UNIVERSIDAD TECNOLÓGICA NACIONAL
Facultad Regional Santa Fe

INGENIERÍA INDUSTRIAL

Proyecto Final de Carrera

Metodología de Optimización Matemática-Algorítmica para la Programación de Operaciones de Plantas Batch

---

Docentes: Ing. Fernando Imaz
Dra. Érica Fernández
Ing. Renzo Piccoli

Director: Dr. Pablo Marchetti

Alumno: Josías Adiel Stürtz Schultheisz
Contacto: josiassturtz@gmail.com

10 de Diciembre, 2021

Índice General

Lista de Algoritmos

Lista de Figuras

Lista de Tablas

Capítulo 1

PRESENTACIÓN DEL PROYECTO

1.1. Introducción

Los avances en el ámbito de la tecnología de la información y su creciente aplicación en diversas áreas brindan, a disposición de la industria, herramientas novedosas generando ambientes más dinámicos y competitivos. Esta competencia produce un incremento en el nivel de exigencia para una eficiente utilización de los recursos la cual, junto a la celeridad con la que ocurren los cambios, exponen la necesidad de incorporar mejoras no solo en los procesos productivos sino también en las herramientas para su gestión.

En las plantas industriales, la flexibilidad y agilidad favorecen la habilidad para realizar las entregas a tiempo y obtener mejoras respecto a productos o servicios de calidad y precio similares. De esta forma, se pone de manifiesto que las herramientas para el soporte y la toma de decisiones en el ámbito productivo son de importancia estratégica y constituyen una ventaja competitiva para las organizaciones.

En la actualidad, en muchas industrias las características del mercado favorecen a una mayor diversificación de productos, cada uno con menor volumen de producción, lo que acrecienta el interés en instalaciones industriales con equipos adaptables. En este tipo de plantas es necesario repartir los recursos entre los diferentes productos que se fabrican y garantizar la utilización óptima de la capacidad productiva reduciendo el tiempo de

inactividad o el uso ineficiente de los mismos. En este contexto, la programación de la producción en la industria de los procesos constituye un campo activo de investigación y desarrollo a fin de optimizar la producción, buscando soluciones para incrementar la eficiencia y rentabilidad de las organizaciones.

1.2. Objetivos del Proyecto

Mediante el estudio y experimentación con el uso de herramientas de modelado y optimización matemática, en el desarrollo del presente Proyecto Final de Carrera se buscó idear, implementar y evaluar una estrategia para la resolución de problemas de programación de operaciones a corto plazo, o problema de *"scheduling"* de carácter predictivo, en plantas industriales discontinuas multiproducto multietapa. El objetivo principal del proyecto se enuncia como:

> *"Desarrollar una metodología para la resolución de problemas de "scheduling" de gran escala, en plantas "batch" multiproducto multietapa, y aplicarlo a un caso de estudio real de la industria farmacéutica obteniendo soluciones eficientes con información precisa de las cotas de calidad de las mismas"*

Seguidamente, se detallan los objetivos específicos que condujeron progresivamente a alcanzar el objetivo general:

(I) *Analizar el estado del arte en la literatura científica abocada al estudio de los problemas de "scheduling" y valorar los aportes más relevantes.*

(II) *Obtener un modelo ajustado del problema utilizando Programación Matemática Mixta Entera Lineal.*

(III) *Desarrollar una estrategia de descomposición del problema y construcción paso a paso de la solución priorizando la asignación de los recursos críticos y conservando las cotas de calidad.*

(IV) *Obtener soluciones para diferentes instancias del caso de estudio incrementando gradualmente su complejidad.*

(V) *Generar figuras y gráficos que permitan analizar el desempeño de la metodología y comparar las soluciones obtenidas.*

(VI) *Discutir los resultados obtenidos.*

1.3. Ámbito que Abarca

El problema abordado en el presente proyecto tiene incumbencia en el ámbito de las decisiones operativas, las cuales impactan concretamente en el piso de planta y en el corto plazo de tiempo. Una vez determinadas las metas específicas de producción, y asegurada la disponibilidad de materiales y capacidad suficiente en la instalación para fabricar los productos requeridos por período, se procede a obtener un programa de operaciones. Dicho programa o *"schedule"* debe especificar el cronograma de trabajo para cada equipo, y el detalle de asignación de los recursos necesarios, que garanticen las cantidades de producto final especificadas en el Plan Maestro de Producción.

El problema de *"scheduling"* predictivo se aplica, entre otros, a industrias de procesos con flujo de materiales discontinuo que se presenta como una secuencia de lotes o *"batches"*. La temática planteada se focaliza además en instalaciones industriales multiproducto multietapa, en las cuales se procesan dos o más productos que tienen recetas de elaboración similares.

En la actualidad, debido a causas como variaciones en la demanda por cambios estacionales o del mercado, o una mayor diversificación de productos desarrollados, un número importante de industrias de manufactura deben adaptarse para operar con flexibilidad. Los procesos *"batch"* son particularmente convenientes para los casos en que se debe fabricar una cantidad de productos similares con bajos volúmenes de producción, ya que ofrecen la posibilidad de procesarlos en la misma instalación. En este tipo de plantas, debido a la existencia de recursos compartidos, es de gran importancia que la agenda de utilización de los mismos o *"schedule"* sea lo más eficiente posible en base al criterio que mejor represente sus necesidades. Entre algunos casos en los que se deben tomar decisiones de *"scheduling"* se pueden mencionar plantas industriales del rubro de alimentos y bebidas, metales, petróleo y gas, química, pulpa y papel, farmacéutica, entre otras. En particular, en esta última industria se enfoca el caso de estudio del presente trabajo.

1.4. Aportes Alcanzados

Dentro de los aportes planificados en este trabajo se encuentran:

- Un modelo matemático para la programación de la producción en plantas *"batch"* multiproducto multietapa que tenga un buen desempeño computacio-

nal.

- Una estrategia eficiente para la búsqueda de soluciones a problemas de programación de la producción priorizando la programación de los recursos críticos compartidos.

Como aspiración amplia, se persiguió lograr una contribución científica valiosa aplicable a diversos casos similares al estudiado, o bien, con características comunes que permitan hacer un paralelismo. Por sí mismo o en conjunción con técnicas ya estudiadas o por desarrollar, se buscó que resulte en un aporte de utilidad para la innovación y el desarrollo tecnológico en la disciplina.

1.5. Estructura del Proyecto

Luego del *Capítulo 1: Presentación del Proyecto*, en el cual se realizó una breve introducción al mismo, en el *Capítulo 2: Marco Teórico y Estado de la Situación Actual*, se presenta una integración y exposición de la información correspondiente al marco conceptual de referencia para el desarrollo del proyecto. Dicho capítulo contiene referencias a la bibliografía y literatura científica sobre la que se sustenta el trabajo presentando un análisis de los aportes y antecedentes más relevantes del área.

En el *Capítulo 3: Formulación Matemática del Problema*, se describe formalmente el problema a abordar planteando las hipótesis y supuestos del mismo. Además, se plantean las bases conceptuales de la formulación a utilizar y se expresa el problema abordado en términos algebraicos mediante un modelo matemático. Por otro lado, el *Capítulo 4: Estrategia Algorítmica de Resolución*, presenta el desarrollo de la metodología propuesta para arribar a una solución al problema, y los detalles requeridos para su aplicación al problema abordado.

A continuación, el *Capítulo 5: Resolución del Caso de Estudio*, expone las soluciones obtenidas para las diferentes instancias de complejidad creciente y las formulaciones planteadas para el caso de estudio. Asimismo, se analiza el desempeño de la metodología propuesta y se comparan los resultados. Para terminar, en el *Capítulo 6: Conclusiones y Trabajos Futuros*, se desarrollan las conclusiones obtenidas durante la realización del presente proyecto y se plantean posibles líneas de trabajo futuro.

Capítulo 2

MARCO TEÓRICO Y ESTADO DE LA SITUACIÓN ACTUAL

2.1. Planificación y Control de la Producción en la Industria

Para introducir la temática del presente Proyecto Final se revisarán, en primer lugar, las decisiones que se toman durante el proceso de planificación en los ambientes industriales. Con el fin de conseguir un resultado óptimo, se establece una clasificación de las mismas según el horizonte temporal en el que tienen efecto (Chase & Jacobs, 2009; Domínguez Machuca, 2003). Como se ilustra en la Figura 2.1, se pueden considerar decisiones que impactan en el largo, mediano y corto plazo, etapas respectivamente nombradas como: (i) diseño, donde se toman decisiones estratégicas relativas a la estructura de la planta, diseño y reingeniería de los productos, entre otros; (ii) planificación, correspondiéndose con decisiones tácticas con relación a alcanzar las metas de producción establecidas para el período, como el número y tipos de productos a elaborar, materiales y recursos necesarios para cubrir la demanda, fechas estimadas de lanzamiento de las órdenes, etc.; y (iii) programación, la cual comprende decisiones operativas y el desarrollo de los mecanismos para su retroalimentación y control. Entre algunas de ellas pueden mencionarse la especificación de las operaciones a realizar, la determinación de los tiem-

pos precisos de comienzo y finalización de estas operaciones, la asignación de recursos, el dimensionamiento y secuenciación de los lotes de producción, la reprogramación de operaciones, etc.

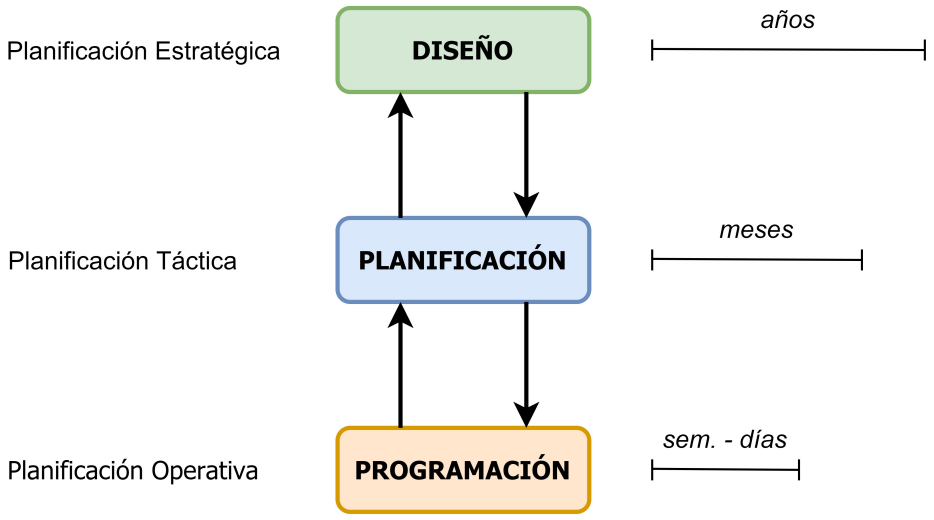

Figura 2.1: Diseño, Planificación y Programación

Fuente: Castro y col. (2018)

En primer lugar, establecer la estrategia empresarial implica definir aspectos ligados de forma estrecha al principal objetivo y metas de la organización. Estos lineamientos le otorgan un sentido y propósito de existencia a la misma y se definen considerando largos plazos de tiempo, generalmente contados en años. Una vez definidos estos caminos de acción, proyectar la producción industrial involucra actividades dentro del área táctica como la planificación agregada de producción, generación de un plan maestro de producción y elaboración de un plan de requerimiento de recursos. Por otro lado, también incluye actividades vinculadas con el área operativa de la industria, entre las cuales se pueden mencionar una planificación de requerimiento de capacidad, elaborar un plan de requerimiento de materiales, la programación de operaciones de corto plazo y el control de producción. Como se indica en la Figura 2.2, dichas actividades se hacen efectivas articulando la planificación de la producción y la planificación de la capacidad productiva, teniendo estas una influencia recíproca, con el fin de garantizar el balance más conveniente.

Siguiendo un orden jerárquico de las actividades, en primer lugar, se desarrolla el *Plan Agregado de Producción* con el objetivo de lograr un nivel de producción que utilice un alto porcentaje de la capacidad productiva proyectada como disponible. En este nivel se trabaja con nociones genéricas para considerar los recursos de toda la planta industrial, y se diferencia entre las familias de productos, es decir, agrupando aquellos que

poseen recetas de fabricación similares. Se busca así establecer los niveles de producción e inventarios por familia de productos, en general, para un horizonte de entre 6 a 24 meses y un *Plan de Requerimiento de Recursos* con las necesidades de personal y de recursos productivos para toda la planta.

A continuación, se efectúa un *Plan Maestro de Producción*, también con un horizonte de análisis a mediano plazo. Teniendo como entrada el *Plan Agregado de Producción*, el resultado serán metas específicas de producción mediante la desagregación de las metas de cada familia. El plan maestro es especificado para cada producto individual y con mayor detalle sobre los plazos de tiempo, pudiendo analizarse períodos en días, semanas, o hasta un mes. En esta instancia, las verificaciones de capacidad son más estrictas para los equipos con alto índice de ocupación y se define un *Plan de Requerimientos de Capacidad Preliminar* trabajando de forma conjunta a la desagregación por productos desarrollada.

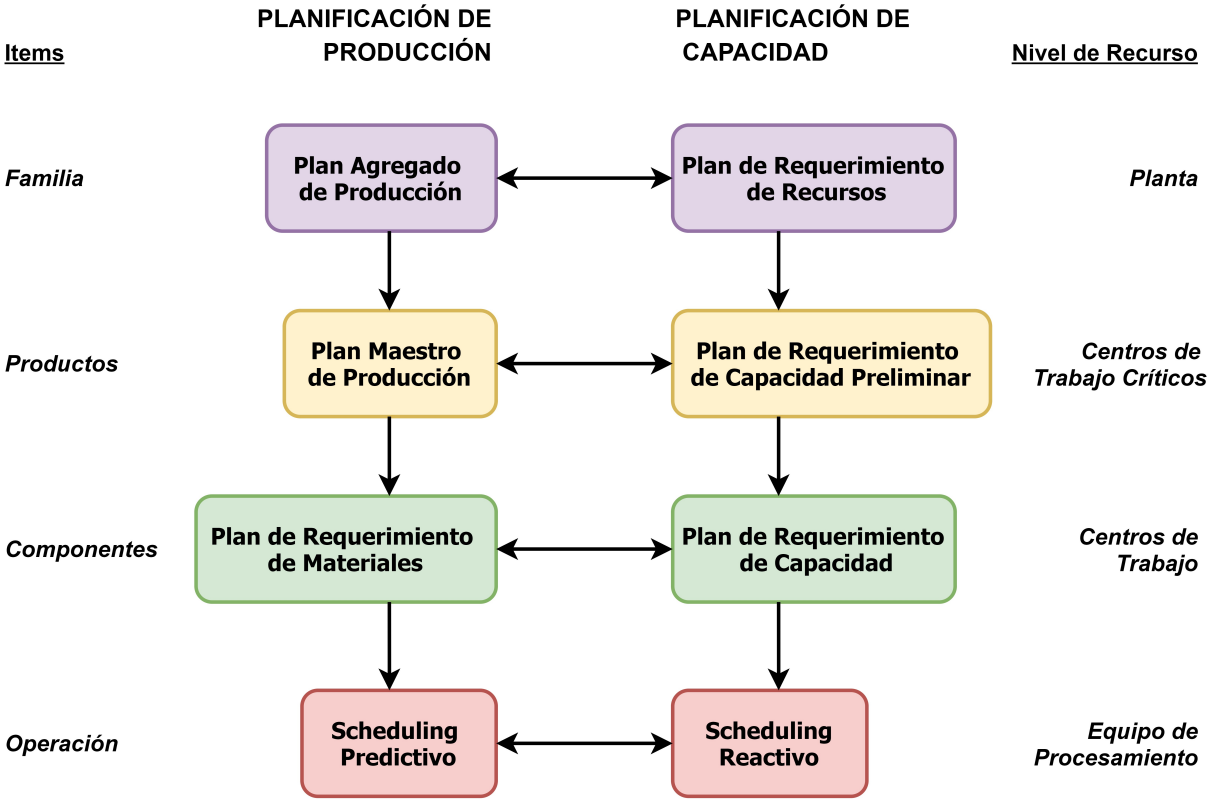

Figura 2.2: Planificación de Producción y Planificación de Capacidad

Fuente: Domínguez Machuca (2003)

La siguiente tarea será formular un *Plan de Requerimiento de Materiales*, con el objetivo de asegurar la disponibilidad de materias primas e insumos necesarios. El mismo se realiza con un horizonte temporal a mediano y corto plazo, abarcando decisiones del

ámbito operativo. Son especificados los requerimientos de todos los componentes individuales necesarios en la fabricación de los productos, incluidos en la lista de materiales (también conocida como BOM proveniente del inglés *"Bill of Materials"*). En conjunto, se trabaja en el *Plan de Requerimiento de Capacidad*, para garantizar la capacidad productiva que se precisa en cada uno de los centros de trabajo por los que pasarán los productos. Es de importancia saber que para formular el *Plan de Requerimiento de Materiales* es necesario trabajar de forma regresiva en el tiempo, partiendo de las fechas de entrega programadas para cada producto, pasando luego por los tiempos necesarios para su producción, y llegando a la estimación de la fecha de emisión de la orden de compra teniendo en cuenta los tiempos necesarios para el arribo y disponibilidad del componente en la planta. Trabajando con el *Plan de Requerimientos de Capacidad* y el *Plan de Requerimiento de Materiales* de forma simultánea, en caso de que la capacidad disponible se vea desbordada, se actualizan el *Plan Maestro de Producción* y el *Plan de Requerimiento de Materiales* o se analiza la ampliación de la misma, por ejemplo, añadiendo mano de obra en tiempo extra o habilitando nuevos equipos en los períodos que sea necesario.

En cuarto y último lugar, se realiza la *Programación de la Producción de Corto Plazo* o *Scheduling Predictivo*, con el objetivo de arribar a un programa detallado para cada equipo de trabajo, especificando la manera en que se asignarán los recursos y el tiempo preciso en que tendrá lugar cada operación. Naturalmente, este proceso toma decisiones a nivel de piso de planta, las cuales pertenecen al ámbito operativo, y tiene un horizonte de planeamiento corto, de no más de quince días.

Durante la ejecución del programa de operación obtenido o *"schedule"*, tiene lugar un control de la producción que verifica posibles discrepancias entre la capacidad de recursos utilizados y la cantidad de productos obtenida que fue programada. En general, en piso de planta pueden ocurrir imprevistos que distorsionen el plan original y obliguen a reprogramar tareas con las nuevas condiciones, por lo que surge la actividad nombrada como *Scheduling Reactivo* o, también llamada *Scheduling Dinámico*. Entre los inconvenientes más frecuentes que pueden causar una discrepancia entre el *Scheduling Predictivo* y los sucesos en tiempo real se encuentran: demoras en la producción por errores en los tiempos programados, cambio en los lotes que se quieren procesar, variación en la disponibilidad de recursos, inconvenientes en ciertos equipos que ocasionen disminución en su productividad o su salida de servicio, etc.

En particular, en este Proyecto Final se aportan avances en el desarrollo de modelos y metodologías de optimización para problemas de *"scheduling"* predictivo en la industria de procesos. Estas instalaciones, también llamadas plantas *"batch"*, presentan un flujo de

materiales intermitente que consiste en una secuencia de lotes o *"batches"*. En la siguiente sección, se desarrollan con mayor grado de detalle las características para este tipo de plantas productivas.

2.2. Caracterización de las Plantas *"Batch"*

Una instalación *"batch"*, o planta de procesamiento por lotes, puede estar compuesta por una única etapa de fabricación, conocida en este caso como planta monoetapa, o incluir múltiples procesos organizados en una serie de etapas de procesamiento, es decir, constituyendo una instalación multietapa. A su vez, cada etapa de la planta está integrada por equipos que operan procesando lotes de distintos tipos y tamaños. Las etapas de la instalación están conectadas entre sí para llevar a cabo una serie de tareas de acuerdo a la receta de procesamiento del producto. En general, cada lote se considera compuesto por un solo tipo de producto y es asignado a un solo equipo para cada operación. Luego de una etapa de procesamiento, se obtiene un producto intermedio o, en el caso de la última etapa de la receta, el resultado es el lote de producto terminado.

Seguidamente, se describen algunas características distintivas para entender el funcionamiento de las plantas *"batch"*.

2.2.1. Estructuras de flujo de los materiales

Una característica fundamental en la clasificación de las plantas industriales es la manera en que se presenta el flujo de materiales, pudiendo ser continuo, discontinuo o semicontinuo. Cuando el flujo de materiales es continuo, el material circula de forma estable durante toda la campaña de producción. En este caso, la duración de cada campaña puede medirse en periodos que se extienden desde semanas hasta meses, y sus interrupciones se dan de forma programada para la puesta a punto de los equipos y cambios del producto a fabricar. Esta estructura es principalmente utilizada en industrias que fabrican un número reducido de productos con alto nivel de estandarización y, por lo general, demanda sostenida en el tiempo.

Por otra parte, si el flujo de materiales se presenta de forma irregular, como una secuencia de lotes o *"batches"*, se trata de una planta discontinua. Cada lote de procesamiento sigue una receta de operaciones y se desplaza pasando por sucesivos estados intermedios hasta que obtiene su presentación final. Para procesar cada *"batch"*, deben

considerarse los tiempos de preparación y limpieza de los equipos, sus tiempos de carga y descarga y la duración del proceso en sí. En los equipos de procesamiento por lotes, normalmente el tamaño de los mismos se verá limitado por la capacidad del equipo o, de otro modo, deberá ser determinado buscando la utilización óptima de todos los recursos compartidos.

En el caso de las instalaciones semicontinuas, existe una combinación de maquinarias de los dos tipos mencionados anteriormente. Esto, dada la necesidad de acoplar los equipos y balancear el flujo de productos, presenta complejidades que a menudo obligan a incluir almacenes intermedios compatibles con el tipo de producto que se manufactura.

2.2.2. Unidades de procesamiento y estrategias de producción

Cuando una planta *"batch"* es utilizada para fabricar una única clase de producto, recibe el nombre de planta *"batch"* monoproducto. En este caso, todas las etapas de la instalación están físicamente dispuestas en el orden en que intervienen en la fabricación del producto, comúnmente con disposiciones en forma de línea recta, S o U. En las líneas de producción o ensamble monoproducto es posible lograr un balance de la capacidad productiva casi perfecto entre todas las etapas, lo cual resulta ventajoso para operar con economías de escala.

Por otro lado, cuando una planta *"batch"* se utiliza para procesar más de un producto pueden surgir dos tipos de plantas: multiproducto y multipropósito. Una planta *"batch"* multiproducto, también llamada *"flowshop"*, procesa *"batches"* que presentan similitud en sus recetas de elaboración y, por lo tanto, siguen básicamente la misma secuencia de operaciones. En este caso, la disposición física de la planta de producción se diseña ubicando las etapas de forma consecutiva, similar a las instalaciones monoproducto.

En las líneas de producción *"batch"* multiproducto, la estrategia base que más se adecúa para la producción se denomina *"make-to-stock"* (MTS), o sistema de producción *"push"*. En este sistema, la producción no se basa en la demanda actual, sino que se fabrica para cubrir órdenes de reposición de inventario anticipando una demanda futura aproximada. Un punto ventajoso de este enfoque consiste en la eliminación de los tiempos de espera para los clientes, lo cual es muy útil cuando los tiempos de producción son altos. En contraste, las desventajas de la estrategia MTS son los altos costos de inventario, el riesgo de obsolescencia de los productos y la dependencia de las previsiones de demanda.

Por otra parte, una planta *"batch"* multipropósito, en inglés *"jobshop"*, permite fabricar productos con recetas disímiles, los cuales pueden obtenerse a partir de etapas diferentes o una secuencia de procesamiento distinta. Los diferentes equipos de procesamiento se organizan en centros de trabajo agrupando operaciones y familias de productos. Es posible que el producto tenga que cumplir con más de una operación en el mismo centro de trabajo, saltear otros o, incluso, visitar en más de una oportunidad el mismo agrupamiento de equipos. La circulación de materiales en estas instalaciones puede volverse compleja ya que suelen aparecer cruces y movimientos no lineales.

En este tipo de instalación, la estrategia de fabricación más conveniente se denomina *"make-to-order"* (MTO) o sistema *"pull"* ya que son los pedidos de los clientes o la demanda real la que tracciona el producto que se va a procesar. Las principales ventajas de este enfoque consisten tanto en la eliminación de los costos de inventario, ya que no se almacena producto terminado ni se generan productos intermedios, como en la no dependencia de los pronósticos de demanda. En contraposición, las principales desventajas son los largos tiempos que el cliente debe esperar para recibir su compra, y las dificultades de programar un tiempo de entrega para la misma y organizar los procesos para su fabricación.

2.2.3. Equipos de procesamiento en paralelo

En una instalación de tipo *"flowshop"*, las etapas que componen la línea de producción están dispuestas según la secuencia de procesamiento especificada en las recetas de los productos. Suponiendo que se trata de una línea de producción monoproducto y cada etapa está integrada por un único equipo de procesamiento, el tiempo de ciclo de dicha línea estará determinado por el mayor de ellos para todos los equipos. La etapa que determina la extensión del ciclado será considerada cuello de botella de la línea y centrará toda la atención para lograr su mayor eficiencia. Estas diferencias en los tiempos de procesamiento resultan en tiempos ociosos para los demás recursos productivos que, por lo tanto, bajan su índice de utilización.

Cuando la capacidad productiva de todos los equipos se encuentra perfectamente balanceada no habrá una etapa en particular que pueda considerarse crítica y el tiempo ocioso de toda la línea podrá ser reducido a su mínima expresión. De otro modo, si la planta productiva presenta marcados desbalances, una opción a considerar es la adición de equipos de procesamiento en paralelo en las etapas cuello de botella. En este caso, para una etapa que ahora está compuesta por dos equipos idénticos trabajando en paralelo se

logrará reducir a la mitad su tiempo de ciclo. Sin embargo, dicho indicador limitante de la línea productiva en su conjunto dependerá además del valor que tome en el resto de las etapas.

Por otra parte, considerando ahora una planta en la que se procesa más de un tipo de producto, el análisis toma mayor grado de complejidad. Esto se debe a que en estas instalaciones, sean multiproducto o multipropósito, los artículos fabricados pueden tener diferentes tamaños de lote, requerimientos de capacidad y tiempos de procesamiento. A su vez, en el caso de las plantas multipropósito, las órdenes que se llevan a cabo pueden diferir entre sí en su secuencia de procesamiento y, por lo tanto, debe considerarse la diversidad de recetas de producción. Por lo tanto, que un equipo o etapa sea considerado cuello de botella, dependerá también de las características de los productos a fabricar. Además, la complejidad del análisis puede ser aún mayor en caso de que los equipos de procesamiento que componen una etapa no sean idénticos entre sí. En conclusión, incluir equipos de procesamiento en paralelo es una opción para incrementar la capacidad productiva de una etapa de procesamiento, y su diseño involucra conocer en detalle diversos aspectos del caso en particular.

2.2.4. Políticas de transferencia y almacenamiento intermedio

En una planta *"batch"* debe definirse la existencia o no de depósitos de almacenamiento intermedio, entre las etapas consecutivas de la línea de producción, de acuerdo con las características de los productos fabricados. A pesar de requerir una inversión económica inicial y, en algunos casos, costos de operación, es posible que su inclusión permita disminuir los tiempos ociosos de los equipos. Añadir depósitos de almacenamiento intermedio posibilita desacoplar las operaciones de un mismo producto en dos etapas consecutivas y amortiguar el tiempo de espera hasta que sea procesado en la etapa siguiente. Como consecuencia, se alcanza mayor flexibilidad en la secuenciación de las tareas para los diferentes equipos.

Entre los diferentes modos de operación y políticas de espera, almacenamiento intermedio y transferencia de los productos semiterminados se encuentran:

- **Transferencia sin espera (*"Zero Wait"* – ZW)**
  Una vez finalizado el procesamiento de un lote en cierta etapa, inmediatamente es transferido y comienza su operación en la etapa siguiente. Este modo de operación no requiere de almacenamiento intermedio y es utilizado en los casos

en que la calidad del producto se ve afectada al interrumpir el procesamiento del mismo.

- **Sin almacenamiento intermedio (*"No Intermediate Storage"* – NIS)**
  No existen dispositivos de almacenamiento intermedio entre las etapas. Al finalizar una operación, el lote permanece en el equipo hasta que haya un lugar disponible para realizar la operación de la etapa siguiente. Mientras el lote espera en el equipo, este último no se encuentra disponible para procesar otra orden, por lo que su tiempo ocioso aumenta.

- **Almacenamiento intermedio finito (*"Finite Intermediate Storage"* – FIS)**
  Se cuenta con depósitos de almacenamiento intermedio de capacidad limitada entre las etapas, pudiendo éstos ser para un único producto o de uso compartido. Esto supone, mínimamente, capacidad suficiente para almacenar un lote de cierto producto.

- **Almacenamiento intermedio ilimitado (*"Unlimited Intermediate Storage"* – UIS)**
  Entre cada etapa consecutiva de la planta existen tanques de almacenamiento intermedio sin restricciones de capacidad, suponiendo que la misma es suficiente para contener todo el material generado en las diferentes etapas de procesamiento.

2.3. Elaboración de Modelos para la Optimización Matemática

Un modelo matemático es la descripción abstracta de un hecho, situación o problema utilizando lenguaje matemático, y es aplicado en diversas áreas tanto de investigación y desarrollo como en aplicaciones industriales concretas. El modelado es considerado una herramienta de gran importancia para la generación de conocimiento, ya que, durante el proceso de construcción de los modelos, con frecuencia se revelan relaciones que no son evidentes a primera vista produciendo una mejor comprensión del fenómeno. Por otra parte, una vez desarrollado un modelo es posible obtener relaciones entre los elementos del mismo y, en algunos casos, llegar a una resolución computacional del problema asociado.

De forma genérica, un modelo matemático se compone de los parámetros o datos del problema, las variables discretas o continuas, que serán utilizadas para representar las decisiones del modelo, y las restricciones en forma de ecuaciones, que incluyen la relación entre los parámetros y variables, y representan la región de soluciones factibles. Además, si es el caso de un modelo matemático de optimización, se buscará maximizar o minimizar un criterio de calidad de las soluciones mediante una función objetivo, la cual establece una relación entre las variables de decisión. Un modelo matemático de optimización puede escribirse en lenguaje algebraico, como expresión general, de la siguiente manera:

$$\begin{aligned} Min \quad & Z = f(x,y) \\ s.a. \quad & h(x,y) = 0 \\ & g(x,y) \leq 0 \\ & x \in X \subseteq \mathbb{R} \\ & y \in 0,1 \end{aligned} \tag{2.1}$$

Donde $f(x,y)$ representa una función objetivo escalar, la cual se busca minimizar o maximizar, y $h(x,y)$ y $g(x,y)$ son las funciones asociadas a las restricciones del problema en forma de igualdades y desigualdades respectivamente. Las variables x son determinadas variables continuas correspondientes a números reales en un cierto rango, mientras que las variables y son variables discretas, que en la mayoría de las aplicaciones están restringidas a valores 0 ó 1, y sirven para tomar decisiones del tipo "sí" o "no".

2.4. Enfoques para la Obtención de Soluciones

En un problema de optimización la selección de una estrategia para la búsqueda de soluciones contempla, en primer lugar, establecer los criterios que son de importancia para su alineación con los objetivos del estudio. Luego, el planteamiento de metas será útil para establecer los niveles mínimos con los que se busca satisfacer estos criterios. Para ello, debe tenerse en cuenta que la solución óptima forma parte de un *"ideal"* el cual, en muchos casos, no es sencillo de hallar. En consecuencia, una solución calificada como *"satisfactoria"* comienza a ser válida cuando supera ciertas medidas establecidas, por ejemplo, en comparación con otras técnicas conocidas u otras opciones disponibles para abordar la situación.

En esta búsqueda de criterios satisfactorios para la toma de decisiones, se han desarrollado variados procedimientos y métodos para obtener soluciones a un modelo matemático. Algunos de ellos, conocidos como métodos descriptivos, se nutren de la simulación, visualizando escenarios posibles y experimentando con variaciones en los parámetros del sistema. Esto permite transitar un proceso de aprendizaje acerca del comportamiento del modelo y arribar a soluciones en base a ello. Por otra parte, los modelos predictivos tienen el objetivo de optimizar uno o más criterios precisos y encuentran soluciones factibles valiéndose de diferentes metodologías, ya sean exactas o aproximadas.

Seguidamente se desarrollan las metodologías predictivas más utilizados en la actualidad en el ámbito de la programación de operaciones a corto plazo.

2.4.1. Metodologías exactas

Las metodologías exactas más utilizadas para resolver problemas de *"scheduling"* emplean Programación Matemática Mixta-Entera Lineal (MILP) o, en menor medida, Programación Matemática Mixta-Entera No-Lineal (MINLP). En ambos casos, se desarrolla un modelo matemático de optimización compuesto por variables discretas y continuas. La diferencia entre ambas metodologías tiene origen en la linealidad o no-linealidad tanto de las restricciones como de la función objetivo.

Los algoritmos con mayor éxito para resolver MILPs (Caballero & Grossmann, 2007) están fundamentalmente basados en los métodos de ramificación y acotamiento (*"Branch and Bound"*), donde cada subproblema lineal se resuelve utilizando el algoritmo simplex (Dakin R. J., 1965). Este método consiste en una enumeración de subproblemas con estructura de árbol, en el cual, el nodo inicial resuelve el problema con las variables enteras relajadas como variables continuas, de tal forma que se les permite tomar valores no enteros. Este nodo inicial, produce una cota inferior global al óptimo del problema (para problemas de minimización). Se debe seleccionar una de las variables que han tomado valores fraccionarios para ramificar y se resuelve el nuevo problema lineal con la variable seleccionada fija. Para la elección del nodo a ramificar se utilizan reglas heurísticas (búsqueda en profundidad, búsqueda en anchura, etc.).

Por otra parte, en la actualidad muchos métodos MILPs corresponden a técnicas del tipo ramificación y corte (*"Branch and Cut"*). Las mismas surgen de la aplicación de planos de corte (Gomory, 1958) luego del desarrollo de la técnica *"lift and project"* propuesta por Balas y col. (1993). Los mismos se utilizan para ajustar la relajación del

subproblema lineal correspondiente a cada nodo del árbol. Inicialmente, se resuelve el problema lineal relajado con el algoritmo simplex y, al obtener una solución óptima no entera, los planos de corte son utilizados para encontrar restricciones lineales adicionales que son satisfechas por todos los puntos enteros factibles pero violadas por la solución no entera actual. En este punto, se continúa con el método *"Branch and Bound"* y los nuevos problemas lineales se resuelven con el método simplex repitiendo el proceso.

Es importante remarcar que en estos métodos, en el proceso de exploración del árbol de alternativas, un nodo cualquiera es una cota inferior para todos los nodos posteriores generados a partir de éste (ramificación), es decir, a medida que se desciende por las ramas del árbol la función objetivo de los nodos va creciendo. Por otro lado, cuando en un nodo se obtiene una solución entera, ésta es un límite superior a la solución óptima del problema, de tal forma que todas las ramas abiertas con valor superior de la función objetivo no necesitan evaluarse (acotamiento). La enumeración continúa hasta que la diferencia entre las cotas inferior y superior están dentro de una tolerancia o bien no existen ramas abiertas. En la actualidad, los principales códigos comerciales para resolver MILPs son CPLEX, XPRESS y GUROBI.

En cuanto a los principales métodos para resolver MINLPs, se pueden mencionar: (i) Ramificación y Acotamiento (Borchers & Mitchell, 1994; O. K. Gupta & Ravindran, 1985) los cuales son una extensión directa de los métodos de ramificación y acotamiento empleados para resolver MILPs. (ii) Métodos de descomposición que iteran entre dos subproblemas, como la Descomposición de Benders Generalizada (Geoffrion, 1972) y el Método de las Aproximaciones Exteriores (Duran & Grossmann, 1986; Fletcher & Leyffer, 1994). (iii) El Método LP-NLP basado en ramificación y acotamiento (Quesada & Grossmann, 1992), el cual es un enfoque intermedio entre los métodos de ramificación y acotamiento y los métodos de descomposición. (iv) El método de Plano de Corte Extendido (Westerlund & Pettersson, 1995), que es una variación que no requiere la solución de NLPs. La principal dificultad de los problemas MINLPs es el tratamiento, si las hubiere, de las no convexidades en la función objetivo o restricciones que generan la aparición de óptimos locales.

2.4.2. Metodologías aproximadas

Las metodologías aproximadas utilizan ideas basadas en la intuición o experiencia para resolver los problemas de optimización. En general, estas técnicas tienen la ventaja de que su implementación puede ser sencilla y permiten, en ocasiones, obtener soluciones

de buena calidad con esfuerzos computacionales razonables. Una característica distintiva de estas técnicas es que, para la solución hallada, no se garantiza su optimalidad ni se establece lo cerca que se está de dicha situación. Las principales metodologías aproximadas utilizadas para resolver problemas de *"scheduling"* son la Programación por Restricciones, Metaheurísticas y las Reglas de Prioridad de Despacho.

Los modelos de Programación por Restricciones (Van Hentenryck, 1989) se basan en el incremento progresivo de restricciones, para cada nodo del árbol de soluciones, mediante la reducción de los dominios de las variables. De este modo, si se encuentra un dominio vacío, el nodo es descartado. Aunque en este apartado se clasifica como aproximada, esta metodología debe considerarse exacta si todas las alternativas posibles son exploradas, si se utiliza la técnica de *"Backtracking"*, la principal diferencia con el método de ramificación y acotamiento es que, tan pronto como se genera un nuevo hijo del nodo en curso, este hijo pasa a ser el nodo en curso. En cambio, el algoritmo *"Branch and Bound"* genera todos los hijos del nodo en curso y selecciona en base a las cotas, a cualquier otro nodo que pasará a ser el nuevo nodo en curso. En consecuencia, en *"Backtracking"* los únicos nodos abiertos son los que están en el camino de la raíz al nodo en curso y en *"Branch and bound"* puede haber más nodos abiertos, ya que, estos se almacenan en una estructura de datos auxiliar. En la Programación por Restricciones, además de la técnica de *"Backtracking"*, existen herramientasque permiten definir algoritmos ad hoc tales como estrategias heurísticas de ramificación específicas del problema.

Las Metaheurísticas, por su parte, son técnicas genéricas que pueden utilizarse en un dominio específico para construir heurísticos adecuados al problema que se esté resolviendo. Entre las más conocidas se puede mencionar: recocido simulado o *"Simulated Annealing"* (SA) (Aarts & Korst, 1989), Algoritmos Genéticos (GA) (Goldberg, 1989) y *"Tabu Search"* (TS) (Glover, 1990). En el caso de SA, genera paso a paso soluciones sin requerir una mejora en cada una de ellas, admitiendo una probabilidad de mantener una solución peor que la anterior. Esta probabilidad disminuye cuando aumenta el deterioro de la función objetivo o cuando crece el número de soluciones ya generadas. La meta de este enfoque es evitar quedar atrapado en un óptimo local. Por otro lado, la Metaheurística GA, utiliza una codificación que imita una cadena de ADN para cada individuo o, en el caso del problema abordado en este proyecto, para cada solución o *"schedule"* factible. De esta manera, la técnica GA conserva una población de individuos sujeta a criterios de supervivencia, mutación y recombinación de características positivas de cada uno de ellos. La evolución de la población asegura la supervivencia de los mejores especímenes imitando un proceso de selección natural y, de este modo, se obtienen soluciones de buena calidad.

Por último, la estrategia TS considera el conocimiento de las soluciones previas y genera una lista, guardada en la memoria, de atributos que están prohibidos o son tabú. Asímismo, conserva un cierto número de las últimas soluciones generadas y realiza una búsqueda de nuevas combinaciones evitando caer en alternativas exploradas anteriormente.

Y, finalmente, las Reglas de Prioridad de Despacho son una de las técnicas clásicas más conocidas para la resolución de problemas de *"scheduling"*. Las mismas permiten definir un orden entre las tareas que están esperando ser procesadas en algún recurso y, con ello, asignan la orden de mayor prioridad al recurso cuando el mismo esté disponible. Entre algunas de las más conocidas se encuentran (Blackstone y col., 1982): *"First In First Out"*, basada en atender primero la primera orden que ingresa al proceso; *"Earliest Due Date"*, que ordena los trabajos en función de su fecha de vencimiento; y *"Shortest Procesing Time"*, la cual considera el tiempo de procesamiento de cada trabajo para establecer su prioridad.

2.4.3. Metodologías de descomposición

Las metodologías de descomposición del problema son estrategias basadas en heurísticos que se utilizan, principalmente, para la simplificación de casos de aplicación reales de gran tamaño que con otros métodos resultan computacionalmente intratables. En general, para los problemas de gran escala se construyen estrategias híbridas, combinando metodologías exactas o aproximadas y técnicas de descomposición. Dentro de los principales enfoques de descomposición se tiene aquellos basados en el tiempo, en los recursos críticos y en los lotes u órdenes de fabricación (Bassett y col., 1996; Harjunkoski y col., 2014).

Al considerar la descomposición basada en el tiempo, aparecen distintas alternativas. Por un lado, una de las formas de descomposición estudiadas busca resolver un problema inicial para un subconjunto de lotes que deben ser procesados en cierta parte del horizonte de tiempo. Seguido a esto, se fijan las decisiones tomadas para dichos lotes y, avanzando en el horizonte de tiempo, se resuelve el problema nuevamente adicionando el próximo subconjunto de *"batches"*, pero teniendo en cuenta el estado actual del *"schedule"*. También considerando el tiempo, otra posibilidad es descomponer el problema completo con cierto nivel de agregación para los recursos, en intervalos de tiempo de duración predefinida, y luego llevar a cabo un ajuste local de la solución construida desagregando los recursos en mayor medida.

En cuanto a las metodologías basadas en la asignación prioritaria de los recursos críticos, éstas resuelven el problema identificando él o los recursos cuello de botella. De este modo, fijan su asignación y secuenciación de forma prioritaria e ignoran total o parcialmente los demás recursos, que aún no han sido programados. Esta forma de construir la programación de operaciones tiene origen en la suposición de que, al programar posteriormente los demás recursos, las decisiones fijadas para el cuello de botella no se verán afectadas.

Finalmente, otra técnica de descomposición estudiada consiste en la inclusión progresiva de las órdenes a fabricar. En este enfoque se incorpora de forma incremental, uno a uno, los lotes a procesar con algún criterio de prioridad. De esta forma, se construye la agenda para cada equipo de procesamiento en forma escalonada, fijando las decisiones de asignación y secuenciación de aquellas órdenes ya insertadas previamente, antes de incrementar el número de éstas.

2.5. Formulaciones MILP para el Problema de *"Scheduling"*

Partiendo de los aspectos ya definidos como la configuración de la planta, los equipos disponibles, las recetas de elaboración que describen las operaciones y sus relaciones de precedencia, los lotes que deben procesarse y sus tiempos de procesamiento en cada equipo; mediante la programación de operaciones se obtiene una agenda detallada indicando la asignación y secuenciación de los lotes, los tiempos precisos de comienzo y finalización de los mismos, los perfiles de utilización de los recursos, niveles de inventario, entre otros. Por lo tanto, la resolución de un problema de programación de operaciones a corto plazo o problema de *"scheduling"* tiene como fin brindar un programa detallado del uso de los recursos.

Durante el proceso de formulación del problema, una vez que fueron definidas todas las variables que representan las alternativas de decisión para el modelo anteriormente mencionadas, deben especificarse, en contraposición, las restricciones que acotan la región factible de soluciones del problema. Entre las más relevantes a considerar se encuentran las relativas a: disponibilidad limitada de recursos (equipos, materia prima, mano de obra, servicios, entre otros), precedencia de tareas y recetas de procesamiento de los productos, posibles prioridades de los lotes a procesar, tiempos de transición dependientes de la secuencia de procesamiento, tiempos a partir de los cuales se encuentran disponibles los

equipos y materias primas, fechas límite de entrega, longitud del horizonte de programación, y políticas de transferencia y almacenamiento de inventario.

En general, la tarea de programación de la producción, además de cumplir con las metas de producción planificadas, busca la utilización eficiente de los recursos compartidos y se ejecuta optimizando alguna medida de calidad de la solución. Entre las más estudiadas se pueden considerar:

- **Tiempo total de producción o *"makespan"*:** es el tiempo total requerido para completar el procesamiento de todas las operaciones, busca la utilización eficiente de los recursos.
- **Anticipación y/o tardanza ponderada total:** considera la suma de los adelantos y retrasos de las órdenes con respecto a sus fechas de finalización y entrega establecidas. Se considera que los costos de inventario están asociados a la finalización anticipada de una tarea, y la tardanza en las fechas de entrega de un producto se vincula con los costos de insatisfacción de los clientes. Utilizando coeficientes de ponderación es posible dar mayor o menor peso a cada adelanto/retraso posible.
- **Costo de operación:** al reducir los costos operacionales se maximizan los beneficios obtenidos con el funcionamiento del sistema productivo.

Una vez definido el único o los múltiples criterios que se busca optimizar, en líneas generales, existen adicionalmente tres aspectos importantes a considerar para generar el modelo de optimización (Méndez y col., 2006). A continuación, se describen las principales formas de representar el tiempo, las opciones para la secuenciación de eventos que estas generan, y las alternativas para abordar el balance de los materiales en la resolución del problema.

2.5.1. Representación de la variable tiempo

Uno de los principales aspectos a considerar al formular matemáticamente el problema de *"scheduling"* es la forma de modelar el tiempo. El mismo se puede definir con una escala continua, en la que es posible que los eventos ocurran en cualquier instante dentro del horizonte de planeamiento o, en contraposición, se puede representar como una escala de valores discretos, forzando a que los acontecimientos tengan lugar en ciertos instantes de tiempo predefinidos.

En el primer caso, las variables que representan el tiempo de inicio y finalización de las tareas son continuas (es decir,definidas en el rango de los números reales), por lo que garantizan que la solución se ajuste de forma precisa a los datos del problema. Una de las principales desventajas es que, en la mayoría de los problemas, su uso acarrea incluir en el modelo restricciones del tipo Big-M las cuales, en muchos casos, tienen impacto negativo en el desempeño computacional de la formulación. Más específicamente, incrementan considerablemente la brecha entre la solución relajada (aquella que se obtiene con las variables discretas del modelo relajadas a valores continuos) y la solución exacta buscada, lo que impacta negativamente incrementando la dificultad de convergencia a la solución óptima.

Por otra parte, cuando se utiliza una representación discreta del tiempo, los eventos se ven obligados a tomar lugar en instantes específicos de una grilla de tiempo preestablecida. El horizonte de tiempo se divide en un número finito de intervalos de duración fija y, las variables de decisión que modelan el *"schedule"*, asocian los eventos al inicio o finalización de éstos. En esta representación, la solución obtenida puede ser una aproximación de la realidad y, en consecuencia, posiblemente será infactible dependiendo del grado de precisión necesario para el problema. Esta formulación es conveniente, brindando mayor facilidad de resolución, cuando existe cierta compatibilidad entre el valor de los parámetros del problema relacionados con el tiempo y la grilla de tiempo discreta utilizada. Sin embargo, debe tenerse en cuenta que una grilla con un número elevado de intervalos incrementa notablemente el tamaño del modelo y su dificultad de resolución computacional.

2.5.2. Representación de eventos

Existen diferentes conceptos utilizados para ordenar los eventos con el propósito de garantizar que la capacidad disponible de los recursos compartidos no se vea excedida en ningún momento. Según la literatura abocada a la formulación matemática de problemas de *"scheduling"*, una clasificación general de estos conceptos de acuerdo a la representación elegida para el tiempo, es la siguiente:

- Para una escala de tiempo discreta puede utilizarse la representación de eventos mediante *intervalos de tiempo globales*, con una grilla fija de tiempo.
- Para escalas de tiempo continuo las principales variantes estudiadas en la literatura son: *puntos de tiempos globales*, *eventos de tiempo asociados a equipos*,

ranuras de tiempo o "time-slots" y *precedencia inmediata* o *general*.

Para las formulaciones discretas el tiempo se modela mediante *intervalos de tiempo globales* (Figura 2.3-(a)), lo que ubica los eventos en una grilla de tiempo fija común para todos los recursos compartidos. Por lo tanto, el problema de *"scheduling"* se resume a un problema de asignación de determinadas tareas a los intervalos de tiempo disponibles mediante variables binarias. Esto tiene como ventaja que ciertos aspectos del problema dependientes del tiempo, como puede ser el costo por utilización de los equipos, pueden ser fácilmente incluidos en las restricciones conservando la linealidad del modelo.

Por otro lado, para una escala de tiempo continua, el enfoque de *puntos de tiempos globales* (Figura 2.3-(b)), se presenta como una generalización de la representación de intervalos de tiempo globales donde la duración de los intervalos de tiempo es tratada como una variable del modelo. De esta forma, se construye una grilla de tiempo variable y global para todos los equipos compartidos y cada inicio o finalización de una tarea es asociada a un punto de ésta. De modo similar, la representación de *eventos de tiempo asociados a equipos* (Figura 2.3-(c)) considera una grilla de tiempo variable única para cada recurso compartido. La eficiencia computacional de estas dos formulaciones depende del número de puntos de tiempo o eventos definidos inicialmente en el modelo. Establecer menos puntos en la grilla da como resultado un número menor de variables binarias, pero puede conducir a una solución subóptima o infactible; en cambio, un número mayor al conveniente deriva en un esfuerzo computacional innecesario.

Por su parte, el enfoque de *ranuras de tiempo o "time-slots"* (Figura 2.3-(d)) busca representar los eventos con un número predefinido de ranuras de tiempo para cada equipo, las cuales tienen una duración (en principio) desconocida, a determinar mediante las decisiones del modelo. En este método, una de las principales dificultades es postular la cantidad de ranuras de tiempo para cada equipo de procesamiento dado que es necesario realizar un balance entre la optimalidad de la solución y el rendimiento computacional obtenido.

Por último, como representación de tiempo continuo, también son utilizadas las formulaciones con base en las nociones de *precedencia inmediata* y *general* de los lotes (Figura 2.3-(e)). En el primer caso, el predecesor inmediato de un *"batch" i* es el *"batch"* i' procesado inmediatamente antes en el mismo equipo, el cual es identificado mediante una variable de decisión. En cambio, el concepto de precedencia general considera no solamente al predecesor inmediato sino también a todos los lotes que utilizan, antes en el tiempo, el mismo recurso compartido. Ambas formas de ordenar los eventos se diferencian

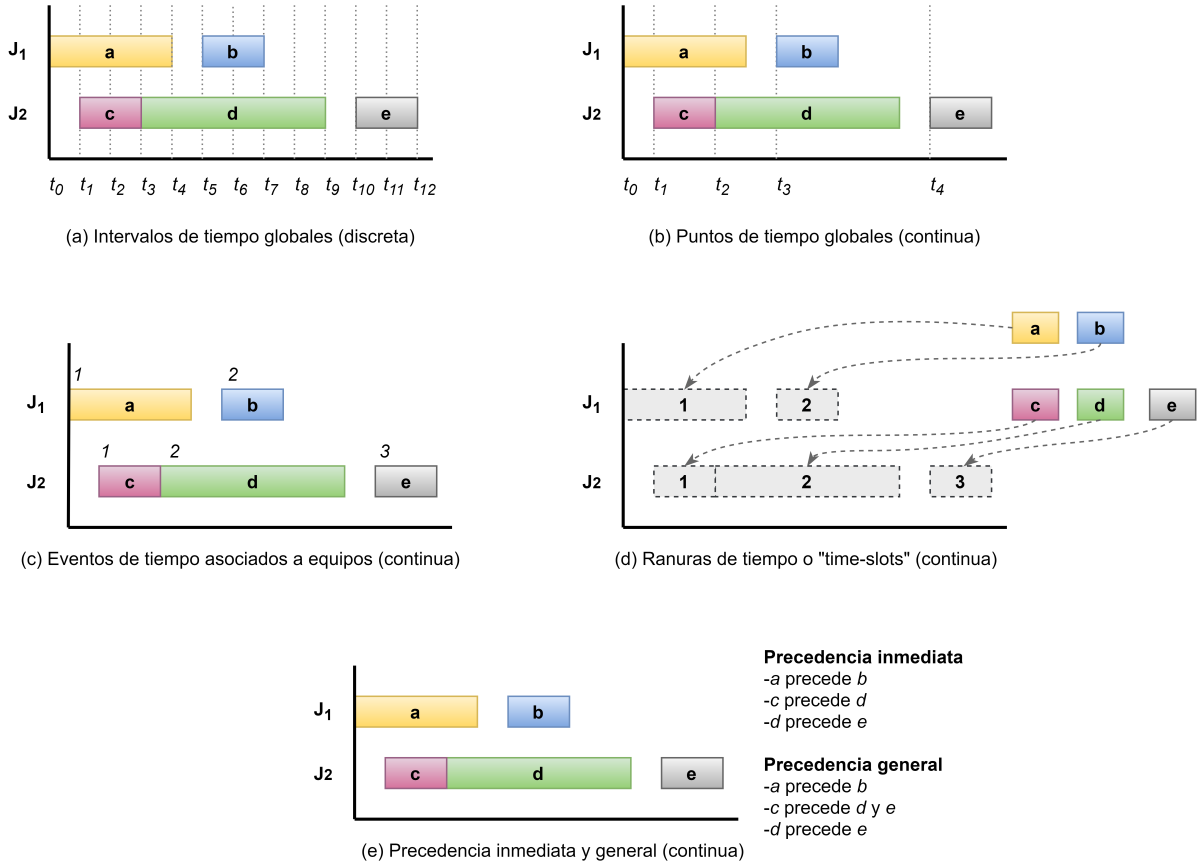

Figura 2.3: Representación de eventos

Fuente: Méndez y col. (2006)

principalmente en que el concepto de precedencia general utiliza una única variable de secuenciación para cada par de tareas asignadas a un mismo equipo, lo cual resulta en un modelo matemático de menor tamaño.

2.5.3. Balance de materiales

Otro aspecto a definir al generar un modelo matemático para un problema de *"scheduling"* es el llamado balance de los materiales. A este se le puede dar un tratamiento monolítico, que resuelve en simultáneo el número y tamaño de los lotes a procesar, su asignación y secuenciamiento; o tratar el problema con un enfoque secuencial, que consiste en dividir el problema general en dos etapas; la primera de dimensionamiento de lotes, y la segunda destinada a tomar las decisiones de asignación y secuenciación asumiendo el tamaño de los lotes predefinido.

En el caso del enfoque secuencial, el dimensionamiento de lotes consiste en asociar las órdenes de producción de cada producto a un conjunto de *"batches"* a ser procesados y, de esta forma, se determina para cada producto el número y tamaño de los lotes a desarrollar. A continuación, para construir el *"schedule"*, se debe resolver el problema de asignación de dichos lotes a los recursos compartidos y definir su secuenciación, cumpliendo con todas las restricciones del problema y optimizando el criterio elegido.

Una formulación monolítica es de utilidad cuando deben resolverse ambas etapas, tanto el dimensionamiento, como la secuenciación y asignación de los *"batches"*, debido a que no es posible saber de antemano la cantidad y tamaño de lotes más conveniente. En general, al utilizar un enfoque secuencial es posible resolver problemas de *"scheduling"* de mayor tamaño, pero de menor complejidad en la receta de procesamiento con relación a aquéllos que pueden ser resueltos con métodos monolíticos.

2.6. Dificultad de Resolución Computacional

Partiendo de la base que un problema computacional constituye una pregunta a ser respondida, las soluciones al mismo se representan generalmente como parámetros, o variables libres, cuyos valores son los que se desea encontrar. Ahora bien, un problema de decisión es un tipo especial de problema cuyas respuestas valen solamente "sí" o "no". Si existe solución al problema, la respuesta a su pregunta es "sí" y en caso contrario "no". Estos problemas tienen gran importancia ya que, en la práctica, casi todos los problemas que se pueden resolver con una computadora pueden transformarse en un problema de decisión.

La programación de operaciones a corto plazo es un problema de optimización, es decir, que incluye la maximización o minimización de una función objetivo. Una solución factible al problema será una asignación de valores a todas las variables de decisión que, a su vez, satisface las restricciones. En cambio, una solución óptima será la programación de todas las tareas, cumpliendo las restricciones y, logrando un valor óptimo para la función objetivo. En este caso, es posible reformular el problema de optimización aplicando un límite al valor a optimizar y, transformándolo así, en un problema de decisión. A modo de ejemplo, se puede considerar que existe un conjunto de n ciudades y otro de m caminos que las conectan, con sus costos cm asociados. Mientras el problema de optimización busca determinar el recorrido que minimiza la función objetivo del costo total CT, el problema de decisión, considera un número no negativo k y se pregunta: ¿existe un recorrido con costo

total CT menor o igual a k? Utilizando esta estrategia puede resolverse cualquier problema de optimización como una secuencia de problemas de decisión de menor complejidad.

Para encontrar respuestas con recursos computacionales a los problemas de decisión se utilizan algoritmos, es decir, procedimientos paso-a-paso para la búsqueda de soluciones. En particular, la Teoría de la Complejidad Computacional estudia la "dificultad" de resolución de los problemas, y considera que los recursos necesarios para resolverlos computacionalmente más comúnmente estudiados son: (i) el tiempo, mediante una aproximación al número de pasos de ejecución que un algoritmo emplea para resolver un problema y (ii) el espacio, mediante una aproximación a la cantidad de memoria utilizada para dicha tarea. Usualmente, de ambos recursos el tiempo es el factor dominante cuando se trata de determinar si un algoritmo es lo suficientemente eficiente para ser utilizado en la práctica, por ello, se hará foco mayormente en éste.

De esta forma, de acuerdo al libro de Garey y Johnson D. S. (1979), la Teoría de la Complejidad Computacional define a P como el conjunto de problemas para los cuales se conoce un algoritmo que permite su resolución en tiempo polinomial. Esto quiere decir que el tiempo de resolución necesario para determinado problema crece, en el peor de los casos, de forma polinomial en relación al tamaño del mismo. Por otro lado, la clase NP incluye los problemas para los cuales una solución puede verificarse en tiempo polinomial. Por lo tanto, los problemas de tipo P son un subconjunto de los problemas NP. Sin embargo, estos últimos pueden ser resueltos con algoritmos en tiempos polinomiales solo en máquinas no deterministas que, básicamente, son máquinas "teóricas" que pueden explorar infinitos caminos de computación en paralelo. En el caso de las máquinas deterministas, que son las computadoras que se conocen, el tiempo de resolución de los mejores algoritmos disponibles para los problemas NP crece, al menos, de forma exponencial con el tamaño del problema. Es importante resaltar que es una incógnita aún abierta en la matemática si existe la posibilidad de desarrollar un algoritmo para resolver los problemas NP en tiempo polinomial, lo que deja sin responder la posibilidad de que se cumpla que $P = NP$.

Dado que son problemas de optimización, los problemas de *"scheduling"* junto con los problemas de programación MILP se clasifican, incluso en los casos más simples, dentro del conjunto de problemas llamados *NP-Hard* (Garey & Johnson D. S., 1979). Para estos problemas puede obtenerse un método de resolución de tiempo polinomial que dependa de la solución de los problemas de decisión asociados. Sin embargo, los problemas de decisión mencionados pertenecen a la clase NP (más específicamente, a una clase denominada *NP-Completo*. Los problemas pertenecientes a esta clase son considerados

computacionalmente intratables y, aunque se han desarrollado innumerables propuestas para su tratamiento sistemático, no existen métodos de solución eficiente para los mismos.

2.7. Abordaje de Problemas de *"Scheduling"* en la Industria de los Procesos

2.7.1. Diagnóstico de las herramientas actuales

Desde los primeros trabajos en el área en la década del '50 fueron estudiadas diversas variaciones del problema general de *"scheduling"* usando diferentes formulaciones matemáticas para el tiempo, así como múltiples restricciones y funciones objetivo. Fuchigami y Rangel (2018) realizaron una compilación de los diversos casos estudiados y sus principales características. Sin embargo, a raíz de su complejidad, debida principalmente a la naturaleza combinatoria del problema, existe aún una notable brecha entre la teoría y su aplicación práctica para la resolución de problemas de escala industrial. Luego de décadas de avances en técnicas de resolución y estudios que consideran entornos teóricos a los problemas planteados, Harjunkoski y col. (2014), sostienen que existe potencial de mejora en la aplicación de las herramientas de optimización actuales para aumentar la eficiencia de la producción en casos de estudio reales. En la actualidad es posible desarrollar herramientas novedosas o combinar técnicas disponibles, generando así nuevas estrategias de resolución del problema.

En el abordaje de la programación de operaciones existen, por un lado, las herramientas exactas (ver Apartado 2.5), las cuales están basadas en modelos matemáticos que exploran el árbol de soluciones de manera ordenada, manteniendo la información del intervalo de integralidad de la solución encontrada. Esto quiere decir que conservan rigurosamente las cotas inferior y superior de la solución dando una medida de cuán "bueno" o "malo" es el resultado obtenido. Además, este tipo de herramientas otorgan información útil para evaluar cuantitativamente su comportamiento y, en definitiva, mayor confiabilidad que aquellas herramientas que no otorgan esta información. Entre algunos datos generalmente disponibles se pueden mencionar la cantidad de nodos del árbol de búsqueda explorados y aquellos abiertos que faltan explorar y, la brecha existente entre la mejor solución posible y aquella con mejor valor de la función objetivo. Por otro lado, los métodos aproximados, realizan la exploración de las soluciones sin garantizar que la mejor entre todas las halladas sea óptima. Sin embargo, son muy utilizados en los casos en que las

técnicas exactas requieren demasiados recursos computacionales, si la situación planteada no requiere soluciones de gran precisión o cuando las limitaciones de tiempo, espacio, entre otros, conducen a desarrollar métodos de respuesta rápida sacrificando información acerca de la optimalidad de la solución.

Las limitaciones de memoria física y velocidad de procesamiento de datos, en problemas de gran tamaño, obligan a programar límites para el cese de la exploración de soluciones, como cotas de tiempo, número de iteraciones, límites para el valor de la solución encontrada, entre otros. Si bien los métodos aproximados muchas veces obtienen buenas soluciones, no aportan ninguna información sobre la calidad de las mismas. En cambio, los métodos exactos brindan una calificación precisa de la solución obtenida, pero obtienen soluciones de poca calidad para problemas de gran escala. El buen rendimiento de la herramienta propuesta dependerá, en general, de que la calidad de solución hallada y el tiempo necesario para su reporte cumplan con las expectativas.

Recientemente Castro y col. (2018) exponen una reseña orientada a expandir el alcance y los desafíos computacionales de los problemas de programación de operaciones. En la misma discuten las principales dificultades actuales que presenta la integración de modelos temporales para los niveles de planificación, programación y control de procesos en una empresa. Además, partiendo de una descripción general del campo de la optimización, los autores aportan una revisión de nuevos paradigmas para la resolución de problemas de *"scheduling"*, como la Programación Disyuntiva Generalizada (GDP), y amplían los casos innovadores de aplicación resaltando posibles líneas de interés futuro. Entre las aplicaciones que reseñan se pueden mencionar la programación de operaciones simultáneas e integración de calor, programación de oleoductos, mezcla de petróleo crudo y productos refinados y optimización robusta que incluye un nivel de incertidumbre en ciertos parámetros del modelo.

2.7.2. Aportes en la literatura que utilizan formulaciones MILP

En los últimos años, diversas técnicas han sido desarrolladas, con el principal objetivo de encontrar la solución óptima con esfuerzo computacional y tiempo razonables. Entre algunas de las publicaciones más recientes relevando las metodologías desarrolladas y casos estudiados hasta el momento se pueden mencionar a Fuchigami y Rangel (2018), Harjunkoski y col. (2014) y Méndez y col. (2006).

En esta sección, se elabora una síntesis sobre las principales metodologías que utilizan formulaciones MILP desarrolladas hasta la fecha para el abordaje del problema de *"scheduling"* de forma predictiva en plantas *"batch"* multiproducto multietapa. Como se ha descripto anteriormente en el Apartado 2.5.3, considerando el balance de los materiales, el problema puede ser tratado de forma monolítica o secuencial. En línea con el caso de estudio de este proyecto se hará foco en este segundo enfoque, el cual asume el dimensionamiento de los lotes ya definido y considera que cada uno de ellos está asociado a un único lote de producto final, por lo que el mismo es procesado de forma individual sin que ocurra una división o mezclado con otro lote.

Por el lado de las formulaciones de tiempo continuo, las propuestas más relevantes emplean representaciones basadas en *ranuras de tiempo o "time slots"* (Chen y col., 2002; Lim & Karimi, 2003; Pinto & Grossmann, 1995), *precedencia inmediata* (Cerdá y col., 1997; S. Gupta & Karimi, 2003) y *precedencia general* (Méndez & Cerdá, 2002; Méndez y col., 2001). Por otra parte, en los últimos años fueron publicados importantes aportes basados en estrategias híbridas que combinan la formulación de ranuras de tiempo con restricciones de secuenciación basadas en relaciones de precedencia (Castro & Grossmann, 2005, 2006; Castro y col., 2012).

El modelo propuesto por Pinto y Grossmann (1995), introduce el concepto de ranuras de tiempo o *"time slots"*, definido como un conjunto de intervalos de tiempo asociadas a un equipo de procesamiento que tienen duración, en principio, desconocida. La formulación considera una escala de tiempo continua y es aplicada a un problema de *"scheduling"* en una instalación *"batch"* multiproducto multietapa con equipos de procesamiento en paralelo. Con el fin de resolver problemas de mayor tamaño, los autores proponen, por un lado, considerar restricciones de pre-ordenamiento de los lotes o, por otro lado, resolver el problema en dos etapas: en primer lugar, tomar las decisiones de asignación y, luego, las decisiones de secuenciación de los lotes. En la misma línea, Pinto y Grossmann (1996), presentan un modelo alternativo en el cual incluyen las restricciones de pre-ordenamiento en la representación subyacente del modelo, en lugar de considerarlas de forma separada. Ambas formulaciones no permiten considerar tiempos de transición dependientes de la secuencia de procesamiento.

Por otra parte, Chen y col. (2002), presentan una formulación considerando la notación de ranuras de tiempo y representando una planta monoetapa con equipos que trabajan en paralelo. Para ello, consideran tiempos de alistamiento y fechas de entrega de las órdenes, tiempos de transición dependientes de la secuencia y tiempos de preparación y alistamiento de los equipos. Asimismo, Lim y Karimi (2003), analizan una planta semi-

continua con líneas de producción en paralelo y recursos compartidos entre los productos. Para esto se definen *"time slots"* asíncronos entre los equipos de procesamiento e incorporan puntos de verificación para asegurar la disponibilidad o no del recurso compartido.

Por otro lado, Cerdá y col. (1997) desarrollaron una formulación basada en la precedencia inmediata, la cual presenta la idea de antecesor y sucesor de una tarea en una planta multiproducto monoetapa con equipos no idénticos en paralelo. Las relaciones de la precedencia inmediata entre las tareas se expresan con variables binarias específicas para cada equipo. Además, para hacer frente a los problemas de mayor tamaño, se informa sobre una estrategia heurística que permite podar parcialmente el conjunto de predecesores factibles para cada lote, reduciendo el tamaño de la representación del problema MILP. Por otra parte, S. Gupta y Karimi (2003), presentan un trabajo en el que resuelven la secuenciación de los *"batches"* con la idea de precedencia inmediata utilizando variables binarias generales, no específicas de cada unidad de procesamiento.

En cuanto a la formulación de precedencia general, la misma fue presentada por Méndez y col. (2001) como estrategia superadora a la de precedencia inmediata e incluye las relaciones de precedencia no solo inmediatamente próximas a la tarea, sino que explicita todas relaciones con las tareas que son ejecutadas previamente en el equipo. En la misma se incluyen restricciones sobre la estructura topológica de la planta y limitaciones de capacidad de recursos discretos. Posteriormente, Méndez y Cerdá (2002), logran reducir notablemente la cantidad de variables binarias y el tiempo de resolución de la formulación tratando los recursos discretos y las unidades de procesamiento de forma uniforme. Mediante las variables de secuenciación, se resuelve el orden en que son asignadas las tareas a un dado equipo o a cualquier otro recurso discreto.

En cambio, si se tiene en cuenta la resolución de problemas de programación de operaciones utilizando una representación de tiempo discreta, en los últimos años se han presentado avances relevantes. Uno de ellos es el aporte de Velez y Maravelias (2013), que pone en escena la posibilidad de encontrar mejores soluciones que las reportadas por los modelos desarrollados previamente bajo estos paradigmas. En su trabajo, se incluyen múltiples grillas de tiempo (posiblemente no comunes) para las tareas, unidades de procesamiento, materiales y demás recursos que se incluyen al modelo. Para determinar las escalas discretas correspondientes se desarrollan dos algoritmos: el primero de ellos determina el intervalo de tiempo más largo que no empeora la solución óptima; el segundo, permite al usuario establecer el nivel de aproximación necesario.

Más tarde, Velez y col. (2015) integran la formulación anteriormente mencionada con otras metodologías desarrolladas, incluyendo restricciones para limitar la producción total y el número de lotes para cada tarea y material en función de la demanda del cliente e incorporando restricciones de límite superior basadas en el inventario y la disponibilidad de recursos en la planta. De esta forma, el trabajo desarrollado es aplicado a problemas de gran escala incluyendo diversas características de procesamiento, políticas de almacenamiento, tiempos de alistamiento y transición de tareas.

Por otra parte, continuando con los aportes basados en una representación discreta del tiempo, Merchan y col. (2016) abordan el problema de *"scheduling"* en plantas *"batch"* multiproducto multietapa. En concreto, se presentan dos nuevos modelos inspirados en el problema de programación de operaciones con restricciones de recursos (*"Resource-Constrained Project Scheduling Problem"- RCPSP*). En esta instancia, se tienen en cuenta escenarios en los que la fuerza laboral empleada para realizar las tareas es limitada y cada trabajo tiene hora de llegada, fecha de entrega y penalizaciones asociadas a su retraso. A su vez, se incluye un algoritmo para aproximar soluciones en problemas de gran escala.

Por su parte, recientemente H. Lee y Maravelias (2018) proponen un método para abordar problemas de *"scheduling"* en entornos de red, donde los lotes se pueden mezclar para formar la entrada de otro lote o varios lotes pueden consumir la salida de otro lote. El método propuesto consta de tres etapas: en la primera, se resuelve un modelo MILP de tiempo discreto para obtener una solución aproximada; en la segunda etapa, se introducen grillas tiempo continuo específicas en los materiales y unidades de procesamiento, utilizando un algoritmo de mapeo; y, en la tercera, se resuelve un modelo de programación lineal de tiempo continuo para mejorar la precisión de la solución de tiempo discreto. El método propuesto aprovecha las fortalezas complementarias de las formulaciones de tiempo discreto y continuo y permite obtener aceleraciones de orden de magnitud en la solución de instancias de gran escala.

2.7.3. Contribuciones relevantes que incluyen metodologías aproximadas

Tal como se ha desarrollado previamente en el Apartado 2.4.2, entre las principales metodologías aproximadas aplicadas para resolver problemas de similares características al abordado en este proyecto se pueden mencionar: la Programación por Restricciones, Metaheurísticas y Reglas de Prioridad de Despacho. Seguidamente se presentan importantes contribuciones abordadas en la bibliografía del área.

En cuanto a los modelos de Programación por Restricciones (CP), se han desarrollado una serie de construcciones y restricciones globales para modelar y resolver problemas específicos de manera eficiente, y las restricciones no necesitan ser lineales ni convexas. Los métodos de CP han demostrado ser bastante efectivos para resolver ciertos tipos de problemas de programación, particularmente aquellos que implican secuenciación y limitaciones de recursos. Recientemente, se han desarrollado enfoques novedosos como Malapert y col. (2012), Novara y col. (2013) y Zeballos y col. (2011). Sin embargo, no siempre es eficaz para resolver una programación óptima más general en problemas que involucran asignaciones de recursos. Por lo tanto, el uso de CP en combinación con técnicas MILP ha recibido atención, logrando una buena relación complementaria. En este sentido, se han reportado avances con ahorros computacionales significativos por Harjunkoski y Grossmann (2002), Jain y Grossmann (2001) y Maravelias y Grossmann (2004).

Siguiendo esta dirección, existen trabajos reportados en la literatura que incluyen metodologías Metaheurísticas como los trabajos de Graells y col. (1998), Y. G. Lee y Malone (2000) y Ryu y col. (2001), usando *"Simulated Annealing"*, He y Hui (2007), Löhl y col. (1998) y Wongthatsanekorn y Phruksaphanrat (2015), empleando Algoritmos Genéticos, y Cavin y col. (2004), aplicando *"Tabu Search"*. Las Metaheurísticas son consideradas como heurísticas de mejora, dado que emplean un procedimiento iterativo el cual comienza con una solución inicial que se mejora gradualmente.

Por otro lado, las Reglas de Prioridad de Despacho son llamadas heurísticas de construcción, ya que usan criterios empíricos para dar un orden de prioridad a los lotes que esperan ser procesados en un equipo. Blackstone y col. (1982), han elaborado un resumen y clasificación de variadas Reglas de Prioridad de Despacho existentes.

2.7.4. Desarrollos importantes incorporando técnicas de descomposición del problema

Las soluciones óptimas en problemas de programación de operaciones de tamaño industrial, en general, no pueden obtenerse con recursos computacionales y tiempo razonables. Dada la importancia del rendimiento de la herramienta para resolver los problemas de gran escala, se han desarrollado avances en técnicas de descomposición del problema. Dentro de las metodologías de descomposición más estudiadas se tiene aquellas basadas en el tiempo, en los recursos críticos y en los lotes u órdenes de fabricación.

En cuanto a una forma de descomposición basada en el tiempo, Bassett y col. (1996), presentaron un enfoque heurístico para lograr soluciones viables al problema general de programación. El método, inicialmente, divide el horizonte del problema en intervalos agregados y otorga una solución que asigna cantidades de producción dentro de cada intervalo. Seguidamente, se generan subproblemas de programación para cada intervalo y se resuelven de forma independiente llegando al nivel de mayor desagregación temporal necesaria.

Por el lado de las metodologías que consideran la asignación prioritaria de los recursos cuello de botella, Balas y col. (1995), propusieron un algoritmo que secuencia las tareas resolviendo problemas de un solo recurso, manteniendo fijas las secuencias previamente determinadas en otras instancias e ignorando las restricciones sobre los recursos aún no programados. Más recientemente, Verbiest y col. (2019) abordan el problema de diseño y programación de una planta *"batch"* con líneas paralelas. Además de un enfoque integral del problema, aplican tres estrategias de descomposición para abordar la complejidad computacional asociada: descomposición de líneas, descomposición de los recursos cuellos de botella y una combinación de ambos. Es así como logran reducir significativamente los tiempos de cálculo en comparación con el enfoque integral, mientras obtienen soluciones casi óptimas.

En cuanto a estrategias basadas en la descomposición del problema considerando un *"batch"* a la vez o un subconjunto del total de estos, Castro y col. (2009) han presentado un avance aplicado en un caso de gran escala. A medida que el algoritmo avanza en las iteraciones, los lotes previamente programados pueden ser parcialmente reprogramados para permitir cierta flexibilidad mientras se mantiene la complejidad combinatoria en un nivel manejable. Una vez que se obtiene el *"schedule"* completo, se aplica el mismo concepto para mejorarlo localmente. En un camino similar, Kopanos y col. (2010) presentan una estrategia de solución iterativa sistemática eficiente para resolver problemas de programación de operaciones del mundo real en plantas *"batch"* multiproducto multietapa. Este tipo de enfoques se ensambla bien en un esquema de horizonte en movimiento, ya que el proceso de construcción considera los *"batches"* dentro de la ventana temporal de programación y el mejoramiento local se puede ejecutar hasta que se deba anexar el nuevo subconjunto de tareas a programar.

Capítulo 3

FORMULACIÓN MATEMÁTICA DEL PROBLEMA

3.1. Presentación del Caso de Estudio

En este proyecto se aborda un problema real de programación de operaciones a corto plazo y gran escala perteneciente a la industria farmacéutica. Este caso de estudio, tomado de la literatura del área de investigación, fue estudiado previamente por Castro y col. (2009) y Kopanos y col. (2010). El mismo trata de una planta industrial discontinua, es decir de procesamiento por lotes, multiproducto multietapa, la cual cuenta con 17 equipos disímiles que trabajan en paralelo distribuidos en 6 etapas de procesamiento. Las etapas de la instalación, integradas por los equipos J, se encuentran dispuestas como se indica en la Figura 3.1 y ordenadas de acuerdo a las recetas de procesamiento de los productos. Como puede observarse, cada equipo está conectado con todos los equipos de la etapa anterior o posterior, lo cual indica que no existen restricciones topológicas en las conexiones disponibles entre equipos pertenecientes a etapas de procesamiento consecutivas. Estas interconexiones se dan acorde a una política de transferencia de los *"batches"* basada en almacenamiento intermedio ilimitado (UIS), es decir que la existencia de capacidad en los tanques se asume suficiente en todos los casos.

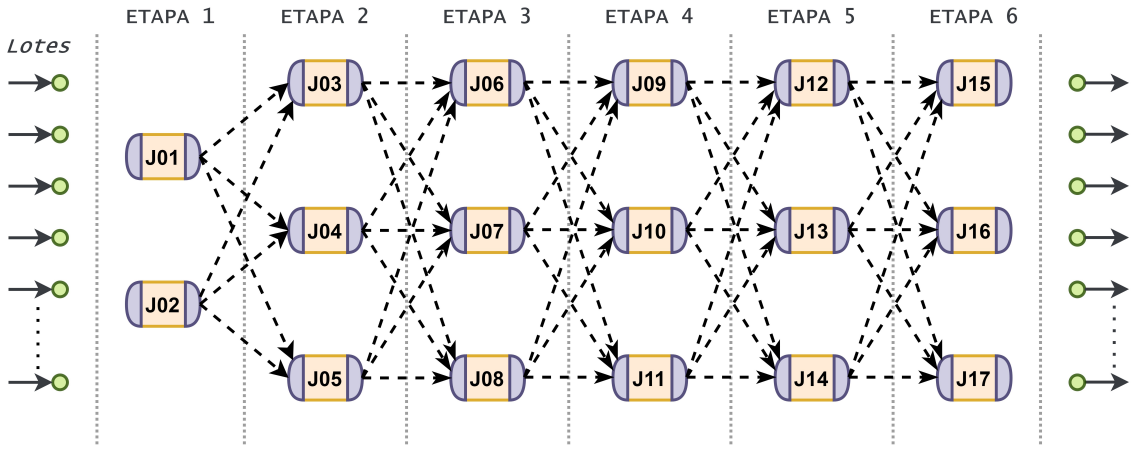

Figura 3.1: Planta *"batch"* multiproducto multietapa - Caso real de la industria farmacéutica

Fuente: Kopanos y col. (2010)

En la planta industrial deben ser procesados un total de 30 *"batches"*, los cuales están asociados a diferentes productos cuyas recetas de elaboración poseen una secuencia de operaciones similar. Cada lote se caracteriza por sus tiempos de procesamiento requeridos en los equipos de cada etapa y los tiempos de transición (*"changeover"*) dependientes de la secuencia que, en algunos casos del problema estudiado, son incluso mayores que los tiempos de elaboración de los productos en cada etapa. Esto último es un factor que incrementa notablemente la dificultad computacional de resolución del problema.

A continuación, en las Tablas 3.1 y 3.2 se expone un fragmento de los datos disponibles para los tiempos de procesamiento y tiempos de transición, respectivamente, para el caso de estudio de este proyecto. Esta información es tomada del material suplementario de Kopanos y col. (2010) y se muestra de forma completa en los Anexos B y A.

Con respecto a los tiempos de procesamiento, en el caso la Etapa 01 los mismos son similares para todos los lotes, indistintamente del equipo al cual sean asignados. Para las etapas siguientes, en cambio, el tiempo requerido para procesar cada lote es propio del lote en sí y del equipo asignado. En este sentido, la Tabla 3.1 mencionada incluye un recorte con la información del tiempo necesario para procesar las primeras cinco órdenes en las Etapas 01 a 03 de la instalación. Por ejemplo, para el caso del lote $P02$ en la Etapa 02, el tiempo de procesamiento es el mismo para los dos equipos en los que puede ser procesado ($J03$ y $J04$) y no se especifica para el equipo $J05$, ya que dicho equipo no puede elaborar el producto intermedio asociado. Nótese además que algunos *"batches"* no necesitan ser procesados en la Etapa 03 con lo que, en estos casos, la misma es pasada

por alto. Esta situación puede observarse en el lote $P04$, el cual saltea la Etapa 03, por lo que los tiempos de procesamiento en sus equipos ($J06$, $J07$ y $J08$) no se indican. La información completa de los tiempos de procesamiento, que en el siguiente Apartado serán representados formalmente con el parámetro $pt_{i,j}$, se encuentra disponible en el Anexo B.

	Etapa 01		**Etapa 02**			**Etapa 03**		
	J01	**J02**	**J03**	**J04**	**J05**	**J06**	**J07**	**J08**
P01	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P02	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P03	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P04	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P05	0,9000	0,9000	1,3050	1,3050	-	-	-	-

Tabla completa en Anexo B

Tabla 3.1: Fragmento de los tiempos de procesamiento $pt_{i,j}$ [hrs]

Fuente: Material suplementario de Kopanos y col. (2010)

	P01	**P08**	**P12**	**P13**	**P16**	**P17**	**P20**	**P21**	**P23**	**P24**	**P26**	**P27**
P01	1,35	1,80	1,80	1,35	1,80	1,35	1,35	1,80	1,80	1,35	1,80	1,80
P08	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80
P12	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80
P13	1,35	1,80	1,80	1,35	1,80	1,35	1,35	1,80	1,80	1,35	1,80	1,80
P16	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80

Tabla completa en Anexo A.3

Tabla 3.2: Fragmento de los tiempos de transición $\tau_{i',i,j}$ [hrs] para la Etapa 03

Fuente: Material suplementario de Kopanos y col. (2010)

En cuanto a los tiempos de transición, los mismos valen cero en la Etapa 01 y, en la Etapa 02, toman un valor de 0,45 [hrs] independientemente de la secuencia asignada a cada equipo. Para el resto de las etapas, el tiempo de limpieza y preparación de un equipo para procesar determinado lote estará asociado al *"batch"* que haya sido elaborado previamente. En la Tabla 3.2 se exponen algunos tiempos de transición, que serán representados luego por el parámetro $\tau_{i',i,j}$, entre dos lotes (i', i) procesados uno a continuación del otro en un mismo equipo j de la Etapa 03. Tomando a modo de ejemplo un caso en que la orden $i = P08$ (ubicada en las columnas de la tabla) sea procesada en el mismo equipo y de forma inmediatamente posterior al lote $i' = P13$ (situado en las filas de la tabla), el tiempo de *"changeover"* entre ambos *"batches"* corresponde a 1,80 [hrs]. Nótese que el tiempo de transición depende únicamente de la secuencia de lotes, y no del equipo al cual

fueron asignados los mismos. La información completa de los tiempos de transición, para todas las etapas, se encuentra disponible en el Anexo A.

3.2. Definición del Problema e Hipótesis

El problema de *"scheduling"* bajo estudio puede ser definido formalmente considerando que inicialmente se tiene:

(I) una planta *"batch"* multiproducto multietapa integrada por equipos $j \in J_l$, los cuales operan en paralelo en cada etapa de procesamiento $l \in L$,

(II) el conjunto de *"batches"* $i \in I$ que deben ser procesados,

(III) la secuencia de etapas de procesamiento $l \in L_i$ requeridas por cada *"batch"* i,

(IV) el subconjunto de equipos $j \in J_{i,l} \subseteq J_l$ en los cuales puede ser procesada cada tarea $(i,\, l)$,

(V) el tiempo de procesamiento $pt_{i,j}$ de cada *"batch"* $i \in I$ en el equipo $j \in J_l$,

(VI) los tiempos de transición dependientes de la secuencia $\tau_{i',i,j}$ para cada *"batch"* $i' \in I$ que se procesa inmediatamente antes al *"batch"* $i \in I$ en el equipo $j \in J_l$.

A su vez, se tienen en cuenta las siguientes consideraciones:

- Los equipos $j \in J$ pertenecientes a etapas de procesamiento $l \in L$ consecutivas están conectados entre sí sin restricciones topológicas.
- Entre las etapas de procesamiento consecutivas existen tanques de almacenamiento con capacidad suficiente para contener los distintos *"batches"*.

Por otra parte, el objetivo que se platea en este estudio es encontrar una agenda de producción o *"schedule"* factible que satisfaga todas las restricciones del problema y, a su vez, minimice el tiempo total necesario para la completar todas las tareas o *"makespan"*.

3.3. Formulación Matemática Propuesta Utilizando Ranuras de Tiempo

Para modelar el problema estudiado se utiliza una nueva versión de la formulación de tiempo continuo basada en ranuras de tiempo o *"time-slots"* desarrollada por Pinto y Grossmann (1995). Mediante la utilización de Programación Mixta-Entera Lineal, esta formulación define ranuras de tiempo, las cuales representan intervalos de longitud variable y alojan las tareas a procesar en un dado equipo. Se debe tener en cuenta que, en esta formulación base, se procesa a lo sumo un producto en cada *"slot"* de un determinado equipo. Además, la longitud de una ranura se vuelve nula si no se le asigna ninguna tarea.

En las formulaciones basadas en ranuras de tiempo se propone inicialmente un conjunto de *"slots"* K_j para cada equipo de procesamiento j. Si se considera el proceso de construcción de una solución factible, cada tarea (i, l) deberá ser asociada a un *"slot"* $k \in K_j$ específico, el cual pertenece al equipo j de la etapa l y se encuentra disponible, es decir, no han sido asignadas otras tareas a dicha ranura previamente. Sin embargo, no necesariamente todos los *"slots"* propuestos para cada equipo son seleccionados para realizar una tarea lo que, consecuentemente, implica un margen entre la cantidad de ranuras propuestas y aquellas que efectivamente son utilizadas. Si bien proponer un número de ranuras suficiente en cada equipo es crítico para analizar la mayor cantidad de posibilidades sin recortar ninguna solución factible de buena calidad, proponer demasiadas suele resultar en una formulación extensa e impactar negativamente en el tiempo computacional requerido para hallar y probar una solución óptima. Por lo tanto, es necesario hacer un balance entre la cantidad de *"slots"* propuestos y el tiempo y recursos computacionales necesarios para encontrar una agenda de producción de buena calidad.

En este trabajo, se introduce el concepto de *"slot"* multivaluado que, en contraste con la ranura convencional a la cual puede ser asignada solo una tarea, este puede alojar múltiples *"batches"* de manera simultánea. Este tipo de *"slot"* es admitido solo una vez para cada equipo y no incluye las restricciones de secuenciación, por lo que los tiempos de transición entre estos *"batches"* son una estimación y la solución obtenida puede considerarse inexacta. De esta forma, si bien se resigna cierto grado de precisión y usualmente se producirán soluciones infactibles, las ranuras multivaluadas (como se mostrará posteriormente) son útiles para construir un modelo "aproximado" que permita obtener propuestas de configuraciones de *"slots"* para cada equipo. A su vez, esta propuesta de cantidad de ranuras por equipo y las cotas inferiores obtenidas de la solución estimada pueden ser uti-

lizadas para resolver un modelo "exacto" el cual permite obtener una solución completa del problema. Esta idea será desarrollada con mayor profundidad en el Capítulo 4 de este proyecto.

Con el fin de sentar las bases de la formulación matemática propuesta y posterior construcción del modelo se definen las variables binarias y variables continuas positivas que son detalladas a continuación:

- $W_{i,j,k,l}$: variable binaria que representa las decisiones de asignación y secuenciación de las tareas. Si vale 1, significa que el *"batch"* $i \in I$ es asignado a la ranura $k \in K_j$ asociada al equipo $j \in J_l$ que pertenece a la etapa de procesamiento $l \in L_i$.
- $ST^1_{i,l}$: variable continua positiva que simboliza el tiempo en que se comienza a procesar el *"batch"* $i \in I$ en la etapa $l \in L_i$.
- $CT^1_{i,l}$: variable continua positiva que define el tiempo de finalización del procesamiento de la tarea $(i,\ l)$.
- $ST^2_{j,k}$: variable continua positiva que representa el tiempo de inicio del *"slot"* $k \in K_j$ asociado al equipo $j \in J_l$.
- $CT^2_{j,k}$: variable continua positiva utilizada para indicar el tiempo de finalización del *"slot"* $k \in K_j$ perteneciente al equipo $j \in J_l$.
- $CO_{j,k}$: variable continua positiva utilizada como complemento para el cálculo del tiempo de transición dependiente de la secuencia necesario antes de que la tarea asignada al *"slot"* $(j,\ k)$ comience.
- MK: variable continua positiva utilizada para definir el *"makespan"*.

A modo de ilustración, en la Figura 3.2 se presenta un ejemplo de las variables de decisión del modelo basado en *"time-slots"* propuesto. En el mismo se cuenta con 1 etapa de procesamiento y 1 un equipo asociado a la misma, el cual debe procesar 3 lotes en total, por lo que son definidas las ranuras k_1, k_2 y k_3. En cuanto la variable binaria de asignación y secuenciación $W_{i,j,k,l}$, la misma toma el valor de 1 para asociar determinado *"batch"* a un *"slot"* disponible. En cuanto esto ocurre, las variables continuas correspondientes utilizadas para representar los tiempos de comienzo y finalización de los pares lote-etapa ($ST^1_{i,l}$ y $CT^1_{i,l}$) y ranura-equipo ($ST^2_{j,k}$ y $CT^2_{j,k}$) toman el mismo valor. A su vez, entre

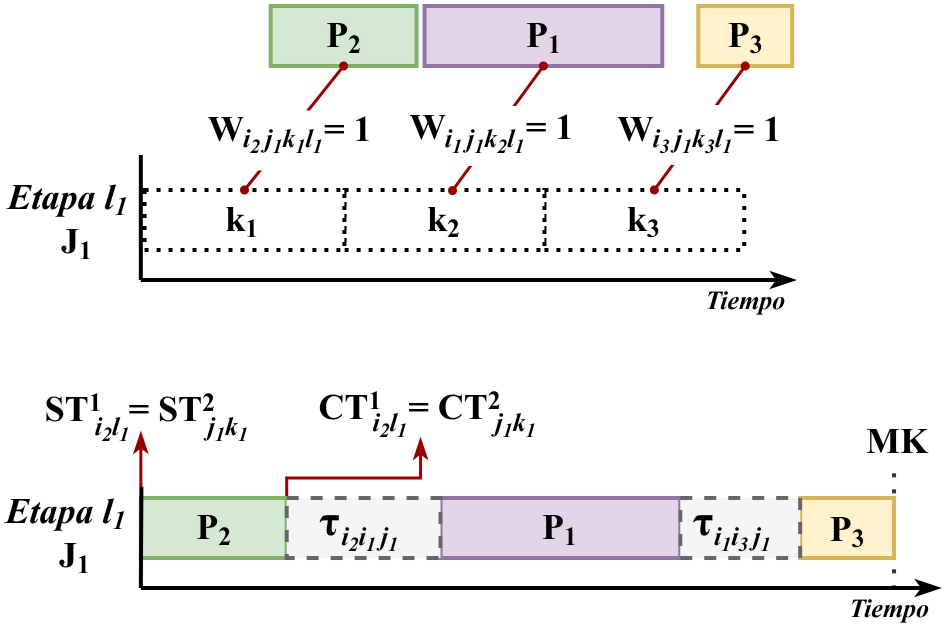

Figura 3.2: Ejemplo - Variables de decisión del modelo

Fuente: Elaboración propia

la finalización de un *"slot"* y el comienzo de la ranura siguiente, se tiene en cuenta el tiempo de *"changeover"* $\tau_{i',i,j}$ que corresponde según la secuencia de procesamiento de los lotes. Luego, una vez finalizado el procesamiento de todas las tareas programadas queda determinado el tiempo total necesario para ello o *"makespan"* (MK).

A continuación se desarrollan las ecuaciones de esta nueva formulación propuesta. Aquí, cabe aclarar que, en las ecuaciones que siguen se utiliza el conjunto L^{P} en lugar del conjunto L. Este conjunto $L^{\mathrm{P}} \subseteq L$ es un subconjunto de las etapas de procesamiento que será introducido en la metodología algorítmica presentada en el Capítulo 4 de este proyecto.

3.3.1. Restricciones de asignación

Las restricciones de asignación sientan las bases de la formulación basada en ranuras de tiempo donde, como se mencionó anteriormente, es propuesto un número fijo de *"slots"* para cada equipo j. Dadas las ranuras $k \in K_j$, la ecuación (3.1) asegura que cada etapa l del *"batch"* i es asignada a un único *"slot"* $k \in K_j$. Dicho *"slot"* debe pertenecer a un equipo $j \in J_{i,l}$ capaz de procesar la tarea (i, l).

$$\sum_{j \in J_{i,l}} \sum_{k \in K_j} W_{i,j,k,l} = 1 \qquad \forall i \in I, l \in L^{\mathrm{P}} \tag{3.1}$$

Para un dado equipo j, no se requiere que sus *"slots"* propuestos y seleccionados sean los mismos. Los *"slots"* propuestos son aquellos disponibles en el modelo para el equipo j. En cambio, aquellos seleccionados en la solución no son necesariamente todos los propuestos para el equipo j. Para obtener una solución factible, la cantidad de ranuras propuestas $m_j = |K_j|$ debe ser elegida en el intervalo $[k_j^{\min}, k_j^{\max}]$, donde:

$$k_j^{\min} = |I_j^1|, \qquad k_j^{\max} = |I_j|, \qquad \forall j \in J \tag{3.2}$$

y

$$I_j^1 = \{i \in I_j : |J_{i,l}| = 1\}, \qquad \forall l \in L, j \in J_l. \tag{3.3}$$

Como se indica en las ecuaciones (3.2) y la (3.3), el valor de $k_j^{\min}$ está dado por la cantidad de *"batches"* $i \in I_j$ que deben ser procesados de forma obligada en un equipo $j \in J_l$ dentro de cierta etapa. Nótese que la condición $|J_{i,l}| = 1$ indica que existe un único equipo al que puede asignarse la tarea (i, l). Por otro lado, $k_j^{\max}$ es la máxima cantidad de tareas (i, l) que admiten ser llevadas a cabo en el equipo $j \in J_l$. Teniendo en cuenta lo anterior y determinado el valor m_j a utilizar, se define al conjunto K_j como:

$$K_j = \{1, \ldots, m_j\}, \qquad \forall j \in J \tag{3.4}$$

"Slots" convencionales

En este trabajo, se define como *"slot"* convencional a aquellos en los que solo puede asignarse un *"batch"*, correspondiéndose con la propuesta original de la formulación de ranuras de tiempo. A continuación, se presenta la restricción (3.5) para asegurar que esta condición se cumpla en cada *"slot"* convencional. Los mismos se seleccionan de forma consecutiva (según el orden de numeración con el cual se identifican) mediante la Ecuación (3.6). Nótese que estas ecuaciones son aplicadas para todas las ranuras convencionales k, excluyendo a aquella denotada como k_j^*. Esta ranura tiene la característica de ser multivaluada y será introducida en la subsección siguiente en este mismo Apartado. A

su vez, vale aclarar que en estas ecuaciones, es excluido el subconjunto de etapas L^{F}. Precisamente, este incluye la etapa analizada en la Iteración Interna durante el subproblema exacto, conceptos que serán desarrollados en el Capítulo 4 de este proyecto.

$$\sum_{i \in I_j} W_{i,j,k,l} \leq 1 \qquad \forall l \in L^{\mathrm{P}} \setminus L^{\mathrm{F}}, j \in J_l, k \in K_j : (k \neq k_j^*) \tag{3.5}$$

$$\sum_{i \in I_j} W_{i,j,k,l} \leq \sum_{i \in I_j} W_{i,j,(k-1),l} \qquad \forall l \in L^{\mathrm{P}} \setminus L^{\mathrm{F}}, j \in J_l, k \in K_j : (k \neq k_j^*) \tag{3.6}$$

"Slots" multivaluados adicionales

En esta instancia se introduce el concepto de *"slot"* multivaluado, al cual pueden ser asociados múltiples tareas (i, l) de forma simultánea. Si bien este tipo de ranura puede alojar varios *"batches"*, no incluye las restricciones de secuenciación apropiadas requeridas entre ellos. Si se asignan dos o más lotes a una ranura multivaluada, dado que no se evita su superposición, el *"schedule"* asociado será (casi siempre) infactible. En otras palabras, las soluciones en las que a una ranura multivaluada se le asigna más de un *"batch"* no están "completas". Sin embargo, estas soluciones son "parciales" y, como tales, brindan información sobre los límites inferiores de la solución óptima (para problemas de minimización).

La inclusión de *"slots"* multivaluados será limitada por las siguientes condiciones:

(a) solo un *"slot"* de este tipo puede ser propuesto para cada equipo j,

(b) este *"slot"* debe ser el último elemento del conjunto K_j,

(c) no puede ser seleccionado antes de que todos los *"slots"* convencionales del equipo j sean también seleccionados.

La ranura multivaluada del equipo j, si existe, será denotada como k_j^*. Si este *"slot"* es definido, entonces pertenece al conjunto K_j, o de otra manera, el mismo está ausente ($k_j^* = none$). La ecuación (3.7) garantiza que cada *"slot"* multivaluado k_j^* no aloje ningún *"batch"* i hasta que el último *"slot"* convencional es seleccionado.

$$W_{i,j,k,l} \leq \sum_{i' \in I_j} W_{i',j,(k-1),l} \qquad \forall l \in L^{\mathrm{P}}, j \in J_l, i \in I_j, k \in K_j : (k = k_j^*) \wedge (k > 1) \tag{3.7}$$

3.3.2. Restricciones de procesamiento y sincronización de tiempos

Como fue desarrollado anteriormente en esta sección, las formulaciones basadas en *"time-slots"* incluyen diferentes conjuntos de variables continuas no negativas, asociadas a cada evento *"batch"*-etapa (i, l) o equipo-*"slot"* (j, k). Por un lado, el tiempo de comienzo y finalización del *"batch"* i en cada etapa l es representado por las variables $ST^1_{i,l}$ y $CT^1_{i,l}$, respectivamente. Por otra parte, las variables $ST^2_{j,k}$ y $CT^2_{j,k}$ reflejan el mismo evento pero ahora para cada *"slot"* k del equipo j. Cabe aclarar que, en el caso de éstas últimas, solo las ranuras convencionales son consideradas. Por otra parte, los *"slots"* multivaluados k_j^* no tienen definidos sus tiempos de comienzo y finalización, ya que se incurre en una solución parcial (como fue descrita anteriormente).

Seguidamente son presentadas las restricciones que asocian los tiempos de comienzo y finalización del procesamiento de las tareas. Las ecuaciones (3.8) y (3.9) calculan los tiempos de finalización ($CT^1_{i,l}$ y $CT^2_{j,k}$, respectivamente) añadiendo el tiempo de procesamiento $pt_{i,j}$ al correspondiente tiempo de comienzo ($ST^1_{i,l}$ y $ST^2_{j,k}$). En la ecuación (3.8), dada la ecuación (3.1), un único tiempo de procesamiento es seleccionado para cada tarea. Nótese que el conjunto K_j considerado para la suma en el término a la derecha de la ecuación (3.8) también incluye al *"slot"* multivaluado, si éste ha sido previamente definido. En la ecuación (3.9), si el *"slot"* k es opcional y no es seleccionado, entonces resulta la igualdad $CT^2_{j,k} = ST^2_{j,k}$. Además, esta ecuación no es definida para el *"slot"* multivaluado k_j^*.

$$CT^1_{i,l} = ST^1_{i,l} + \sum_{j \in J_{i,l}} \sum_{k \in K_j} pt_{i,j}\, W_{i,j,k,l} \qquad \forall i \in I, l \in L_i \tag{3.8}$$

$$CT^2_{j,k} = ST^2_{j,k} + \sum_{i \in I_j} pt_{i,j}\, W_{i,j,k,l} \qquad \forall l \in L, j \in J_l, k \in K_j : (k \neq k_j^*) \tag{3.9}$$

Por otra parte, basados en las decisiones de asignación, restricciones de sincronización de tiempos son requeridas para igualar los tiempos de inicio *"batch"*-etapa y equipo-*"slot"* ($ST^1_{i,l}$ y $ST^2_{j,k}$). Las ecuaciones (3.10) y (3.11) son del tipo Big-M y se vuelven activas solo cuando la tarea $(i,\ l)$ es asignada a la ranura $(j,\ k)$, es decir cuando se verifica la condición $W_{i,j,k,l} = 1$:

$$ST^1_{i,l} - ST^2_{j,k} \geq -M\,(1 - W_{i,j,k,l}) \qquad \forall i \in I, l \in L_i, j \in J_l, k \in K_j : (k \neq k^*_j) \tag{3.10}$$

$$ST^1_{i,l} - ST^2_{j,k} \leq M\,(1 - W_{i,j,k,l}) \qquad \forall i \in I, l \in L_i, j \in J_l, k \in K_j : (k \neq k^*_j) \tag{3.11}$$

3.3.3. Restricciones de secuenciación

En primer lugar, acorde a la receta de elaboración de los productos procesados en la planta, la ecuación (3.12) garantiza que las etapas consecutivas para cada *"batch"* i sean procesadas en el orden correcto. El elemento l^{last}_i se refiere a la última etapa de procesamiento de la instalación requerida por el *"batch"* i.

$$CT^1_{i,l} \leq ST^1_{i,(l+1)} \qquad \forall i \in I, l \in L_i : l < l^{\text{last}}_i \tag{3.12}$$

Mientras que la ecuación (3.12) es suficiente para secuenciar las etapas de cada *"batch"*, cuando se consideran los *"slots"* consecutivos de un dado equipo j, las restricciones de secuenciación no son sencillas dado que, en el caso de estudio abordado, se deben tener en cuenta los tiempos de transición dependientes de la secuencia.

En este trabajo se introducen dos esquemas alternativos para la secuenciación temporal de las ranuras del conjunto K_j. Más precisamente, las mismas se acomodan (en el tiempo) hacia adelante, comenzando desde el elemento 1 y ascendiendo, o hacia atrás, comenzando desde la ranura 1 y descendiendo. En otras palabras, teniendo en cuenta la evolución del tiempo en el *"schedule"*, en el primer caso cada *"slot"* k debe ser completado antes de que comience el elemento $k+1$ y, para el segundo caso, cada ranura k debe comenzar después de que se complete el procesamiento de aquella de orden $k+1$. Con el fin de ilustrar ambos esquemas, en la Figura 3.3 se incluye un ejemplo con 2 etapas de

procesamiento secuenciadas de forma diversa. En cada equipo se define una determinada cantidad de ranuras convencionales y se incluye un *"slot"* multivaluado para cada uno de ellos, según las premisas desarrolladas en el Apartado 3.3.1.

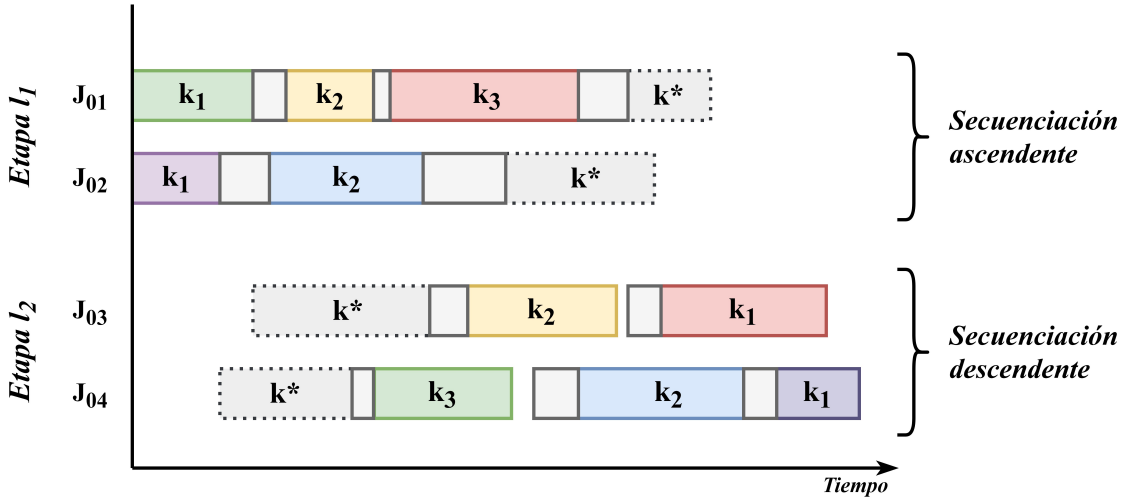

Figura 3.3: Ejemplo - Esquemas de secuenciación temporal de ranuras

Fuente: Elaboración propia

Dada la ecuación (3.5) para la asignación de las ranuras K_j, esta decisión de secuenciar hacia adelante o hacia atrás en el tiempo solamente es relevante si se ha definido una cantidad de *"slots"* convencionales o multivaluados (k_j^*) incluidos en K_j mayor a $k_j^{\min}$. Estos *"slots"* se seleccionan y asignan en orden consecutivo según el índice k, como se indica en las ecuaciones (3.6) y (3.7), pero son ubicados en el tiempo hacia adelante o hacia atrás dependiendo del esquema utilizado. Cuando $m_j = k_j^{\min}$, dado que todos ellos serán seleccionados, ambos esquemas de secuenciación temporal son equivalentes. Puede decirse que, en ese caso, estos esquemas alternativos solo difieren en la forma de ordenamiento de los números que son utilizados para identificar cada elemento del conjunto K_j.

Con el fin de definir cuál de estos criterios de secuenciación temporal se aplicará en cada equipo, se divide el conjunto de etapas L en dos conjuntos disjuntos, L^1 y L^2. Para cada etapa $l \in L^1$, los equipos $j \in J_l$ que la integran secuencian sus ranuras en orden ascendente en el tiempo. En cambio, para cada etapa $l \in L^2$, los equipos $j \in J_l$ acomodan sus ranuras con orden temporal descendente. Usando estos subconjuntos, más adelante se presentan diferentes grupos de restricciones de secuenciación de *"slots"*, diferenciando si éstos son convencionales o multivaluados.

Además de los esquemas de secuenciación presentados, para las ecuaciones que siguen es necesario definir los parámetros $\tau_{\cdot i,j}^{\min}$, $\tau_{i'\cdot,j}^{\min}$, $\tau_{\cdot i,j}^{\max}$ y $\tau_{i'\cdot,j}^{\max}$. Los mismos representan

el mínimo o máximo tiempo de transición que debe tenerse en cuenta antes o después de asignar cierta tarea a una ranura. Entonces, los parámetros quedan expresados como:

$$\tau^{\min}_{\cdot i,j} = min_{i' \in I_j \,:\, i' \neq i}\{\tau_{i',i,j}\}\,, \quad \tau^{\max}_{\cdot i,j} = max_{i' \in I_j \,:\, i' \neq i}\{\tau_{i',i,j}\} \qquad \forall j \in J, i \in I_j \qquad (3.13)$$

$$\tau^{\min}_{i' \cdot,j} = min_{i \in I_j \,:\, i \neq i'}\{\tau_{i',i,j}\}\,, \quad \tau^{\max}_{i' \cdot,j} = max_{i \in I_j \,:\, i \neq i'}\{\tau_{i',i,j}\} \qquad \forall j \in J, i' \in I_j \qquad (3.14)$$

Más adelante, estos parámetros son utilizados junto a la variable positiva $CO_{j,k}$ para determinar el tiempo de transición requerido al procesar el *"batch"* i a continuación del i'. En esta línea, recordando que en las ranuras multivaluadas k^*_j no hay precisiones acerca del orden de procesamiento entre sus tareas asignadas, $CO_{j,k}$ es definida solamente para las ranuras convencionales.

Ahora bien, a fin de establecer la convención a utilizar, tanto los parámetros anteriormente definidos como también la variable positiva $CO_{j,k}$ estarán asociados al *"slot"* ubicado más adelante en el tiempo, independientemente de cual esquema de secuenciación se utilice para el equipo $j \in J_l$. Por lo tanto, los mismos estarán ubicados a la izquierda de la ranura $k \in K_j$ para el caso en que se cumpla $j \in J_l$ y $l \in L^1$. Del mismo modo, cuando el *"slot"* $(k-1)$ esté definido en un equipo $j \in J_l$ y el esquema para dicha etapa sea L^2, la variable $CO_{j,(k-1)}$ junto a $\tau^{\min}_{i' \cdot,j}$ se ubicarán a la izquierda del mismo. Esta disposición es ilustrada en la Figura 3.4, la cual presenta un ejemplo mínimo con dos equipos de procesamiento $j \in J_l$, uno para cada esquema de secueciación temporal, y dos *"slots"* convencionales definidos en cada uno de ellos. Si se analiza, por ejemplo, el equipo $j_2 \in J_l$ donde $l \in L^2$, cierto *"batch"* i es asignado a la ranura $(k-1)$ y debe definirse la variable $CO_{j,(k-1)}$ que complementa al $\tau^{\min}_{i' \cdot,j}$, para garantizar el valor total τ_{i',i,j_2} del tiempo de transición correspondiente.

Por otra parte, antes de definir las restricciones de secuenciación de *"slots"* es necesario explicar los subconjuntos llamados $SeqDep^1_j$ y $SeqDep^2_j$, introducidos en la Figura 3.4. Los mismos serán utilizados en las ecuaciones que siguen según el esquema de secuenciación elegido para el equipo $j \in J_l$. Como fue mencionado en el Apartado 3.1, el caso de estudio incluye algunas etapas de procesamiento con tiempos de transición fijos, y otras con tiempos de transición dependientes de la secuencia. Por ello, se busca diferenciar los equipos j, capaces de procesar tareas $(i,\ l)$, en los que la secuencia de

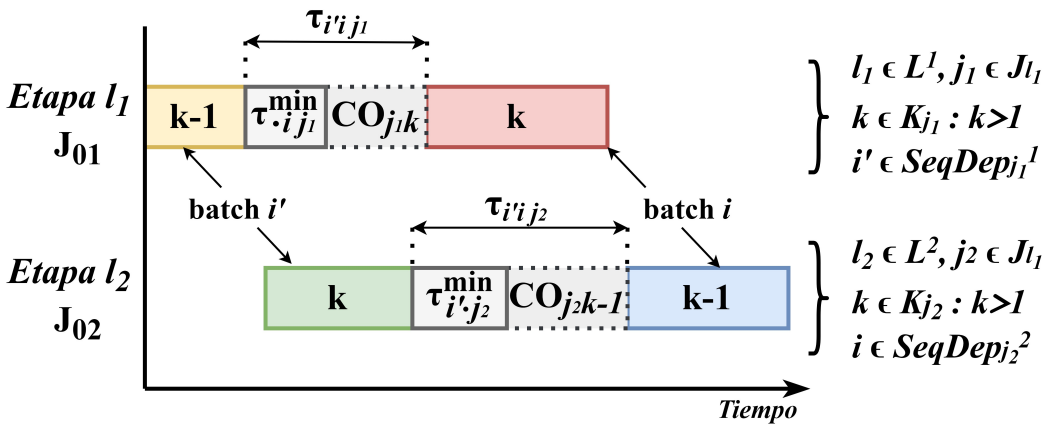

Figura 3.4: Ejemplo - Tiempos de transición en equipos pertenecientes a $SeqDep_j^1$ o $SeqDep_j^2$

Fuente: Elaboración propia

las tareas sea determinante para definir su tiempo de *"changeover"*. Para aquellos que no cumplan esta condición, será suficiente considerar como tiempo de transición fijo al parámetro $tau_{.i,j}^{\min}$ para L^1, o $tau_{i'.,j}^{\min}$ si se corresponde con el esquema L^2. Es así que, como se verá más adelante, las ranuras definidas para los equipos que no sean incluidos en estos subconjuntos no necesitarán de la definición de la variable $CO_{j,k}$ mencionada. La definición de los subconjuntos en cuestión es la siguiente:

$$SeqDep_j^1 = \{i \in I_j : \tau_{.i,j}^{\min} < \tau_{.i,j}^{\max}\} \qquad \forall j \in J_l, l \in L^1 \tag{3.15}$$

$$SeqDep_j^2 = \{i \in I_j : \tau_{i'.,j}^{\min} < \tau_{i'.,j}^{\max}\} \qquad \forall j \in J_l, l \in L^2 \tag{3.16}$$

Ahora bien, para comenzar con el desarrollo de las restricciones de secuenciación, las ecuaciones (3.17) y (3.18) son definidas considerando, respectivamente, los conjuntos L^1 y L^2 para secuenciar los *"slots"* convencionales (es decir, $k \neq k_j^*$).

$$CT_{j,(k-1)}^2 + \sum_{i \in I_j} \tau_{.i,j}^{\min} W_{i,j,k,l} + CO_{j,k}\Big|_{SeqDep_j^1 \neq \emptyset} \leq ST_{j,k}^2 \tag{3.17}$$
$$\forall l \in L^1, j \in J_l, k \in K_j : (k > 1) \wedge (k \neq k_j^*)$$

$$CT^2_{j,k} + \sum_{i' \in I_j} \tau^{\min}_{i'\cdot,j} W_{i',j,k,l} + CO_{j,(k-1)} \Big|_{SeqDep^2_j \neq \emptyset} \leq ST^2_{j,(k-1)} \qquad (3.18)$$
$$\forall l \in L^2, j \in J_l, k \in K_j : (k > 1) \wedge (k \neq k^*_j)$$

Estas ecuaciones garantizan que el tiempo de comienzo $ST^2_{j,k}$ o $ST^2_{j,(k-1)}$, según se sitúe en una etapa incluida dentro del esquema L^1 o L^2 respectivamente, sea posterior al valor del tiempo de finalización del *"slot"* anterior (en el tiempo) y tiempo de transición correspondiente.

Como es indicado en las ecuaciones (3.17) y (3.18), cuando los subconjuntos $SeqDep^1_j$ y $SeqDep^2_j$ son no vacíos, se introducen las variables no negativas $CO_{j,k}$ y $CO_{j,(k-1)}$ respectivamente. Estas variables, en conjunto con el valor de $\tau^{\min}_{\cdot i,j}$ en el esquema de secuenciación implementado en L^1 o $\tau^{\min}_{i'\cdot,j}$ en el esquema utilizado para L^2, representan el tiempo de transición dependiente de la secuencia requerido entre las tareas asignadas a dos *"slots"* consecutivos de un mismo equipo $j \in J_l$. Cuando los subconjuntos $SeqDep^1_j$ y $SeqDep^2_j$, el lado izquierdo de las ecuaciones (3.17) y (3.18) para un dado equipo j no incluyen las variables $CO_{j,k}$ y $CO_{j,(k-1)}$. En este caso la totalidad del tiempo de transición fijo entre los *"batches"* procesados en los *"slots"* k y $(k-1)$ es considerado por el parámetro $tau^{\min}_{\cdot i,j}$ para la ecuación (3.17) y $tau^{\min}_{i'\cdot,j}$ en la ecuación (3.18). Nótese que, de acuerdo a la ecuación (3.15) y las definiciones dadas en (3.13), si $i \notin SeqDep^1_j$ entonces $\tau^{\min}_{\cdot i,j} = \tau_{i',i,j} = \tau^{\max}_{\cdot i,j}$, $\forall i' \in I_j : i' \neq i$. De modo similar, según las definiciones (3.16) y (3.14), si $i' \notin SeqDep^2_j$ entonces $\tau^{\min}_{i'\cdot,j} = \tau_{i',i,j} = \tau^{\max}_{i'\cdot,j}$, $\forall i \in I_j : i \neq i'$.

Para calcular el valor de $CO_{j,k}$, se utilizan las siguientes restricciones:

$$CO_{j,k} \geq \left(\tau^{\max}_{\cdot i,j} - \tau^{\min}_{\cdot i,j}\right) W_{i,j,k,l} - \sum_{i' \in I_j \,:\, i' \neq i} \left(\tau^{\max}_{\cdot i,j} - \tau_{i',i,j}\right) W_{i',j,(k-1),l} \qquad (3.19)$$
$$\forall l \in L^1, j \in J_l, i \in SeqDep^1_j, k \in K_j : (k > 1) \wedge (k \neq k^*_j)$$

$$CO_{j,(k-1)} \geq \left(\tau^{\max}_{i'\cdot,j} - \tau^{\min}_{i'\cdot,j}\right) W_{i',j,k,l} - \sum_{i \in I_j \,:\, i \neq i'} \left(\tau^{\max}_{i'\cdot,j} - \tau_{i',i,j}\right) W_{i,j,(k-1),l} \qquad (3.20)$$
$$\forall l \in L^2, j \in J_l, i' \in SeqDep^2_j, k \in K_j : (k > 1) \wedge (k \neq k^*_j)$$

De forma resumida, se puede analizar la ecuación (3.19) que se aplica a las etapas $l \in L^1$, mediante la cual se selecciona apropiadamente la cota inferior para $CO_{j,k}$ basada en los *"batches"* i' e i asignados a los *"slots"* $(k-1)$ y k, respectivamente. Para un dado $(i,\ j,\ k,\ l)$, si $W_{i,j,k,l} = 1$ entonces la ecuación (3.19) se puede simplificar como:

$$CO_{j,k} \geq -\tau^{\min}_{\cdot i,j} + \sum_{i' \in I_j\,:\,i' \neq i} \tau_{i',i,j}\, W_{i',j,(k-1),l}$$

Donde $W_{i',j,(k-1),l}$ será igual a 1 para algún $i' \in I_j$ como consecuencia de la restricción (3.6). Por lo tanto, nótese aquí que la variable $CO_{j,k}$ representará la porción excedente del tiempo de transición que supera al parámetro $\tau^{\min}_{\cdot i,j}$. Este último, incluido en la ecuación (3.17), en todos los casos será menor o igual al $\tau_{i',i,j}$ seleccionado con lo cual la variable $CO_{j,k}$ resultará cero o positiva.

Por el contrario, si $W_{i,j,k,l} = 0$ y por definición $\tau^{\max}_{\cdot i,j} \geq \tau_{i',i,j}$ $(\forall i' \in I_j : i' \neq i)$, el lado derecho de la ecuación (3.19) se convierte en cero o negativo y la variable $CO_{j,k}$ no se ve afectada. Un análisis similar puede aplicarse a la ecuación (3.20) usada para las etapas $l \in L^2$.

Finalmente, cuando el *"slot"* considerado es multivaluado, se debe tener presente que no serán definidas las variables positivas asociadas a sus tiempos de comienzo y finalización. Por ello, en este caso se relacionan los tiempos de cada tarea $(i,\ l)$ o $(i',\ l)$ asignada a dicho *"slot"* (si las hubiere) con los tiempos de comienzo o finalización del *"slot"* consecutivo, según corresponda dado el esquema de secuenciación temporal aplicado. Las restricciones que consideran las ranuras multivaluadas son las siguientes:

$$\begin{aligned} CT^2_{j,(k-1)} + \sum_{i' \in I_j\,:\,i' \neq i} \tau_{i',i,j}\, W_{i',j,(k-1),l} \leq ST^1_{i,l} + M\ (1 - W_{i,j,k,l}) \\ \forall l \in L^1, j \in J_l, i \in I_j, k \in K_j : (k = k^*_j) \wedge (k > 1) \end{aligned} \tag{3.21}$$

$$\begin{aligned} CT^1_{i',l} + \sum_{i \in I_j\,:\,i \neq i'} \tau_{i',i,j}\, W_{i,j,(k-1),l} \leq ST^2_{j,(k-1)} + M\ (1 - W_{i',j,k,l}) \\ \forall l \in L^2, j \in J_l, i' \in I_j, k \in K_j : (k = k^*_j) \wedge (k > 1) \end{aligned} \tag{3.22}$$

La ecuación (3.21) se puede analizar con un breve ejemplo teniendo en cuenta que $k = k_j^*$ y, por lo tanto, el *"slot"* k puede alojar más de un *"batch"* i. Asumiendo que se cumpla $W_{i,j,k,l} = 1$ para un (i, j, k, l) dado, la ecuación queda:

$$CT^2_{j,(k-1)} + \sum_{i' \in I_j : i' \neq i} \tau_{i',i,j} \, W_{i',j,(k-1),l} \leq ST^1_{i,l}$$

Con lo cual la tarea $(i,\ l)$ asignada al slot multivaluado $(j,\ k^*)$ comienza su procesamiento luego del tiempo de finalización del *"slot"* anterior $(k-1)$ y del tiempo de transición correspondiente. Nótese que se ignoran las demás tareas $(i,\ l)$ que podrían también haber sido asignadas a la misma ranura ya que, en principio, no se toma decisión alguna respecto a su secuenciación. En caso contrario, si se verifica que $W_{i,j,k,l} = 0$, el segundo término del lado derecho de la ecuación (3.21) resulta en un valor grande (es decir, la restricción Big-M no se activa), con lo cual la variable $ST^1_{i,l}$ no se ve afectada. Un análisis similar podría presentarse para la ecuación (3.22), que se aplica al subconjunto de etapas $l \in L^2$.

3.3.4. Función objetivo

En este proyecto la función objetivo bajo estudio es la minimización del *"makespan"*, el cual es el mínimo tiempo requerido para completar todas las tareas. Este objetivo puede ser modelado fácilmente por la ecuación (3.23), donde se considera el tiempo de finalización de la última etapa l_i^{last} de cada *"batch"* i. En las formulaciones de *"time-slots"* también es útil incluir la ecuación (3.24), considerada para el último *"slot"* convencional (k_j^{last}) de cada equipo.

$$CT^1_{i,l} \leq MK \qquad \forall i \in I, l \in L_i : l = l_i^{\text{last}} \tag{3.23}$$

$$CT^2_{j,k} \leq MK \qquad \forall j \in J, k \in K_j : k = k_j^{\text{last}} \tag{3.24}$$

3.3.5. Restricciones de ajuste adicionales

En este apartado se presentan algunas restricciones que mejoran la convergencia del proceso de resolución del modelo, para la función objetivo en particular aplicada.

En primer lugar, se definen parámetros para los tiempos más tempranos de comienzo y finalización, $EST_{i,l}$ (*"Earliest Start Time"*) y $ETC_{i,l}$ (*"Earliest Time to Complete"*) respectivamente, para cada tarea (i, l). En el caso de $EST_{i,l}$, para cada *"batch"* i puede ser definido, en primera instancia, como la suma de los mínimos tiempos de procesamiento $pt_{i,j}$ necesarios en todas las etapas anteriores a la etapa en análisis, tal como se muestra en la Ecuación (3.25). Por el lado de $ETC_{i,l}$, se define inicialmente aplicando un razonamiento similar en la Ecuación (3.26) pero, esta vez, con las etapas posteriores de la tarea (i, l) en cuestión. Luego, estos parámetros serán actualizados con una ecuación específica en cada iteración del algoritmo principal que se presenta en el Apartado 4.3.

$$EST_{i,l} = \sum_{l' \in L \,:\, l' < l} min_{j \in J_{i,l'}} \{pt_{i,j}\} \qquad \forall i \in I, l \in L_i \tag{3.25}$$

$$ETC_{i,l} = \sum_{l' \in L \,:\, l' > l} min_{j \in J_{i,l'}} \{pt_{i,j}\} \qquad \forall i \in I, l \in L_i \tag{3.26}$$

Utilizando las definiciones anteriores, se presentan las ecuaciones (3.27) y (3.28). Las mismas acotan la región de búsqueda de soluciones restringiendo los valores del tiempo de comienzo y finalización de las ranuras (j, k). Nótese que estas ecuaciones de ajuste no son incluidas para los *"slots"* multivaluados, dado que, como ya fue mencionado, no tienen tiempos de comienzo y finalización asociados.

$$\sum_{i \in I_j} EST_{i,l} \, W_{i,j,k,l} \leq ST^2_{j,k} \qquad \forall l \in L, j \in J_l, k \in K_j : k \neq k^*_j \tag{3.27}$$

$$MK - \sum_{i \in I_j} ETC_{i,l} \, W_{i,j,k,l} \geq CT^2_{j,k} \qquad \forall l \in L, j \in J_l, k \in K_j : k \neq k^*_j \tag{3.28}$$

Nótese que la sumatoria presente en el lado izquierdo de estas restricciones podrá incluir, como mucho, una única variable binaria $W_{i,j,k,l} = 1$, correspondiéndose con los índices (l, j, k) de la ecuación. Esto se debe a que solo puede ser seleccionada una tarea (i, l) para ser procesada en cada ranura convencional.

Por otra parte, mediante la definición de los parámetros EST_j y ETC_j se introduce una extensión de los conceptos de tiempo más temprano de comienzo y menor tiempo pendiente de finalización respectivamente, aplicable en este caso a los equipos de proce-

samiento $j \in J_l$. En el caso del parámetro EST_j, se considera como el mínimo $EST_{i,l}$ entre todas las tareas (i,l) que pueden ser procesadas en dicho equipo. De forma similar, ETC_j será el mínimo $ETC_{i,l}$ entre todas las tareas (i,l) factibles de ser realizadas en j. Expresado en forma algebraica queda como sigue:

$$EST_j = min_{i \in I_j} \{EST_{i,l}\} \qquad \forall j \in J_l, l \in L \tag{3.29}$$

$$ETC_j = min_{i \in I_j} \{ETC_{i,l}\} \qquad \forall j \in J_l, l \in L \tag{3.30}$$

A su vez, se definen también los subconjuntos $\overleftarrow{K}_j^k$ y $\overrightarrow{K}_j^k$. En el primer caso, $\overleftarrow{K}_j^k$ incluye al *"slot"* $k \in K_j$ y a todas las ranuras que se ubican previamente en el tiempo dentro del mismo equipo $j \in J_l$. Para ello debe hacerse una diferenciación según el equipo j integre una etapa $l \in L$ secuenciada temporalmente de forma ascendente (L^1) o de forma descendente (L^2). De la misma manera, este criterio debe aplicarse al subconjunto $\overrightarrow{K}_j^k$ que incluye al *"slot"* $k \in K_j$ y todas las ranuras posteriores en el tiempo en determinado equipo $j \in J_l$. Estas definiciones pueden expresarse como:

$$\overleftarrow{K}_j^k = \{k' \in K_j : k' \leq k\} \qquad \forall l \in L^1, j \in J_l, k \in K_j \tag{3.31}$$

$$\overrightarrow{K}_j^k = \{k' \in K_j : k' \geq k\} \qquad \forall l \in L^1, j \in J_l, k \in K_j \tag{3.32}$$

$$\overleftarrow{K}_j^k = \{k' \in K_j : k' \geq k\} \qquad \forall l \in L^2, j \in J_l, k \in K_j \tag{3.33}$$

$$\overrightarrow{K}_j^k = \{k' \in K_j : k' \leq k\} \qquad \forall l \in L^2, j \in J_l, k \in K_j \tag{3.34}$$

Los subconjuntos anteriores son utilizados en las ecuaciones de ajuste (3.35) y (3.36) presentadas a continuación:

$$EST_j \left(\sum_{i' \in I_j} W_{i',j,(k_1),l} \right) + (EST_{i,l} - EST_j) \left(\sum_{k' \in \overleftarrow{K}_j^k} W_{ijk'l} \right) \leq ST_{j,k}^2 \qquad (3.35)$$
$$\forall i \in I, l \in L_i, j \in J_l, k \in K_j : k \neq k_j^*$$

$$CT_{j,k}^2 + ETC_j \left(\sum_{i' \in I_j} W_{i',j,(k_1),l} \right) + (ETC_{i,l} - ETC_j) \left(\sum_{k' \in \overrightarrow{K}_j^k} W_{i,j,k',l} \right) \leq MK \qquad (3.36)$$
$$\forall i \in I, l \in L_i, j \in J_l, k \in K_j : k \neq k_j^*$$

Analizando la ecuación (3.35), al ser ocupado el primer *"slot"* k_1, el primer término del lado izquierdo vale EST_j. Luego, dado el caso en que la restricción es analizada para un índice $i \in I_j$ que, a su vez, otorga $W_{i,j,k',l} = 1$ para algún $k' \in \overleftarrow{K}_j^k$, la ecuación queda:

$$EST_{i,l} \leq ST_{j,k}^2$$

Considerando primeramente el caso en que $k \neq k' \in \overleftarrow{K}_j^k$, la expresión anterior indica que la tarea (i, l) asignada al *"slot"* $k' \in \overleftarrow{K}_j^k$ debe comenzar a procesarse antes de la tarea asignada al *"slot"* $k \in \overleftarrow{K}_j^k$, el cual es ubicado de forma posterior en el tiempo. Por otro lado, si se verificara que $k = k' \in \overleftarrow{K}_j^k$, la ecuación (3.35) también ajusta la región de búsqueda de soluciones a que la tarea (i, l) asignada a la ranura (j, k) debe tener un tiempo de comienzo $ST_{j,k}^2$ mayor o igual a su $EST_{i,l}$ correspondiente.

Por otro lado, nuevamente en la sumatoria presente en el segundo término del lado izquierdo de la restricción (3.35) podrá seleccionarse únicamente una variable binaria $W_{i,j,k',l} = 1$, según coincida con los índices de la ecuación. En caso contrario, cuando la variable $W_{i,j,k',l} = 0 \forall k \in \overleftarrow{K}_j^k$, el segundo término del lado izquierdo se anula y, si $W_{i,j,(k_1),l} = 1$, la ecuación también se verifica para cualquier *"slot"* $k \in K_j$. Este desarrollo analizando la ecuación (3.35) puede ser planteado, de forma análoga, para el caso de la ecuación (3.36). Vale mencionar que ambas restricciones son compatibles con ambos esquemas de secuenciación temporal, dada la definición previa de los subconjuntos $\overrightarrow{K}_j^k$ y $\overleftarrow{K}_j^k$.

Por último, se introducen dos restricciones de ajuste para los *"slots"* multivaluados. Con ellas se busca aproximar todos los tiempos involucrados en procesar las tareas $(i,\ l)$ asignadas al *"slot"* $k_j^* \in K_j$.

$$CT^2_{j,(k-1)} + \sum_{i \in I_j} \left(\tau^{\min}_{\cdot i,j} + pt_{i,j}\right) W_{i,j,k,l} + ETC_j \left(\sum_{i' \in I_j} W_{i',j,(k_1),l}\right) \leq MK \qquad (3.37)$$
$$\forall l \in L^1, j \in J_l, k = k_j^* : (k > 1)$$

$$EST_j \left(\sum_{i \in I_j} W_{i,j,(k_1),l}\right) + \sum_{i' \in I_j} \left(\tau^{\min}_{i'\cdot,j} + pt_{i',j}\right) W_{i',j,k,l} \leq ST^2_{j,(k-1)} \qquad (3.38)$$
$$\forall l \in L^2, j \in J_l, k = k_j^* : (k > 1)$$

En primer lugar, la ecuación (3.37) se aplica para las etapas secuenciadas de la forma L^1 y, consecuentemente, el *"slot"* $(k-1)$ es el anterior a la ranura multivaluada k_j^*. El primer término de la ecuación es, por lo tanto, el tiempo de finalización del *"slot"* $(k-1)$. Por otra parte, si se observa el segundo término del lado izquierdo, este incluye los tiempos de procesamiento y el mínimo tiempo de transición para todos los *"batches"* asignados en $k_j^* \in K_j$. En cuanto al parámetro ETC_j, el mismo es incluido cuando el equipo $j \in J_l$ esta activo, es decir que $W_{i',j,(k_1),l} = 1$ para algún $i' \in I_j$.

Para el caso de la ecuación (3.38), su aplicación es similar pero ahora considerando las etapas $l \in L^2$. En el caso en que el equipo esté activo, la variable $W_{i,j,(k_1),l}$ valida el tiempo más temprano de comienzo de la actividad (EST_j) para el equipo j. Luego, la suma de los tiempos de procesamiento $pt_{i',j}$ y mínimos tiempos de transición $\tau^{\min}_{i'\cdot,j}$ para cada *"batch"* $i' \in I_j$ representa una estimación del tiempo mínimo necesario para procesar todas las tareas asignadas al *"slot"* multivaluado $k_j^* \in K_j$. Este lado izquierdo de la restricción debe ser menor o igual al tiempo de comienzo de la ranura $(k-1) \in K_j$, la cual se ubica de forma posterior en el tiempo al *"slot"* k_j^* dentro de un esquema de secuenciación L^2.

Capítulo 4

METODOLOGÍA ALGORÍTMICA DE RESOLUCIÓN

4.1. Descripción General

En este capítulo se presenta una metodología matemática-algorítmica que apunta a resolver la programación de operaciones a corto plazo en ámbitos de tamaño industrial. La estrategia se basa en la construcción progresiva del *"schedule"* mediante la resolución de una secuencia de subproblemas más pequeños que el problema completo. En sucesivas iteraciones, se busca encontrar una buena propuesta de ranuras para los equipos de procesamiento y se fija de forma prioritaria la programación de las tareas en los recursos críticos compartidos. Para la resolución de los subproblemas se utiliza la formulación MILP basada en *"time-slots"* expuesta en el Capítulo 3. Durante el proceso resolutivo se conservan las cotas de integralidad de la solución con el fin de llevarlo a cabo de manera eficiente y brindar una medida de calidad para la solución encontrada.

El procedimiento iterativo propuesto se basa en analizar alternativas para el número de *"slots"* de cada equipo en una dada etapa de procesamiento. Durante este proceso, se resuelven subproblemas aproximados y exactos de manera alternada centrando la atención en una de las etapas y considerando un subconjunto de tareas previas y posteriores ubicadas aguas arriba y aguas abajo de la misma. En los pasos sucesivos, se busca iden-

tificar y fijar, una a una, la etapa considerada crítica y construir la solución de forma gradual.

Concretamente, el algoritmo comprende dos procedimientos: una Iteración Interna y una Iteración Principal. La primera aborda el desafío de encontrar un número de ranuras para cada equipo que conduzca a una buena solución. La segunda, por su parte, identifica la etapa cuello de botella y fija las asignaciones de la misma, repitiendo el ciclo hasta que todas las etapas son completadas. Aquí vale mencionar que, a medida que se avanza en las Iteraciones Principales, la incertidumbre decrece ya que se dispone de más información sobre las etapas restantes. En lo que sigue, se describe en detalle la estructura de la metodología matemática-algorítmica diseñada.

4.2. Iteración Interna: Encontrar la Mejor Configuración de *"Slots"*

La Iteración Interna, analiza distintas alternativas para el número de *"slots"* de cada equipo considerando una única etapa identificada como l^{act}, con el objetivo de encontrar la mejor solución posible para esa etapa. Esta iteración es identificada con el índice b, el cual representa el número de ciclo en curso. De este modo, inicialmente b es puesto en 1 y, en cada repetición, se resuelven distintos subproblemas: un modelo aproximado y un modelo exacto, alternadamente. En los subproblemas aproximados, la formulación incluye ranuras multivaluadas en la etapa l^{act} para hacer posible la asignación de una cantidad de tareas diferente a la configuración de *"slots"* incluidos por equipo. Por otro lado, para la resolución de cada subproblema exacto se establece una cantidad fija de ranuras convencionales en cada equipo de la etapa l^{act}.

Seguidamente, se muestra un ejemplo simplificado para ilustrar la resolución del primer ciclo de la Iteración Interna ($b = 1$) en una instalación compuesta por 3 etapas, cada una con 2 equipos que trabajan en paralelo. Comenzando por el subproblema aproximado y considerando que la etapa en análisis corresponde a $l^{\text{act}} = l_2$, en la Figura 4.1 inicialmente, todos los equipos pertenecientes a etapas aún no fijadas por la Iteración Principal (que por ende, como se verá más adelante, no pertenecen al subconjunto L^{P}) comienzan el proceso con una propuesta de ranuras inicial, en este caso $m_j^0 = 3$. Para el caso de la etapa l^{act}, las ranuras multivaluadas servirán para obtener propuestas del número de *"slots"* a evaluar posteriormente en el subproblema exacto siguiente. Por otra parte, el resto de las etapas ($l \neq l^{\text{act}} \wedge l \notin L^{\text{P}}$) son incluidas en el modelo con un número re-

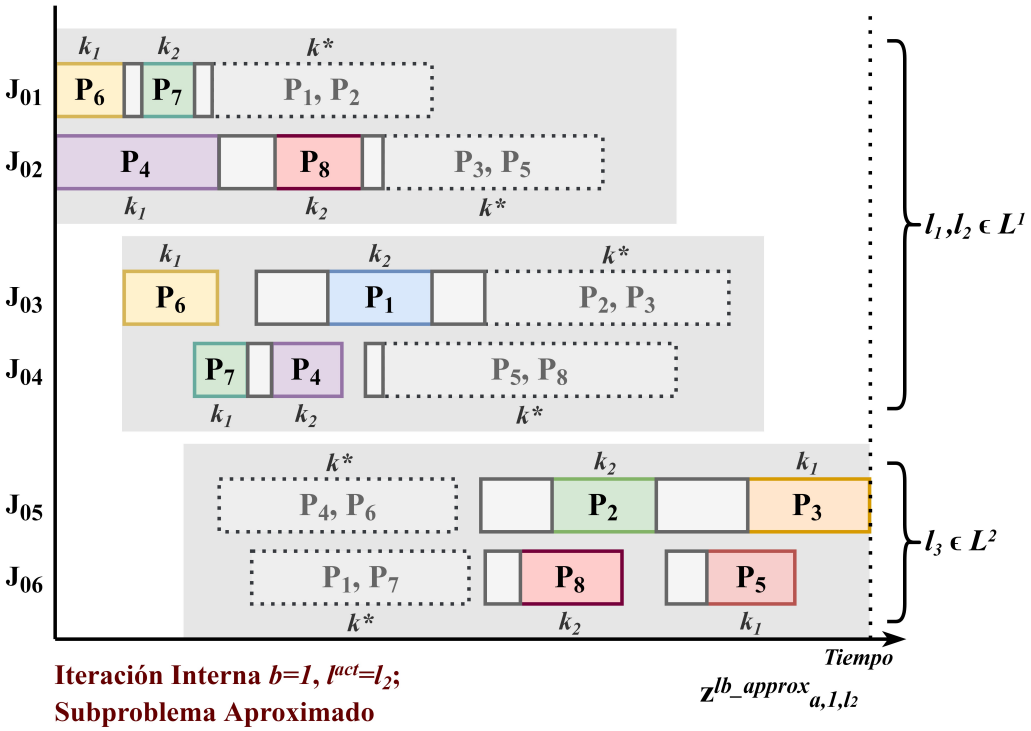

Figura 4.1: Ejemplo Iteración Interna - subproblema aproximado: análisis de propuestas de *"slots"* para la etapa l^{act}

Fuente: Elaboración propia

ducido de *"slots"* incorporando ranuras multivaluadas que ayudan a obtener información aproximada sobre su ocupación. Esta característica se incluye en la resolución de todos los subproblemas, sean aproximados o exactos, con lo cual se logra un ahorro notable en la cantidad de variables binarias del modelo y se mejora su convergencia. De esta manera, al encontrar la mejor solución para el subproblema aproximado quedan determinadas la cota inferior para dicha etapa ($z^{\text{lb\_approx}}_{a,1,l_2}$) y el valor de la función objetivo ($z^{\text{sol\_approx}}_{a,1,l_2}$).

En esta instancia, cabe mencionar que para definir los esquemas de secuenciación temporal utilizados en cada etapa se consideran dos estadios. El primero de ellos, es cuando aún no se ha fijado la solución de ninguna etapa (es decir, $a = 1$). En este caso, se utiliza el esquema L^1 para la etapa l^{act} y todas las anteriores ($l < l^{\text{act}}$). Para el caso de las etapas posteriores a la actual analizada ($l > l^{\text{act}}$) se aplicará el esquema de secuenciación L^2. Por otra parte, para los ciclos de la Iteración Principal siguientes ($a > 1$), se considera el parámetro l^{critic}, el cual se refiere a la etapa que presenta las mayores dificultades para encontrar una buena solución. Esto significa que la etapa l^{critic} es aquella seleccionada como l^{sel} en la primera Iteración Principal ($a = 1$). De este modo, el segundo estadio queda definido para las iteraciones en las cuales se cumple que $a > 1$ y se aplica el esquema

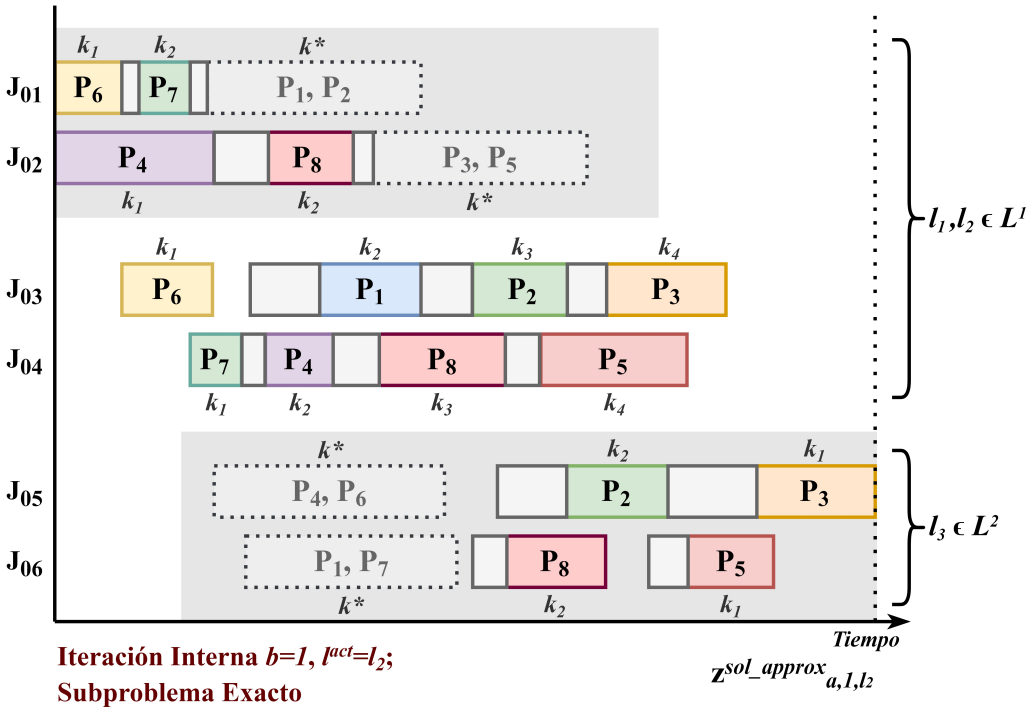

Figura 4.2: Ejemplo Iteración Interna - subproblema exacto: evaluación de una configuración de *"slots"* para la etapa l^{act}

Fuente: Elaboración propia

de secuenciación L^1 para la etapa crítica identificada y todas las anteriores ($l \leq l^{\text{critic}}$) o el esquema de secuenciación L^2 para las etapas restantes ($l > l^{\text{critic}}$). El detalle de los esquemas de secuenciación utilizados se incluye en la estructura algorítmica de la Iteración Principal, desarrollada posteriormente en el Apartado 4.3.1.

Continuando con la descripción del ejemplo, el paso siguiente es la resolución del subproblema exacto. El mismo es ilustrado con la Figura 4.2 y, como puede observarse, en este caso la totalidad de ranuras presentes en la etapa l^{act} son convencionales. La cantidad de *"slots"* presentes en cada equipo de la etapa l^{act} es la correspondiente a la cantidad de *"batches"* asignados a dichos equipos en el subproblema aproximado anterior, lo cual es desarrollado en detalle más adelante dentro de este mismo Apartado. Como fue mencionado anteriormente, en esta instancia las etapas aguas arriba y aguas abajo de l^{act}, que aún no fijadas por la Iteración Principal, se incluyen con un número reducido de ranuras con el fin de obtener estimaciones de los tiempos asociados a las tareas (i, l) que deben ser procesadas. Para concluir con el ejemplo, para el modelo exacto se almacenan las cotas obtenidas: la mejor solución posible ($z^{\text{lb}}_{a,1,l_2}$) y el valor de la función objetivo ($z^{\text{sol}}_{a,1,l_2}$).

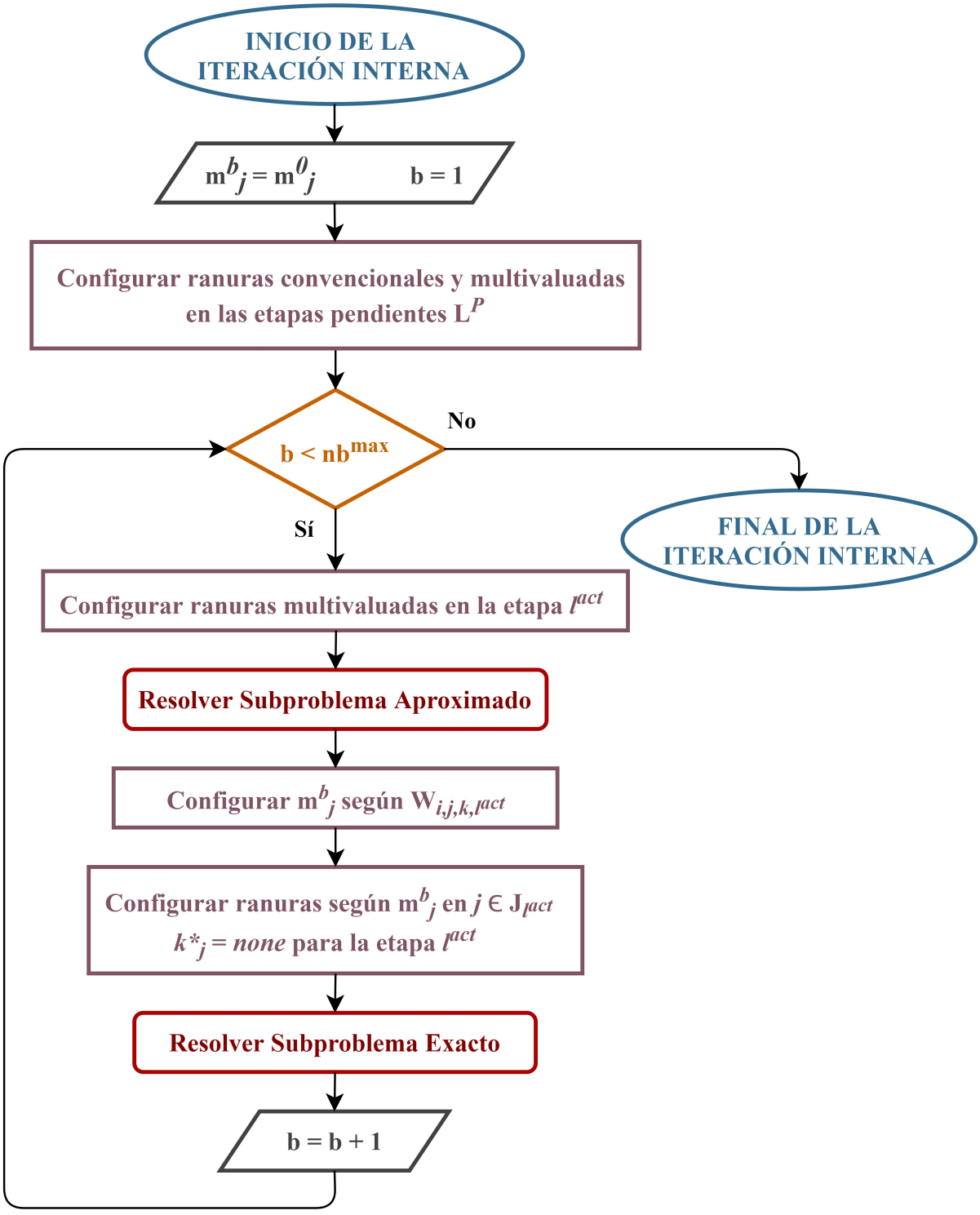

Figura 4.3: Diagrama de Flujo de la Iteración Interna

Fuente: Elaboración propia

Por otra parte, en la Figura 4.3 se presenta la estructura general de la Iteración Interna en un diagrama de flujo. Comenzando con la descripción del mismo, primeramente para el ciclo $b = 1$ se parte de un número inicial de ranuras m_j^0 en cada equipo perteneciente a las etapas pendientes (es decir, $j \in J_l, l \in L^{\mathrm{P}}$), y se realiza la configuración de los conjuntos de ranuras convencionales y multivaluadas. Luego, comenzando con la iteración en la etapa l^{act}, el modelo aproximado busca una buena configuración para el número de *"slots"* de sus equipos $j \in J_{l^{\mathrm{act}}}$ evitando las selecciones previamente consideradas. La cantidad de ranuras obtenida se almacena en el parámetro m_j^b de acuerdo a la definición dada en la ecuación (4.1). En cada iteración b, se obtiene una nueva entrada para m_j^b basada en las variables binarias de asignación de la solución actual del modelo aproximado.

$$m_j^b = \sum_{i \in I_j} \sum_{k \in K_j} W_{i,j,k,l} \qquad \forall j \in J_{l^{\mathrm{act}}} \tag{4.1}$$

Luego, el subproblema exacto inspecciona el número de ranuras propuestas para la etapa l^{act}, por lo que resuelve un modelo con m_j^b ranuras convencionales en cada equipo j de la etapa en análisis. Véase que, con el fin de forzar el número de ranuras utilizado en cada equipo, el total de propuestas es igual a la cantidad de tareas (i, l) que deben procesarse en dicha etapa. Además, recuérdese que en este subproblema ya no se incluyen ranuras multivaluadas en estos equipos. Continuando con las iteraciones, el siguiente modelo aproximado parte de la última configuración de ranuras analizada e incluye nuevamente *"slots"* multivaluados para los equipos de la etapa l^{act}, considerando al último de ellos como k_j^*.

Cada modelo aproximado recortará las propuestas de número de ranuras de las iteraciones previas en la misma etapa, para no recaer en una alternativa ya analizada. Con este fin, se introduce la variable binaria $Q_{j,m}$ para indicar que m *"batches"* son alojados en el equipo j, cuando $Q_{j,m} = 1$. En cada modelo aproximado, el resultado obtenido para m puede tomar un valor entre $k_j^{\min}$ y $k_j^{\max}$. Esta restricción está establecida por la ecuación (4.2), de modo que se selecciona una sola variable $Q_{j,m}$ para cada equipo j en cada Iteración Interna b. A continuación, la ecuación (4.3) asegura que la variable $Q_{j,m}$ corresponde con el número de *"batches"* asignados por el modelo. Por otro lado, la ecuación (4.4) restringe a que, durante la Iteración Interna b en la etapa l^{act}, no se repita la misma configuración de ranuras propuesta en otra Iteración Interna anterior b'. Asumiendo que se considera actualmente la Iteración Interna b, esto se logra evitando que sea factible seleccionar $m_j^{b'}$ *"baches"* en cada equipo j, considerando simultáneamente a todos

los equipos $j \in J_{l^{\text{act}}}$, para cada iteración interna previa b'. Para lograr que no sea posible asignar cierta configuración de ranuras, la suma de las variables $Q_{j,m_j^{b'}}$ debe ser menor a la cantidad total de equipos menos uno ($|J_{l^{\text{act}}}| - 1$).

$$\sum_{m=k_j^{\min}}^{k_j^{\max}} Q_{j,m} = 1 \qquad \forall j \in J_{l^{\text{act}}} : b > 1 \tag{4.2}$$

$$\sum_{i \in I_j} \sum_{k \in K_j} W_{i,j,k,l} = \sum_{m=k_j^{\min}}^{k_j^{\max}} m_j^b \cdot Q_{j,m} \qquad \forall j \in J_{l^{\text{act}}} : b > 1 \tag{4.3}$$

$$\sum_{j \in J_{l^{\text{act}}}} Q_{j,m_j^{b'}} \leq |J_{l^{\text{act}}}| - 1 \qquad \forall b' = 1, \ldots, (b-1) \tag{4.4}$$

Para finalizar la Iteración Interna en la etapa l^{act}, uno de los mecanismos utilizados es la inclusión del parámetro $nb^{\max}$, que indica el máximo de Iteraciones Internas permitidas en una misma etapa. Más adelante, en la Apartado 4.3, serán desarrolladas estrategias adicionales para concluir esta iteración y lograr ahorros en el tiempo invertido para la ejecución del modelo.

4.2.1. Algoritmo para obtener la mejor solución y cotas para la etapa actual

Para introducir el guardado sistemático de las soluciones y cotas halladas, se hace uso de los caracteres z o Z según corresponda a los atributos de un subproblema en particular dentro de la Iteración Interna b o, respectivamente, las cotas halladas para una etapa l en el contexto de la Iteración Principal a. A su vez, se utiliza el superíndice *sol* para indicar el valor de la función objetivo y el superíndice *lb* para referir a la cota inferior hallada para la etapa, esta última también llamada mejor solución posible. Por otro lado, los superíndices *lb_approx* y *sol_approx* indican que los límites considerados se refieren a soluciones halladas en cierto subproblema aproximado. De esta forma, se declaran los siguientes parámetros:

$$Z_{a,l}^{\text{lb\_approx}} \qquad Z_{a,l}^{\text{sol}} \qquad Z_{a,l}^{\text{lb}} \tag{4.5}$$

$$z_{a,b,l}^{\text{sol\_approx}} \qquad z_{a,b,l}^{\text{lb\_approx}} \qquad z_{a,b,l}^{\text{sol}} \qquad z_{a,b,l}^{\text{lb}} \tag{4.6}$$

Ahora bien, antes de comenzar con la descripción ordenada del pseudocódigo presentado en el Algoritmo 4.1, es necesario mencionar que el parámetro $Z_{a,l}^{\text{lb\_approx}}$ conserva la máxima cota inferior obtenida para todos los subproblemas aproximados de la Iteración Interna en cierta etapa l en el contexto de una dada Iteración Principal a. De manera similar, el parámetro $Z_{a,l}^{\text{sol}}$ guardará el menor valor de la función objetivo para los modelos exactos de una dada etapa l. Por otra parte, $Z_{a,l}^{\text{lb}}$ considera la menor solución posible que puede garantizarse para la etapa en la que se itera, en ese momento l^{act}.

En primer lugar, al inicio del algoritmo, es necesario asignar en 1 el índice de la Iteración Interna b y comenzar con un valor grande M para $Z_{a,l^{\text{act}}}^{\text{sol}}$ y $Z_{a,l^{\text{act}}}^{\text{lb}}$, así como un valor de *cero* para el parámetro $Z_{a,l^{\text{act}}}^{\text{lb\_approx}}$. A su vez, debe ser definida la cantidad de ranuras iniciales m_j^0 para todos los equipos de las etapas pendientes distintas de la etapa actual ($l \in L^{\text{P}} : l \neq l^{\text{act}}$). Seguidamente, se configuran los *"slots"* iniciales, sean convencionales o multivaluados, así como los conjuntos K_j, $\overleftarrow{K}_j^k$ y $\overrightarrow{K}_j^k$, utilizando las ecuaciones correspondientes para todas las etapas pendientes $l \in L^{\text{P}}$.

A continuación, comenzando con el ciclo iterativo en la etapa l^{act} el mismo se repetirá, como primera condición, mientras no se supere el máximo de repeticiones admitidas para una misma etapa $nb^{\max}$. La segunda condición será que la mejor solución posible hallada entre todos los subproblemas aproximados $Z_{a,l^{\text{act}}}^{\text{lb\_approx}}$ no alcance a dicho límite para la etapa en las soluciones exactas $Z_{a,l^{\text{act}}}^{\text{lb}}$, admitiendo entre éstos una diferencia pequeña positiva ε para evitar errores de redondeo. Esto tiene sentido ya que la inclusión de *"slots"* multivaluados en el primero de los modelos conduce a una aproximación subestimada de la solución. Por lo tanto, si se considera el no cumplimento de esta condición, y recordando que ciertas alternativas de propuestas de ranuras fueron descartadas en iteraciones previas, continuar con la exploración conduce a una cota inferior del siguiente modelo aproximado igual o mayor que aquella mejor solución posible hallada en una Iteración Interna anterior. Si esto sucede, el análisis de configuración de ranuras adicionales no permitirá incrementar la cota inferior $Z_{a,l^{\text{act}}}^{\text{lb}}$. Dado este caso, puede considerarse que las propuestas de *"slots"* más adecuadas ya han sido probadas y, dando continuidad al proceso, comienzan a ser evaluadas configuraciones de ranuras menos convenientes.

Algoritmo 4.1: Pseudocódigo de la Iteración Interna en la etapa l^{act}

Input: Cantidad de ranuras iniciales $m_j^0 \ \forall\ j \in J_l, l \in L^{\text{P}}$ y etapa actual l^{act}

Output: Mejor solución para la etapa l^{act}

$b \leftarrow 1$

$Z^{\text{sol}}_{a,l^{\text{act}}} \leftarrow M \qquad Z^{\text{lb}}_{a,l^{\text{act}}} \leftarrow M \qquad Z^{\text{lb\_approx}}_{a,l^{\text{act}}} \leftarrow 0$

loop $j \in J_l,\ l \in L^{\text{P}} : l \neq l^{\text{act}}$

  Actualizar $(K_j,\ \overleftarrow{K}^k_j,\ \overrightarrow{K}^k_j)$ *según* m_j^0 ▷ ecs. 3.4 y 3.31 a 3.34

  $k_j^* \leftarrow |K_j|$

while $b < nb^{\max}$ ***and*** $Z^{\text{lb}}_{a,l^{\text{act}}} - Z^{\text{lb\_approx}}_{a,l^{\text{act}}} > \varepsilon$ **do**

  /* PREPARACIÓN PARA EL SUBPROBLEMA APROXIMADO */

  loop $j \in J_{l^{\text{act}}}$

    if $b = 1$ ***or*** $m_j^{b-1} = 0$ **then**

      Actualizar $(K_j,\ \overleftarrow{K}^k_j,\ \overrightarrow{K}^k_j)$ *según* m_j^0 ▷ ecs. 3.4 y 3.31 a 3.34

    $k_j^* \leftarrow |K_j|$

  solve Subproblema Aproximado **minimizing** MK

  Guardar $(z^{\text{sol\_approx}}_{a,b,l^{\text{act}}},\ z^{\text{lb\_approx}}_{a,b,l^{\text{act}}})$

  $Z^{\text{lb\_approx}}_{a,l^{\text{act}}} \leftarrow \max\{Z^{\text{lb\_approx}}_{a,l^{\text{act}}},\ z^{\text{lb\_approx}}_{a,b,l^{\text{act}}}\}$

  /* PREPARACIÓN PARA EL SUBPROBLEMA EXACTO */

  $L^{\text{F}} = \{l^{\text{act}}\}$

  loop $j \in J_{l^{\text{act}}}$

    Actualizar (m_j^b) *según* $W_{i,j,k,l^{\text{act}}}$ ▷ ec. 4.1

    $k_j^* \leftarrow none$

    Actualizar $(K_j,\ \overleftarrow{K}^k_j,\ \overrightarrow{K}^k_j)$ *según* m_j^b ▷ ecs. 3.4 y 3.31 a 3.34

  solve Subproblema Exacto **minimizing** MK *with* LIMIT $= Z^{\text{sol}}$

  if *Solver status* = USER_OBJ_LIMIT **then**

    return *(-,-,-,*INTERRUPTED*)*

  Guardar $(z^{\text{sol}}_{a,b,l^{\text{act}}},\ z^{\text{lb}}_{a,b,l^{\text{act}}})$

  if $Z^{\text{sol}}_{a,l^{\text{act}}} > z^{\text{sol}}_{a,b,l^{\text{act}}}$ **then**

    $Z^{\text{sol}}_{a,l^{\text{act}}} \leftarrow z^{\text{sol}}_{a,b,l^{\text{act}}}$

    $\beta_{i,j,k,l^{\text{act}}} \leftarrow W_{i,j,k,l^{\text{act}}}$

  $Z^{\text{lb}}_{a,l^{\text{act}}} \leftarrow \min\{Z^{\text{lb}}_{a,l^{\text{act}}},\ \max\{Z^{\text{lb\_approx}}_{a,l^{\text{act}}},\ z^{\text{lb}}_{a,b,l^{\text{act}}}\}\}$

  $L^{\text{F}} = \emptyset$

  $b \leftarrow b + 1$

return $(\beta_{i,j,k,l^{\text{act}}},\ Z^{\text{lb}}_{a,l^{\text{act}}},\ Z^{\text{sol}}_{a,l^{\text{act}}},$ FEASIBLE*)*

Luego, en la *línea 7* del pseudocódigo se realiza un ciclo considerando todos los equipos j de la etapa l^{act}. En caso de que el algoritmo se encuentre en la primera iteración ($b = 1$), deben ser configurados los conjuntos que contienen las ranuras de dichos equipos con las ecuaciones allí referenciadas (*línea 9*). Además, se considera aquí también una salvedad en el caso de que se provenga de una propuesta que considere *cero* ranuras en algún equipo, es decir $m_j^{b-1} = 0$ para algún equipo $j \in J_l^{\text{act}}$. Si esto sucede, el siguiente modelo aproximado debe incluir nuevamente una cantidad de ranuras m_j^0 para dicho equipo, con el fin de evitar que éste quede anulado por el resto de la Iteración Interna en dicha etapa. Seguidamente, se configuran las últimas ranuras de cada equipo como *"slots"* multivaluados k_j^*. De esta forma, la asignación de conjuntos y parámetros queda preparada para resolver el subproblema aproximado y, posteriormente, son guardan los límites $z_{a,b,l^{\text{act}}}^{\text{lb\_approx}}$ y $z_{a,b,l^{\text{act}}}^{\text{sol\_approx}}$. A su vez, en la *línea 13* el parámetro $Z_{a,l^{\text{act}}}^{\text{lb\_approx}}$ almacena la mayor solución posible hallada en todos los subproblemas aproximados para la presente etapa.

Posteriormente, en la *línea 14* del algoritmo se incluye el subconjunto L^{F}, el cual se utiliza para resolver la asignación de tareas a las ranuras K_j pertenecientes a la etapa l^{act} durante el subproblema exacto. Estos *"slots"* habían sido excluidos de la ecuación (3.5) que garantiza la asignación de como máximo un *"batch"* por ranura y la ecuación (3.6) que obliga la elección de *"slots"* consecutivos. Esto se debe a que, en este caso, el total de *"slots"* incluidos en la etapa l^{act} es igual a la cantidad de tareas que deben procesarse en la misma y, por esto, durante la resolución del modelo exacto todos ellos serán seleccionados. Es así que, en lugar de la primera restricción mencionada, se utiliza la ecuación (4.7) y ya no será necesario forzar a que la asignación de ranuras sea de forma consecutiva.

$$\sum_{i \in I_j} W_{i,j,k,l} = 1 \qquad \forall l \in L^{\text{F}}, j \in J_l, k \in K_j \tag{4.7}$$

Luego, en la *línea 15* se configura la cantidad de ranuras para cada equipo obtenidas de las variables binarias de asignación $W_{i,j,k,l^{\text{act}}}$ del subproblema aproximado anterior. Recuérdese que en este subproblema no se incluyen *"slots"* multivaluados para la etapa en análisis l^{act}. Seguidamente, se actualizan nuevamente los conjuntos referidos a las ranuras de estos equipos con las ecuaciones correspondientes y se resuelve el modelo exacto para obtener los valores de $z_{a,b,l^{\text{act}}}^{\text{lb}}$ y $z_{a,b,l^{\text{act}}}^{\text{sol}}$.

Ahora bien, considerando el algoritmo en el contexto de la resolución de un subproblema exacto en cierta Iteración Interna b, el *"Solver"* incluye una cota o límite (LIMIT)

para un posible cese de la iteración en la etapa actual. Este corte de la búsqueda de soluciones puede darse en el caso de que un subproblema exacto encuentre una solución inferior al Z^{sol} guardado previamente. Si esto sucede, significa que la solución para la etapa l^{act} tendrá un valor de MK menor al de la solución encontrada para la l^{sel} seleccionada anteriormente, aún de forma transitoria, para ser fijada. Por ello, no será necesario continuar con la Iteración Interna en dicha etapa l^{act}. La cota del cese de la exploración de propuestas de ranuras para cierta etapa es entonces el valor de Z^{sol} fijado para una etapa explorada previamente dentro del mismo ciclo de la Iteración Principal. En este proyecto se utiliza el resolvedor GUROBI, el cual permite implementar esta cota durante el proceso de ejecución del algoritmo *"Branch and Bound"*. De esta manera, se busca conseguir ahorros significativos de tiempo durante la ejecución de la Iteración Interna.

De este modo, si el corte de la exploración en l^{act} se concreta el algoritmo devuelve la información para dicha etapa como (-,-,-,INTERRUPTED) y se continúa con la siguiente etapa a explorar dentro del subconjunto L^{P}, si existe. En caso de que este mecanismo de corte no se halla activado, se guardan las cotas del problema en la *línea 22* del algoritmo. Además, como se desarrolla en el condicional de la *línea 23*, si el valor de la función objetivo para dicha etapa analizada es menor al valor guardado hasta el momento en $Z^{\text{sol}}_{a,l^{\text{act}}}$, este último es reemplazado y se guardan las variables binarias de asignación de la solución en el parámetro $\beta_{i,j,k,l^{\text{act}}}$.

El paso siguiente será almacenar el valor de la mejor solución posible para la etapa en $Z^{\text{lb}}_{a,l^{\text{act}}}$ como se muestra en la *línea 26*. Este valor, considera el mínimo entre todas estas cotas guardadas anteriormente y el máximo entre la mejor solución del subproblema exacto actual y la cota inferior de los subproblemas aproximados. Este último término se incluye debido a que, si los modelos aproximados con ranuras multivaluadas alcanzan la cota inferior $Z^{\text{lb\_approx}}_{a,l^{\text{act}}}$, entonces no será posible que una solución exacta con un valor de función objetivo menor a dicha cota. Para finalizar la iteración en curso solo queda excluir a la etapa en análisis del subconjunto L^{F} e incrementar el índice de la Iteración Interna como $b = b + 1$ mediante la *línea 28*.

El ciclo anteriormente descripto se repite y su conclusión se da cuando no se cumple alguna de las dos condiciones definidas en la iteración *"while"* al inicio del algoritmo en la *línea 6*. A su vez, es necesario establecer un límite de tiempo para la resolución de cada subproblema, denotado T, y otro para el total de la Iteración Interna en una misma etapa, referenciado como t. Al concluir la prueba de diferentes propuestas de cantidades de *"slots"* para cierta etapa l^{act} y no habiéndose activado el corte de operación del *"Solver"* (status =FEASIBLE), el algoritmo devuelve la mejor solución exacta hallada, las

variables de asignación de ranuras correspondientes y el límite de la mejor solución posible en esta instancia de construcción del *"schedule"*.

4.3. Iteración Principal: Identificar y Fijar la Etapa Cuello de Botella

La Iteración Principal, identificada con el índice a, es la encargada de seleccionar y fijar una a una las etapas que aún no hayan sido guardadas y, por lo tanto, pertenecen al conjunto L^{P}. De esta forma, en cada ciclo concluido una nueva etapa l^{sel} será elegida para fijar sus asignaciones mediante el parámetro $\beta_{i,j,k,l^{\mathrm{sel}}}$. Al cabo de $a = |L|$ Iteraciones Principales, se obtendrá una solución completa al problema con valor de función objetivo representado en Z^{sol} y una cota inferior para la mejor solución posible Z^{lb}.

Con el fin de exponer de forma sencilla la idea de funcionamiento de la Iteración Principal, a continuación se presenta un ejemplo en el cual se construye una agenda de producción para una planta integrada por 4 equipos, distribuidos en 2 etapas, que deben procesar un total de 8 *"batches"*. Inicialmente, a instancias de la primera Iteración Principal $a = 1$, en la Figura 4.4 se muestra la mejor solución hallada por la Iteración Interna (subproblema exacto) para la etapa l_1 de la instalación. Para llegar a la misma, fueron tenidos en cuenta diversos parámetros de cada lote y equipo, como son el $ETC_{i,l}$, $EST_{i,l}$, EST_j y ETC_j. Además, como fue descrito en el Apartado anterior, en el desarrollo de la Iteración Interna también se incluye una estimación para la etapa l_2, la cual se analiza con una cantidad de ranuras iniciales definida al inicio del ciclo para todas las etapas pendientes (en este caso, $m_j^0 = 3$). Al concluir la Iteración Interna en esta etapa, la Iteración principal guarda como Z^{sol}_{1,l_1} la cota para la mejor solución parcial correspondiente a la etapa $l^{\mathrm{act}} = l_1$ y el valor para la mejor solución posible Z^{lb}_{1,l_1} hallada hasta el momento.

Continuando con el primer ciclo $a = 1$, el procedimiento se lleva a cabo de forma similar para la etapa 2 (ahora $l^{\mathrm{act}} = l_2$) y el mismo es ilustrado en la Figura 4.5. La cota para la mejor solución de la etapa es hallada y representada en Z^{sol}_{1,l_2} y su límite inferior correspondiente se guarda como Z^{lb}_{1,l_2}. Como puede observarse se verifica la condición $Z^{\mathrm{sol}}_{1,l_1} > Z^{\mathrm{sol}}_{1,l_2}$, con lo cual, exploradas todas las etapas pendientes el algoritmo elige fijar la solución para la peor de ellas (en términos del valor de la función objetivo) y queda que $l^{\mathrm{sel}} = l_1$ con $Z^{\mathrm{sol}} = Z^{\mathrm{sol}}_{1,l_1}$. De este modo, la primera iteración $a = 1$ finaliza guardando la solución obtenida para l_1 y fijando las decisiones de asignación y secuenciación correspondientes en β_{i,j,k,l_1}.

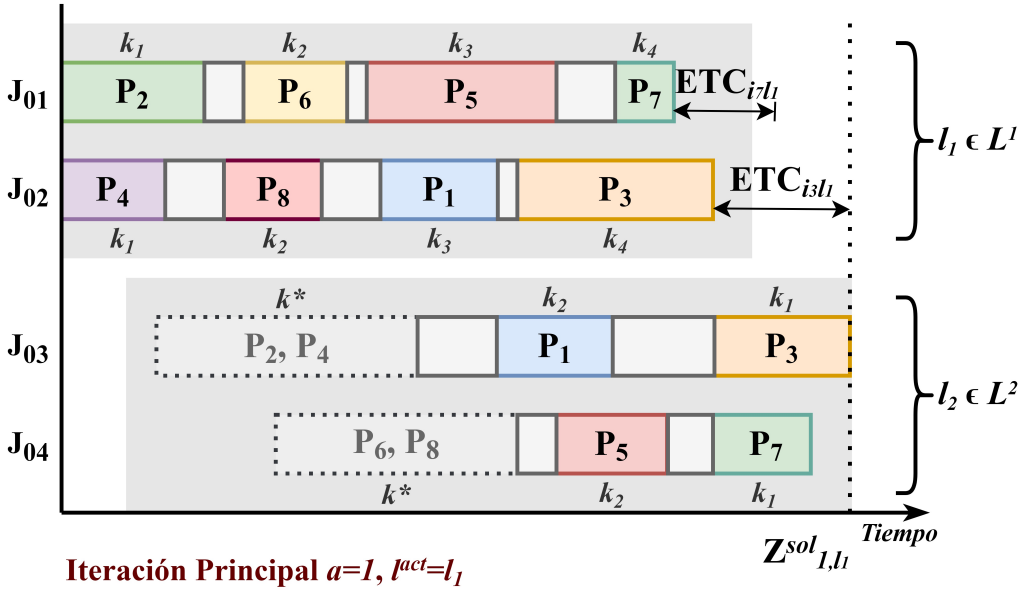

Figura 4.4: Ejemplo Iteración Principal $a = 1$, $l^{\text{act}} = l_1$: construcción progresiva del *"schedule"*

Fuente: Elaboración propia

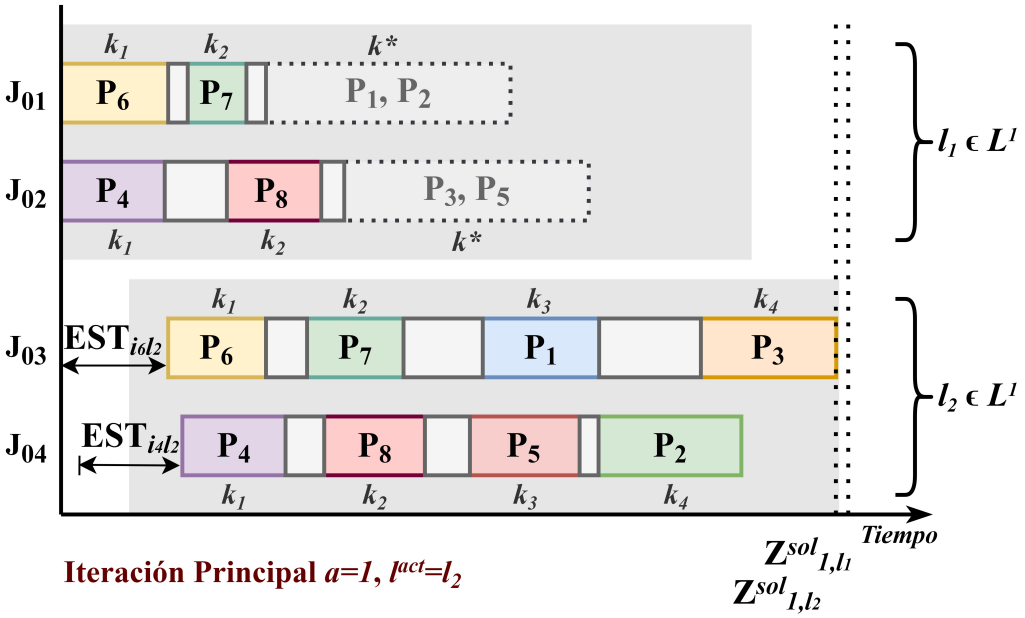

Figura 4.5: Ejemplo Iteración Principal $a = 1$, $l^{\text{act}} = l_2$: construcción progresiva del *"schedule"*

Fuente: Elaboración propia

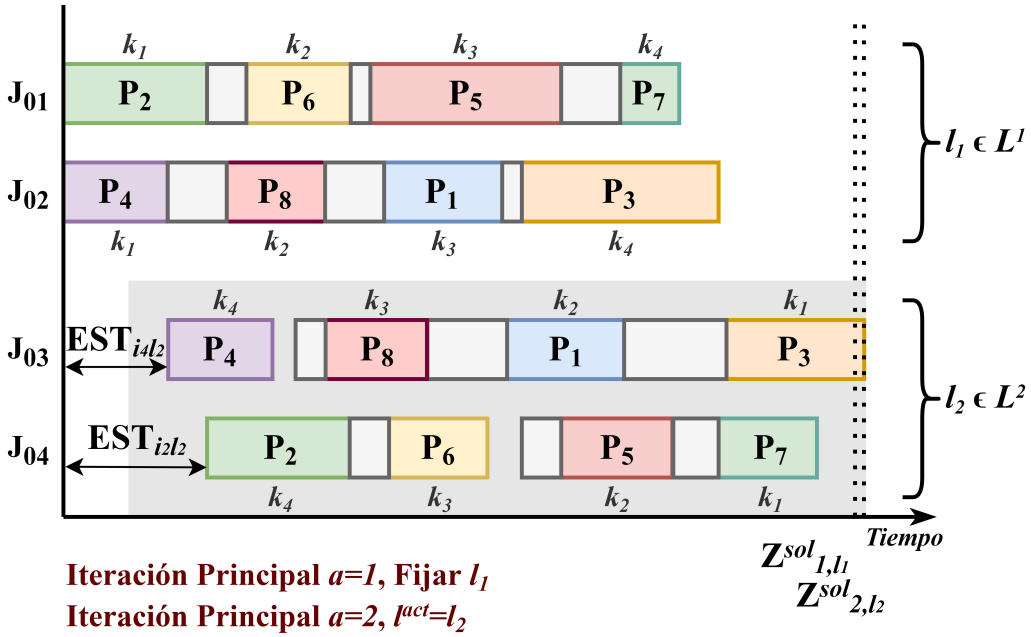

Figura 4.6: Ejemplo Iteración Principal $a = 2$, $l^{\text{act}} = l_2$: construcción progresiva del *"schedule"*

Fuente: Elaboración propia

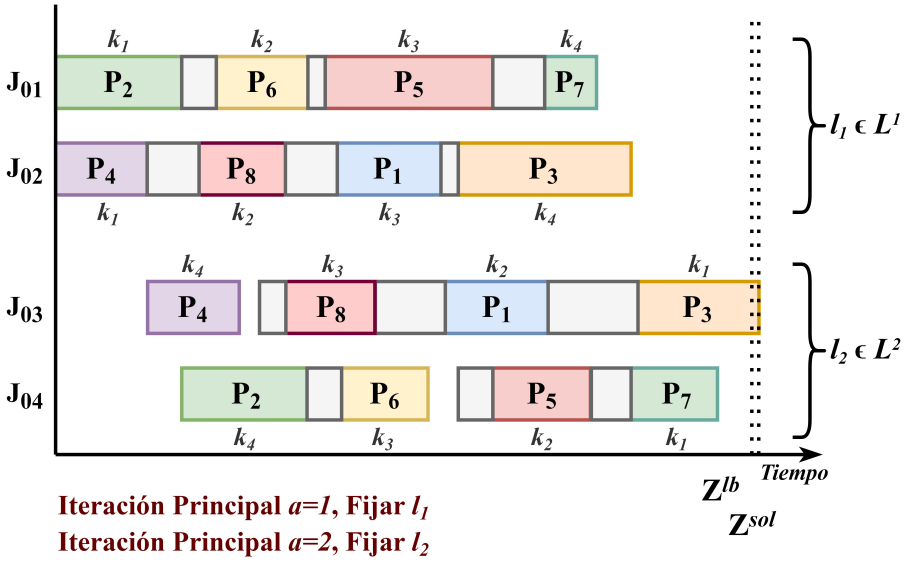

Figura 4.7: Ejemplo Iteración Principal: mejor *"schedule"* obtenido

Fuente: Elaboración propia

Posteriormente, comienza la segunda Iteración Principal $a = 2$ y la Iteración Interna devuelve la mejor solución hallada para la etapa $l^{\text{act}} = l_2 \in L^{\text{P}}$. La misma es ilustrada en la Figura 4.6. En este momento, se obtiene la cota de la mejor solución hallada para esta etapa y el valor de la función objetivo para la solución construida de forma progresiva evoluciona a $Z^{\text{sol}} = Z^{\text{sol}}_{2,l_2}$. Nuevamente, todas las etapas pendientes han sido analizadas (quedando que $l^{\text{sel}} = l_2$) y se guarda la solución correspondiente a la etapa l_2 en β_{i,j,k,l_2}.

De esta forma, se ha construido una solución completa para todas las etapas de la instalación y, como puede verse en la Figura 4.7, el valor para el *"makespan"* (MK) corresponde al $Z^{\text{sol}} = Z^{\text{sol}}_{2,l_2}$ obtenido en la última Iteración Principal ($a = 2$). Asimismo, puede calcularse el margen pendiente para probar la optimalidad de la solución ($GAP\,\%$) con el valor de la mejor solución posible obtenido para la etapa fijada en la primera Iteración Principal $a = 1$, es decir, $Z^{\text{lb}} = Z^{\text{lb}}_{1,l_1}$.

A continuación, con el fin de explicar el proceso de forma más genérica se presenta el diagrama de flujo de la Iteración Principal en la Figura 4.8. De aquí se puede ver que, inicialmente, todas las etapas están pendientes y la iteración se repetirá mientras existan etapas en L^{P}. Al comienzo, se asigna 0 al parámetro Z^{sol} y se setea l^{sel} como una etapa no especificada. Luego, para cada etapa pendiente que pertenece al subconjunto L^{P} se realiza la Iteración Interna y se obtienen los límites $Z^{\text{sol}}_{a,l}$, $Z^{\text{lb}}_{a,l}$ y los parámetros $\beta_{i,j,k,l}$ de cada una de ellas. En caso de que cierta etapa empeore el valor de la función objetivo guardado hasta el momento ($Z^{\text{sol}}_{a,l^{\text{act}}} > Z^{\text{sol}}$), dicha etapa se convierte en la etapa seleccionada para fijar (l^{sel}) y se almacenan sus respectivas variables de asignación de ranuras. A continuación, se analiza la etapa de orden siguiente repitiendo el procedimiento de manera similar mientras falten explorar elementos del subconjunto L^{P}.

Al finalizar el primer ciclo, la etapa elegida como l^{sel} es removida del subconjunto L^{P} y quedan fijas sus decisiones de asignación y cantidad de ranuras para cada equipo, tal como se introduce en las ecuaciones (4.8) y (4.9) respectivamente.

$$\beta_{i,j,k,l} = W_{i,j,k,l} \qquad \forall i \in I_l, j \in J_l, k \in K_j, l \in L \setminus L^{\text{P}} \tag{4.8}$$

$$m_j = \sum_{k \in K_j} \beta_{i,j,k,l} \qquad \forall i \in I, j \in J_l, l \in L \setminus L^{\text{P}} \tag{4.9}$$

A continuación, comienza una nueva Iteración Principal (ahora $a = a+1$) y se repite el ciclo hasta que todas las etapas hayan sido fijadas, obteniendo el *"schedule"* completo

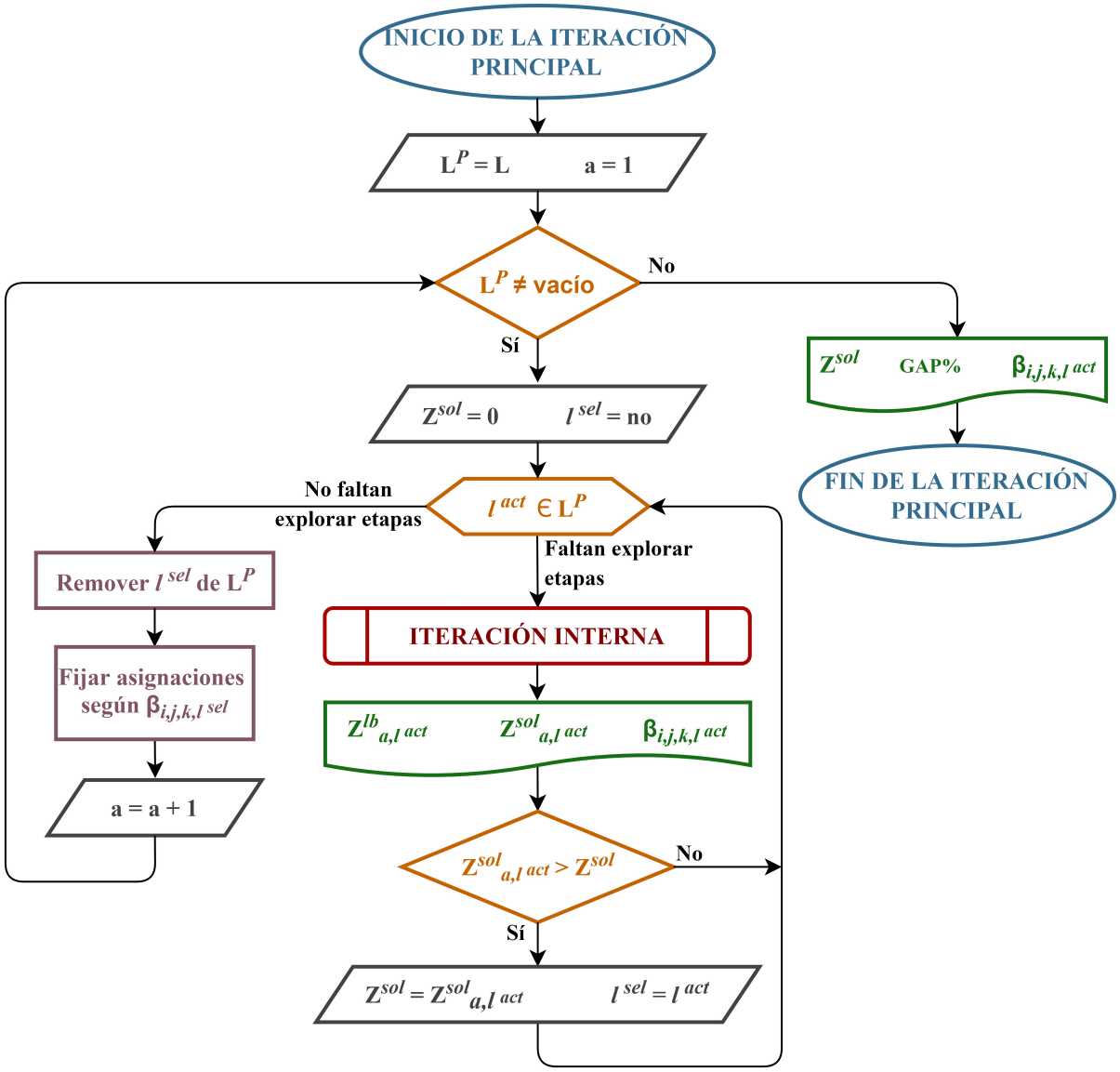

Figura 4.8: Diagrama de Flujo de la Iteración Principal

Fuente: Elaboración propia

con el equipo asignado y la secuencia en que es procesada cada tarea (i, l). Además, como fue visto anteriormente, cada tarea tiene asociado sus tiempos de comienzo ST^1_{il}, y finalización, CT^1_{il}. Respecto al *"makespan"*, el mismo se ve reflejado en el parámetro Z^{sol} y puede ser calculado su margen porcentual con respecto a la mejor solución posible, o margen relativo para probar la optimalidad de la solución $(GAP\,\%)$.

4.3.1. Estructura algorítmica para la construcción del *"schedule"*

Con el fin de brindar mayor detalle respecto al procedimiento propuesto para la Iteración Principal, se presenta el pseudocódigo del Algoritmo 4.2. Para comenzar, en las *líneas 1* y *2*, se parte de considerar a todo el conjunto de etapas como pendientes ($L = L^{\mathrm{P}}$) y se inicia el índice del proceso en $a = 1$. Luego, se realizará una iteración mientras que el conjunto de etapas pendientes no se encuentre vacío, es decir, se repetirá el proceso una cantidad $a = |L|$ veces. Al inicio de cada repetición será necesario actualizar los parámetros $EST_{i,l}$, $ETC_{i,l}$, EST_j y ETC_j utilizados en las restricciones de ajuste del modelo. Los mismos, luego de completar cada ciclo podrán aprovechar la información disponible de una nueva etapa fijada para acotar la región factible con mayor precisión.

Las definiciones para los parámetros $EST_{i,l}$ y $ETC_{i,l}$ fueron desarrolladas en el capítulo previo mediante las ecuaciones (3.25) y (3.26). Las mismas serán utilizadas únicamente cuando ninguna etapa ha sido fijada, pero a medida que avancen las iteraciones su actualización se realizará mediante las ecuaciones (4.10) y (4.11). Estas últimas consideran que los índices j^{s}, i^{p} e i^{n} representan el equipo seleccionado, el *"batch"* previo y el *"batch"* siguiente en dicho equipo, respectivamente, para cada tarea (i, l). Como salvedad, los parámetros con índices i^{p}, i^{n}, $(l-1)$ o $(l+1)$ no definidos toman el valor *cero* y, además, se utiliza el parámetro $\widetilde{pt}_{i,l}$ definido en la ecuación (4.12) para representar el menor tiempo estimado para procesar la tarea (i, l). Este parámetro será equivalente al tiempo de procesamiento exacto si $l \notin L^{\mathrm{P}}$ y j^{s} está definido o bien, será una estimación del menor tiempo de procesamiento para los equipos disponibles.

Para la actualización del parámetro $EST_{i,l}$ se consideran dos casos. El primero de ellos, en el cual la etapa l está dentro de las pendientes, el valor para el *"Earliest Start Time"* de la tarea (i, l) es este mismo considerado para la etapa anterior $(l-1)$ más el menor tiempo $\widetilde{pt}_{i,(l-1)}$ para procesar dicho *"batch"* en algún equipo de la etapa precedente. Para el segundo caso, cuando la etapa l ya fue fijada en una Iteración Principal previa ($l \notin L^{\mathrm{P}}$), debe considerarse el mínimo entre el término analizado para el caso anterior y la estimación para el $EST_{i^{\mathrm{p}},l}$ de la tarea previa en el tiempo (i^{p}, l) dentro del mismo equipo j^{s} en el cual se procesa el *"batch"* i, más el tiempo de procesamiento para dicho lote $pt_{i^{\mathrm{p}},j^{\mathrm{s}}}$ y sumado al tiempo de transición entre ambas tareas $\tau_{i^{\mathrm{p}},i,j^{\mathrm{s}}}$.

Por el lado de la actualización del parámetro $ETC_{i,l}$, el análisis puede hacerse de forma similar considerando la etapa siguiente a la analizada $(l+1)$ y los parámetros del

tiempo de procesamiento y tiempo de *"changeover"* para el lote i, el *"batch"* siguiente i^{n} y el equipo seleccionado para procesar dichas tareas j^{s}. Por otra parte, los parámetros EST_j y ETC_j, dado que dependen directamente de sus pares para las combinaciones *"batch"*-etapa (i, l), bastará con recalcularlos con su definición inicial de las ecuaciones (3.29) y (3.30).

$$EST_{i,l} = \begin{cases} EST_{i,(l-1)} + \widetilde{pt}_{i,(l-1)} & \forall i \in I, l \in L^{\mathrm{P}} \\ max\{EST_{i,(l-1)} + \widetilde{pt}_{i,(l-1)}, EST_{i^{\mathrm{P}},l} + pt_{i^{\mathrm{P}},j^{\mathrm{s}}} + \tau_{i^{\mathrm{P}},i,j^{\mathrm{s}}}\} & \forall i \in I, l \in L \setminus L^{\mathrm{P}} \end{cases} \tag{4.10}$$

$$ETC_{i,l} = \begin{cases} ETC_{i,(l+1)} + \widetilde{pt}_{i,(l+1)} & \forall i \in I, l \in L^{\mathrm{P}} \\ max\{ETC_{i,(l+1)} + \widetilde{pt}_{i,(l+1)}, ETC_{i^{\mathrm{n}},l} + pt_{i^{\mathrm{P}},j^{\mathrm{s}}} + \tau_{i,i^{\mathrm{n}},j^{\mathrm{s}}}\} & \forall i \in I, l \in L \setminus L^{\mathrm{P}} \end{cases} \tag{4.11}$$

$$\widetilde{pt}_{i,l} = \begin{cases} min_{j \in J_l}\{pt_{i,j}\} & \forall i \in I, l \in L^{\mathrm{P}} \\ pt_{i,j^{\mathrm{s}}} & \forall i \in I, l \in L \setminus L^{\mathrm{P}} \end{cases} \tag{4.12}$$

Seguidamente, cada Iteración Principal a comienza con el parámetro Z^{sol} en *cero* y, a su vez, l^{sel} se configura como no especificada. Posteriormente, la iteración de la *líneas 7* es utilizada para recorrer todas aquellas etapas pertenecientes al subconjunto L^{P}. Para ello se asigna un orden de exploración a cada etapa (Ord_l), el cual es explicado con mayor detalle en el siguiente Apartado 4.3.2. Luego, en la *línea 8*, se definen los esquemas de secuenciación de ranuras para cada una de las etapas en el contexto de la primera Iteración Principal ($a = 1$), es decir, cuando aún no fue fijada ninguna etapa de la instalación. La lógica empleada en este paso fue explicada con anterioridad en el Apartado 4.2. Posteriormente, en la *línea 11* se efectúa la Iteración Interna para la etapa actual.

Algoritmo 4.2: Pseudocódigo de la Iteración Principal

Input: Orden de exploración de etapas Ord_l

Output: Mejor solución encontrada para el problema completo

1 $L^{\mathrm{P}} \leftarrow L$
2 $a \leftarrow 1$
3 **while** $L^{\mathrm{P}} \neq \emptyset$ **do**
4 *Actualizar* $(EST_{i,l},\ ETC_{i,l},\ EST_j,\ ETC_j)$ ▷ ecs. 4.10, 4.11, 3.29 y 3.30
5 $Z^{\mathrm{sol}} \leftarrow 0$
6 $l^{\mathrm{sel}} \leftarrow none$
7 **loop** $l^{\mathrm{act}} \in L^{\mathrm{P}}$ ***ordered by*** Ord_l
8 **if** $a = 1$ **then**
9 $L^1 = \{l \in L : l \leq l^{\mathrm{act}}\}$
10 $L^2 = \{l \in L : l > l^{\mathrm{act}}\}$
 /* COMIENZO DE LA ITERACIÓN INTERNA */
11 $(\beta_{i,j,k,l^{\mathrm{act}}}, Z^{\mathrm{lb}}_{a,l^{\mathrm{act}}}, Z^{\mathrm{sol}}_{a,l^{\mathrm{act}}}, status) \leftarrow$ **Iteración Interna** (l^{act})
 /* FINAL DE LA ITERACIÓN INTERNA */
12 **if** $status \neq$ INTERRUPTED **then**
13 **if** $l^{\mathrm{sel}} = none$ ***or*** $Z^{\mathrm{sol}}_{a,l^{\mathrm{act}}} > Z^{\mathrm{sol}}$ **then**
14 $l^{\mathrm{sel}} \leftarrow l^{\mathrm{act}}$
15 $Z^{\mathrm{sol}} \leftarrow Z^{\mathrm{sol}}_{a,l^{\mathrm{act}}}$
 /* CONFIGURACIÓN DE LA ETAPA A FIJAR */
16 $L^{\mathrm{P}} = L^{\mathrm{P}} \setminus \{l^{\mathrm{sel}}\}$
17 **loop** $j \in J_{l^{\mathrm{sel}}}$
18 *Actualizar* $(m_j,\ K_j,\ \overleftarrow{K}^k_j$ y $\overrightarrow{K}^k_j)$ *según* $\beta_{i,j,k,l^{\mathrm{sel}}}$ ▷ ecs. 4.8, 3.4 y 3.31 a 3.34
19 $k^*_j \leftarrow none$
20 **if** $a = 1$ **then**
21 $l^{\mathrm{critic}} \leftarrow l^{\mathrm{sel}}$
22 $L^1 = \{l \in L : l \leq l^{\mathrm{critic}}\}$
23 $L^2 = \{l \in L : l > l^{\mathrm{critic}}\}$
24 $Z^{\mathrm{lb}} \leftarrow Z^{\mathrm{lb}}_{a,l^{\mathrm{critic}}}$
25 $a \leftarrow a + 1$
26 **return** $(MK,\ GAP\,\%,\ \beta_{i,j,k,l},\ ST^1_{i,l},\ CT^1_{i,l})$

A continuación, en la *línea 12* del pseudocódigo, se introduce una condición para el caso en que el *"Solver"* no haya interrumpido la exploración de soluciones en la etapa l^{act}, es decir, *status* $\neq$ INTERRUPTED. En dicho caso, a través de la Iteración Interna se habrán obtenido las decisiones de asignación de ranuras $\beta_{i,j,k,l^{\text{act}}}$ y los límites de la solución $Z^{\text{sol}}_{a,l^{\text{act}}}$ y $Z^{\text{lb}}_{a,l^{\text{act}}}$. A su vez, como se introduce en la *línea 13*, en el caso de que aún no haya sido seleccionada ninguna etapa ($l^{\text{sel}} = none$) o el valor de la función objetivo obtenido sea mayor al guardado hasta el momento en Z^{sol}, la etapa actual l^{act} se convierte en la etapa seleccionada l^{sel}. Además, se almacena el valor estimado para el *"makespan"* disponible como $Z^{\text{sol}}_{a,l^{\text{act}}}$ en el parámetro interno del algoritmo Z^{sol}. De esta forma, en cada Iteración Principal se selecciona una etapa a fijar posteriormente (l^{sel}), entre todas las $l \in L^{\text{P}}$ en base al valor que tome el parámetro $Z^{\text{sol}}_{a,l}$. De las soluciones obtenidas en cada etapa pendiente, el algoritmo selecciona aquella con peor valor para el *"makespan"* ya que representa la etapa limitante o cuello de botella siguiente a considerar.

Una vez finalizado el recorrido de todas las etapas pendientes, como se muestra en la *línea 16* del algoritmo, la etapa seleccionada l^{sel} es extraída del conjunto L^{P}. Posteriormente, en la *línea 17*, se asigna el número de *"slots"* m_j seleccionado y se realizan las configuraciones de los conjuntos de ranuras (K_j, $\overleftarrow{K}^k_j$, $\overrightarrow{K}^k_j$) para los equipos pertenecientes a dicha etapa. A su vez, dado que para las etapas fijadas ($l \in L \setminus L^{\text{P}}$) no se considera la existencia de ranuras multivaluadas, la *línea 19* anula las mismas.

Seguidamente, en la *línea 20* se aplican los esquemas de secuenciación temporal correspondientes para las iteraciones siguientes ($a > 1$). Para ello, se introduce el índice l^{critic}, el cual indica la etapa fijada (l^{sel}) en el primer ciclo de la Iteración Principal. A su vez, debe almacenarse el valor de la cota inferior de la solución completa del problema, la cual se define como el valor $Z^{\text{lb}}_{1,l^{\text{critic}}}$. Para concluir cada Iteración Principal, el índice correspondiente debe tomando el valor $a = a + 1$, tal como se escribe en la *línea 25*. Finalmente, luego de $a = |L|$ Iteraciones Principales concluidas, el proceso concluye y se obtiene la construcción del *"schedule"* completo, el valor para el *"makespan"*, el margen porcentual pendiente para probar la optimalidad de la solución ($GAP\,\%$) y los tiempos de inicio y finalización asociados a cada tarea (i, l) que debe ser programada.

4.3.2. Orden de exploración de etapas

Con el fin de establecer un orden de prioridad para la exploración de las etapas se ha diseñado un procedimiento que, en un corto período de tiempo, busca reflejar de forma aproximada la dificultad de programación de cada una de ellas. Este orden, identificado

en el pseudocódigo de la Algoritmo 4.2 como Ord_l, es respetado al momento de explorar las etapas pertenecientes a L^{P} en cada Iteración Principal a. Es así que, comenzando por la etapa con $Ord_l = 1$, la Iteración Principal recorre todas las etapas de forma consecutiva y, en caso de que aquella en determinado orden ya haya sido programada (esto significa que $l \notin L^{\mathrm{P}}$) la misma es omitida para continuar con la de prioridad siguiente.

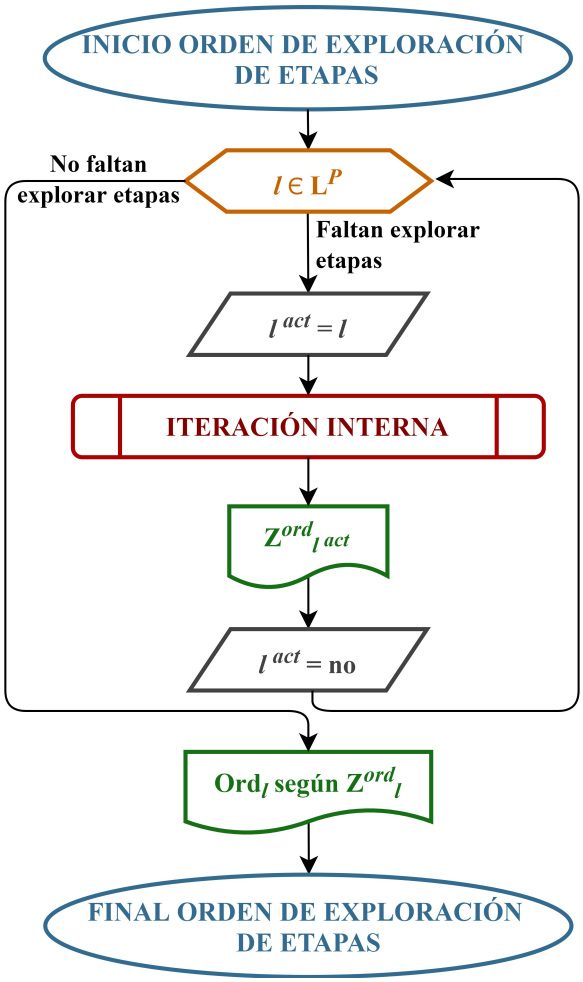

Figura 4.9: Diagrama de Flujo Orden de Exploración de Etapas

Fuente: Elaboración propia

Como se muestra en el diagrama de flujo de la Figura 4.9, para establecer el parámetro Ord_l para cada etapa se lleva a cabo un proceso que involucra un ciclo de la Iteración Interna. Al comienzo de este proceso se establecen el límite de tiempo T^{ini} para la resolución de cada subproblema aproximado o exacto, y un máximo de un solo ciclo de la Iteración Interna, es decir $nb^{\mathrm{max\_ini}} = 1$. En este caso, el parámetro T^{ini} tomará un valor pequeño de forma que el impacto de la implementación de este módulo no deteriore el desempeño computacional de la metodología propuesta. Por otra parte, cabe recordar que

para la resolución del primer modelo aproximado de cada etapa será necesario partir de una propuesta inicial de *"slots"* m_j^0.

Al finalizar el proceso de exploración, se establece el orden de prioridad según las soluciones del subproblema exacto almacenadas en el parámetro Z_l^{ord}. De esta forma, para determinado tiempo de ejecución de este módulo, se asignará como etapa $l \in L$ con $Ord_l = 1$ a aquella que presente la peor solución en términos del valor de la función objetivo. La implementación de esta regla para el recorrido de las etapas cobra sentido ante la inclusión de cortes durante el proceso de construcción de la solución, lo cual fue presentado anteriormente en este mismo capítulo en el Algoritmo 4.1.

Capítulo 5

RESOLUCIÓN DEL CASO DE ESTUDIO

5.1. Presentación de Modelos Alternativos

Con el fin de llevar a cabo una valoración del desempeño y resultados de la metodología matemática-algorítmica propuesta se evaluarán, además del citado método, dos variantes de la formulación de *"time-slots"* desarrollada por Pinto y Grossmann (1995). Ambas variantes incluyen solo ranuras convencionales y difieren entre sí por la cantidad de *"slots"* que son definidos para cada equipo de procesamiento j. Estos modelos tienen menor complejidad de desarrollo y, por lo tanto, su implementación para resolver el caso de estudio resulta más directa. En contraparte con la estrategia matemática-algorítmica desarrollada en los Capítulos 3 y 4, estos modelos no exploran subproblemas aproximados o parciales, sino que consideran el problema completo al momento de optimizar la función objetivo.

El primero de ellos, denominado FULL, considera para cada equipo una cantidad de ranuras igual al máximo número de *"batches"* que pueden ser procesados en el mismo, acorde a lo expresado en la ecuación (5.1) y considerando la definición de $k_j^{\max}$ dada en la ecuación (3.2). Por otro lado, la formulación llamada FULL FIXED utiliza la configuración de ranuras hallada para la mejor solución obtenida con la metodología matemática-algorítmica propuesta, tal como se muestra en la ecuación (5.2).

$$m_j = |K_j| = k_j^{\max} \qquad \forall j \in J \tag{5.1}$$

$$m_j = |K_j| = \sum_{i \in I_j} \sum_{k \in K_j} \beta_{i,j,k,l} \qquad \forall j \in J_l, l \in L \tag{5.2}$$

Para la construcción de ambos modelos completos se utilizan las ecuaciones desarrolladas para la descripción de la metodología propuesta, con la inclusión de algunas salvedades. La primera de ellas es que deberá considerarse a todas las etapas de la planta como incluidas en el conjunto de etapas pendientes, es decir $L^{\mathrm{P}} = L$. Luego la formulación FULL incluye las restricciones de asignación de ranuras convencionales dada por las ecuaciones (3.1), (3.5) y (3.6). En este caso, el número de *"slots"* considerados en cada etapa l no se define estrictamente en concordancia con la cantidad de tareas (i, l) que deben efectuarse en la misma. En su lugar, se asigna $L^{\mathrm{F}} = \emptyset$ de modo que en todas las etapas de procesamiento puedan existir equipos con ranuras adicionales, las cuales serán utilizadas o no para obtener las posibles soluciones del problema. Por otro lado, en el modelo FULL FIXED se establece que la cantidad de ranuras por equipo esté definida con precisión en todas las etapas de la instalación. En consecuencia, las restricciones de asignación se consideran determinando previamente que se cumple la igualdad $L^{\mathrm{F}} = L$ y, de esta manera, se incluyen en el modelo las ecuaciones (3.1) y (4.7).

En cuanto a los siguientes grupos de restricciones, se añaden en ambas formulaciones de forma similar. Por el lado de las ecuaciones de procesamiento y sincronización de tiempos, deben ser incluidas las ecuaciones (3.8), (3.9), (3.10) y (3.11). En cuanto a las restricciones de secuenciación, será necesario procesar las etapas en el orden requerido por cada *"batch"* con la ecuación (3.12). Además, dado que se desea encontrar la solución analizando todas las etapas en simultáneo, se aplica un único esquema de secuenciación temporal (L^1) con la ecuación (3.17). De este contexto, también se incorpora la definición de la variable $CO_{j,k}$ para las etapas que poseen tiempos de transición dependientes de la secuencia de procesamiento, incluyendo para ello la ecuación (3.19). La definición del *"makespan"*, utilizado en la función objetivo a minimizar, se incorpora mediante las ecuaciones (3.23) y (3.24). Por último, en cuanto a las restricciones de ajuste de la región factible, deberán incluirse en ambas formulaciones la ecuación (3.27), (3.28), (3.35), (3.36) y (3.37). A modo de síntesis se presenta el Anexo C, en el cual se resumen las restricciones a considerar en la metodología matemática-algorítmica propuesta y en las formulaciones alternativas FULL y FULL FIXED.

5.2. Definición de Parámetros y Diferentes Instancias del Problema

Con el objeto de comparar la formulación matemática-algorítmica propuesta con respecto a las formulaciones alternativas presentadas en la sección anterior, se definen seis instancias de evaluación del caso de estudio. Se busca, de esta forma, incrementar gradualmente la complejidad del problema abordado y realizar un análisis del impacto que esto tiene en la performance computacional y los resultados obtenidos. Para ello, se determina en primer término que las instancias a evaluar serán aquellas teniendo en cuenta los primeros 5, 10, 15, 20, 25 y, finalmente, el total de 30 lotes del problema.

Para el caso de la metodología desarrollada en este proyecto, cada una de estas instancias es evaluada con una cantidad de ranuras iniciales por equipo de $m_j^0 = 2$ y $m_j^0 = 3$. Recuérdese que esto es posible dada la inclusión de *"slots"* multivaluados en este modelo. Además, debe recordarse que estos valores son también los utilizados para considerar los equipos de las etapas pendientes L^{P} ubicadas aguas arriba y aguas abajo de la etapa analizada en cierta Iteración Interna. A su vez, al comenzar con la evaluación de cada instancia, la estrategia de resolución desarrollada requiere definir parámetros para la ejecución del módulo encargado de establecer el orden de exploración de etapas Ord_l. La duración de tiempo máxima para resolver cada subproblema en esta instancia se establece, $T^{\mathrm{ini}} = 100[s]$ y la cantidad de ciclos permitidos en esta Iteracion Interna, como ya fue explicado en la Apartado 4.3.2, será de $nb^{\mathrm{max\_ini}} = 1$.

Por otra parte, también se definen los valores para los límites de tiempo en la ejecución de la metodología propuesta y las formulaciones completas adicionales. Para el caso de la estrategia de resolución propuesta se define la duración máxima para la Iteración Interna en cierta etapa $t = 1[hrs]$ y el límite de tiempo para la resolución de cada subproblema con valores de $T = 300,\ 600\ y\ 1200[s]$. Así también, se establece la cantidad máxima de ciclos para el proceso de Iteración Interna en cada etapa como, $nb^{\mathrm{max}} = 10$. Por el lado de las formulaciones FULL y FULL FIXED, se las evalúa con un tiempo límite establecido en $T = 5[hrs]$.

5.3. Exposición y Discusión de Resultados

Las formulaciones desarrolladas fueron implementadas en GAMS (*"General Algebraic Modeling System"*), utilizándose el resolvedor GUROBI 7.5 para la resolución de los subproblemas de la Iteración Interna, o bien, las formulaciones alternativas completas. Todas las evaluaciones computacionales se realizaron en un equipo PC genérico con procesador Intel Core i7 3.2 GHz y 16 GB de RAM. En los siguientes apartados, se presentan los resultados obtenidos en la investigación.

5.3.1. Soluciones para los diferentes instancias del problema

En la Tabla 5.1 se presenta, en primer lugar, los resultados obtenidos mediante la metodología matemática-algorítmica propuesta. En ella se muestra el valor de la función objetivo, la diferencia porcentual entre este valor y la mejor solución posible (*"Relative GAP, GAP %*), con la mejor solución posible y el tiempo necesario para arribar a esta solución considerando los valores de los parámetros T y m_j^0 en cada instancia. Para cada cantidad de órdenes procesadas se resalta en negrita la opción con el mejor resultado obtenido teniendo en cuenta el *"makespan"* y tiempo total de resolución, en ese orden, correspondientes con determinados valores de T y m_j^0. Nótese que, para el caso particular de 10 *"batches"*, considerar $T = 300[s]$ es suficiente para resolver a optimalidad todos los subproblemas planteados hasta llegar a construir la solución completa del problema. Por ello, esta instancia no es resuelta para los límites de T mayores.

Ahora bien, considerando un cierto límite T, al ir incrementando el tamaño del problema puede observarse una mejoría en los resultados obtenidos para los modelos con una menor cantidad de ranuras iniciales m_j^0 definidas por equipo. Para el caso de $T = 300[s]$ este cambio ocurre en la instancia de 20 *"batches"*, para $T = 600[s]$ se da en el caso de 30 órdenes y para $T = 1{,}200[s]$ esto ocurre para una cantidad de 25 *"batches"*. Esto se debe a que considerar una menor cantidad de *"slots"* iniciales para los equipos de las etapas L^{P} impacta en una cantidad menor de variables discretas del modelo. Para cada tamaño del problema, puede encontrarse un balance entre la precisión, variando la cantidad de ranuras en las etapas pendientes aguas arriba y aguas abajo de la etapa analizada, y el tiempo destinado a resolver cada subproblema, con el fin de obtener una solución de calidad aceptable en un tiempo de computo razonable.

Cantidad de órdenes	m_j^0	T = 300[s] Makespan [hrs]	GAP %	Tiempo [s]	T = 600[s] Makespan [hrs]	GAP %	Tiempo [s]	T = 1.200[s] Makespan [hrs]	GAP %	Tiempo [s]
10	**2**	11,86	3,75 %	32	-	-	-	-	-	-
	3	**11,61**	**11,65 %**	**129**	-	-	-	-	-	-
15	**2**	15,87	16,47 %	2.108	15,87	16,47 %	2.551	15,87	16,47 %	2.559
	3	**14,27**	**9,97 %**	**3.813**	14,52	9,68 %	6.755	14,99	12,56 %	11.662
20	**2**	18,75	8,16 %	1.337	18,85	8,65 %	1.910	18,57	7,29 %	4.355
	3	20,10	15,60 %	2.856	**18,50**	**6,91 %**	**8.789**	18,50	6,91 %	11.162
25	**2**	23,19	20,50 %	3.271	24,64	25,07 %	6.253	**22,05**	**16,12 %**	**9.224**
	3	23,50	21,57 %	1.582	24,21	23,85 %	2.439	23,83	22,65 %	4.554
30	**2**	27,03	18,81 %	2.431	**25,38**	**13,55 %**	**5.259**	25,38	13,55 %	8.821
	3	30,55	29,02 %	3.073	29,79	27,23 %	4.705	27,14	19,13 %	11.063

Tabla 5.1: Soluciones para las diferentes instancias del problema obtenidas con la metodología propuesta

Fuente: Elaboración propia

Por otro lado, con la aplicación de la estrategia desarrollada puede afirmarse que, debido a la resolución parcial por etapas del problema y construcción progresiva del *"schedule"*, mayor tiempo de cómputo para cada subproblema T no garantiza que la configuración de ranuras y asignaciones propuestas para cada etapa sean las mejores para el valor final de la función objetivo. De todos modos, para el caso del problema de 30 *"batches"*, al aumentarse el límite T se observa una tendencia a mejorar la calidad de las soluciones encontradas.

Seguidamente, se presenta la Tabla 5.2 con las características de la solución obtenida para los modelos alternativos FULL y FULL FIXED, teniendo en cuenta los diferentes tamaños del problema. En la primera instancia evaluada, ambas formulaciones resuelven el problema asegurando la optimalidad de la solución encontrada. A pesar de ello, como era de esperarse el modelo FULL presenta un incremento notable del tiempo invertido respecto a la formulación ajustada FULL FIXED. Luego, a medida que crece la dimensión del caso de estudio, los tiempos de resolución aumentan de forma pronunciada (incluso para el problema pequeño de 15 órdenes) llegando a forzar el corte de la exploración del *"Solver"* en el tiempo límite establecido: $T = 5[hrs]$, equivalente a $T = 18{,}000[s]$. En estos casos, resulta evidente la dificultad de convergencia de ambas formulaciones, siendo muy elevados los valores de la función objetivo así como los margenes pendientes para probar la optimalidad de las soluciones.

Cantidad de órdenes	**Modelo alternativo**	**Makespan [hrs]**	**GAP %**	**Tiempo [s]**
10	FULL	**11,42**	**0,00 %**	**228**
	FULL FIXED	**11,42**	**0,00 %**	**4**
15	FULL	15,78	18,55 %	18.000
	FULL FIXED	**14,42**	**9,05 %**	**18.000**
20	FULL	19,99	15,15 %	18.000
	FULL FIXED	**18,68**	**9,20 %**	**18.000**
25	FULL	27,73	33,52 %	18.000
	FULL FIXED	**25,88**	**28,77 %**	**18.000**
30	FULL	33,84	35,14 %	18.000
	FULL FIXED	**33,47**	**32,95 %**	**18.000**

Tabla 5.2: Soluciones obtenidas para los modelos alternativos FULL y FULL FIXED

Fuente: Elaboración propia

Con la formulación FULL FIXED, que incluye una configuración de ranuras ajustada y fija, a medida que aumenta la cantidad de órdenes a procesar no se observa una mejora

sustancial respecto al modelo con cantidad de *"slots"* holgada, denominado FULL. Ambos se ven limitados por la cantidad de variables discretas y la expansión del árbol combinatorio a explorar por el *"Solver"*. Por lo tanto, puede decirse que ambas formulaciones de *"time-slots"* tradicionales presentan dificultades para ser utilizadas como estrategia de solución de problemas de tamaño industrial.

Finalmente, tomando como base la información de las tablas anteriores, en la Figura 5.1 se presenta, para cada cantidad de *"batches"*, una comparación entre las formulaciones FULL y FULL FIXED con las mejores soluciones obtenidas por la metodología propuesta. La altura total de cada columna corresponde al valor hallado de la función objetivo, y está compuesto por la mejor solución posible en azul y el intervalo pendiente para probar optimalidad de dicha solución en rojo. Además, se aprecia este margen relativo porcentual ($GAP\,\%$) en las etiquetas de cada columna.

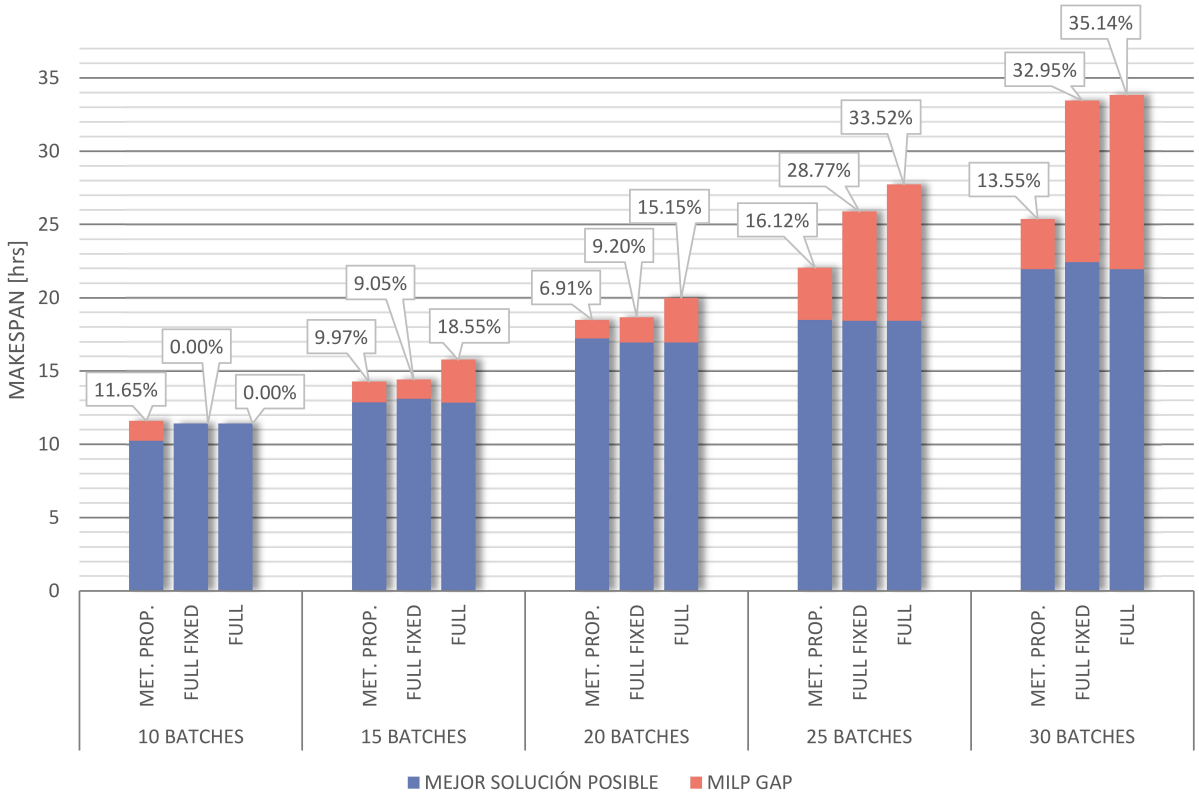

Figura 5.1: Comparación de las mejores soluciones obtenidas por la metodología propuesta con los modelos completos FULL y FULL FIXED

Fuente: Elaboración propia

En esta figura puede verse que, al aumentar la cantidad de lotes, la estrategia desarrollada permitió obtener soluciones de mejor calidad y cotas inferiores similares a

las formulaciones completas, a pesar de resolver únicamente subproblemas parciales y construir la solución de forma progresiva fijando una etapa a la vez. En particlar, puede observarse claramente que, al aumentar la cantidad de órdenes a procesar, la degradación de la función objetivo de los modelos completos respecto a la metodología presentada es cada vez mayor.

Para el caso particular de la instancia de 10 *"batches"*, las estimaciones realizadas en las sucesivas iteraciones de la estrategia algorítmica desarrollada conducen a un valor de función objetivo levemente mayor a aquella hallada para las formulaciones completas FULL y FULL FIXED. Sin embargo, a modo de comparación general entre las formulaciones evaluadas puede decirse que en problemas de mayor tamaño la metodología matemática-algorítmica propuesta se destaca por su capacidad de encontrar soluciones de buena calidad requiriendo un tiempo de cómputo menor.

5.3.2. Dimensiones de los modelos matemáticos

Cantidad de órdenes	Modelo	Var. Binarias		Var. Continuas		Ecuaciones	
		min	max	min	max	min	max
10	T=300[s]; $m_j^0 = 3$	90	412	190	254	1.593	2.418
	FULL	1.105		433		5.901	
	FULL FIXED	458		246		2.430	
15	T=300[s]; $m_j^0 = 3$	16	705	256	382	2.395	5.108
	FULL	2.524		675		12.935	
	FULL FIXED	1.012		377		5.153	
20	T=600[s]; $m_j^0 = 3$	49	1.057	321	510	3.287	8.945
	FULL	4.557		919		22.826	
	FULL FIXED	1.831		510		9.097	
25	T=1.200[s]; $m_j^0 = 2$	100	1.173	335	644	2.464	13.718
	FULL	7.134		1.153		35.212	
	FULL FIXED	2.842		644		13.929	
30	T=600[s]; $m_j^0 = 2$	120	1.556	395	775	2.935	19.583
	FULL	10.264		1.384		50.190	
	FULL FIXED	4.090		775		19.847	

Tabla 5.3: Dimensiones de los modelos utilizados para obtener las mejores soluciones con la metodología propuesta y los modelos alternativos FULL y FULL FIXED

Fuente: Elaboración propia

En este apartado, se exponen los atributos de los modelos generados, como son la cantidad de variables binarias, de variables continuas y de ecuaciones. De éstos, el primero es el que determina en mayor medida la dificultad de resolución computacional de este tipo de problemas. En la Tabla 5.3 puede observarse una comparación de las características mencionadas entre las formulaciones completas y las evaluaciones utilizadas para obtener mejor solución con la metodología propuesta, para cada instancia del problema.

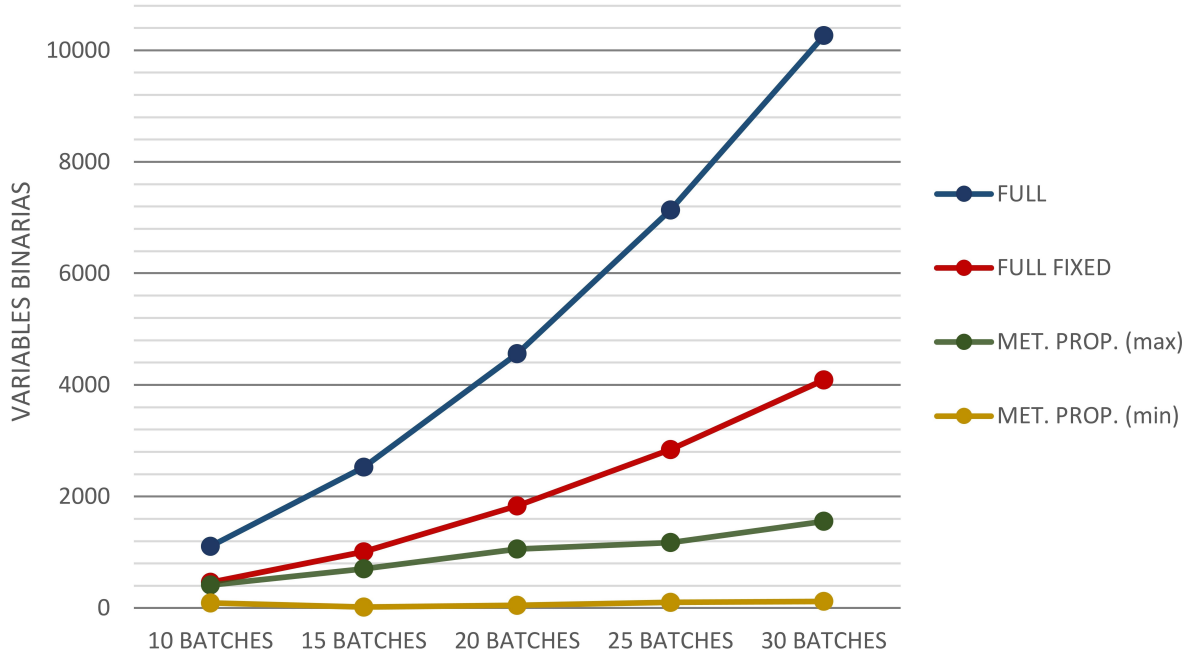

Figura 5.2: Comparación de la cantidad de variables binarias mínima y máxima en las mejores soluciones obtenidas por la metodología propuesta con los modelos completos FULL y FULL FIXED

Fuente: Elaboración propia

Nótese, en primer lugar, que para la metodología desarrollada se exponen las cantidades mínimas y máximas de estos atributos, dado que se resuelve una secuencia de subproblemas y no un único modelo. Aquí se observa que se itera entre problemas más pequeños que las formulaciones completas llegando, en el caso de 30 órdenes, a un máximo de 1.556 variables binarias. Por otra parte, en cuanto a la formulación ajustada FULL FIXED, puede observase que al utilizar la misma se reduce a más de la mitad la cantidad de variables binarias del modelo FULL. A pesar de esto, como fue expuesto previamente, las mejoras en el valor de la función objetivo para el mismo tiempo de cómputo entre ambos modelos son prácticamente nulas. De lo anterior puede concluirse que la posibilidad de resolver el problema completo explorando diferentes configuraciones de *"slots"*, sean

fijos u opcionales, no se considera una opción viable para afrontar problemas de tamaño industrial.

Adicionalmente, en la Figura 5.2 se realiza una comparación entre la cantidad de variables binarias para las tres formulaciones que, como fue mencionado previamente, es el atributo de mayor influencia en la complejidad de resolución computacional de este problema combinatorio. En la misma se observa que, a medida que aumenta la cantidad de órdenes a procesar, existe un crecimiento significativo de éstas para los modelos completos FULL y FULL FIXED. En contraste, por el lado de la metodología propuesta existe un aumento menor de las cantidades, ya sean mínimas o máximas, de variables binaras utilizadas en los subproblemas. Esta diferencia permite que dicha característica se mantenga entre límites notablemente inferiores a las formulaciones completas, y explica la capacidad para encontrar buenas soluciones en tiempos de resolución menores.

5.3.3. Mejor solución encontrada para el problema de 30 *"batches"*

En la Figura 5.3 se presenta el Diagrama de Gantt de la mejor solución hallada con la estrategia matemática-algorítmica desarrollada en este proyecto. La misma fue obtenida con una cantidad de ranuras iniciales en cada equipo de $m_j^0 = 2$ y un límite de tiempo para la Iteración Interna establecido en $T = 600[s]$. A su vez, el orden de exploración de etapas resultante para el problema completo de 30 *"batches"* fue la secuencia l_4, l_6, l_2, l_1, l_3 y l_5. De esta manera, como se indicó en la Tabla 5.1 el valor del *"makespan"* hallado es de $MK = 25,38[hrs]$ en un total de 5.259 [s], equivalente a 1h 27' de ejecución de las iteraciones. Además, en este problema se garantiza una mejor solución posible de 21,94 [hrs], con lo cual puede determinarse un intervalo relativo de integralidad de $GAP\,\% = 13,55\,\%$ restante para probar la optimalidad de la solución. A fines comparativos, en los Anexos E y F se exponen los Diagramas de Gantt de la mejor solución hallada para los problemas de 20 y 25 *"batches"*, respectivamente.

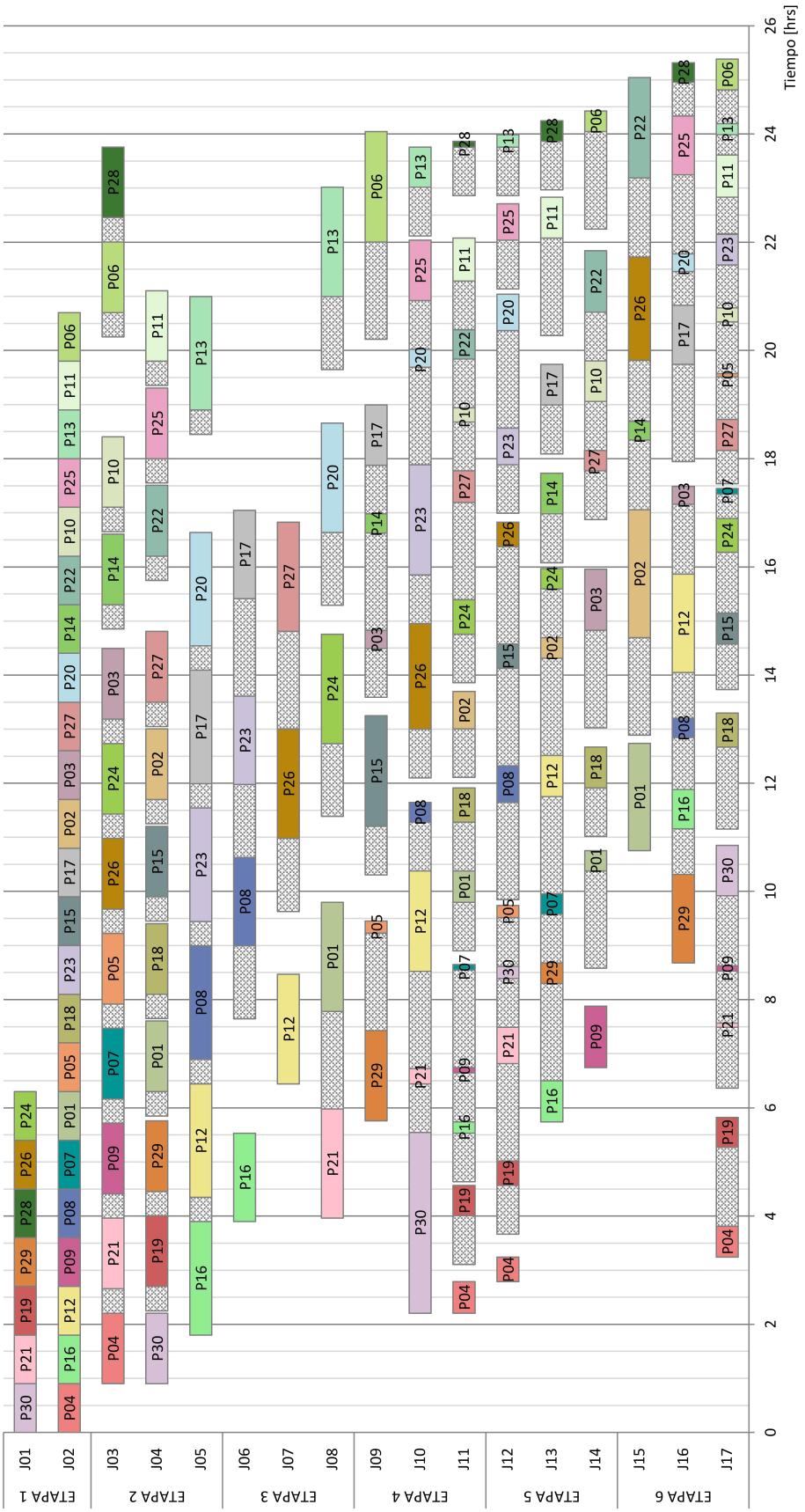

Figura 5.3: Diagrama de Gantt de la mejor solución obtenida ($MK = 25,38[hrs]$) para el problema de 30 lotes, con parámetros $m_j^0 = 2$ y $T = 600[s]$

Fuente: Elaboración propia

Por otra parte, la Tabla 5.4 muestra el detalle de los parámetros de la Iteración Principal durante la construcción progresiva del *"schedule"*. Tal como se observa en esta tabla la etapa fijada en la primera Iteración Principal es la l_4. En ella, las asignaciones y secuenciación de tareas, sumadas a las aproximaciones inferidas para el resto de las etapas pendientes, resultan con un valor transitorio de la función objetivo de $MK = 23,65[hrs]$. En particular, de los datos del problema se desprende que esta etapa presenta un cuello de botella en el equipo de procesamiento $J11$, debido a que una gran cantidad de tareas necesitan ser procesadas de forma exclusiva en este equipo. Continuando con el proceso, la siguiente etapa elegida para fijar es la l_6, con la cual se eleva el mínimo tiempo total de procesamiento a $MK = 24,84[hrs]$. En los equipos de esta etapa, los tiempos de transición dependientes de la secuencia son, en algunos casos, mucho mayores que los tiempos de procesamiento. Además, esta etapa presenta dificultades de resolución debido a la amplia diferencia entre los tiempos de transición mínimos y aquellos asociados a una secuencia de *"batches"* (i, i') específica.

Iteración Principal	**Etapa** l^{act}	$Z^{\text{lb\_approx}}_{a,l^{\text{act}}}$	$Z^{\text{lb}}_{a,l^{\text{act}}}$	$Z^{\text{sol}}_{a,l^{\text{act}}}$	**Etapa** l^{sel}	**Tiempo de resolución [s]**
$a = 1$	l_4	**21,9447**	**21,9447**	**23,6574**	l_4	2.502,40
	l_6	21,9447	-	-	-	82,28
	l_2	21,9447	-	-	-	24,20
	l_1	21,9447	-	-	-	16,46
	l_3	21,6386	-	-	-	601,74
	l_5	21,9447	-	-	-	280,97
$a = 2$	l_6	**23,6574**	**23,6574**	**24,8472**	l_6	602,52
	l_2	23,6574	-	-	-	6,99
	l_1	23,6574	-	-	-	4,76
	l_3	23,6574	-	-	-	3,73
	l_5	23,6574	-	-	-	18,41
$a = 3$	l_2	**25,2234**	**25,2234**	**25,2234**	l_2	17,10
	l_1	24,8472	-	-	-	2,93
	l_3	24,8472	-	-	-	3,09
	l_5	24,8472	-	-	-	38,74
$a = 4$	l_1	25,2234	25,2234	25,2234	l_1	3,43
	l_3	25,2234	25,2612	25,2612	l_3	34,66
	l_5	**25,3836**	**25,3836**	**25,3836**	l_5	1.007,04
$a = 5$	l_1	25,3836	25,3836	25,3836	l_1	2,87
	l_3	**25,3836**	**25,3836**	**25,3836**	l_3	2,29
$a = 6$	l_1	**25,3836**	**25,3836**	**25,3836**	l_1	2,40

Tabla 5.4: Construcción progresiva de la solución para el problema de 30 lotes

Fuente: Elaboración propia

Seguidamente, en la tercera Iteración Principal se fija la etapa l_2, en la cual las asignaciones de las tareas $(i,\ l)$ tienen como resultado un valor de función objetivo de $MK = 25,22[hrs]$. Observando el camino crítico de la solución, esta etapa presenta una influencia significativa para el tiempo total de procesamiento. Sin embargo, la complejidad combinatoria para la secuenciación de *"batches"* es menor debido a que los tiempos de procesamiento y transición son iguales para todas las tareas. Posteriormente, se fija la etapa l_5 con la cual el *"makespan"* evoluciona a su valor definitivo de $MK = 25,38[hrs]$. Para finalizar el proceso, son fijadas consecutivamente las etapas l_3 y l_1, las cuales completan la solución exacta del problema y mantienen el mínimo tiempo total de procesamiento encontrado anteriormente. En el Anexo D.1, puede observarse el detalle de la evolución de las cotas utilizadas durante la Iteración Interna para la solución expuesta anteriormente.

Capítulo 6

CONCLUSIONES Y TRABAJOS FUTUROS

6.1. Conclusiones del Trabajo Realizado

En este proyecto se presentó una metodología matemática-algorítmica desarrollada para resolver problemas de programación de operaciones de corto plazo, o problema de *"scheduling"* de carácter predictivo, en instalaciones industriales discontinuas multiproducto multietapa. En su desarrollo, se considera que la instalación se compone de etapas ordenadas de forma secuencial y las mismas están integradas por equipos disímiles que trabajan en paralelo. A su vez, se considera una política de almacenamiento intermedio ilimitado (*"Unlimited Intermediate Storage"* - *UIS*) en todas las etapas y tiempos de transición dependientes de la secuencia de procesamiento en algunas de ellas.

Para la construcción de la estrategia desarrollada, se utilizó un modelo de tiempo continuo novedoso basado en la formulación de *"time-slots"* presentada por Pinto y Grossmann (1995). La metodología formulada incluye el concepto de ranura multivaluada, a la cual pueden ser asignadas más de una tarea a la vez y permite, mediante aproximaciones, resolver un conjunto de subproblemas de menor tamaño que el problema completo. Haciendo uso de este nuevo recurso y en conjunto con técnicas algorítmicas, se buscó evaluar diferentes alternativas de configuración para el número de *"slots"* por equipo y construir

la solución paso a paso fijando una a una las etapas de la instalación.

La metodología propuesta, está estructurada como una Iteración Principal y una Iteración Interna. Por un lado, en cada instancia de resolución, el algoritmo utilizado para la Iteración Interna tiene como objetivo encontrar la mejor solución considerando una de las etapas y un subconjunto de tareas previas y posteriores correspondientes a las etapas aguas arriba y aguas abajo de la misma. En cada ciclo iterativo se resuelve un subproblema aproximado y otro exacto: el primero de ellos incorpora ranuras multivaluadas con el fin de encontrar una propuesta de cantidad de *"slots"* para cada equipo de la etapa en análisis y, el segundo, analiza esa propuesta evaluando la etapa con cantidad de ranuras convencionales resultante del subproblema anterior.

Por otra parte, el algoritmo denominado Iteración Principal se desarrolló para construir paso a paso una agenda de producción factible o *"schedule"*, con el tiempo total de producción mínimo, fijando progresivamente las etapas de la instalación según su dificultad decreciente de programación. Para ello, se utilizan cotas que caracterizan las soluciones parciales halladas para cada etapa en la Iteración Interna previamente mencionada. A medida que concluye cada ciclo de la Iteración Principal, cierta etapa es elegida y se fijan las decisiones de asignación y secuenciación de las tareas en los equipos que la componen. El proceso se repite hasta que todas las etapas hayan sido fijadas y quede construida la solución completa del problema. A su vez, se implementa un tercer algoritmo al inicio del proceso, con el cual se otorga un orden de prioridad para la exploración de las etapas en la Iteración Principal. Éste fue desarrollado con el fin de conseguir importantes ahorros de tiempo en la resolución de los modelos, haciendo uso de cotas para el cese de la exploración del árbol de alternativas de ciertos subproblemas parciales resueltos en la Iteración Interna. Con ello, se busca evitar el análisis de casos en cuales la etapa no resulta crítica para determinar la solución de acuerdo al camino de construcción del *"schedule"* propuesto.

La metodología propuesta fue aplicada a un problema real de gran escala perteneciente a la industria farmacéutica y se compara su rendimiento con dos modelos tradicionales basados en la formulación de *"time-slots"* desarrollada por Pinto y Grossmann (1995). La primera de ellas, llamada FULL, considera una cantidad de ranuras igual al máximo número de *"batches"* que pueden ser procesados en cada equipo. La segunda es llamada FULL FIXED y utiliza una cantidad de *"slots"* por equipo igual al resultante para la mejor solución obtenida con la metodología propuesta. Esto significa que, para este último modelo, todas las ranuras definidas para cada equipo serán necesariamente utilizadas, a diferencia de la primera formulación en la cual es posible que existan algunas

de ellas ociosas en la solución encontrada.

Para evaluar el rendimiento de las formulaciones, fueron definidas diferentes instancias del problema con tamaño y complejidad creciente incrementando la cantidad de *"batches"* a programar. Los resultados reportados revelaron que el enfoque tradicional de *"time-slots"* es una opción poco viable para tratar problemas de gran tamaño ya que luego de altos tiempos de cómputo no se obtienen soluciones de calidad aceptable. Debido a la complejidad combinatoria del problema, resulta de gran dificultad afrontar casos de estudio de tamaño industrial con métodos exactos, lo que lleva a implementar estrategias alternativas para la construcción de soluciones. En esta línea, la metodología matemática-algorítmica desarrollada plantea una estrategia de descomposición del problema que busca construir gradualmente la solución, identificando e incorporando las decisiones críticas en cada paso. Esta metodología permite obtener soluciones de buena calidad en tiempos competitivos para la industria, conservando además información rigurosa de las cotas de integralidad asociadas.

En este sentido, es de importancia mencionar que la metodología propuesta permitió obtener una notable reducción en el tamaño de los modelos resueltos respecto a las formulaciones que consideran el problema completo. Esto se logra fundamentalmente a través de la implementación de ranuras multivaluadas y la idea de construir la agenda de producción analizando la solución de una etapa de la instalación a la vez. La resolución de modelos matemáticos de tamaño reducido en cuanto a cantidad de variables binarias, denominados subproblema aproximado y exacto, resulta de gran impacto en el rendimiento de los recursos computacionales requeridos para resolver el problema.

Cabe destacar que la estrategia planteada fue aplicada al caso de estudio obteniendo resultados altamente satisfactorios. En este contexto, el desarrollo realizado y los resultados obtenidos han sido presentados en un congreso internacional, seleccionado para un capítulo de libro de la editorial Springer y se proyecta su publicación un artículo de revista internacional con referato del área de investigación.

Finalmente, quiero agradecer al director de este proyecto, el Dr. Pablo Andrés Marchetti por sus aportes en los espacios de debate y dedicación permanente. Sus acompañamiento, confianza hacia mi trabajo y apoyo constante durante mis últimos años de carrera académica han aportado a un gran enriquecimiento personal. A su vez, expreso mi agradecimiento a la Universidad Tecnológica Nacional por el financiamiento brindado a través de los proyectos PID 4932 y PID 7768, de los cuales formé parte en el marco de las actividades realizadas como becario de Investigación y Desarrollo.

6.2. Trabajos Futuros

A partir del proyecto desarrollado, se dejan abiertas posibles líneas de trabajos futuros que permiten proyectar desafíos y conseguir nuevos avances en la temática. El buen rendimiento de la metodología desarrollada da lugar a considerar su adaptación e implementación como soporte en procesos de toma de decisiones de programación de operaciones en diversos ambientes de producción industrial a gran escala. Entre las posibles alternativas de líneas de investigación se plantea aplicar la metodología generada a otros casos de estudio y avanzar con técnicas para la identificación y ramificación del camino crítico de la solución.

En el primer caso, se plantea la iniciativa de adaptar la estrategia desarrollada a requerimientos particulares de otros problemas reales de gran tamaño para la programación de operaciones de corto plazo. En este sentido, es posible añadir complejidades a la formulación incluyendo fechas límites de entrega para las tareas, adaptándola para su actualización dinámica con la operación en tiempo real de la planta industrial, implementando políticas de almacenamiento e inventario diferentes a la estudiada, o aplicándola a casos de estudio que consideren niveles de incertidumbre en ciertos parámetros del modelo.

Por otra parte, como segunda línea de trabajo se propone el desarrollo de un algoritmo que explore diversas alternativas a partir de los camino críticos hallados para las mejores soluciones construidas con la metodología desarrollada en este trabajo. A través de un proceso de ramificación se buscará mejorar el valor de la función objetivo y desempeño computacional del modelo, manteniendo información rigurosa de las cotas.

Nomenclatura

Índices y conjuntos

$i, i' \in I$:	*"batches"* a procesar.
$l, l' \in L_i$:	etapas de procesamiento requeridas por cada *"batch"* i.
$j \in J_l$:	equipos asociados a la etapa l.
$k, k' \in K_j$:	*"slots"* pertenecientes a cada equipo j.
a :	número de ciclo para la Iteración Principal.
b :	repetición realizada en la Iteración Interna.
m :	cantidad de *"batches"* alojados en cierto equipo.

Elementos

k_j^* :	*"slot"* multivaluado en el equipo j.
l_i^{last} :	última etapa para el *"batch"* i que existe en L_i.
k_j^{last} :	ranura ubicada más adelante en el tiempo para el equipo j.
l^{act} :	etapa considerada para la Iteración Interna.
l^{sel} :	etapa seleccionada para fijar en un ciclo de la Iteración Principal.
l^{critic} :	etapa seleccionada para fijar en la primera Iteración Principal.

Subconjuntos

$I_j \subseteq I$: tareas (i, l) que pueden ser procesadas en el equipo $j \in J_l$.

$L^{\mathrm{P}} \subseteq L$: etapas pendientes para ser fijadas.
$L^1 \subseteq L$: etapas con esquema de secuenciación temporal ascendente.
$L^2 \subseteq L$: etapas con esquema de secuenciación temporal descendente.
$L^{\mathrm{F}} \subseteq L^{\mathrm{P}}$: etapa pendiente con número de ranuras ajustado.

$J_{i,l} \subseteq J_l$: equipos disponibles para la tarea (i,l).
$SeqDep_j^1 \subseteq J_l$: equipos con tiempos de transición dependientes de la secuencia para las etapas $l \in L^1$.
$SeqDep_j^2 \subseteq J_l$: equipos con tiempos de transición dependientes de la secuencia para las etapas $l \in L^2$.

$\overleftarrow{K}_j^k \subseteq K_j$: ranura k y todas las ubicadas a la izquierda de ésta en el equipo j.
$\overrightarrow{K}_j^k \subseteq K_j$: *"slot"* k y todos los ubicados a la derecha del mismo asociados al equipo j.

Parámetros

$pt_{i,j}$: tiempo de procesamiento del *"batch"* i en el equipo j.

$\tau_{i',i,j}$: tiempos de transición para cada *"batch"* i' que se procesa anteriormente al *"batch"* i en el equipo j.
$\tau_{\cdot i,j}^{\min}$ / $\tau_{\cdot i,j}^{\max}$: mínimo / máximo tiempo de transición asociado al *"batch"* i en un equipo $j \in J_l : l \in L^1$.
$\tau_{i'\cdot,j}^{\min}$ / $\tau_{i'\cdot,j}^{\max}$: mínimo / máximo tiempo de transición asociado al *"batch"* i' en un equipo $j \in J_l : l \in L^2$.

$k_j^{\min}$ / $k_j^{\max}$: mínimo / máximo número de *"slots"* que se deben / pueden asociar al equipo j.

m_j : cantidad de ranuras propuestos para el equipo j.
m_j^0 : número de ranuras iniciales propuestas para el equipo j.
m_j^b : cantidad de *"slots"* evaluados en la Iteración Interna b para el equipo j.

$EST_{i,l}$: tiempo más temprano para el cual el *"batch"* i puede ser procesado en la etapa l.

EST_j: mínimo tiempo de comienzo de operación para el equipo j.

$ETC_{i,l}$: tiempo más temprano de finalización para el *"batch"* i en la etapa l.

ETC_j : mínimo tiempo de finalización de operación para el equipo j.

$\widetilde{pt}_{i,l}$: mínimo tiempo de procesamiento considerado para la actualización de $EST_{i,l}$ y $ETC_{i,l}$

$nb^{\max}$: máxima cantidad de Iteraciones Internas para la misma etapa.

$nb^{\max\_ini}$: máxima cantidad ciclos para la Iteración Interna al establecer el orden de exploración de etapas.

t : duración de tiempo máxima para la Iteración Interna en una etapa.

T : límite de tiempo para cada subproblema resuelto.

T^{ini} : límite de tiempo para cada subproblema al establecer el orden de exploración de etapas.

Ord_l : orden para el recorrido de las etapas pendientes.

$Z_{a,l}^{\text{lb\_approx}}$: máxima cota inferior para los subproblemas aproximados de cierta etapa en una repetición de la Iteración Principal.

$Z_{a,l}^{\text{sol}}$: mejor solución exacta obtenida para una etapa la Iteración Principal a.

$Z_{a,l}^{\text{lb}}$: cota inferior de la solución exacta encontrada para la etapa l en un ciclo de la Iteración Principal.

$z_{a,b,l}^{\text{sol\_approx}}$: valor de la función objetivo obtenida en un subproblema aproximado para la etapa l.

$z_{a,b,l}^{\text{lb\_approx}}$: cota inferior para una solución aproximada obtenida en la Itearación Interna b.

$z_{a,b,l}^{\text{sol}}$: valor de la función objetivo obtenida en un subproblema exacto de la Iteración Interna.

$z_{a,b,l}^{\text{lb}}$: cota inferior para una solución exacta dentro de la Iteración Interna.

Z^{sol} : valor de la función objetivo para la solución del problema construida de forma progresiva.

Z^{lb} : cota inferior para la solución completa del problema.

Z_l^{ord} : valor de la función objetivo para establecer el orden de exploración de etapas.

$\beta_{i,j,k,l}$: decisiones de asignación y secuenciación guardadas para una etapa l.

Variables continuas

$ST^1_{i,l}$ / $CT^1_{i,l}$: tiempo de comienzo/finalización para el *"batch"* i en la etapa l.
$ST^2_{j,k}$ / $CT^2_{j,k}$: tiempo de comienzo/finalización para el *"slot"* k asociado al equipo j.

$CO_{j,k}$: complemento para determinar el tiempo de transición necesario antes de que comience el *"slot"* k asociado al equipo j.

MK : valor para el *"makespan"*.

Variables binarias

$W_{i,j,k,l}$: decisión de asignación y secuenciación de las tareas.

$Q_{j,m}$: indica que m *"slots"* son evaluados en el equipo j.

Bibliografía

Aarts, E. H. L. & Korst, J. (1989). Simulated annealing and boltzmann machines: A stochastic approach to combinatorial optimization and neural computing. *Discret. Appl. Math.*, *26*(1), 131. https://doi.org/10.1016/0166-218x(90)90039-f

Balas, E., Lenstra, J. K. & Vazacopoulos, A. (1995). One-machine problem with delayed precedence constraints and its use in job shop scheduling. *Manage. Sci.*, *41*(1), 94-109. https://doi.org/10.1287/mnsc.41.1.94

Balas, E., Ceria, S. & Cornuéjols, G. (1993). A lift-and-project cutting plane algorithm for mixed 0–1 programs. *Math. Program. 1993 581*, *58*(1), 295-324. https://doi.org/10.1007/BF01581273

Bassett, M. H., Dave, P., Doyle, F. J., Kudva, G. K., Pekny, J. F., Reklaitis, G. V., Subrahmanyam, S., Miller, D. L. & Zentner, M. G. (1996). Perspectives on model based integration of process operations. *Comput. Chem. Eng.*, *20*(6-7), 821-844. https://doi.org/10.1016/0098-1354(95)00184-0

Blackstone, J. H., Phillips, D. T. & Hogg, G. L. (1982). A state-of-the-art survey of dispatching rules for manufacturing job shop operations. *Int. J. Prod. Res.*, *20*(1), 27-45. https://doi.org/10.1080/00207548208947745

Borchers, B. & Mitchell, J. E. (1994). An improved branch and bound algorithm for mixed integer nonlinear programs. *Comput. Oper. Res.*, *21*(4), 359-367. https://doi.org/10.1016/0305-0548(94)90024-8

Caballero, J. A. & Grossmann, I. E. (2007). A review of the state of the art in optimization. *RIAI - Rev. Iberoam. Autom. e Inform. Ind.*, *4*(1), 5-23. https://doi.org/10.1016/s1697-7912(07)70188-7

Castro, P. M. & Grossmann, I. E. (2005). New continuous-time MILP model for the short-term scheduling of multistage batch plants. *Ind. Eng. Chem. Res.*, *44*(24), 9175-9190. https://doi.org/10.1021/ie050730l

Castro, P. M. & Grossmann, I. E. (2006). An efficient MILP model for the short-term scheduling of single stage batch plants. *Comput. Chem. Eng.*, *30*(6-7), 1003-1018. https://doi.org/10.1016/j.compchemeng.2005.12.014

Castro, P. M., Harjunkoski, I. & Grossmann, I. E. (2009). Optimal short-term scheduling of large-scale multistage batch plants. *Ind. Eng. Chem. Res.*, *48*(24), 11002-11016. https://doi.org/10.1021/ie900734x

Castro, P. M., Zeballos, L. J. & Méndez, C. A. (2012). Hybrid time slots sequencing model for a class of scheduling problems. *AIChE J.*, *58*(3), 789-800. https://doi.org/10.1002/aic.12609

Castro, P. M., Grossmann, I. E. & Zhang, Q. (2018). Expanding scope and computational challenges in process scheduling. *Comput. Chem. Eng.*, *114*, 14-42. https://doi.org/10.1016/J.COMPCHEMENG.2018.01.020

Cavin, L., Fischer, U., Glover, F. & Hungerbühler, K. (2004). Multi-objective process design in multi-purpose batch plants using a Tabu Search optimization algorithm. *Comput. Chem. Eng.*, *28*(4), 459-478. https://doi.org/10.1016/j.compchemeng.2003.07.002

Cerdá, J., Henning, G. P. & Grossmann, I. E. (1997). A Mixed-Integer Linear Programming Model for Short-Term Scheduling of Single-Stage Multiproduct Batch Plants with Parallel Lines. *Ind. Eng. Chem. Res.*, *36*(5), 1695-1707. https://doi.org/10.1021/ie9605490

Chase, R. B. & Jacobs, F. R. (2009). *Administración de producción y operaciones: manufactura y servicios* (13a). McGraw Hill.

Chen, C. L., Liu, C. L., Feng, X. D. & Shao, H. H. (2002). Optimal short-term scheduling of multiproduct single-stage batch plants with parallel lines. *Ind. Eng. Chem. Res.*, *41*(5), 1249-1260. https://doi.org/10.1021/ie010465d

Dakin R. J. (1965). A tree-search algorithm for mixed integer programming problems. *Comput. J.*, *8*(3), 250-255. https://doi.org/10.1093/comjnl/8.3.250

Domínguez Machuca, J. A. (2003). *Dirección de operaciones: aspectos tácticos y operativos en la producción y los servicios* (1a). McGraw-Hill.

Duran, M. A. & Grossmann, I. E. (1986). An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math. Program.*, *36*(3), 307-339. https://doi.org/10.1007/BF02592064

Fletcher, R. & Leyffer, S. (1994). Solving mixed integer nonlinear programs by outer approximation. *Math. Program.*, *66*(1-3), 327-349. https://doi.org/10.1007/BF01581153

Fuchigami, H. Y. & Rangel, S. (2018). A survey of case studies in production scheduling: Analysis and perspectives. *J. Comput. Sci.*, *25*. https://doi.org/10.1016/j.jocs.2017.06.004

Garey, M. R. & Johnson D. S. (1979). *Computers and intractability : a guide to the theory of NP-completeness* (Freeman W. H., Ed.).

Geoffrion, A. M. (1972). Generalized Benders decomposition. *J. Optim. Theory Appl.*, *10*(4), 237-260. https://doi.org/10.1007/BF00934810

Glover, F. (1990). Tabu Search: A Tutorial. *Interfaces (Providence).*, *20*(4), 74-94. https://doi.org/10.1287/inte.20.4.74

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning* (Addison - Wesley, Ed.).

Gomory, R. E. (1958). Outline of an Algorithm for Integer Solutions to Linear Programs. *Bull. Am. Math. Soc.*, *64*(5), 275-278. https://doi.org/10.1090/S0002-9904-1958-10224-4

Graells, M., Cantón, J., Peschaud, B. & Puigjaner, L. (1998). General approach and tool for the scheduling of complex production systems. *Comput. Chem. Eng.*, *22*(1), 395-402. https://doi.org/10.1016/s0098-1354(98)00080-5

Günther, H.-O. & Van Beek, P. (2003). *Advanced planning and scheduling in process industry*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-05607-3_1

Gupta, O. K. & Ravindran, A. (1985). Branch and Bound Experiments in Convex Nonlinear Integer Programming. *Manage. Sci.*, *31*(12), 1533-1546. https://doi.org/10.1287/mnsc.31.12.1533

Gupta, S. & Karimi, I. A. (2003). An improved MILP formulation for scheduling multiproduct, multistage batch plants. *Ind. Eng. Chem. Res.*, *42*(11), 2365-2380. https://doi.org/10.1021/ie020180g

Harjunkoski, I. & Grossmann, I. E. (2002). Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. *Comput. Chem. Eng.*, *26*(11), 1533-1552. https://doi.org/10.1016/S0098-1354(02)00100-X

Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., Hooker, J., Méndez, C., Sand, G. & Wassick, J. (2014). Scope for industrial applications of production scheduling models and solution methods. *Comput. Chem. Eng.*, *62*. https://doi.org/10.1016/j.compchemeng.2013.12.001

He, Y. & Hui, C. W. (2007). Genetic algorithm for large-size multi-stage batch plant scheduling. *Chem. Eng. Sci.*, *62*(5), 1504-1523. https://doi.org/10.1016/j.ces.

2006.11.049

Hillier, F. S. & Lieberman, G. J. (2015). *Introducción a la investigación de operaciones* (10th). McGraw-Hill/Interamericana.

Jain, V. & Grossmann, I. E. (2001). Algorithms for Hybrid MILP/CP Models for a Class of Optimization Problems. *INFORMS J. Comput.*, *13*(4), 258-276. https://doi.org/10.1287/ijoc.13.4.258.9733

Kopanos, G. M., Méndez, C. A. & Puigjaner, L. (2010). MIP-based decomposition strategies for large-scale scheduling problems in multiproduct multistage batch plants: A benchmark scheduling problem of the pharmaceutical industry. *Eur. J. Oper. Res.*, *207*(2). https://doi.org/10.1016/j.ejor.2010.06.002

Lee, H. & Maravelias, C. T. (2018). Combining the advantages of discrete- and continuous-time scheduling models: Part 1. Framework and mathematical formulations. *Comput. Chem. Eng.*, *116*, 176-190. https://doi.org/10.1016/J.COMPCHEMENG.2017.12.003

Lee, Y. G. & Malone, M. F. (2000). Flexible batch process planning. *Ind. Eng. Chem. Res.*, *39*(6), 2045-2055. https://doi.org/10.1021/ie990185m

Lim, M. F. & Karimi, I. A. (2003). A slot-based formulation for single-stage multiproduct batch plants with multiple orders per product. *Ind. Eng. Chem. Res.*, *42*(9), 1914-1924. https://doi.org/10.1021/ie020536o

Löhl, T., Schulz, C. & Engell, S. (1998). Sequencing of batch operations for a highly coupled production process: Genetic algorithms versus mathematical programming. *Comput. Chem. Eng.*, *22*(1), 579-585. https://doi.org/10.1016/s0098-1354(98)00103-3

Malapert, A., Guéret, C. & Rousseau, L. M. (2012). A constraint programming approach for a batch processing problem with non-identical job sizes. *Eur. J. Oper. Res.*, *221*(3), 533-545. https://doi.org/10.1016/j.ejor.2012.04.008

Maravelias, C. T. & Grossmann, I. E. (2004). A hybrid MILP/CP decomposition approach for the continuous time scheduling of multipurpose batch plants. *Comput. Chem. Eng.*, *28*(10), 1921-1949. https://doi.org/10.1016/j.compchemeng.2004.03.016

Méndez, C. A. & Cerdá, J. (2002). An MILP framework for short-term scheduling of single-stage batch plants with limited discrete resources. *Comput. Aided Chem. Eng.*, *10*(100), 721-726. https://doi.org/10.1016/S1570-7946(02)80148-1

Méndez, C. A., Cerdá, J., Grossmann, I. E., Harjunkoski, I. & Fahl, M. (2006). State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput. Chem. Eng.*, *30*(6-7), 913-946. https://doi.org/10.1016/j.compchemeng.2006.02.008

Méndez, C. A., Henning, G. P. & Cerdá, J. (2001). An MILP continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. *Comput. Chem. Eng.*, *25*(4-6), 701-711. https://doi.org/10.1016/S0098-1354(01)00671-8

Merchan, A. F., Lee, H. & Maravelias, C. T. (2016). Discrete-time mixed-integer programming models and solution methods for production scheduling in multistage facilities. *Comput. Chem. Eng.*, *94*, 387-410. https://doi.org/10.1016/J.COMPCHEMENG.2016.04.034

Novara, F., Novas, J. & Henning, G. (2013). Planificación de la Producción de Corto Plazo en Plantas "Batch" Multiproducto Multietapa: Un Enfoque CP Novedoso. *Iberoam. J. Ind. Eng.*, *5*(10), 203-218. https://doi.org/10.13084/2175-8018.v05n10a15

Pinto, J. M. & Grossmann, I. E. (1995). A continuous time MILP model for short term scheduling of batch plants with pre-ordering constraints. *Comput. Chem. Eng.*, *20*(2), 1197-1202. https://doi.org/10.1016/0098-1354(96)00207-4

Pinto, J. M. & Grossmann, I. E. (1996). A continuous time MILP model for short term scheduling of batch plants with pre-ordering constraints. *Comput. Chem. Eng.*, *20*(2), 1197-1202. https://doi.org/10.1016/0098-1354(96)00207-4

Quesada, I. & Grossmann, I. E. (1992). An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Comput. Chem. Eng.*, *16*(10-11), 937-947. https://doi.org/10.1016/0098-1354(92)80028-8

Ryu, J. H., Lee, H. K. & Lee, I. B. (2001). Optimal scheduling for a multiproduct batch process with minimization of penalty on due date period. *Ind. Eng. Chem. Res.*, *40*(1), 228-233. https://doi.org/10.1021/ie000375t

Van Hentenryck, P. (1989). *Constraint Satisfaction in Logic Programming*. The MIT Press.

Velez, S. & Maravelias, C. T. (2013). Multiple and nonuniform time grids in discrete-time MIP models for chemical production scheduling. *Comput. Chem. Eng.*, *53*, 70-85. https://doi.org/10.1016/J.COMPCHEMENG.2013.01.014

Velez, S., Merchan, A. F. & Maravelias, C. T. (2015). On the solution of large-scale mixed integer programming scheduling models. *Chem. Eng. Sci.*, *136*, 139-157. https://doi.org/10.1016/J.CES.2015.05.021

Verbiest, F., Pinto-Varela, T., Cornelissens, T., Springael, J. & Barbosa-Povoa, A. (2019). Decomposition approaches for the design and scheduling of multiproduct multistage batch plants with parallel lines. *Comput. Chem. Eng.*, *127*, 111-126. https://doi.org/10.1016/j.compchemeng.2019.05.001

Westerlund, T. & Pettersson, F. (1995). An extended cutting plane method for solving convex MINLP problems. *Comput. Chem. Eng.*, *19*(1), 131-136. https://doi.org/10.1016/0098-1354(95)87027-X

Williams, P. (2013). *Model Building in Mathematical Programming* (5th). Wiley; Sons.

Wongthatsanekorn, W. & Phruksaphanrat, B. (2015). Genetic algorithm for short-term scheduling of make-and-pack batch production process. *Chinese J. Chem. Eng.*, *23*(9), 1475-1483. https://doi.org/10.1016/j.cjche.2015.04.021

Zeballos, L. J., Novas, J. M. & Henning, G. P. (2011). A CP formulation for scheduling multiproduct multistage batch plants. *Comput. Chem. Eng.*, *35*(12), 2973-2989. https://doi.org/10.1016/j.compchemeng.2011.01.043

Anexos

A. Tiempos de transición $\tau_{i',i,j}$ [hrs] para cada etapa de la instalación

A.1. Etapa 01

$$\tau_{i',i,j} = 0$$

Fuente: Material suplementario de Kopanos y col. (2010)

A.2. Etapa 02

$$\tau_{i',i,j} = 0,45$$

Fuente: Material suplementario de Kopanos y col. (2010)

A.3. Etapa 03

	P01	**P08**	**P12**	**P13**	**P16**	**P17**	**P20**	**P21**	**P23**	**P24**	**P26**	**P27**
P01	1,35	1,80	1,80	1,35	1,80	1,35	1,35	1,80	1,80	1,35	1,80	1,80
P08	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80
P12	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80
P13	1,35	1,80	1,80	1,35	1,80	1,35	1,35	1,80	1,80	1,35	1,80	1,80
P16	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80
P17	1,35	1,80	1,80	1,35	1,80	1,35	1,35	1,80	1,80	1,35	1,80	1,80
P20	1,35	1,80	1,80	1,35	1,80	1,35	1,35	1,80	1,80	1,35	1,80	1,80
P21	1,80	1,80	1,80	1,80	1,80	1,80	1,80	1,35	1,80	1,80	1,80	1,35
P23	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80
P24	1,35	1,80	1,80	1,35	1,80	1,35	1,35	1,80	1,80	1,35	1,80	1,80
P26	1,80	1,35	1,35	1,80	1,35	1,80	1,80	1,80	1,35	1,80	1,35	1,80
P27	1,80	1,80	1,80	1,80	1,80	1,80	1,80	1,35	1,80	1,80	1,80	1,35

Tabla A.1: Tiempos de transición $\tau_{i',i,j}$ [hrs] para la Etapa 03

Fuente: Material suplementario de Kopanos y col. (2010)

A.4. Etapa 04 y Etapa 05

	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10
P01	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P02	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P03	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P04	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P05	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P06	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P07	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P08	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P09	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P10	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P11	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P12	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P13	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P14	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P15	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P16	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P17	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P18	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P19	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P20	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P21	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P22	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P23	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P24	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P25	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P26	1,8	1,8	1,8	0,9	1,8	0,9	1,8	0,9	0,9	1,8
P27	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P28	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9
P29	0,9	0,9	1,8	1,8	1,8	1,8	0,9	1,8	1,8	1,8
P30	1,8	1,8	0,9	1,8	0,9	1,8	1,8	1,8	1,8	0,9

Continúa en página siguiente

Tabla A.2: Tiempos de transición $\tau_{i',i,j}$ [hrs] para las Etapa 04 y Etapa 05

Fuente: Material suplementario de Kopanos y col. (2010)

Continúa de página anterior

	P11	**P12**	**P13**	**P14**	**P15**	**P16**	**P17**	**P18**	**P19**	**P20**
P01	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P02	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P03	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P04	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P05	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P06	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P07	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P08	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P09	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P10	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P11	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P12	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P13	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P14	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P15	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P16	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P17	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P18	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P19	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P20	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P21	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P22	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P23	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P24	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P25	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P26	1,8	0,9	1,8	1,8	1,8	0,9	1,8	1,8	0,9	1,8
P27	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P28	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8
P29	1,8	1,8	0,9	0,9	1,8	1,8	0,9	0,9	1,8	0,9
P30	0,9	1,8	1,8	1,8	0,9	1,8	1,8	1,8	1,8	1,8

Continúa en página siguiente

Tabla A.2: Tiempos de transición $\tau_{i',i,j}$ [hrs] para las Etapa 04 y Etapa 05

Fuente: Material suplementario de Kopanos y col. (2010)

Continúa de página anterior

	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30
P01	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P02	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P03	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P04	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P05	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P06	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P07	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P08	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P09	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P10	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P11	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P12	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P13	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P14	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P15	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P16	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P17	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P18	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P19	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P20	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P21	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P22	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P23	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P24	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P25	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P26	1,8	1,8	0,9	1,8	1,8	0,9	1,8	1,8	1,8	1,8
P27	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P28	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9
P29	1,8	1,8	1,8	0,9	0,9	1,8	1,8	1,8	0,9	1,8
P30	0,9	0,9	1,8	1,8	1,8	1,8	0,9	0,9	1,8	0,9

Tabla A.2: Tiempos de transición $\tau_{i',i,j}$ [hrs] para las Etapa 04 y Etapa 05

Fuente: Material suplementario de Kopanos y col. (2010)

A.5. Etapa 06

	P01	P02	P03	P04	P05	P06	P07	P08
P01	0,0000	1,8000	1,2933	1,5183	1,5183	1,8000	1,8000	1,1250
P02	1,8000	0,0000	1,4625	1,4625	1,8000	1,8000	1,2933	1,1808
P03	1,2933	1,4625	0,0000	0,7875	1,4625	1,4625	1,4625	1,8000
P04	1,5183	1,4625	0,7875	0,0000	1,1808	0,9558	1,4625	1,2933
P05	1,5183	1,8000	1,4625	1,1808	0,0000	1,1250	1,1250	1,8000
P06	1,8000	1,8000	1,4625	0,9558	1,1250	0,0000	1,1250	1,2933
P07	1,8000	1,2933	1,4625	1,4625	1,1250	1,1250	0,0000	1,8000
P08	1,1250	1,1808	1,8000	1,2933	1,8000	1,2933	1,8000	0,0000
P09	0,9558	1,8000	1,2933	1,8000	1,4625	1,4625	1,4625	1,4625
P10	1,4625	1,8000	1,1250	1,1250	0,9558	1,4625	1,4625	1,4625
P11	1,8000	1,8000	1,4625	0,9558	1,4625	0,6183	1,4625	1,2933
P12	0,9558	0,8433	1,2933	1,8000	1,8000	1,8000	1,8000	0,8433
P13	1,8000	1,8000	1,4625	0,9558	1,4625	0,6183	1,4625	1,2933
P14	1,4625	1,2933	1,4625	1,4625	1,8000	1,8000	1,2933	1,4625
P15	1,1808	1,8000	1,8000	1,5183	1,0125	1,8000	1,4625	1,4625
P16	1,4625	1,8000	1,8000	1,2933	1,8000	0,6750	1,8000	0,9558
P17	1,1250	1,8000	1,8000	1,8000	0,9558	1,1808	1,4625	1,4625
P18	1,5183	1,8000	1,8000	1,0125	1,1808	0,9558	1,1250	1,2933
P19	1,8000	1,1250	1,1808	1,4625	1,2933	1,8000	1,8000	1,8000
P20	1,4625	1,8000	1,8000	1,8000	0,9558	1,4625	1,1250	1,4625
P21	1,2933	1,4625	0,6183	1,1250	1,4625	1,4625	1,4625	1,8000
P22	1,8000	1,4625	1,8000	1,2933	1,4625	1,0125	1,8000	1,2933
P23	1,4625	1,8000	1,1250	0,6183	1,4625	0,9558	1,4625	0,9558
P24	1,4625	1,2933	1,4625	1,4625	1,4625	1,4625	0,4500	1,4625
P25	1,4625	1,4625	1,1250	0,6183	1,8000	1,0125	1,8000	0,9558
P26	1,2933	1,8000	0,9558	1,4625	1,8000	1,8000	1,5183	1,8000
P27	1,8000	1,0125	1,4625	1,4625	0,7875	1,1250	0,6183	1,5183
P28	1,4625	1,1250	1,4625	0,9558	1,8000	1,2933	1,8000	0,6183
P29	1,4625	1,8000	1,4625	1,4625	1,2933	1,5183	1,8000	1,4625
P30	0,6750	1,4625	1,2933	1,5183	1,1808	1,8000	1,8000	1,4625

Continúa en página siguiente

Tabla A.3: Tiempos de transición $\tau_{i',i,j}$ [hrs] para la Etapa 06

Fuente: Material suplementario de Kopanos y col. (2010)

Continúa de página anterior

	P09	P10	P11	P12	P13	P14	P15	P16
P01	0,9558	1,4625	1,8000	0,9558	1,8000	1,4625	1,1808	1,4625
P02	1,8000	1,8000	1,8000	0,8433	1,8000	1,2933	1,8000	1,8000
P03	1,2933	1,1250	1,4625	1,2933	1,4625	1,4625	1,8000	1,8000
P04	1,8000	1,1250	0,9558	1,8000	0,9558	1,4625	1,5183	1,2933
P05	1,4625	0,9558	1,4625	1,8000	1,4625	1,8000	1,0125	1,8000
P06	1,4625	1,4625	0,6183	1,8000	0,6183	1,8000	1,8000	0,6750
P07	1,4625	1,4625	1,4625	1,8000	1,4625	1,2933	1,4625	1,8000
P08	1,4625	1,4625	1,2933	0,8433	1,2933	1,4625	1,4625	0,9558
P09	0,0000	1,8000	1,5183	1,2933	1,8000	1,1808	1,8000	1,8000
P10	1,8000	0,0000	1,4625	1,4625	1,4625	1,4625	0,9558	1,4625
P11	1,5183	1,4625	0,0000	1,8000	0,3753	1,1808	1,8000	0,9558
P12	1,2933	1,4625	1,8000	0,0000	1,8000	1,8000	1,4625	1,4625
P13	1,8000	1,4625	0,3753	1,8000	0,0000	1,4625	1,8000	0,9558
P14	1,1808	1,4625	1,1808	1,8000	1,4625	0,0000	1,8000	1,8000
P15	1,8000	0,9558	1,8000	1,4625	1,8000	1,8000	0,0000	1,1250
P16	1,8000	1,4625	0,9558	1,4625	0,9558	1,8000	1,1250	0,0000
P17	1,1250	1,2933	1,8000	1,8000	1,8000	1,4625	1,2933	1,5183
P18	1,4625	1,8000	1,2933	1,8000	1,2933	1,8000	0,8433	0,9558
P19	1,4625	1,2933	1,8000	1,4625	1,8000	1,8000	1,2933	1,8000
P20	1,1250	1,2933	1,8000	1,8000	1,5183	1,4625	0,9558	1,8000
P21	0,9558	1,4625	1,4625	1,2933	1,4625	1,8000	1,8000	1,8000
P22	1,8000	1,8000	0,9558	1,4625	0,9558	1,4625	1,8000	1,0125
P23	1,5183	0,7875	0,6750	1,4625	0,9558	1,1808	1,4625	0,9558
P24	1,8000	1,1250	1,4625	1,4625	1,4625	1,2933	1,1250	1,4625
P25	1,4625	1,4625	1,2933	1,8000	1,2933	1,1250	1,8000	1,0125
P26	1,2933	1,4625	1,4625	1,2933	1,4625	1,1250	1,4625	1,4625
P27	1,4625	1,4625	1,4625	1,5183	1,4625	1,2933	1,8000	1,8000
P28	1,4625	1,8000	1,2933	1,4625	1,2933	1,4625	1,8000	1,2933
P29	1,8000	0,6183	1,8000	1,4625	1,8000	1,4625	0,6183	0,8433
P30	1,2933	1,4625	1,8000	0,6183	1,8000	1,8000	1,1808	1,4625

Continúa en página siguiente

Tabla A.3: Tiempos de transición $\tau_{i',i,j}$ [hrs] para la Etapa 06

Fuente: Material suplementario de Kopanos y col. (2010)

Continúa de página anterior

	P17	**P18**	**P19**	**P20**	**P21**	**P22**	**P23**
P01	1,1250	1,5183	1,8000	1,4625	1,2933	1,8000	1,4625
P02	1,8000	1,8000	1,1250	1,8000	1,4625	1,4625	1,8000
P03	1,8000	1,8000	1,1808	1,8000	0,6183	1,8000	1,1250
P04	1,8000	1,0125	1,4625	1,8000	1,1250	1,2933	0,6183
P05	0,9558	1,1808	1,2933	0,9558	1,4625	1,4625	1,4625
P06	1,1808	0,9558	1,8000	1,4625	1,4625	1,0125	0,9558
P07	1,4625	1,1250	1,8000	1,1250	1,4625	1,8000	1,4625
P08	1,4625	1,2933	1,8000	1,4625	1,8000	1,2933	0,9558
P09	1,1250	1,4625	1,4625	1,1250	0,9558	1,8000	1,5183
P10	1,2933	1,8000	1,2933	1,2933	1,4625	1,8000	0,7875
P11	1,8000	1,2933	1,8000	1,8000	1,4625	0,9558	0,6750
P12	1,8000	1,8000	1,4625	1,8000	1,2933	1,4625	1,4625
P13	1,8000	1,2933	1,8000	1,5183	1,4625	0,9558	0,9558
P14	1,4625	1,8000	1,8000	1,4625	1,8000	1,4625	1,1808
P15	1,2933	0,8433	1,2933	0,9558	1,8000	1,8000	1,4625
P16	1,5183	0,9558	1,8000	1,8000	1,8000	1,0125	0,9558
P17	0,0000	1,4625	1,2933	0,6183	1,8000	1,5183	1,8000
P18	1,4625	0,0000	1,8000	1,1250	1,8000	1,2933	1,2933
P19	1,2933	1,8000	0,0000	1,2933	1,1250	1,4625	1,8000
P20	0,6183	1,1250	1,2933	0,0000	1,8000	1,8000	1,8000
P21	1,8000	1,8000	1,1250	1,8000	0,0000	1,8000	1,4625
P22	1,5183	1,2933	1,4625	1,8000	1,8000	0,0000	1,2933
P23	1,8000	1,2933	1,8000	1,8000	1,4625	1,2933	0,0000
P24	1,8000	1,4625	1,8000	1,4625	1,4625	1,8000	1,1250
P25	1,1808	1,2933	1,4625	1,4625	1,4625	1,0125	0,9558
P26	1,8000	1,4625	1,8000	1,8000	1,2933	1,4625	1,4625
P27	1,4625	1,4625	1,8000	1,4625	1,4625	1,4625	1,4625
P28	1,4625	1,2933	1,4625	1,4625	1,1808	1,2933	1,2933
P29	1,0125	1,4625	1,2933	1,2933	1,8000	1,5183	1,1250
P30	1,8000	1,5183	1,4625	1,8000	1,2933	1,1250	1,4625

Continúa en página siguiente

Tabla A.3: Tiempos de transición $\tau_{i',i,j}$ [hrs] para la Etapa 06

Fuente: Material suplementario de Kopanos y col. (2010)

Continúa de página anterior

	P24	P25	P26	P27	P28	P29	P30
P01	1,4625	1,4625	1,2933	1,8000	1,4625	1,4625	0,6750
P02	1,2933	1,4625	1,8000	1,0125	1,1250	1,8000	1,4625
P03	1,4625	1,1250	0,9558	1,4625	1,4625	1,4625	1,2933
P04	1,4625	0,6183	1,4625	1,4625	0,9558	1,4625	1,5183
P05	1,4625	1,8000	1,8000	0,7875	1,8000	1,2933	1,1808
P06	1,4625	1,0125	1,8000	1,1250	1,2933	1,5183	1,8000
P07	0,4500	1,8000	1,5183	0,6183	1,8000	1,8000	1,8000
P08	1,4625	0,9558	1,8000	1,5183	0,6183	1,4625	1,4625
P09	1,8000	1,4625	1,2933	1,4625	1,4625	1,8000	1,2933
P10	1,1250	1,4625	1,4625	1,4625	1,8000	0,6183	1,4625
P11	1,4625	1,2933	1,4625	1,4625	1,2933	1,8000	1,8000
P12	1,4625	1,8000	1,2933	1,5183	1,4625	1,4625	0,6183
P13	1,4625	1,2933	1,4625	1,4625	1,2933	1,8000	1,8000
P14	1,2933	1,1250	1,1250	1,2933	1,4625	1,4625	1,8000
P15	1,1250	1,8000	1,4625	1,8000	1,8000	0,6183	1,1808
P16	1,4625	1,0125	1,4625	1,8000	1,2933	0,8433	1,4625
P17	1,8000	1,1808	1,8000	1,4625	1,4625	1,0125	1,8000
P18	1,4625	1,2933	1,4625	1,4625	1,2933	1,4625	1,5183
P19	1,8000	1,4625	1,8000	1,8000	1,4625	1,2933	1,4625
P20	1,4625	1,4625	1,8000	1,4625	1,4625	1,2933	1,8000
P21	1,4625	1,4625	1,2933	1,4625	1,1808	1,8000	1,2933
P22	1,8000	1,0125	1,4625	1,4625	1,2933	1,5183	1,1250
P23	1,1250	0,9558	1,4625	1,4625	1,2933	1,1250	1,4625
P24	0,0000	1,8000	1,5183	0,9558	1,8000	1,4625	1,4625
P25	1,8000	0,0000	1,4625	1,8000	0,6183	1,1808	1,8000
P26	1,5183	1,4625	0,0000	1,8000	1,8000	1,1250	1,2933
P27	0,9558	1,8000	1,8000	0,0000	1,8000	1,8000	1,4625
P28	1,8000	0,6183	1,8000	1,8000	0,0000	1,8000	1,8000
P29	1,4625	1,1808	1,1250	1,8000	1,8000	0,0000	1,4625
P30	1,4625	1,8000	1,2933	1,4625	1,8000	1,4625	0,0000

Tabla A.3: Tiempos de transición $\tau_{i',i,j}$ [hrs] para la Etapa 06

Fuente: Material suplementario de Kopanos y col. (2010)

B. Tiempos de procesamiento $pt_{i,j}$ [hrs]

	Etapa 01		Etapa 02			Etapa 03		
	J01	**J02**	**J03**	**J04**	**J05**	**J06**	**J07**	**J08**
P01	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P02	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P03	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P04	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P05	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P06	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P07	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P08	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P09	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P10	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P11	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P12	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P13	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P14	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P15	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P16	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P17	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P18	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P19	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P20	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P21	0,9000	0,9000	1,3050	1,3050	-	-	2,0205	2,0205
P22	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P23	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P24	0,9000	0,9000	1,3050	1,3050	-	-	2,0205	2,0205
P25	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P26	0,9000	0,9000	1,3050	1,3050	2,0979	1,6335	2,0205	2,0205
P27	0,9000	0,9000	1,3050	1,3050	-	-	2,0205	2,0205
P28	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P29	0,9000	0,9000	1,3050	1,3050	-	-	-	-
P30	0,9000	0,9000	1,3050	1,3050	-	-	-	-

Continúa en página siguiente

Tabla B.1: Tiempos de procesamiento $pt_{i,j}$ [hrs]

Fuente: Material suplementario de Kopanos y col. (2010)

Continúa de página anterior

	Etapa 04			Etapa 05			Etapa 06		
	J09	**J10**	**J11**	**J12**	**J13**	**J14**	**J15**	**J16**	**J17**
P01	-	-	0,5778	0,2250	0,3780	0,3780	1,9818	1,9818	0,5661
P02	-	-	0,6894	0,2250	0,3780	0,3780	2,3634	2,3634	0,6750
P03	0,3339	0,3339	-	0,6750	1,1340	1,1340	0,3276	0,3276	0,0936
P04	-	-	0,5832	0,4500	0,7560	0,7560	1,9998	1,9998	0,5715
P05	0,2223	0,2223	-	0,2250	0,3780	0,3780	0,2178	0,2178	0,0621
P06	2,0403	2,0403	0,5832	0,2250	0,3780	0,3780	1,9998	1,9998	0,5715
P07	-	-	0,1062	0,2250	0,3780	0,3780	0,3636	0,3636	0,1035
P08	-	0,3708	0,1062	0,6750	1,1340	1,1340	0,3636	0,3636	0,1035
P09	-	-	0,1008	0,6750	1,1340	1,1340	0,3456	0,3456	0,0990
P10	-	-	0,2646	0,4500	0,7560	0,7560	0,9090	0,9090	0,2601
P11	-	-	0,7947	0,4500	0,7560	0,7560	2,7270	2,7270	0,7794
P12	-	1,8549	-	0,4500	0,7560	0,7560	1,8180	1,8180	0,5193
P13	-	0,7416	-	0,2250	0,3780	0,3780	0,7272	0,7272	0,2079
P14	0,3528	0,3528	0,1008	0,4500	0,7560	0,7560	0,3456	0,3456	0,0990
P15	2,0403	2,0403	0,5832	0,4500	0,7560	0,7560	1,9998	1,9998	0,5715
P16	-	-	0,2124	0,4500	0,7560	0,7560	0,7272	0,7272	0,2079
P17	1,1133	1,1133	-	0,4500	0,7560	0,7560	1,0908	1,0908	0,3114
P18	-	-	0,6363	0,4500	0,7560	0,7560	2,1816	2,1816	0,6237
P19	-	-	0,5616	0,4500	0,7560	0,7560	1,9269	1,9269	0,5508
P20	-	0,3339	0,0954	0,6750	1,1340	1,1340	0,3276	0,3276	0,0936
P21	-	0,2781	-	0,6750	1,1340	1,1340	0,2727	0,2727	0,0783
P22	-	-	0,5409	0,6750	1,1340	1,1340	1,8549	1,8549	0,5301
P23	-	2,0403	0,5832	0,6750	1,1340	1,1340	1,9998	1,9998	0,5715
P24	-	2,2257	0,6363	0,2250	0,3780	0,3780	2,1816	2,1816	0,6237
P25	-	1,1133	-	0,6750	1,1340	1,1340	1,0908	1,0908	0,3114
P26	-	1,9476	0,5562	0,4500	0,7560	0,7560	1,9089	1,9089	0,5454
P27	-	2,0403	0,5832	0,2250	0,3780	0,3780	1,9998	1,9998	0,5715
P28	-	-	0,1062	0,2250	0,3780	0,3780	0,3636	0,3636	0,1035
P29	1,6695	1,6695	-	0,2250	0,3780	0,3780	1,6362	1,6362	0,4671
P30	-	3,3390	-	0,2250	0,3780	0,3780	3,2724	3,2724	0,9351

Tabla B.1: Tiempos de procesamiento $pt_{i,j}$ [hrs]

Fuente: Material suplementario de Kopanos y col. (2010)

C. Restricciones que se incluyen para cada modelo

	Restricciones del modelo					
	Asignación	**Procesamiento y sincronización de tiempos**	**Secuenciación**	**Función objetivo**	**Ajuste adicional**	**Iteración Interna entre modelos**
METODOLOGÍA PROPUESTA	ec. (3.1); ec. (3.5); ec. (3.6); ec. (3.7); ec. (4.7)	ec. (3.8); ec. (3.9); ec. (3.10); ec. (3.11)	ec. (3.12); ec. (3.17); ec. (3.18); ec. (3.19); ec. (3.20); ec. (3.21); ec. (3.22)	ec. (3.23); ec. (3.24)	ec. (3.27); ec. (3.28); ec. (3.35); ec. (3.36); ec. (3.37); ec. (3.38)	ec. (4.2); ec. (4.3); ec. (4.4)
FULL	ec. (3.1); ec. (3.5); ec. (3.6)		ec. (3.12); ec. (3.17); ec. (3.19)		ec. (3.27); ec. (3.28); ec. (3.35); ec. (3.36); ec. (3.37)	-
FULL FIXED	ec. (3.1); ec. (4.7)					

Tabla C.1: Ecuaciones que se incluyen en la metodología matemática-algorítmica propuesta y las formulaciones FULL y FULL FIXED

Fuente: Elaboración propia

D. Construcción progresiva de la solución durante la Iteración Interna para el problema de 30 lotes

It. Principal	**Etapa** l^{act}	**It. Interna**	$z^{\text{lb\_approx}}_{a,b,l^{\text{act}}}$	$z^{\text{sol\_approx}}_{a,b,l^{\text{act}}}$	$z^{\text{lb}}_{a,b,l^{\text{act}}}$	$z^{\text{sol}}_{a,b,l^{\text{act}}}$	**Etapa** l^{sel}
$a=1$	l_4	$b=1$	21,9447	21,9447	22,0440	23,8743	l_4
		$b=2$	**21,6573**	**23,5116**	**23,6574**	**23,6574**	
		$b=3$	21,7945	23,9436	21,8996	24,0624	
	l_6	$b=1$	21,9447	21,9447	-	-	-
	l_2	$b=1$	21,9447	21,9447	-	-	-
	l_1	$b=1$	21,9447	21,9447	-	-	-
	l_3	$b=1$	21,6386	21,9447	-	-	-
	l_5	$b=1$	21,9447	21,9447	-	-	-
$a=2$	l_6	$b=1$	**23,6574**	**23,6574**	**23,6574**	**24,8472**	l_6
	l_2	$b=1$	23,6574	23,6574	-	-	-
	l_1	$b=1$	23,6574	23,6574	-	-	-
	l_3	$b=1$	23,6574	23,6574	-	-	-
	l_5	$b=1$	23,6574	23,6574	-	-	-
$a=3$	l_2	$b=1$	24,8472	24,8472	26,1018	26,1018	l_2
		$b=2$	**25,2234**	**25,2234**	**25,2234**	**25,2234**	
	l_1	$b=1$	24,8472	24,8472	-	-	-
	l_3	$b=1$	24,8472	24,8472	-	-	-
	l_5	$b=1$	24,8472	24,8472	-	-	-

Continúa en página siguiente

Tabla D.1: Evolución de las cotas de la Iteración Interna para el problema de 30 lotes

Fuente: Elaboración propia

Continúa de página anterior

It. Principal	**Etapa** l^{act}	**It. Interna**	$z^{\text{lb\_approx}}_{a,b,l^{\text{act}}}$	$z^{\text{sol\_approx}}_{a,b,l^{\text{act}}}$	$z^{\text{lb}}_{a,b,l^{\text{act}}}$	$z^{\text{sol}}_{a,b,l^{\text{act}}}$	**Etapa** l^{sel}
$a=4$	l_1	$b=1$	25,2234	25,2234	25,2234	25,2234	l_1
	l_3	$b=1$	25,2234	25,2234	26,0631	26,0631	l_3
		$b=2$	25,2234	25,2234	25,2612	25,2612	
		$b=3$	25,2234	25,2234	25,4142	25,4142	
		$b=4$	25,2234	25,2234	25,2612	25,2612	
		$b=5$	25,2234	25,2234	25,2612	25,2612	
		$b=6$	25,2234	25,2234	26,0478	26,0478	
		$b=7$	25,2234	25,2234	26,0172	26,0172	
		$b=8$	25,2234	25,2234	25,2612	25,2612	
		$b=9$	25,2234	25,2234	25,2612	25,2612	
		$b=10$	25,2234	25,2234	25,2612	25,2612	
	l_5	$b=1$	25,2234	25,2234	27,0864	27,0864	l_5
		$b=2$	25,3242	25,3242	25,4322	25,4322	
		$b=3$	25,3836	25,3836	25,6023	25,6023	
		$b=4$	25,3836	25,3836	25,6545	25,6545	
		$b=5$	25,3836	25,3836	25,4322	25,4322	
		$b=6$	**25,3836**	**25,3836**	**25,3836**	**25,3836**	
$a=5$	l_1	$b=1$	25,3836	25,3836	25,3836	25,3836	l_1
	l_3	$b=1$	**25,3836**	**25,3836**	**25,3836**	**25,3836**	l_3
$a=6$	l_1	$b=1$	**25,3836**	**25,3836**	**25,3836**	**25,3836**	l_1

Tabla D.1: Evolución de las cotas de la Iteración Interna para el problema de 30 lotes

Fuente: Elaboración propia

E. Diagrama de Gantt para el problema de 20 lotes

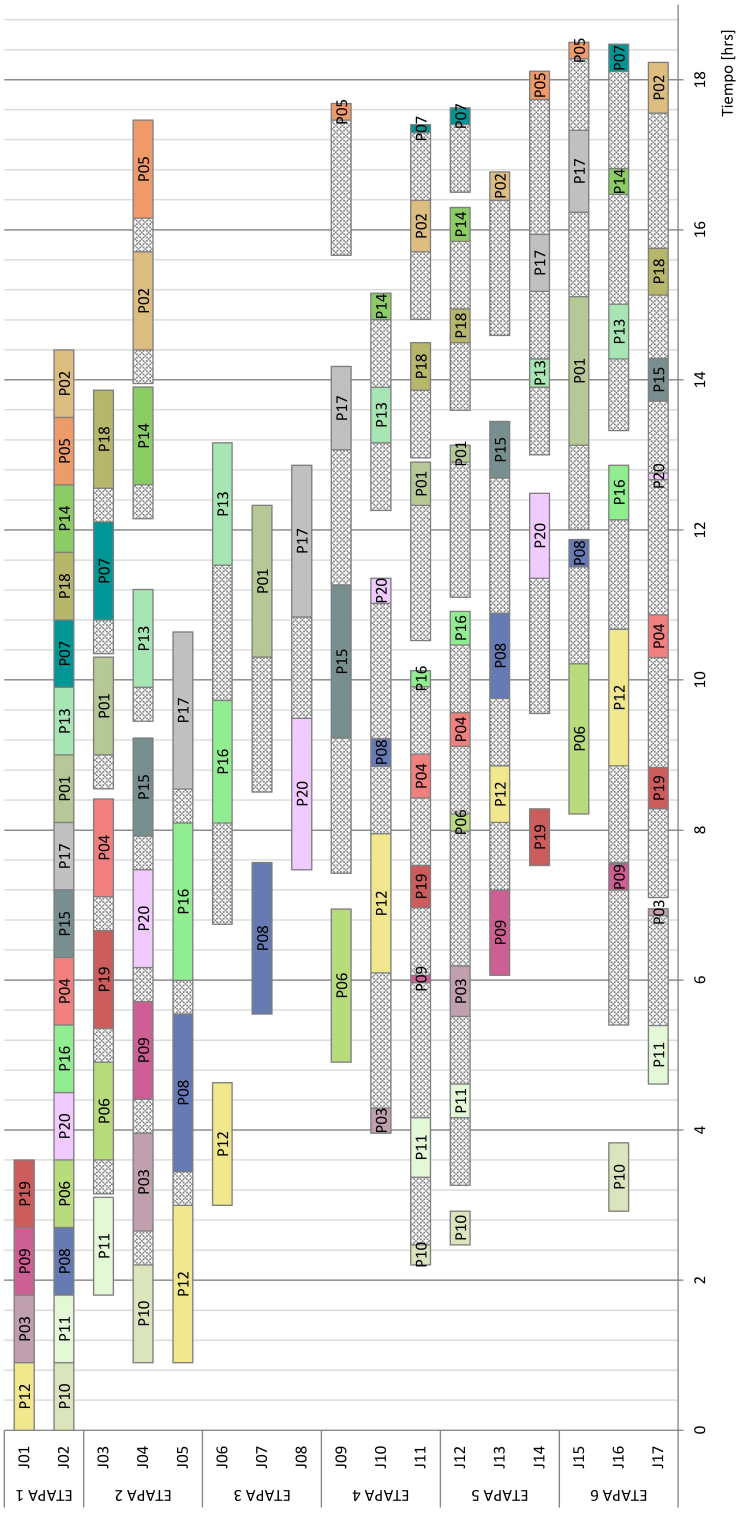

Figura E.1: Diagrama de Gantt de la mejor solución obtenida ($MK = 18,50[hrs]$) para el problema de 20 lotes, con parámetros $m_j^0 = 3$ y $T = 600[s]$

Fuente: Elaboración propia

F. Diagrama de Gantt para el problema de 25 lotes

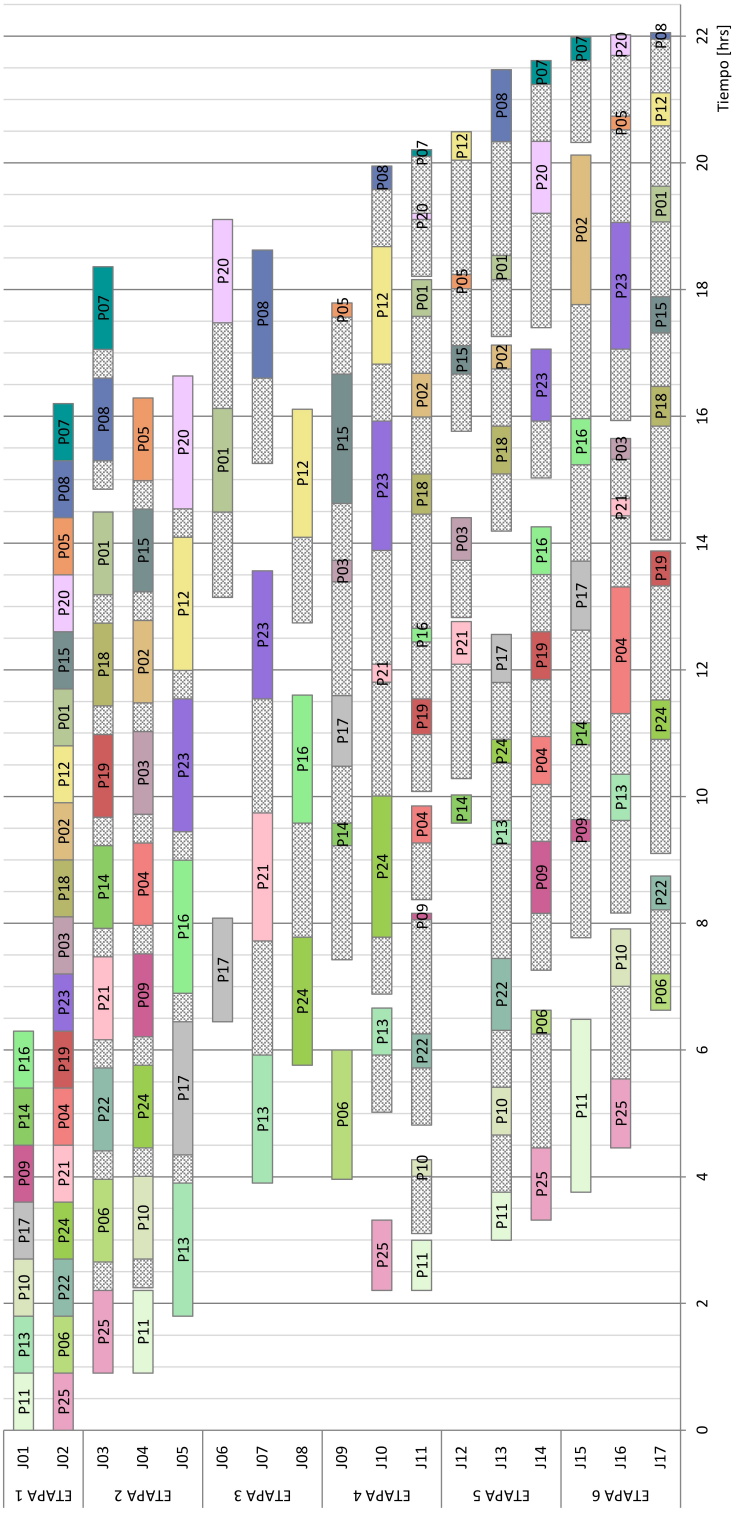

Figura F.1: Diagrama de Gantt de la mejor solución obtenida ($MK = 22,05[hrs]$) para el problema de 25 lotes, con parámetros $m_j^0 = 2$ y $T = 1{,}200[s]$

Fuente: Elaboración propia