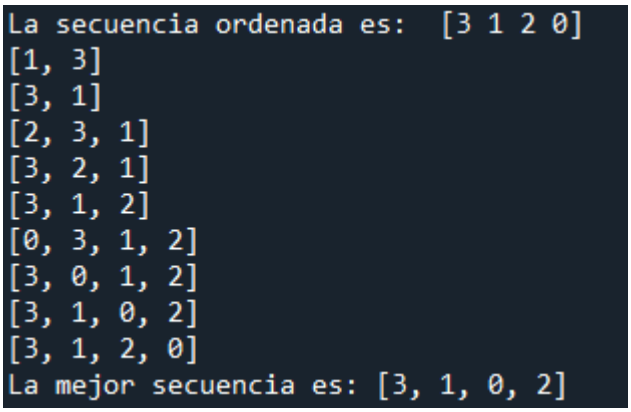

Ilustración 27: NEH en funcionamiento. Fuente: elaboración propia a través del programa.

Primero ordenaría los trabajos según la suma de la matriz, resultado en la secuencia ordenada que aparece en la imagen, a continuación, realizaría el proceso iterativo como se observa, probando cada valor en todas las posiciones según se va añadiendo a la secuencia. Finalmente devolvería la mejor secuencia

5.3. ITERACIÓN GRAVITATORIA

Como se ha explicado anteriormente, una iteración consiste en la prueba de opciones y/o configuraciones en pro de refinar o mejorar un resultado. Una iteración aleatoria consiste en realizar este proceso obteniendo las diferentes configuraciones mediante una generación aleatoria.

Los procesos iterativos se usan comúnmente para resolver resultados a la fuerza, es decir, probando todas las posibles soluciones y seleccionando la mejor de ellas. El problema reside en que, en la mayoría de las ocasiones, la cantidad de resultados a probar supera por mucho la capacidad de tiempo humana, por lo que se imposibilita comprobar todos los resultados (Stützle, 1998)

En el caso de los problemas con secuencias aleatorias, la cantidad de soluciones es, por definición, la exponencial de la cantidad de trabajos. De forma que las posibles maneras de ordenar una secuencia de 10 trabajos, serán 10! = 3.628.800 posibles opciones.

En cuanto el problema escala a dimensiones mayores, como en casos de 20 trabajos, la cantidad de posibles configuraciones es del orden de: 24.329.020.080.000.000.000 posibles configuraciones.

Como se observa, según se añaden trabajos a la secuencia el número de resultados aumenta exponencialmente.

La iteración gravitatoria se encarga de reducir el espacio iterado. Dada una secuencia, el programa itera una cantidad de veces sobre el espacio total de soluciones y almacena los resultados. Posteriormente, selecciona la secuencia que mejora más el parámetro elegido y fija el primer valor de dicha secuencia. A continuación, realiza una segunda iteración, siendo el primer valor de la secuencia, el fijado en la instancia anterior.

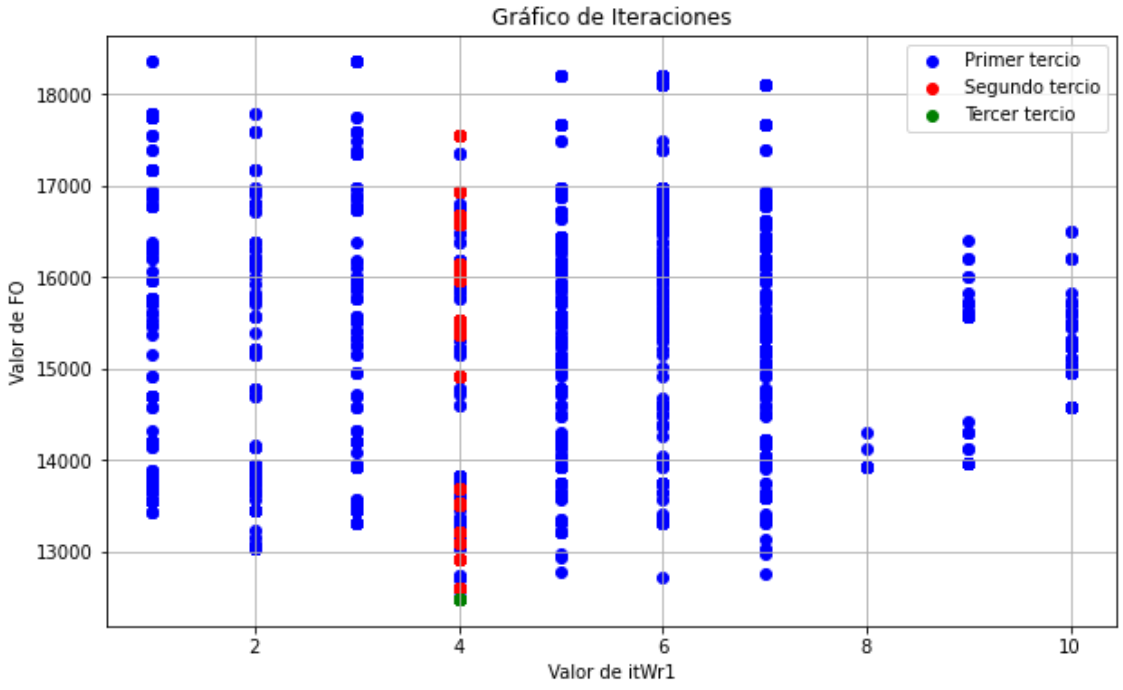

Ilustración 28: Estudio muestral de la mejora de las secuencias con la iteración. Fuente: elaboración propia a través del programa.

En este gráfico se observan las soluciones en el eje de ordenadas, en este caso, el valor de Cmax de cada solución aleatoria, respecto al primer valor de la secuencia. Como se observa, en el primer conjunto iterativo (azul) el programa trabaja sobre todas las opciones, y fija el valor 4 como el primer valor de la secuencia. En el segundo conjunto (rojo) todos los puntos se encuentran en la coordenada 4, debido a que se ha fijado este valor como principio de la secuencia. Finalmente, en el tercer conjunto (verde), como se observa, todas las soluciones se concentran en un punto, el óptimo comprobado.

De esta manera con cada iteración se reduce el espacio de posibles soluciones. Esto permite que con un conjunto muy reducido de iteraciones (6000 en total) se puedan encontrar rápidamente buenas soluciones en espacios que son cientos de veces superiores.

Este proceso funciona tan bien debido a que, los parámetros que más afectan a los resultados de la secuencia completa son los dos primeros trabajos de la secuencia. Como se observa en la ilustración 28, la barra de coordenadas de 4 esta considerablemente más baja que las del resto de los parámetros desde el primer conjunto iterado. Por lo que, al fijar el 4 como primer valor de la secuencia, todas las iteraciones siguientes, van a estar orientadas a encontrar una mejor solución.

Este hecho se confirmó al realizar miles de iteraciones con diferentes matrices. Desde el primer conjunto iterado, aquellos valores iniciales de la secuencia que a priori la mejoraban más, llegado el tercero conjunto iterativo, siempre aportaban valores óptimos o próximos al óptimo. Debido a la configuración de la empresa, sus características, la distribución de trabajos... no se dan óptimos aislados o picos. La llegada al óptimo siempre viene dada por una pendiente moderada, es decir, no se dan casos en los que, un valor que el 99,99% al situarse primero en la secuencia da malos resultados, el 0,01% de las veces es un óptimo asilado.

La progresión de los resultados y la mejora de los mismo viene dada de forma progresiva, los valores que situados al principio dan malos resultados, siempre dan malos resultados, y aquellos valores que, desde el inicio, al situarse primeros en la secuencia, dan buenos resultados, suelen ser los que capitanean el vector óptimo o cercano al óptimo. De esta manera el fijar los valores del inicio de la secuencia una vez han mostrado que dan buenos resultados, permiten sumergirse de manera precisa y efectiva en la búsqueda de un óptimo, de lo contrario, el seguir usando valores que desde el inicio dan malos resultados al situarse primeros en la secuencia, es perder fuerza, no tener dirección ni rumbo.

Continuando con lo anterior, de no haber fijado el 4 como primer valor, el propio proceso iterativo tendría fugas, por estadística, habría secuencias con valores como el 10, 9 u 8 como primer valor, que, como se puede observar en el gráfico, se observa que nunca darán buenos resultados, y al trabajar con una cantidad fijada de iteraciones (2000 en cada conjunto) una gran cantidad de iteraciones serían desperdiciadas.

Asumiendo probabilidades equiprobables en cada trabajo, habría un 10% de posibilidades de que la secuencia empezará por 8, 10% por 9 y 10% por 10. Un 30% de las iteraciones, 600, serían de entrada iteraciones que se desechan ya que nunca van a generar un buen resultado.

Sin embargo, al fijar valores en la secuencia, el 100% de las iteraciones a partir del segundo conjunto, son potenciales óptimos, lo que permite ser mucho más eficiente y en definitiva preciso. Y una vez llegado al tercer conjunto todas las soluciones son buenas.

Al realizar tres pruebas con conjuntos de 20 trabajos se obtienen los siguientes resultados:

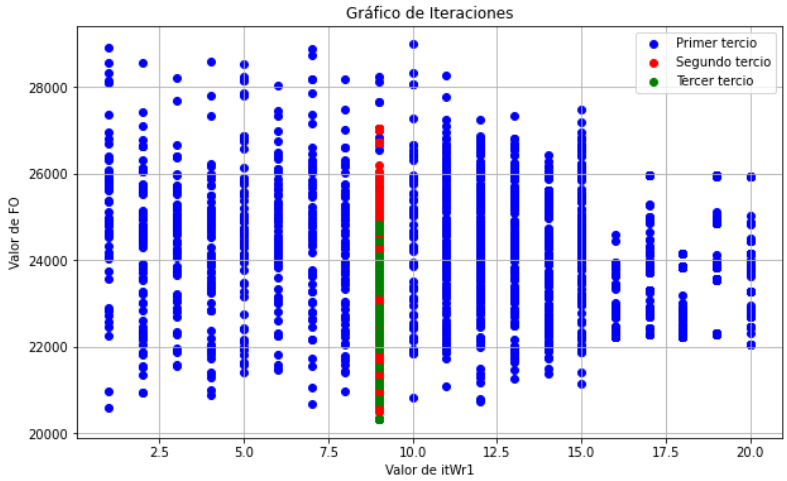

Ilustración 29: Primera prueba. Fuente: elaboración propia a través del programa.

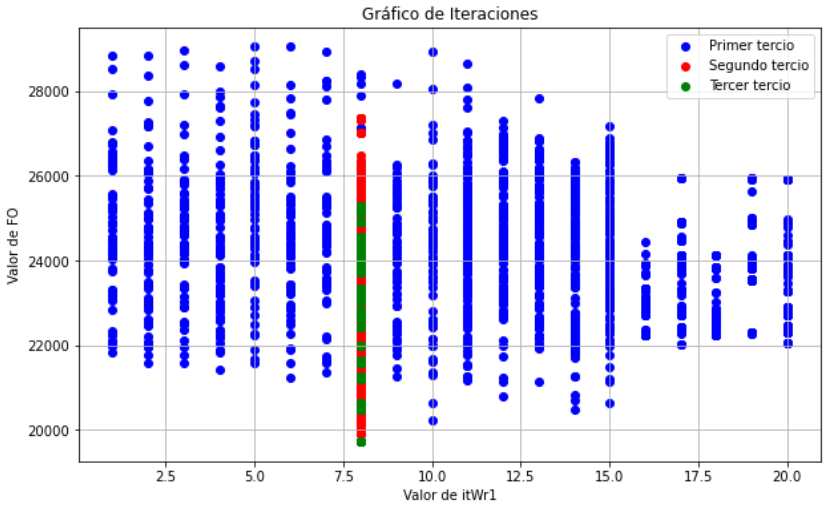

Ilustración 30: Segunda prueba. Fuente: elaboración propia a través del programa.

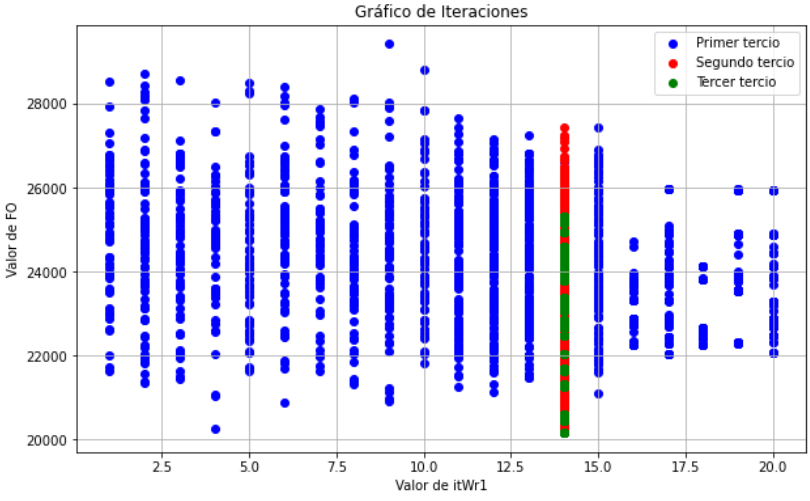

Ilustración 31: Tercera prueba. Fuente: elaboración propia a través del programa.

Pese a que las secuencias empiezan en los tres casos con valores diferentes, los valores optimizados difieren en menos de un 2% entre ellos. A efectos prácticos, la solución tiende a ser la misma.

Por otra parte, se observa que la distribución es cualitativamente igual. De todo el conjunto dado hay una serie de valores aislados que siempre dan malos resultados (del 15 al 20), así mismo se observa que en los tres casos, en cada iteración los resultados se ven mejorados.

El conjunto azul trabaja en todos los valores, y da resultados diversos. Una vez fijado el punto, el conjunto rojo siempre mejora los resultados, la forma de comprobar esto, es que el peor valor rojo, siempre es menor o igual que el peor valor azul (del valor fijado). En las ilustraciones 28 y 29, se observa claramente que el conjunto rojo tiende a ser inferior que el azul dentro de la coordenada fijada.

Finalmente, el conjunto verde **siempre,** mejora la solución roja. Lo cual se observa en las tres ilustraciones de la misma forma, por una parte, el peor valor verde (el más alto) siempre es más bajo que el peor valor rojo y, por otra parte, la barra verde tiene a ser inferior que la roja.

Se realizaron docenas de pruebas extra, y todas ellas confirmaban lo planteado, fijar los dos primeros valores de la secuencia no solo ayuda a centrar las iteraciones y que éstas sean en sí mismas más efectivas, sino que además con cada valor fijado se mejora sustancialmente la solución.

5.4. CASOS DE USO: Pedido urgente

Además de los dos problemas nombrados, tanto el de las fechas como el de la mejora de las soluciones, la empresa nos nombró un tercer problema, y es el de los pedidos urgentes.

Ocasionalmente, los clientes exigían acortamientos de fechas de entrega o hacían pedidos urgentes, situación que alteraba la organización de la empresa.

Para solucionar este problema, el programa incorpora una opción de trabajar con hojas de Excel.

Dado el conjunto de trabajos:

Trabajo	Ruta	Metros a procesa	Tipo de tejido
1	1	500	1
2	1	880	1
3	1	600	1
4	1	1000	1
5	1	600	2
6	2	380	2
7	1	650	2
8	2	2000	3
9	1	350	3
10	2	960	3

Ilustración 32: Matriz de ejemplo. Fuente: elaboración propia a través del programa.

Al realizar la optimización de la secuencia con Cmax mediante iteración graviatoria, la secuencia dada y su Gantt serian:

Secuencia: [4, 1, 3, 2, 9, 10, 8, 7, 6, 5]

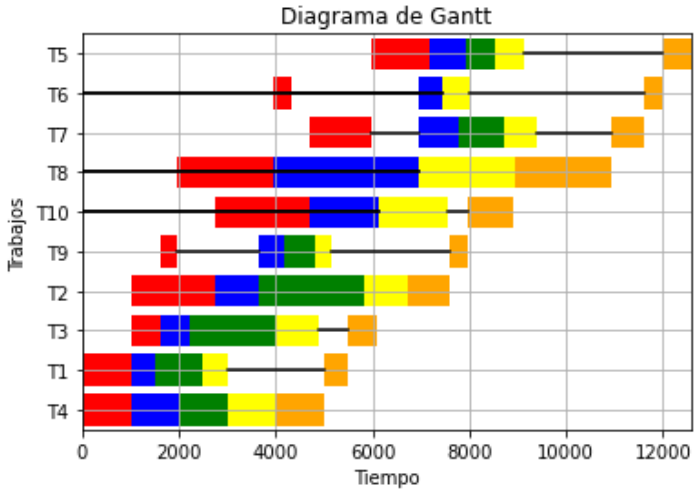

Ilustración 33: Gráfico de la matriz de ejemplo. Fuente: elaboración propia a través del programa.

El cliente A realiza un pedido urgente de 2500m de tejido 1:

Trabajo	Ruta	Metros a procesa	Tipo de tejido
1	1	500	1
2	1	880	1
3	1	600	1
4	1	1000	1
5	1	600	2
6	2	380	2
7	1	650	2
8	2	2000	3
9	1	350	3
10	2	960	3
11	1	2500	1

Ilustración 34: Presencia de pedido urgente. Fuente: elaboración propia a través del programa.

La forma de proceder sería incluir el pedido urgente en el primer puesto de la secuencia ordenada, es decir, la nueva secuencia seria: [11, 4, 1, 3, 2, 9, 10, 8, 7, 6, 5]

Estos valores serían trasladados a Excel:

Trabajo	Ruta	Metros a pro	Tipo de tejid
11	1	2500	1
4	1	1000	1
2	1	880	1
1	1	500	1
3	1	600	1
9	1	350	3
10	2	960	3
8	2	2000	3
6	2	380	2
5	1	600	2
7	1	650	2

Ilustración 35: Ajuste de la secuencia en Excel. Fuente: elaboración propia a través del programa.

En el programa se seleccionaría la opción de trabajar con Cmax en Excel:

```
Cmax seleccionado.
Desea que el programa itere para encontrar una solución mejor o desea visualizar
la información de los datos de excel
1. Excel
2. Iterar
3. Usar NEH
```

Ilustración 36: Visualización de la consola del programa. Fuente: elaboración propia a través del programa.

El Gantt resultante sería:

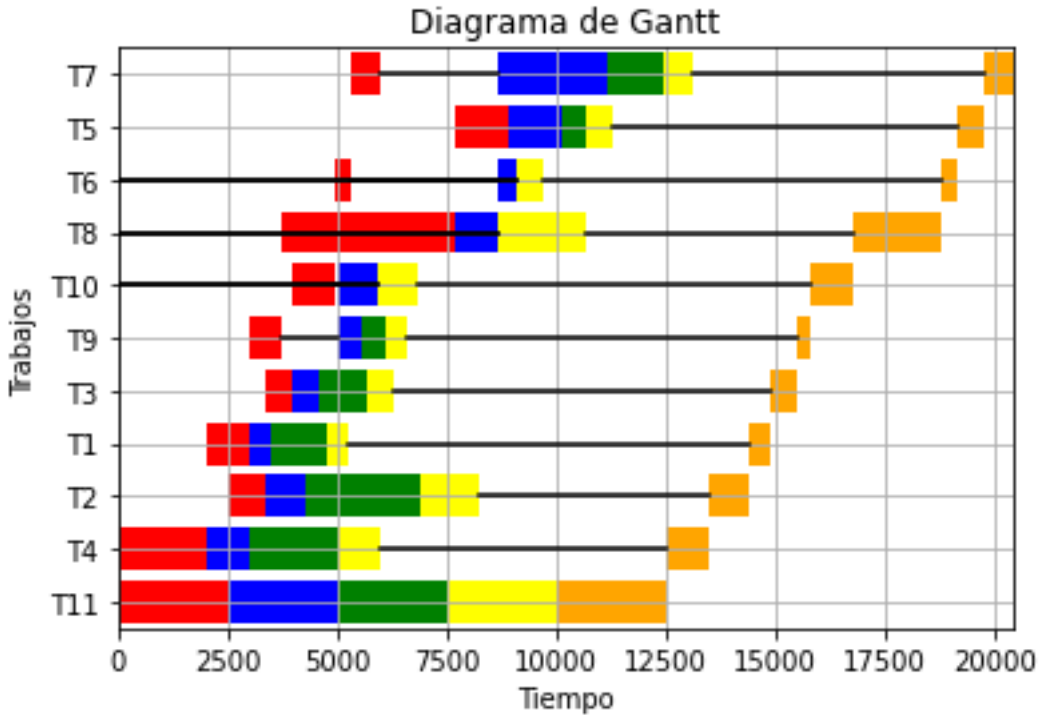

Ilustración 37: Gantt del pedido urgente. Datos de partida en la ilustración 35. Fuente: elaboración propia a través del programa.

Como se observa, el pedido urgente se procesaría el primero, a continuación, se mantendría la secuencia previamente mejorada para reducir lo máximo posible las desviaciones.

5.5. CASOS DE USO: Nuevas funciones

El programa se puede dividir en dos bloques principales como ya se ha comentado. Bloque de cálculos (formado por las funciones de apoyo, las activas y las funciones basadas en leer datos y realizar gráficas) y Bloque de resultados (el NEH y la iteración)

Como se ha visto, tanto el NEH como la iteración aportan resultados diferentes debido a que se basan en procesos lógicos diferentes, sin embargo, computacionalmente, la única diferencia que poseen es la forma de gestionar los datos de entrada.

De forma que, el programa en su conjunto funciona de la siguiente forma:

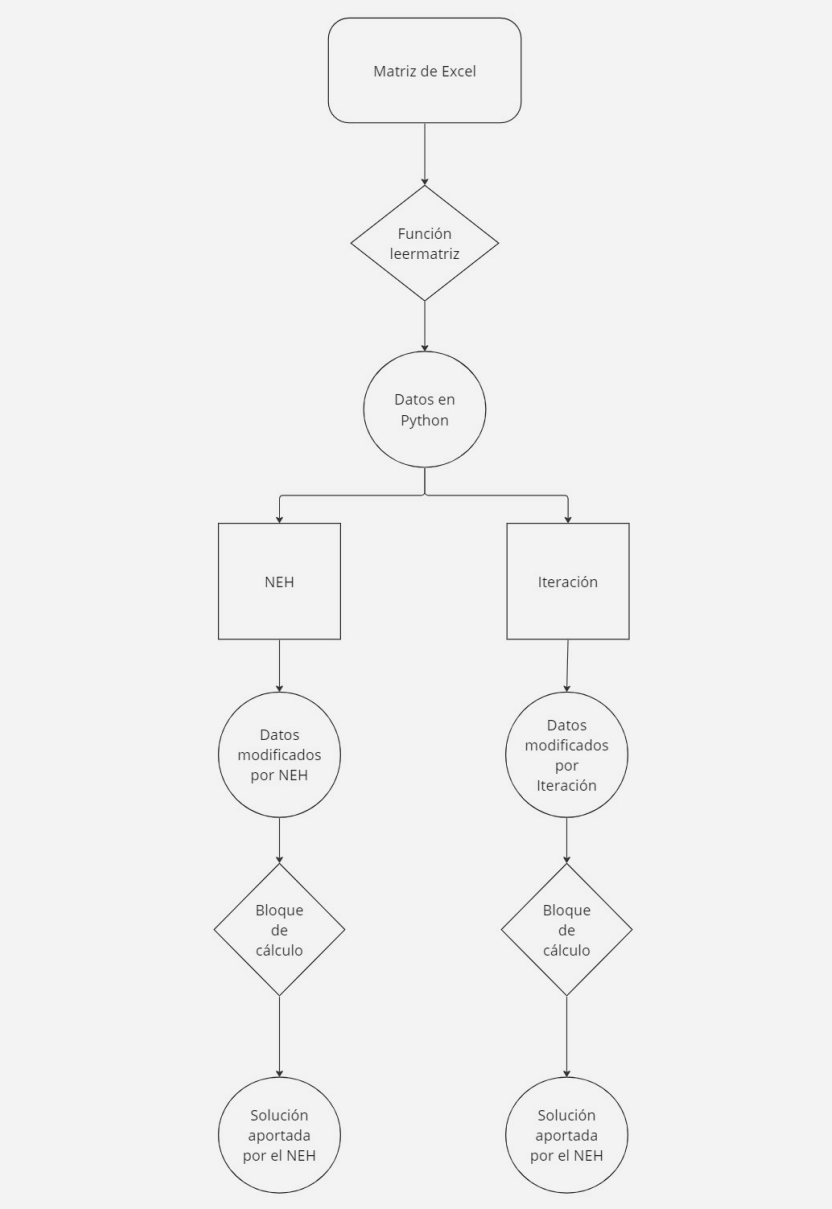

Ilustración 38:Funcionamiento del programa. Fuente: Elaboración propia

Siendo el desglose del bloque de cálculo el expuesto en la ilustración 9.

De forma que, como se observa, tanto el NEH como la iteración, se basan en una modificación de datos de partida, para posteriormente enviar los datos a través del bloque de cálculos y que éste, devuelva los resultados.

Computacionalmente la visualización es la siguiente:

```
for pos in range(i+1):  # Asegurémonos de recorrer la longitud correcta
    sequence = NEH_sequence[:pos] + [current_job] + NEH_sequence[pos:]

    worksNEH=works[sequence]
    NEHWm1 = calcWM(worksNEH[:len(sequence)], len(sequence))
    NEHWr1 = calcWR(worksNEH[:len(sequence)], len(sequence))
    NEHWtej1=calcWTEJ(worksNEH[:len(sequence)],len(sequence))
    nehruta=calcvector(worksNEH[:len(sequence)],len(sequence))
    emburrad1, lavad1, tint1, rame1, presanch1, emburrad1m, lavad1m, tint1m, rame1m, presanch1m = ruta1(
    embfin1, lavfin1, tintfin1, ramefin1,  presanchfin1 = metertotime(len(sequence), emburrad1m, lavad1m
    NEHfechafinR1, NEHfecha_ini1, NEHtimes1 = calcular_fechasal(embfin1, lavfin1, tintfin1, ramefin1,  p
```

Ilustración 39: NEH, gestión de datos y flujo de información. Fuente: elaboración propia a través del programa.

```
for i in range(2000):
    # Simulación de datos y cálculos
    works = permutaciones(works)
    itWm1 = calcWM(works, Njobs)
    itWr1 = calcWR(works, Njobs)
    itWtej1=calcWTEJ(works, Njobs)
    VectorRutaIT=calcvector(works, Njobs)
    emburrad1, lavad1, tint1, rame1, presanch1, emburrad1m, lavad1m, tint1m, rame1m, presanch1m = ruta1
    embfin1, lavfin1, tintfin1, ramefin1,  presanchfin1 = metertotime(Njobs, emburrad1m, lavad1m, tint1
    itfechafinR1, itfecha_ini1, ittimes1 = calcular_fechasal(embfin1, lavfin1, tintfin1, ramefin1,  pre
```

Ilustración 40: Iteración, gestión de datos y flujo de información. Fuente: elaboración propia a través del programa.

Tanto en la ilustración 39 y 40, se observa lo expuesto anteriormente, los datos iniciales son gestionados de forma diferente, siendo la matriz de datos iniciales del NEH “worksNEH” basada en la construcción según el algoritmo, y en el caso de la iteración, siendo una permutación de las filas “Works”.

Así mismo, a continuación, se observa que, una vez modificados los datos iniciales, el proceso es el mismo. Se crean los 4 vectores de apoyo en función de los datos iniciales, y se sigue el flujo:

f. ruta1→f. metertotime → f. calcular_fechasal

Hasta lograr los 3 resultados: Matriz de fechas de fin (NEHfechafinR1, itfechafinR1 respectivamente), matriz de fechas de inicio (NEHfecha_ini1, itfecha_ini1 respectivamente) y matriz de tiempos (NEHtimes1, ittimes1 respectivamente).

En esta instancia se observa la utilidad de trabajar con funciones. Una vez definidas y programadas, y habiendo sido elaborado un proceso sólido (ilustración 9), el cálculo de soluciones de basa en la modificación de datos iniciales para posteriormente seguir una ruta de funciones que calculan los resultados.

Por otra parte, al trabajar la función del diagrama de Gantt con los datos de salida de "calcular_fechasal" la graficación de resultados no requiere de modificaciones. En su conjunto es un proceso sólido y eficiente.

De esta forma, la incorporación de nuevas funciones o algoritmos se basaría en lo siguiente:

1. Definir una forma de gestionar los datos iniciales de Excel.
2. Crear los 4 vectores de apoyo para la función concreta y llamar a sus respectivas funciones.
3. Enviar datos devueltos a "ruta1".
4. Enviar los datos devueltos por "ruta1" a "metertotime" junto a los vectores de apoyo necesarios.
5. Enviar los datos devueltos por "metertotime" a "calcular_fechasal" junto a los vectores de apoyo necesarios.
6. Evaluar los resultados.

En caso de desear graficar resultados, se enviarían los datos devueltos por "calcular_fechasal" juntos a los vectores de apoyo correspondientes a la función "dibujar_gantt".

```
dibujar_gantt(NEHfechafinR1, NEHfecha_ini1, NEHtimes1, Njobs, NEHWr1)
```

Ilustración 41: Graficar con Gantt el resultado devuelto por NEH. Fuente: elaboración propia a través del programa.

Siendo los tres primeros valores los devueltos por la función "calcular_fechasal" y NEHWr1 el vector de apoyo correspondiente.

En conclusión, la lógica a seguir para implementar nuevas funciones se basa en la modificación de los datos de entrada y el seguimiento del flujo de información:

vectores de apoyo→ f. ruta1→f. metertotime → f. calcular_fechasal

Como se ha visto a lo largo de esta memoria, se han realizado numerosos estudios y muestreos para confirmar o desmentir patrones, en todos esos muestreos, se ha seguido la lógica anteriormente vista a la hora de realizar los cálculos.

Por ejemplo, para la realización del estudio de la relación de las fechas de fin de los trabajos respecto a los parámetros del Cmed y del Cmax (punto 5.1), se creó una función que generaba matrices aleatorias (modificación de datos de entrada) posteriormente se siguió el flujo de información anteriormente expuesto, y se evaluaron los resultados de la forma que se requería, en ese caso, mediante el recuento de aquellos trabajos se veían mejorados.

Para el estudio de la relación de los tiempos ociosos con el Cmed y el Cmax, también se creó otra función exactamente igual a la anterior, con la diferencia en la evaluación de resultados. En el caso del estudio del NEH y de la Iteración y su rendimiento en las mejoras del Cmed y del Cmax, el cual se explicará en el siguiente punto, también se realizó una función igual a las anteriores. Evidentemente con un análisis distinto de resultados. Las tres funciones de estudio previamente nombradas se basan en el cálculo de fechas del bloque de cálculo, y se sirven de este de la misma forma que el NEH y la iteración lo hacen.

De esta forma se exponen tres ejemplos extra de tres funciones distintas generadas siguiendo la lógica anteriormente expuesta, de forma que también se refleja que la incorporación de funciones es sencilla no sólo en el caso de mejora de resultados, si no también en el estudio de muestras.

```
CMAX=[]
CMED=[]
cmaxMayor=0
cmedMejora=0
for i in range (4000):
    works=generar_matriz()
    Njobs1=works.shape[0]

    NEHWmC = calcWM(works, Njobs1)
    NEHWrC = calcWR(works, Njobs1)
    NEHWtejC=calcWTEJ(works, Njobs1)
    nehrutaC=calcvector(works, Njobs1)

    emburrad1, lavad1, tint1, rame1, presanch1, emburrad1m, lavad1m, tint1m, rame1m, presanch1m = ruta1(NEHW
    embfin1, lavfin1, tintfin1, ramefin1, presanchfin1 = metertotime(Njobs1, emburrad1m, lavad1m, tint1m, ra
    NEHfechafinR1, NEHfecha_ini1, NEHtimes1 = calcular_fechasal(embfin1, lavfin1, tintfin1, ramefin1, presar

    seq_NEH_M, makespan_NEH_M, NEHfechafinR1_M, NEHfecha_ini1_M, NEHtimes1_M= neh_algorithmCMED(Njobs1, 5, N
    seq_NEH, makespan_NEH, NEHfechafinRM, NEHfecha_iniM, NEHtimesM= neh_algorithm(Njobs1, 5, NEHtimes1, work


    for i in range (Njobs1-1):

            if NEHfechafinRM[i,4]>NEHfechafinR1_M[i,4]:
                cmaxMayor=cmaxMayor+1
            else:
                cmedMejora=cmedMejora+1



total=cmedMejora+cmaxMayor
print(total)
print(cmaxMayor/total)
```

Ilustración 42: Ejemplo de aplicación de una nueva función. Fuente: elaboración propia a través del programa.

En la ilustración 42 se refleja el primer ejemplo expuesto, el estudio de la relación de las fechas de fin de los trabajos respecto a los parámetros del Cmed y del Cmax. Como se observa, la información sigue el flujo previamente nombrado:

vectores de apoyo→ f. ruta1→f. metertotime → f. calcular_fechasal

Pero en este caso, los datos de entrada son matrices aleatorias (Works), y el análisis de resultados es mediante ese bucle en el que se comparan las fechas.

6. ANÁLISIS ECONÓMICO:

Una vez realizado el programa, se estudiará en qué medida ayuda a la empresa a mejorar costes o aumentar beneficios. Se analizarán los resultados aportados por el programa desde la perspectiva numérica y la económica, se presupuestará el coste del programa y se analizará la inversión que supondría para la empresa.

Así mismo para elaborar el presupuesto del programa se parte de estos datos donde se indican la hora destinada a cada actividad:

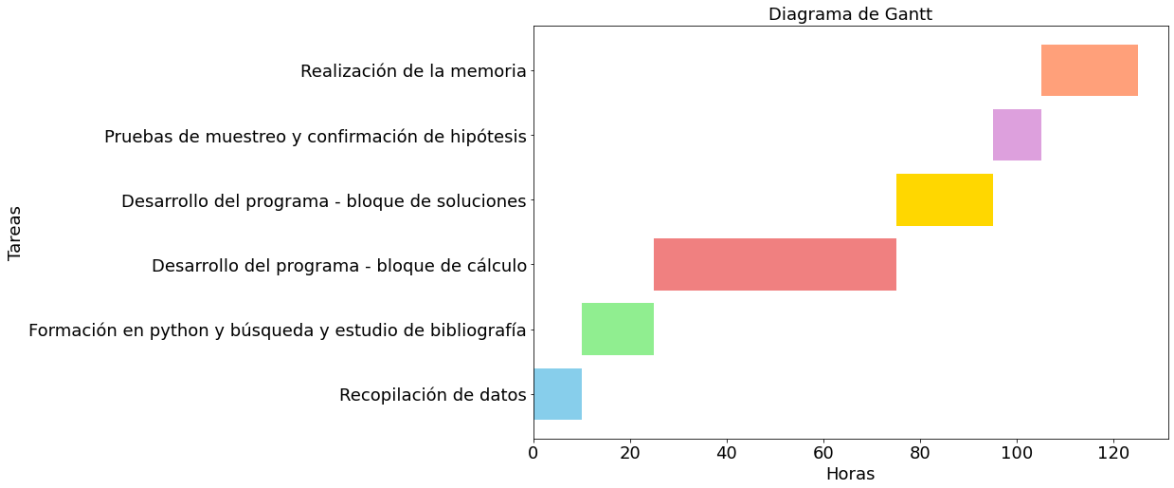

Ilustración 43: Fuente: elaboración propia con Python.

Para la realización del TFG se destinaron 85h aproximadamente en acciones relacionadas con Python, ya fueran de programación, de aprendizaje, de pruebas, de borradores, de entrenamientos, de estudio de muestras... Siendo el conjunto mayor el dedicado al bloque de cálculo, extendiéndose este 50h.

La elaboración de un flujo de información sostenible y robusto que funcionara en todos los escenarios posibles, sumado a la función “calcular_fechasal” y todas las restricciones que debía tener en cuenta, produjo que esta labor ocupara la mayor parte del tiempo del trabajo en su conjunto.

Se destinaron por otra parte 20h a la realización de la memoria y 15h a recopilación de información ya fuera de la empresa o de otras fuentes, dando un total de 120h de trabajo. De las cuales el 30% se disipó a lo largo del curso, y el 70% restante se concentró desde mediados de abril hasta junio

6.1. ANÁLISIS DE RESULTADOS

Hasta el momento en el que se escribe este TFG, la empresa trabaja sin una política fijada se secuenciación. Los diferentes operarios y sobre todo los jefes de planta realizan ajustes visuales a las secuencias dependiendo de diferentes factores, como la llegada de pedidos urgentes o la agrupación de trabajos por tipo de tejido.

Pese a eso, el modelo de secuenciación, aún sin haber sido pensado, se puede aproximar a un FIFO. Los primeros pedidos en llegar se ponen en cola, y salvo ajustes situacionales, la cola dada se sigue sin modificaciones, de forma que los primeros en llegar, usualmente, son los primeros en salir.

Dada una matriz de ejemplo:

Trabajo	Ruta	Metros a procesar	Tipo de tejido
1	1	500	1
2	1	880	1
3	1	600	1
4	1	1000	1
5	1	600	2
6	2	380	2
7	1	650	2
8	2	2000	3
9	1	350	3
10	2	960	3

Ilustración 44: Matriz de ejemplo. Fuente: elaboración propia a través del programa.

Se asume que una serie de trabajos han llegado en ese orden y que la empresa ha ordenado los trabajos por tipo de tejido. En caso de realizar la secuenciación según FIFO los resultados obtenidos en ausencia de pedidos urgentes serían:

Cmax= 16330 segundos = 4,53h.

Cmed= 10239 segundos = 2,84 h.

Con el Gantt:

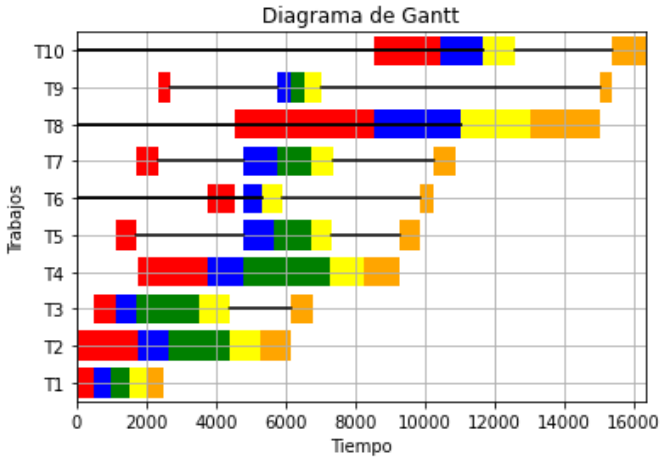

Ilustración 45: Graficación de la matriz de ejemplo. Fuente: elaboración propia a través del programa.

No obstante, si construyera la secuencia mediante iteración gravitatoria y orientando la mejora al Cmax, los resultados obtenidos serían:

Cmax= 12480 segundos =3,47h.

Con el Gantt:

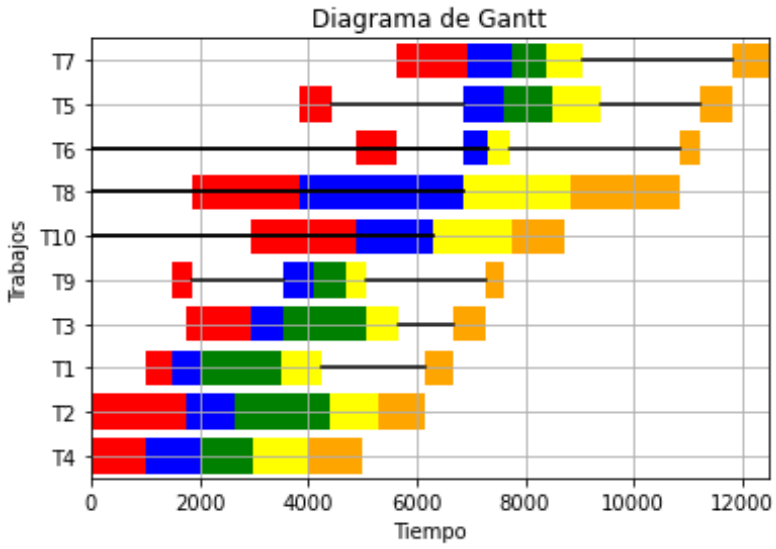

Ilustración 46: Mejora del Cmax mediante iteración gravitatoria. Fuente: elaboración propia a través del programa.

Por una parte, se ha reducido en una hora el tiempo necesario para completar los trabajos, una mejora del 25%.

Por otra parte, si en 3,47h se han realizado 10 trabajos, los trabajos se han procesado a una velocidad de 2,88 trabajos/ h; 0,35h/ trabajo.

Lo que supone que, en el mismo tiempo que FIFO realiza 10 trabajos, el programa es capaz de organizar la producción para realizar 12 trabajos.

1 hora extra a 0,35h/ trabajo = 2,85 trabajos = +2 trabajos.

Es decir, si se quiere acabar la batería de trabajos en el menor tiempo posible, el programa generaría ahorros del 25%.

En el caso de mejorar la secuencia con el Cmed, los resultados serian también satisfactorios.

El Cmed mejorado pasaría de 2,84h a 2,01h con el Gantt:

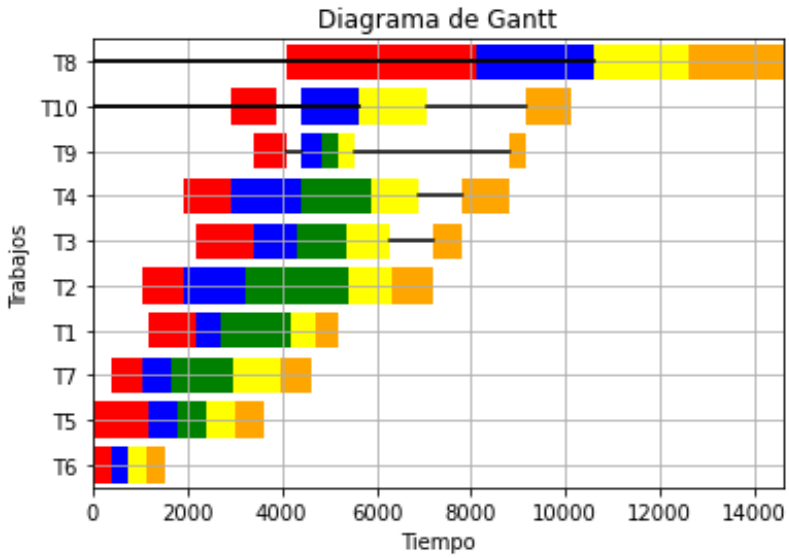

Ilustración 47: Secuencia mejorada atendiendo a Cmed. Fuente: elaboración propia a través del programa.

Una mejora del 30% en tiempos.

De esta manera queda explicado el proceso que se sigue para la evaluación de la mejora del rendimiento. Dada una matriz, se calculan sus parámetros sin modificaciones previas. Seguidamente, se elige que parámetro mejorar y que algoritmo usar, una vez aplicado, se comparan los resultados iniciales y finales. Debido a que con un caso de ejemplo no es posible extraer conclusiones, se realizó un estudio muestral.

Para ello, se generaron varias tandas de 2000 matrices de datos iniciales para trabajar con el NEH. Estas matrices oscilaban entre diez y veinte trabajos, con una cantidad de metros a procesar que oscilaba entre 500 y 2500 metros por trabajo.

Así mismo, tanto el tipo de tejido como la ruta y, la cantidad de productos/ruta y productos/tejido, también se generaban aleatoriamente en la construcción de la matriz. La única modificación que se hizo a las matrices una vez generadas fue la agrupación de trabajos por tipo de tejidos, de la forma que se muestra en la ilustración 43. Es decir, mismas condiciones que los estudios realizados en el punto 5.1.

Una vez generadas esas 2000 matrices, se elegía aleatoriamente un parámetro a valorar (Cmed o Cmax) y se guardaba el valor del parámetro de la matriz.

Para la primera tanda de 2000 matrices, fue elegido el parámetro del Cmax, y los datos obtenidos fueron los siguientes:

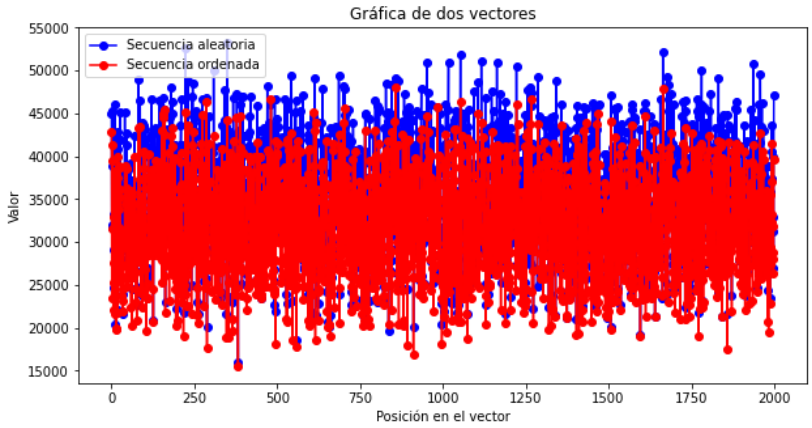

Ilustración 48: Tanda de 2000 matrices mejorando Cmax con NEH. Fuente: elaboración propia a través del programa.

En esta gráfica se observa la comparación entre el Cmax de las secuencias aleatorias (azul) y las mejoradas (rojo). Siendo el valor de Y el valor que alcanza Cmax y siendo el valor de X la matriz a la que corresponde. De modo que, para cada matriz generada, habrá dos puntos, uno rojo y uno azul.

Debido a la cantidad de datos, no se visualizan todos los puntos, pero si se puede observar que el conjunto rojo se sitúa inferiormente respecto al azul, lo que supone que el valor de Cmax tiende a ser menor en las secuencias mejoradas.

De las 2000 matrices generadas, 1847 fueron mejoradas por el programa, el resto, partían de una solución que no podía ser mejorada por el NEH. Es decir, el 92,35% de las secuencias se pudieron mejorar, mientras que el 7,65% ya partían de una solución, al menos, al nivel del NEH. Así mismo, en las 1847 matrices mejoradas, la mejora media fue del 10,3% en reducción de tiempos

Por parte, se realizó otra tanda de 2000 matrices aleatorias con las mismas características que las anteriores, y se evaluó el rendimiento del Cmed mejorándolas mediante el NEH. Se obtuvieron los siguientes resultados.

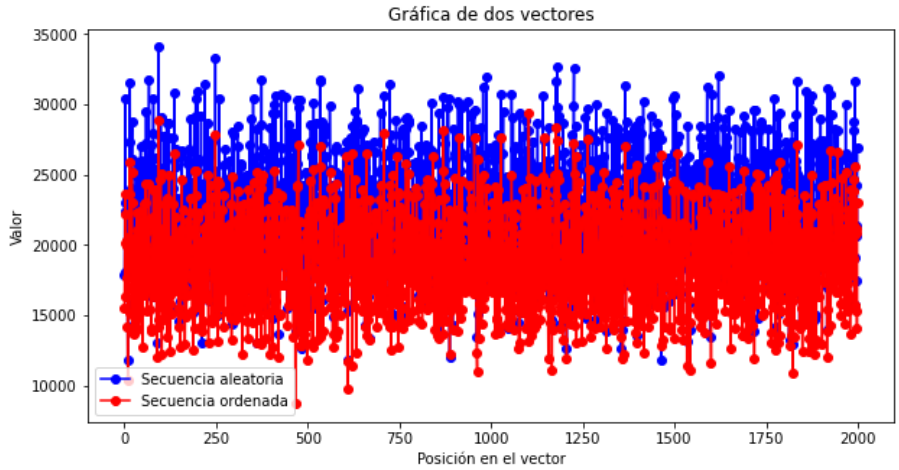

Ilustración 49: Tanda de 2000 matrices mejorando Cmax con NEH. Fuente: elaboración propia a través del programa.

De las 2000 matrices evaluadas, se consiguió mejorar el parámetro en 1924 de ellas, es decir, el 96,2% de las secuencias fueron mejoradas con el programa, mientras que el 3,8% de ellas tenían una situación de partida que era, al menos, tan buena como la aportada por el NEH. En cuanto al rendimiento, aquellas matrices que fueron mejoradas lo fueron en un 16,14% de media.

En el caso de la iteración, debido al tiempo de computación necesario se debió reducir el tamaño de muestra de 2000 a 50. El NEH, tardó en evaluar bloques de 2000 matrices menos de dos minutos por bloque, mientras que la iteración, tardó aproximadamente 10 minutos en evaluar cada bloque de 50 matrices.

Los resultados obtenidos tras el estudio de la muestra de 50 matrices mejorando CMax mediante iteración fueron los siguientes:

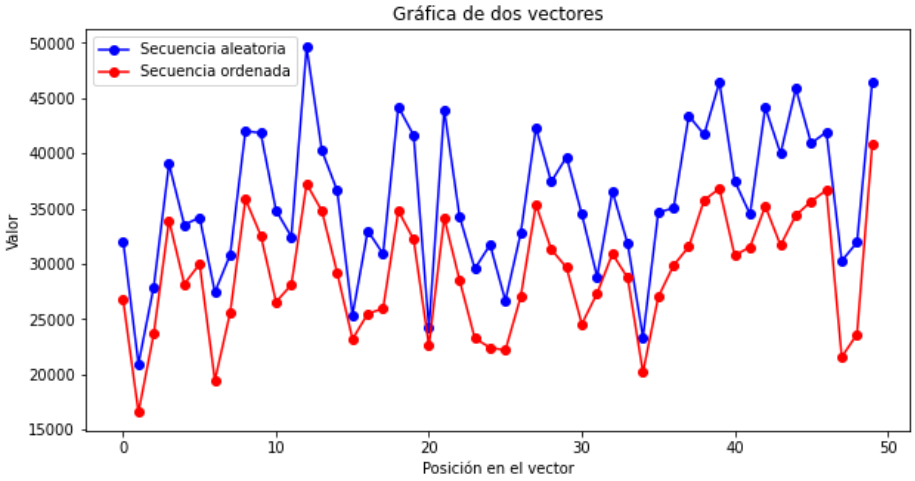

Ilustración 50: Muestra de 50 matrices mejorando Cmax con Iteración. Fuente: elaboración propia a través del programa.

En el caso de la iteración mejorando Cmax, de las 50 matrices dadas, el programa mejoró los tiempos de 50, es decir, un 100% de casos mejorados, como se observa en los puntos, todos los

rojos (secuencia ordenada) se encuentran situados por debajo de los azules (secuencia sin ordenar). Así mismo, la media de mejora del parámetro fue del 18,43%.

También se realizó un estudio de iguales características que el anterior para el caso del Cmed, se obtuvieron los siguientes resultados.

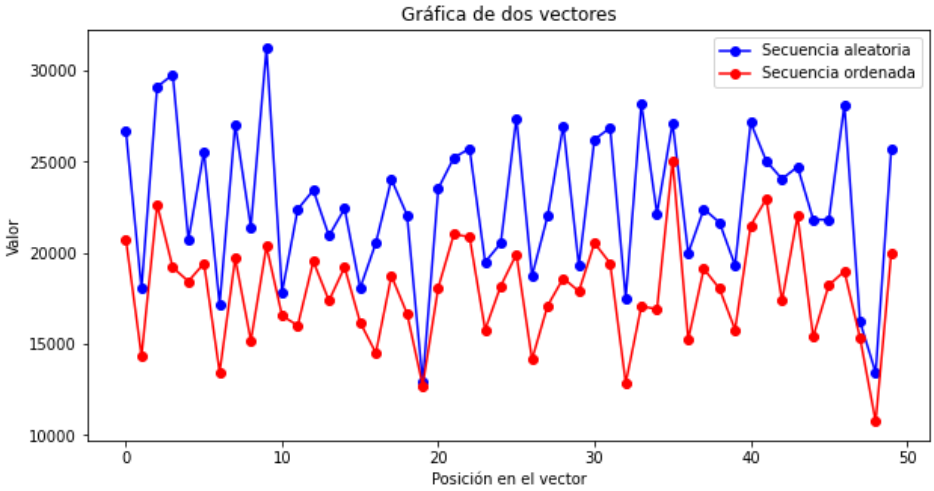

Ilustración 51: Muestra de 50 matrices mejorando Cmed con Iteración. Fuente: elaboración propia a través del programa.

De las 50 matrices generadas de forma aleatoria el programa logró mejorar el tiempo de 50 de ellas, al igual que con el caso del Cmax, se logró un 100% de resultados mejorados, como se observa además visualmente al estar todos los puntos rojos debajo de su respectivo punto azul.. Además, la mejora media del parámetro fue del 21,4%.

De esta forma se realizaron varias pruebas, en el caso del NEH en baterías de 2000 matrices y en el caso de la iteración en baterías de 50. Tanto el parámetro a mejorar como el algoritmo usado eran elegidos de forma aleatoria, después de dos horas de muestreo y más de 25.000 matrices evaluadas se llegó a las conclusiones resumidas en la siguiente tabla.

	Porcentaje de matrices mejoradas	Porcentaje de mejora	Mejora bruta
Iteración gravitatoria	100%	19,7%	19,7%
NEH	95,8%	13,7%	13,12%

Tabla 1: Resumen de datos. Fuente: elaboración propia.

La iteración no sólo permite mejorar todas las secuencias, sino que, además, las mejora en mayor medida que el NEH, sin embargo, el NEH tiene mayor capacidad de procesamiento de información que la iteración y es capaz de analizar mayor cantidad de datos en menos tiempo.

En este estudio se ve reflejado también el hándicap del NEH relacionado con los datos de partida (punto 5.2). La capacidad de maniobra del NEH se ve limitada por el problema de las máquinas no paralelas y eso no sólo hace que su campo de aplicación sea menor, sino que además mejora en

menor medida. Siendo la iteración más polivalente y precisa y, en resumen, mejor. El hecho de que haya secuencias aleatorias que el NEH no haya lograda mejorar, es un mal indicador del mismo, y sobre todo cuando se da en el porcentaje dado.

Debido a la cantidad de datos generados, no habría sido extraño que el NEH no hubiera logrado mejorar aluna matriz, pero que aproximadamente el 4% de las matrices aleatorias hayan sido mejores o iguales, que aquellas mejoradas por el NEH, dejan en evidencia las carencias de este algoritmo ente problema dado.

También de esta forma se concluye que el programa puede aportar soluciones que mejoran los parámetros, de media, entorno al 20% vía iteración y entorno al 13% vía NEH, lo que evidencia una vez más, la superioridad de la iteración frente al NEH.

6.2. MEJORA DEL RENDIMIENTO ECONÓMICO Y PRESUPUESTO

Como se comentó en el apartado "Selección del área de aplicación", este programa se basa en dos rutas que suponen el 25% de la facturación de la empresa. Esto se debe a que a través de estas dos rutas se procesa un gran porcentaje de los productos.

Este 25% de la facturación se traduce en más de 1,2 millones de euros que pueden ser potencialmente mejorados por el programa. Como se ha expuesto en el punto anterior, el programa permite mejoras de entorno al 20% en las secuencias vía iteración. Ello se puede observar desde dos puntos de vista, por una parte, se puede hacer capaz a la empresa de hacer el mismo trabajo en un 20% menos de tiempo, o se puede gestionar la producción de forma que se puedan procesar hasta un 20% más de productos, una quinta parte.

Debido a que la empresa no va a reducir su jornada, se asume que se podría aumentar hasta en un 20% la cantidad de productos procesados. De forma que, si este 20% es aplicado sobre el 25% de la facturación, que constituye la parte equivalente a las rutas, se puede inducir que la facturación podría verse aumentada en un 5%, lo que supone un aumento valorado en 240.000€.

Para analizar el retorno de la inversión se debe presupuestar el coste de este programa, cuyo coste el primer año sería:

Costes totales				
Concepto	**Unidad**	**Precio unitario**	**Cantidad**	**Importe total**
Consultoría y mejora	horas	45 €	100	4.500 €
Mantenimiento	horas	25 €	25	625 €
Desarrollo (PVP)	horas	30 €	120	3.600 €
Ahorro aportado	%	240.000 €	0,05	12.000 €
			Total primer año	20.725 €

Tabla 52: Coste primer año. Fuente: elaboración propia.

El presupuestado se basa en 4 conceptos. El PVP ha sido calculado con las horas dedicadas al desarrollo del programa, tanto mantenimiento como consultoría y mejora se basan en el tiempo dedicado a lo largo del primer año a la implantación, solución de problemas y nuevas funcionalidades. Finalmente se añade el concepto de referenciar el rendimiento del programa en forma económica.

El programa va a permitir ahorrar o aumentar la facturación. De ese aumento de la facturación logrado tras la implantación, el 5% será cobrado, de forma que a más ahorre el cliente, más cobrará el desarrollador.

El margen de las empresas en España alcanzó el 12,3% (Ferrari, 2023), asumiendo ese margen, los beneficios netos aportados por el programa ascenderían a 29.520€, por lo que se recuperaría la inversión en el primer año de uso.

7. CONCLUSIONES, LIMITACIONES Y FUTURAS LÍNEAS DE TRABAJO

7.1. CONCLUSIONES

En este trabajo final de grado se ha solucionado el problema que tenía una empresa sobre la incertidumbre de las fechas de entrega, así como se han aportado soluciones para la mejora del rendimiento.

En primer lugar, se contextualizó la empresa en el sector y se describió como es su proceso productivo. Se describieron las diferentes secciones y en que se basaban, que máquinas tenían y de qué manera gestionaban la información.

A continuación, se analizó el problema desde el marco teórico, por una parte, se solucionaron las incertidumbres y por otra se abordó la mejora de resultados desde cuatro frentes distintos mediante la mejora de dos parámetros a través de dos algoritmos diferentes.

Se expusieron dos casos de uso, en uno se explicó cómo tratar los pedidos urgentes y en el otro como incorporar y ampliar la forma de buscar soluciones.

Se realizó un estudio para valorar en qué medida ayudaba a la empresa a mejorar su situación y se presupuestó el desarrollo de este programa.

A lo largo de este TFG se ha explicado el funcionamiento del programa y en las líneas anteriores se ha resumido su proceso de elaboración, sin embargo, no se ha hecho hincapié en ningún momento en el esfuerzo que ha supuesto que el programa simplemente funcionase correctamente. El proceso de programación es muy delicado y cualquier acción sencilla puede ser mal ejecutada si el proceso no es sólido.

En numerosas ocasionas se ha utilizado la terminología “función sólida”, las funciones sólidas son aquellas que siempre funcionan en independencia de los datos iniciales y en independencia del contexto, en definitiva, son funciones fiables y robustas. En el momento que se escriben estas líneas con el programa acabado, es posible decir que el 100% de las funciones son sólidas.

Por poner un ejemplo de lo anterior, en el punto en el que se habló del NEH no se hizo referencia, pero pese a que la lógica que sigue este algoritmo es sencilla, y la implementación en sí no fue especialmente compleja, lo complicado fue el hacer de la función del NEH una función robusta y sólida.

La función interna del NEH, en un inicio, ocupaba a penas 25 líneas de programación, pero trabajaba con una serie de datos iniciales constantes, siempre 10 trabajos, con al menos un trabajo de cada ruta, y al menos un trabajo de cada tipo de tejido. Pero al ser este caso irreal, el proceso de tornar esa función en una función sólida (que trabajara con cualquier número de trabajos, con cualquier combinación de rutas y tipos de tejido, con cualquier orden...) hizo que la función pasase de las 25 líneas a las 250 aproximadamente, 10 veces más. Este proceso se realizó en todas las funciones, incluidas las que leen datos.

Si solamente se hubiese trabajado con una matriz de ejemplo a lo largo de todo este TFG, el programa habría ocupado entre 250-350 líneas programadas, y la información aportada con el

maquillaje adecuado, podría ser de similares características: un programa que da soluciones y funciona bien (a excepción de los estudios inmersivos, los análisis de muestras y las conclusiones transversales de las que se hablará a continuación).

Pero el hecho de hacer un programa serio y sólido ha hecho que ocupe aproximadamente 2000 líneas contando las funciones de estudio de muestras.

Sin embargo, este trabajo que se ha hecho es difícil de reflejar, ya que, como mínimo, se espera que el programa funcione bien, pero el hacer que funcione bien es diferente al hacer que funcione bien siempre y en cualquier contexto.

Como ya se ha nombrado, el proceso de elaboración del bloque de cálculos fue el más complicado. En especial en la modelación de la función de cálculos, apenas hay bibliografía que hable sobre máquinas NO paralelas, por lo que la forma de solucionarlo y de modelarlo se basó en el propio ingenio del alumno.

Ningún artículo que se haya revisado, buscado y comprobado, habla sobre la forma de solucionar el problema de máquinas NO paralelas en un flowshop híbrido a través de una matriz_Z pívot. En el punto en cuestión se comentó en que se basó la solución, pero la línea de trabajo estaba enfocada en explicar “una función que calcula fechas finales”, cuando en el proceso de elaboración de esa función, también se elaboró la solución al problema de cálculo de fechas finales en máquinas no paralelas.

En pro de ayudar a la visualización, este es el bucle final de la función “calcular_fechasal”, el bucle en el que realmente se calculan las fechas.

```
for i in range (1, Njobs):
        for j in range(1, 5):

            if j==0  or j==3:

                fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j])
            elif j==2 :
                fechafinR[i,j]= max(fechaf22[i], fechafinR[i, j-1]+ times1[i,j])
            elif j==1 and Wtej1[i]==Wtej1[i-1] and Wtej1[i]==ordentipo[0]:
                fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j])
            elif j==1 and Wtej1[i]==Wtej1[i-1] and Wtej1[i]==ordentipo[1]:
                    fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j], np.max(lavadora[0])+times1[i,j])
            elif j==1 and Wtej1[i]==Wtej1[i-1] and Wtej1[i]==ordentipo[2]:
                        fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j], np.max(lavadora[1])+times1[i,j])

            elif j==1 and Wtej1[i]!=Wtej1[i-1] and Wtej1[i-1]==ordentipo[0]:
                fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j], np.max(lavadora[0])+times1[i,j])
            elif j==1 and Wtej1[i]!=Wtej1[i-1] and Wtej1[i-1]==ordentipo[1]:
                    fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j], np.max(lavadora[1])+times1[i,j])
            elif j==1 and Wtej1[i]!=Wtej1[i-1] and Wtej1[i-1]==ordentipo[2]:
                        fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j], np.max(lavadora[2])+times1[i,j])
            else :
                fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, j-1]+ times1[i,j], fechafinR[i-1,j]+times1[i,j])

            if VectorRuta[i]==2 and j==2:
                fechafinR[i,j]= 0


            elif VectorRuta[i]==2 and j==3 :
                fechafinR[i,j]= max(fechaf[i, j], fechafinR[i, 1]+ times1[i,j])


            if Wtej1[i]==ordentipo[0]:
                lavadora[0,i]=fechafinR[i,1]
                alfa=i+1
            elif Wtej1[i]==ordentipo[1]:
                lavadora[1,i-alfa]=fechafinR[i,1]
                beta=i+1
            else:
                lavadora[2,i-beta]=fechafinR[i,1]
```

Ilustración 53: Bucle final de la función del cálculo de fechas. Fuente: elaboración propia a través del programa

Este bucle debía tener en cuenta las máquinas paralelas, el orden de tipos de la lavadora para evitar solapes, las diferentes rutas (para determinar que valores son 0 de la forma que se observa en las ilustraciones 19 y 20 por ejemplo), el hecho de que había estaciones con máquinas paralelas y estaciones con una máquina y además debía de ser un proceso sólido. Lo importante para este TFG era que funcionara para poder continuar con estudio de resultados, pero detrás de este funcionamiento se dieron tardes dedicadas íntegramente a la prueba error y al proceso de parcheo. Por ello, se deseaba dejarlo reflejado de algún modo.

Además, el hecho de que en la bibliografía no se hable especialmente sobre las máquinas NO paralelas (que casualmente son las que se dan en el 99% de los casos reales) hace que el propio proceso de darse cuenta de los errores que se cometerían en caso de realizar los cálculos "como dicen los libros" también fuera del 7alumno. Así como el hecho de trabajar en un entorno real y no hacer la simplificación: "se asume que todas las máquinas son paralelas" que son de las que hablaban Ruíz y Vázquez-Rodríguez en 2009 en su artículo sobre el flowshop híbrido.

Continuando con el asunto de la bibliografía, no se encontraron artículos que estudiaran las relaciones que en este caso sí se han estudiado. En este TFG se han realizado más de 5 estudios de muestras que en sí mismos podrían haber sido otro TFG. Se elaboraron numerosos indicadores, se estudiaron, se comprobaron y se afirmaron: el Cmed reduce los tiempos ociosos más que el Cmax, el Cmed reduce las fechas de finalización de todos los trabajos a excepción del último en mayor medida que el Cmax, el NEH funciona mal y tiene desviaciones por el caso de las máquinas NO paralelas, el NEH tiene un hándicap respecto a la iteración gravitatoria por la forma de llegada de los datos de entrada, fijar valores al inicio de las secuencias ayuda a mejorar las soluciones... todas ellas entre otras, son conclusiones que se han confirmado en el caso dado y que han debido ser elaboradas individualmente a través de estudios personalizados ya que no había bibliografía que hablara de los comportamientos de estos parámetros en este contexto.

Por lo que, no solamente se ha elaborado un programa que calcula fechas y genera y mide soluciones, se ha elaborado un programa que en todos los contextos funciona bien, que se mueve en un entorno real, que mejora soluciones con fundamento teórico, se ha elaborado una nueva forma de solucionar el problema del cálculo de las máquinas NO paralelas y en el proceso se han elaborado una serie de conclusiones que permiten continuar la mejora de esta herramienta en una línea de trabajo futura marcada.

Con todo lo anterior, se puede decir que este trabajo final de grado está totalmente alienado con el noveno ODS, "industria, innovación e infraestructuras".

Como se expuso en el inicio de este trabajo, está empresa ha sufrido los efectos de las guerras y de la inflación, acción que se ha visto repercutida en el trabajador en forma de ERTEs y en la empresa en forma de una reducción del margen. Este TFG permitiría a la empresa ser más resiliente al mejorar su eficiencia de forma directa y mediante una industrialización sostenible, ya que no requiere de nuevas máquinas ni de infraestructura alguna.

Además de esta forma, también se podrían evitar nuevos ERTEs en caso de nuevas situaciones como las dadas, estando también alineado con el octavo ODS, "trabajo decente y crecimiento económico". Finalmente, se ha innovado a todos los niveles, tanto con el estudio de parámetros enfrentados a diferentes estímulos como en la forma de abordar el problema de las máquinas NO paralelas

7.2. LIMITACIONES Y FUTURAS LÍNEAS DE TRABAJO

En la realización de este TFG se ha trabajado sobre una serie de supuestos y se han dejado abiertas varias posibles continuaciones sobre el tema dado.

El principal supuesto que hace este TFG en el proceso del cálculo de tiempos es la asunción de que no importa el color del textil ni la anchura, así como que se asume que, "todos los productos pueden procesarse en todas las máquinas dentro de una misma sección". Exceptuando la sección de lavado donde si se respeta esta distinción por tipo de tejido.

En la realidad, es de vital importancia conocer y respetar el orden de procesamiento en función de los colores. No se puede procesar una prenda de color claro después de una prenda de color oscuro sin una penalización en forma de limpieza de máquina. Ni las máquinas de tintura que procesan algodón pueden procesar poliéster.

Así mismo se han ignorado estas limpiezas de máquina, sobre todo en la lavadora. Al procesar determinados tipos de textil es necesaria una limpieza en la sección de lavado, de lo contrario los siguientes productos podrían ser defectuosos.

Por otra parte, se ha asumido que no había retrabajos ni productos defectuosos, siendo que en la realidad un porcentaje de los productos deben procesarse dos veces por errores. También se ha suprimido el control de calidad por la propia asunción de que no se producen errores.

Se ha asumido que las máquinas poseen capacidad infinita y, que la capacidad entre máquinas también es ilimitada. Por ejemplo, después de lavado, solo hay 8 soportes para tejido húmedo, por lo que se imposibilita que los trabajos puedan acumularse de forma infinita después de lavado.

Finalmente, se ha asumido que los productos pueden pasar de una estación a otra sin esperas, en la realidad, por ejemplo, después de lavados o de tinturas, se requieren tiempos variables en función de material y la cantidad de metros para secar los productos.

Todos y cada uno de los aspectos previamente mencionados pueden servir para una ampliación del software en pro de hacerlo más realista. El tener en cuenta los colores y las limpiezas especialmente pueden aproximar los datos a un escenario más real.

Así mismo, a nivel teórico se abre otra posible línea de trabajo en el estudio de los datos de partida del NEH. Como se comentó en el apartado "Regla NEH", los datos de partida afectan en desviaciones de hasta el 12% en el rendimiento del NEH, y en el estudio de los rendimientos de los diferentes algoritmos se observó que aproximadamente el 4% de las secuencias no podían ser mejoradas de esta forma. De ahí que se puedan mejorar las soluciones dadas y aumentar el rendimiento de este algoritmo mediante la gestión previa de los datos de entrada.

Finalmente, otra posible continuación, y en definitiva la más interesante, es que, debido a las numerosas restricciones de la empresa como la capacidad entre máquinas, secuenciación por color, agrupación de tipo de tejidos... El conjunto de soluciones real no se trata de la factorial del número de trabajos, sino un número menor como se explicará a continuación.

Dada la matriz:

Trabajo	Ruta	Metros a procesar	Tipo de tejido
1	1	500	1
2	1	880	1
3	1	600	1
4	1	1000	1
5	1	600	2
6	2	380	2
7	1	650	2
8	2	2000	3
9	1	350	3
10	2	960	3

Ilustración 54: Matriz de ejemplo. Fuente: elaboración propia a través del programa.

Tan solo haciendo la asunción de que los tipos de tejido deben ir siempre agrupados, las posibles combinaciones no son del orden de 10!

Por una parte, tenemos 4! para tipo de tejido 1, y 3! para tipo 2 y tipo 3. Y luego 3! Para las posibles ordenaciones de tipo 1, tipo 2 y tipo 3.

Por lo que, en realidad, el conjunto de soluciones no es 10!=3.628.800, si no 4!*3!*3!*3!=5184 combinaciones.

Lo que supone que se puede transformar este problema de uno donde es imposible encontrar óptimos por encima de 15 trabajos, a uno donde prácticamente cualquier secuencia puede ser optimizada.

Si se añade la restricción de colores, de cantidad de trabajos por capacidad y de anchura, el programa sería capaz de encontrar óptimos al 100% de las secuencias dadas debido a que el espacio de resultados se reduce exponencialmente con cada restricción.

A modo personal, considero que esta debe ser la línea de trabajo futura.

8. BIBLIOGRAFÍA

Andrés Romano, C. (s.f.) *Apuntes de Programación y Control de Producción y Operaciones*

Chen, W.J,. (2009) "Scheduling with dependent setups and maintenance in a textile company"

El Amri, A. (2024) "Open AI GPT for Python Developers"

Euratex (2022), "Fatcs & Key Figures of the european textile and clothing industry"

Fink, A., Voss, S (2003) "Solving the continuous flow-shop scheduling problem by metaheuristics", *European Journal of Operational Research,* 151 (2) pp 400-414

Nawaz, M., Enscore, E. Ham, I. (1983) "Applying Iterated Local Search to the Permutation Flow Shop Problem", *Omega*, 11(1) pp. 91–95.

Ferrari, J, (2023) "El margen de beneficios de las empresas españolas toca máximos históricos en 2023", *ON Economia*

Ruiz, R. Vázquez Rodríguez, J.A. (2009) "The hybrid Flow shop scheduling problem".

Stützle, T. (1998) "Applying Iterated Local Search to the Permutation Flow Shop Problem"

Shore, B. and Warden, J., (2007). "A holistic approach to project scheduling using Gantt charts," *Journal of Project Management*, 25(3), pp. 50-62.

Sipser, M*.,* (2013). "Introduction to the Theory of Computation". *3rd ed. Boston: Cengage Learning.*

Schurmann, N. (2024), "Ultimate Python"

Tavakkoli-Moghaddam, R., Rahimi-Vahed, A., Mirzaei, A.H, (2007). "A hybrid multi-objective immune algorithm for a flow shop scheduling problem with bi-objectives: Weighted mean completion time and weighted mean tardiness"