UNIVERSIDAD NACIONAL DEL SUR

TESIS DE DOCTORADO EN INGENIERÍA

Diseño y desarrollo de estructuras de planificación eficientes a través de técnicas de simulación y optimización aplicables a entornos productivos complejos

Daniel Alejandro Rossit

BAHIA BLANCA ARGENTINA

2017

PREFACIO

Esta tesis se presenta como parte de los requisitos para optar al grado Académico de Doctor en Ingeniería, de la Universidad Nacional del Sur y no ha sido presentada previamente para la obtención de otro título en esta Universidad u otra. La misma contiene los resultados obtenidos en investigaciones llevadas a cabo en el ámbito del Departamento de Ingeniería durante el período comprendido entre el 10 de julio de 2013 y el 22 de febrero de 2017, bajo la dirección del Dr. Fernando Tohmé y del Dr. Mariano Frutos.

……..
Daniel Alejandro Rossit

UNIVERSIDAD NACIONIAL DEL SUR
Secretaría General de Posgrado y Educación Continua

La presente tesis ha sido aprobada el ……01…/……06……/……2017……,
mereciendo la calificación de ……10……(……diez……)

AGRADECIMIENTOS

A mis directores, los Dres. Fernando Tohmé y Mariano Frutos por ser guías incondicionales y apoyo permanente.

A mis compañeros de trabajo: Diego Rossit, Antonella Cavallin, Daniel Carbone, Adrián Toncovich, Nancy López y Gabriel Recabarren.

Al Dr. Diego Broz por su generosidad y amistad.

Al Departamento de Ingeniería, tanto docentes como no docentes, por darme su apoyo.

Al CONICET por darme esta posibilidad de formarme.

A docentes y personal de la UNS por brindarme lugar de trabajo y su colaboración.

Al Dr. Óscar Vásquez por contribuir en mi formación y a los miembros de la USACH por recibirme.

Al Dr. Andrés Weintraub y el Mg. Cristóbal Pais por su colaboración y a miembros de la Universidad de Chile por recibirme.

A docentes y compañeros de mi carrera de grado.

A la sociedad argentina por su apoyo.

A mis amigos; a mis hermanos Diego y José; a mis padres, Diana y Carlos; a mi esposa, Antonela y nuestro hijo; a Jesús y la Virgen María; a todos ellos mi más profundo agradecimiento por su acompañamiento y amor.

"… aunque conociera todos los misterios y toda la ciencia, … si no tengo amor, no soy nada."

1 Corintios 13:2

Contenidos:

RESUMEN 6

ABSTRACT 8

1. Capítulo 1: INTRODUCCIÓN 9

1.1. Sistemas de Planificación y control de la Producción 10

1.1.1. Scheduling de la producción 12

1.2. Configuración Flow shop 13

1.3. Flow shop no-permutativo 15

2. Capítulo 2: ESTADO DEL ARTE DE PROBLEMAS FLOW SHOP NO-PERMUTATIVOS 17

2.1. Notación y presentación de los distintos tipos de problemas NPFS 18

2.2. Revisión de la literatura NPFS 21

2.2.1. Objetivos basados en tiempos de finalización 22

2.2.2. Objetivos basados en fechas de entrega 28

2.2.3. Objetivos basados en costos 28

2.2.4. Estudios experimentales mono-objetivos 29

2.2.5. Estudios experimentales multi-objetivos 30

2.3. Un análisis cuantitativo de la literatura 35

2.3.1. Análisis bibliométrico 37

2.3.2. Casos Especiales 39

2.4. Conclusiones del capítulo 40

3. Capítulo 3: IMPLEMENTACIÓN DE SUBLOTEO EN PROBLEMAS NPFS 43

3.1. Lotes de transferencia 44

3.2. Metodología de subloteo 45

3.3. Modelos matemáticos 48

3.3.1. Modelo mixto entero NPFS 49
3.3.2. Modelo mixto entero NPFS con subloteo 51
3.3.3. Modelo mixto entero PFS con subloteo 53
3.4. Experimentación Computacional 54
3.5. Resultados 55
3.5.1. Costo computacional del subloteo 55
3.5.2. Evaluación del makespan aplicando subloteo 57
3.6. Conclusiones del capítulo 63
4. Capítulo 4: ESTUDIO DE CAMINOS CRÍTICOS DE PROBLEMAS NPFS Y PFS 64
4.1. Estudio de caminos críticos 65
4.2. Definiciones y ejemplo base 65
4.3. Propiedades estructurales y de dominancia 68
4.4. Experimentación no-paramétrica 79
4.4.1. Experimentos 80
4.4.2. Resultados 80
4.5. Experimentación paramétrica 83
4.5.1. Formulaciones mixto-enteros 83
4.5.2. Escenarios de prueba 85
4.5.3. Resultados 86
4.6. Conclusiones del capítulo 89
5. Capítulo 5: CONCLUSIONES 91
5.1. Conclusiones 92
Referencias 95

Lista de Figuras

Figura 1-2. Representación de los conjuntos de soluciones PFS y NPFS. 15

Figura 1-3. Representación Gantt de problema PFS y NPFS para un caso de dos trabajos n = 2, y cuatro máquinas, m = 4. 16

Figura 2-1. Número de artículos NPFS publicados en periodos de 5 años. 36

Figura 2-2. Distribución de las funciones objetivos consideradas en la literatura. 37

Figura 2-3. Distribución de los métodos de optimización utilizados. 37

Figura 3-1. Ilustraciones de una solución sin aplicar subloteo a), y de una solución aplicando subloteo b). 46

Figura 3-2. Tiempos de cómputo respecto al incremento en el número de sublotes. 57

Figura 3-3. Descenso del makespan al incrementar el número máximo de sublotes disponibles 59

Figura 3-4. Uso promedio de sublotes por producto respecto al número máximo de sublotes admitidos por producto. 60

Figura 3-5. Reducción del makespan en relación al máximo número de sublotes admitidos para el caso de 3 productos. 62

Figura 3-6. Reducción del makespan en relación al máximo número de sublotes admitidos para el caso de 5 máquinas. 62

Figura 4-1. Representación de Gantt de las soluciones PFS y NPFS para el Ejemplo 1 66

Figura 4-2. Representación en grafo de las soluciones PFS y NPFS para el caso de dos trabajos n=2 y cuatro máquinas m=4 mostrado en la Figura 4.1. 67

Figura 4-3. Representación en diagrama de Gantt y en grafo para el caso de n = 2 trabajos, m máquinas y $|S| = 1$, definiendo a M_k como la máquina de mutación. 68

Figura 4-4. Ilustraciones de la solución de un problema NPFS con dos trabajos n = 2, cinco máquinas m = 5, y dos máquinas de mutación $S = \{M_3, M_4\}$. 73

Figura 4-5. Dominancia de caminos críticos. 75

Figura 4-6. Caminos críticos del Ejemplo 1, el camino crítico superior corresponde al ordenamiento PFS (Figura 4.1.a), y de abajo al ordenamiento NPFS (Figura 4.1.b)..... 78

Lista de Tablas

Tabla 2.1. Funciones objetivos para problemas flow shop. 21

Tabla 2.2. Revisión de la literatura NPFS. 31

Tabla 2.3. Lista de las revistas que han publicado dos o más artículos sobre NPFS. 38

Tabla 2.4. Citas de los artículos NPFS tomadas de Google Scholar, agosto 2016..... 39

Tabla 2.5. Resultados especiales NPFS, considerando makespan como objetivo. 39

Tabla 3.1. Tamaño de las instancias evaluadas en términos de restricciones, variables y tiempos promedio de cómputo. 56

Tabla 3.2. Resultados NPFS, valores de makespan y número promedio de sublotes. ... 58

Tabla 3.3. Resultados PFS, valores de makespan y número promedio de sublotes. 58

Tabla 3.4. Reducción del makespan expresada en porcentajes respecto al caso sin subloteo..... 61

Tabla 4.1. Tiempos de procesamiento del Ejemplo 1..... 66

Tabla 4.2. Pseudocódigo del algoritmo exhaustivo ExA..... 80

Tabla 4.3. Resultados experimentales no-paramétricos obtenidos con algoritmo ExA. 81

Tabla 4.4. Resultados experimentales paramétricos obtenidos con programación matemática..... 86

Tabla 4.5. Cantidad de instancias para las cuales NPFS es mejor que PFS y la mejora promedio..... 88

RESUMEN

La tesis aborda problemas de secuenciamiento en entornos productivos del tipo flow shop en los que se retira la condición de ordenamientos permutativos. Este tipo de problemas se encuentran inmersos dentro de los sistemas de Planificación y Control de la Producción que dan soporte en la toma de decisiones a las organizaciones o empresas que producen bienes del tipo manufactura. Como primera aproximación al problema se presenta una revisión exhaustiva de la literatura científica sobre problemas flow shop no permutativos (NPFS). De esta forma se pudo enmarcar la tesis doctoral en la literatura de la temática y se definió concretamente la contribución a la literatura del tema. Como resultado del estudio de la literatura se planteó abordar los problemas NPFS desde una perspectiva que permitiera estudiar la estructura de las soluciones para así poder compararlos con los resultados de los problemas flow shop permutativos (PFS). Primeramente, se propuso estudiar los problemas NPFS con makespan como función objetivo bajo un nuevo enfoque de planificación. Para ello se utilizará la metodología de lotes de transferencia o *lot streaming*, la cual modifica el problema clásico de secuenciamiento incorporando nuevas variables de decisión al problema a optimizar. Las nuevas variables de decisión van asociadas al dimensionamiento del tamaño del lote de producción. Este estudio reportó resultados para NPFS y PFS bastante similares, aunque el caso NPFS obtuvo leves mejoras para las instancias más grandes. No obstante, el esfuerzo computacional requerido para resolver el caso NPFS fue considerablemente mayor que requerido para PFS. A partir de estos resultados, se planteó un estudio conceptual de las soluciones NPFS y PFS para el caso de dos trabajos en términos de caminos críticos (conjunto de actividades que definen el makespan) que posibilitaron caracterizar ambos conjuntos de soluciones de forma no-paramétrica, es decir, independizarse de los parámetros que definen un escenario. De este estudio de caminos críticos, se pudieron analizar una serie de propiedades y definir criterios de dominancia entre las soluciones NPFS y PFS que permitirían reducir el espacio factible. A su vez, el estudio no-paramétrico permitió realizar una serie experimentaciones computacionales innovadoras, que dieron sustento al desarrollo de algunas hipótesis sobre la relación de las soluciones NPFS y PFS para el caso de que los problemas sean evaluados en escenarios paramétricamente definidos. Para evaluar estas hipótesis se implementaron

experimentaciones paramétricas a través de programación matemática, las cuales validaron las hipótesis planteadas.

ABSTRACT

This dissertation focuses on non-permutation scheduling problems in flow shop production settings. These problems, proper of systems of Production Planning and Control, are central to the decision making processes in organizations or firms producing manufactured goods. A first look into these problems requires a thorough review of the scientific literature on non-permutation flow shop (NPFS) problems. This review provides a background on this issue and defines precisely the contribution of this thesis to the literature. A novel and interesting approach to address NPFS problems is by studying the structure of the solutions, comparing it to the corresponding structure of permutation flow shop (PFS) problems. In this light, we study NPFS problems where makespan is minimized considering a special planning technique involving lot streaming. This technique modifies the regular scheduling problem adding new decision variables, related to production lot sizing. From the implementation of lot streaming on these problems we obtain new results. The main conclusion is that the makespans of NPFS and PFS problems are quite similar, although NPFS yields a better makespan for larger instances. The computational effort required by NPFS problems is much larger than that of solving PFS ones. Up from these results, we develop a new approach to the analysis of solutions to NPFS and PFS problems. We center on the two jobs case, and on the concept of critical path (enumerating the set of activities that defines makespan). This allows the non-parametric characterization of the solutions, freeing them from the dependence on particular parameters. We analyze a family of properties that yield dominance criteria for the comparison between NPFS and PFS solutions, reducing, in general, the number of feasible solutions. In addition, this non-parametric method allows the design of novel computational experimental frameworks, yielding new insights on the relation between NPFS and PFS solutions for parametric scenarios. To assess these hypotheses, we obtain via an application of mathematical programming a set of parametric results that we test in experiments that confirm the aforementioned hypotheses.

1. CAPÍTULO 1: INTRODUCCIÓN

Introducción

1.1. Sistemas de Planificación y control de la Producción

Los sistemas de Planificación y control de la Producción (PyCP) tienen por objetivo dar soporte a todo el proceso de producción de manufactura. Esencialmente, administran el flujo de materiales y la información con el fin de cumplir el objetivo de la organización de satisfacer la demanda del cliente. Para ello utilizan los recursos de la organización (personal, equipamiento, capital, etc.), información interna (gestión de la información) y externa (demanda de clientes). En este sentido Vollmann et al (2005) define la tarea fundamental de los sistemas de PyCP como "*administrar con eficiencia el flujo de materiales, la utilización del personal y del equipo y responder a los requerimientos de los clientes utilizando la capacidad de los proveedores, de las instalaciones internas y (en algunos casos) la de los propios clientes para cumplir la demanda del cliente*". Los sistemas PyCP dan soporte a la organización tanto a nivel estratégico (largo plazo), táctico (mediano) como al operativo (corto plazo).

A nivel estratégico los sistemas PyCP brindan la suficiente información como para determinar la capacidad que va a ser requerida para satisfacer la demanda de los clientes en el futuro (largo plazo), como también las características de esa capacidad (dónde instalar nuevas instalaciones, qué tipo de producto debería abastecerse desde cada centro, etc.), y la selección de proveedores también se asocia a este nivel. Generalmente estas decisiones involucran un fuerte desembolso de capital, y los horizontes temporales suelen comprender varios años. Una vez definida la estructura de la producción a nivel estratégico, se deben tomar otro tipo de decisiones que son de nivel táctico. El nivel táctico considera horizontes temporales más cortos, generalmente menores a dos años, incluso para algún tipo de industria puede ser alrededor de un año. Y para el corto plazo, se toman decisiones de nivel operativo que suelen involucrar de uno a dos meses.

En la etapa táctica se busca, principalmente, empatar la demanda con la oferta, considerando las necesidades en volúmenes de mezcla de productos, como también capacidad exacta de materiales y de producción. Para definir el volumen de mezcla de productos a producir es necesario conocer la demanda de los clientes para el horizonte temporal contemplado. Generalmente, no se cuenta con esa información, al menos de forma certera, por lo que suele ser estimada utilizando criterios alineados con la estrategia de la organización y conocimientos del mercado, muchas de las herramientas de apoyo a esta estimación analizan información histórica. Para definir la capacidad exacta de

materiales y de producción, se utiliza como información primaria la demanda (estimada) y se planifica el uso de los recursos de producción para el año siguiente. De esta forma se especifica la capacidad laboral requerida, el uso de horas extras, los contratos con los proveedores, si es necesario subcontratar alguna actividad, etc. Es claro que las estimaciones de la demanda pierden calidad al pretender ser muy detalladas, por lo que resulta más fehaciente estimar las ventas anuales totales de un producto, que estimar las ventas semanales del mismo. Es por eso que, generalmente, se suele hacer un plan agregado de producción, en donde se utiliza como información la venta estimada de una familia o grupo de productos (en lugar de pormenorizar por cada producto individualmente). De esta forma se puede planificar la capacidad agregada necesaria para producir esta cantidad.

En el corto plazo, a nivel operativo, la planificación incorpora un nivel de detalle mucho mayor para cumplir con los requerimientos de producción. Esta planificación involucra tiempo, personal, material, equipo e instalaciones. En este nivel es donde la programación de actividades o *scheduling* tiene lugar. El scheduling se encarga de la asignación de recursos a las tareas que deben realizarse durante un horizonte de tiempo para concretar un plan de producción, y los horizontes temporales suelen ser menores a una semana.

Los tres niveles de planificación considerados en los sistemas PyCP tienen diferentes horizontes de tiempos y alcances, en cuanto a decisiones. Esto conlleva a que el tipo de información requerida en cada nivel para tomar decisiones sea distinto. Por un lado, las decisiones de horizontes largos se toman en base a especulaciones sobre la evolución del mercado y de los recursos internos de la organización, por lo que no se requieren técnicas cuantitativas muy sofisticadas para dar apoyo. Además, intentar formalizar este proceso de decisión puede tener poco o nulo sentido, ya que los entornos dinámicos y cambiantes en los que están inmersas las organizaciones muy difícilmente requieran, a nivel estratégico, tomar dos veces las mismas decisiones (Framinan et al. 2014). Se suele utilizar el juicio de expertos y la información de algunos indicadores claves para el negocio. En contrapartida, en las decisiones de horizontes cortos, los escenarios suelen ser más repetitivos por lo que formalizar y generar técnicas/modelos para dar apoyo a la toma de decisiones cobra mayor interés para la organización. Es por eso que se suelen utilizar métodos cuantitativos para estos plazos, dada la alta

repetitividad de escenarios y decisiones, como también por la gran cantidad de información que se requiere procesar y analizar (tiempos de procesamiento, fechas de entrega, lista de órdenes, etc).

1.1.1. Scheduling de la producción

Tal como ya lo hemos definido de modo amplio, el scheduling es la asignación de recursos a las tareas que definen el plan de producción. Dada esta definición, es posible extender la aplicabilidad del scheduling a otros sistemas distintos de los PyCP, tales como asignación de enfermeras a los turnos de los hospitales, programación de los aviones en los vuelos de los aeropuertos, unidades de procesamientos en centros de cómputo, etc. En nuestro caso nos centraremos en su implementación en los sistemas PyCP. Dado que el scheduling implica directamente la asignación de recursos, su complejidad e importancia crecerá en la medida en que el uso de los recursos entre en conflicto.

El scheduling es un proceso de planificación fuertemente ligado a la naturaleza y estructura de una organización, lo que implica que su proceso de toma de decisiones es muy dependiente de la organización. Sin embargo, algunas características pueden considerarse independientes del tipo de organización, en Framinan et al (2014) señalan las siguientes condiciones comunes del scheduling para todos los sistemas de producción:

- Son decisiones complejas. Y su complejidad seguirá aumentando debido a la tendencia actual de incrementar la flexibilidad en los pedidos de los clientes, lo que deriva en aumentar la complejidad del proceso productivo.
- Tiene horizonte temporal corto, el ciclo de vida de un ordenamiento o *schedule* suele ser muy corto.
- A pesar de ser decisiones de horizontes cortos, suelen ser muy importantes para la línea de producción de una organización ya que impactan directamente sobre los costos de producción y los tiempos de entrega, lo que afecta a la competitividad de la empresa (tiempo de entrega y costos)
- Por estar situados en el centro de las operaciones de producción, las restricciones y objetivos del scheduling son muy dependientes del tipo de empresa.
- Suelen ser decisiones estructuradas, lo que permite que la información, los criterios y las limitaciones sean fáciles de formalizar.

Generalmente los problemas de scheduling se dan en producciones con lotes de tamaño chicos y medianos, con una heterogeneidad de media a alta. En el caso de producciones masivas, no suele haber mayores complicaciones de scheduling, lo mismo que en producciones de muy alto volumen, ya que se suele contar con líneas de producción diseñadas y con disponibilidad exclusivas para dichas producciones. A su vez el modelo de scheduling adoptado dependerá muy significativamente del tipo de configuración productiva que tiene la organización. La configuración productiva es la forma en que se relacionan los trabajos a procesar y el conjunto de recursos a asignar. Las configuraciones más comúnmente utilizadas son job shop, flow shop y sus distintas variantes. En la configuración job shop, cada trabajo a ser procesado posee una ruta particular de operaciones a realizarse en un conjunto de máquinas. En el caso de flow shop todos los trabajos siguen la misma secuencia de máquinas. Esta última es la configuración que se estudiará en este trabajo.

1.2. Configuración Flow shop

La configuración productiva del tipo flow shop consiste en un grupo de máquinas que realizan sus tareas de forma secuencial. Primero debe realizarse la primera tarea en la primera máquina, luego en la segunda máquina, y así sucesivamente. La Figura 1.1 ilustra de forma esquemática como es una configuración flow shop. En la Figura 1.1 se muestran 4 centros de trabajo representados por cuatro máquinas. Las flechas indican el sentido que recorren los trabajos que son procesados en dicho sistema. Todos los trabajos pasan por todas las máquinas y en la misma secuencia.

Figura 1.1. Diagrama de una configuración de producción flow shop

En un sentido más formal un sistema flow shop consiste en un conjunto $J = \{J_1, \ldots, J_n\}$ de n trabajos y un conjunto $M = \{M_1, \ldots, M_m\}$ de m máquinas. Cada trabajo consiste de un conjunto $O_j = \{O_{j,1}, \ldots, O_{j,m}\}$ de m operaciones, donde la i - *ésima* operación se ejecuta en la máquina M_i. Además, la operación $O_{j,i+1}$ puede comenzar si y solo si la

operación $O_{j,i}$ fue completada. Cada operación $O_{j,i}$ tiene un tiempo de procesamiento $p_{j,i}$ ϵ $\mathbb{N}$. Cada máquina M_i puede procesar sólo una operación a la vez. Un ordenamiento de los trabajos σ se define como la secuencia en que cada máquina procesará los trabajos, y en consecuencia las operaciones que deberá realizar. Si este ordenamiento de los trabajos en cada máquina es el mismo, el problema de scheduling se denomina *problema permutativo de flow shop* (PFS). En otro caso se lo llama *problema no-permutativo de flow shop* (NPFS). En ambos casos el objetivo es encontrar un ordenamiento σ^* de los *n* trabajos para las *m* máquinas de manera de optimizar alguna función objetivo.

Los sistemas flow shop suelen ser utilizados para producciones del tipo estandarizada, en donde los productos y sus variaciones están especificados y definidas de ante mano. Esto permite, generar sistemas de producción de propósitos específicos que permiten que tengan alto rendimiento. Algunos casos de productos producidos en sistemas flow shop son los helados, las comidas rápidas, el ensamble de los electrodomésticos, envasado de vinos y bebidas gaseosa, fabricación de muebles, entre otros.

El grado de especialización que suelen tener los sistemas flow shop, no implica que resolver su programación de actividades sea sencilla. Estos problemas suelen ser para un gran espectro de variantes del tipo de los problemas NP-hard (Garey et al. 1976). Solo en algunos casos particulares existen algoritmos capaces de resolver los problemas en tiempos polinomiales, como es el caso de la regla de Johnson (Johnson 1954) para los casos de 2 y 3 máquinas con función objetivo de tiempo máximo de finalización. Esto ha llevado a que, se concentren muchos esfuerzos de la comunidad científica en el estudio de los métodos de resolución, tal como lo muestran en sus revisiones (Lin and Zhang (1999); Framinan et al. (2004); Ruiz y Maroto (2005) y Pan y Ruiz 2013).

Un aspecto a considerar en los sistemas flow shop es cómo se organizan las secuencias entre las sucesivas etapas o máquinas del sistema productivo. Tradicionalmente, se ha abordado esta organización a través de una simplificación: la misma secuencia para todas las máquinas. Es decir, hay una única secuencia que se mantiene en cada máquina. Sin embargo, esto no es estrictamente necesario en la gran mayoría de las aplicaciones reales de flow shop, y, por consiguiente, tampoco por la definición formal del flow shop. Pero cuando se flexibiliza esta simplificación se da lugar a que el ordenamiento en las etapas puede ser modificado lo que implica que obtener la

solución óptima del problema sea considerablemente más difícil. En la siguiente sección se profundizará en la presentación de los ordenamientos no-permutativos como también en sus características.

1.3. Flow shop no-permutativo

El problema hasta ahora definido es el secuenciamiento de un conjunto de trabajos en un sistema productivo flow shop. La decisión que debe tomarse es en qué orden los trabajos atravesarán el sistema. En el caso más general ese orden puede ser independiente para cada etapa o máquina del proceso (formato NPFS), mientras que el caso más limitado exige el mismo orden para todas las máquinas (formato PFS). El principal argumento para abordar el problema de forma permutativa es que la cantidad posible de secuencias factibles en este formato es *n!*, mientras que para el caso no-permutativo la cantidad asciende a $n!^m$, lo que es considerablemente mayor, volviéndose inmanejable aun para instancias reducidas. Por ejemplo, para una instancia de 10 trabajos y 5 máquinas, el formato permutativo comprende algo más de 3,5 millones de soluciones, mientras que el no-permutativo posee 6,3 exp 32 soluciones. Es de interés aclarar, que los sistemas flow shop no-permutativos (NPFS), incluyen todas las soluciones de un sistema flow shop permutativo (PFS), tal como lo muestra la Figura 1.2. Por lo tanto, para obtener la mejor solución posible u óptima de un problema dado es necesario resolver el NPFS.

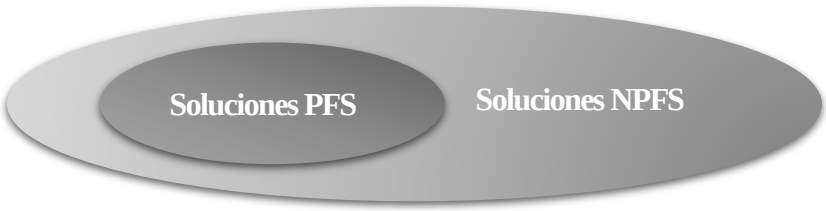
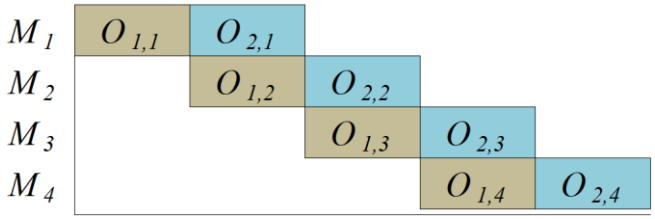


Figura 1-1. Representación de los conjuntos de soluciones PFS y NPFS.

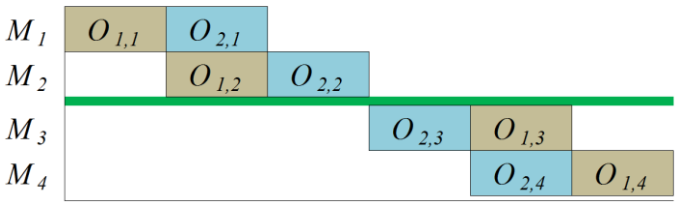
Sin embargo, hay un par de condiciones particulares que permiten asegurar que una solución PFS es la óptima global del problema. Esas condiciones son (i) que la función objetivo a minimizar sea el tiempo máximo de finalización (*makespan*) y (ii) que el sistema no tenga más de 3 máquinas. Estos resultados, propuestos por primera vez por Conway et al. (1967), se basan en que para un sistema flow shop con makespan como objetivo, el modificar la secuencia de los trabajos de la primer a la segunda máquina no generará ningún beneficio en términos de la calidad de la solución final. Lo mismo aplica

para la última y ante-última máquinas. Por lo tanto, en el caso de hasta 3 máquinas para optimizar el makespan, basta analizar ordenamientos permutativos.

Por fuera de esas condiciones, es evidente la importancia y necesidad de un enfoque no-permutativo a la hora de resolver problemas de secuenciamiento en configuraciones del tipo flow shop. Sin embargo, no puede obviarse el incremento del costo computacional. Este es el principal motivo por el cual en gran parte de la literatura de flow shop, el enfoque NPFS no ha sido prácticamente estudiado. En conclusión, para obtener soluciones de muy alta calidad en los sistemas flow shop se requiere el enfoque NPFS. Dado que éste no ha sido prácticamente estudiado, se propone esta tesis doctoral con el fin de poder generar una contribución en la literatura del flow shop, enriqueciendo los aportes de NPFS.



a) Diagrama de Gantt de una solución PFS.



b) Diagrama de Gantt de una solución NPFS.

Figura 1-2. Representación Gantt de problema PFS y NPFS para un caso de dos trabajos $n = 2$, y cuatro máquinas, $m = 4$.
Referencia: la línea horizontal verde indica el cambio en el ordenamiento de los trabajos.

2. CAPÍTULO 2: ESTADO DEL ARTE DE PROBLEMAS FLOW SHOP NO-PERMUTATIVOS

Estado del arte de problemas flow shop no-permutativos

2.1. Notación y presentación de los distintos tipos de problemas NPFS

Con el objetivo de generar un contexto homogéneo para abordar la literatura de NPFS, primeramente, presentaremos una descripción estándar de los problemas NPFS, donde esencialmente todos los problemas NPFS tienen en común las siguientes características:

- Cada máquina puede procesar de a un trabajo a la vez.
- Cada trabajo tiene un tiempo de procesamiento $p_{j,m}$ en cada máquina.
- La capacidad de los almacenamientos intermedios debe ser lo suficientemente grande como para permitir el reordenamiento de los trabajos en la secuencia.

Estas características básicas aplican para la mayoría de los problemas PFS y NPFS, por lo que podría considerarse una descripción básica de flow shop. Sin embargo, la tercera característica es la que presenta la mayor de las diferencias entre NPFS y PFS. En algunos casos, los problemas PFS asumen una capacidad infinita de los almacenamientos intermedios, siendo esto totalmente compatible con los sistemas NPFS. En el caso de la ausencia de estos almacenamientos, el enfoque NPFS deja de ser aplicable (lo mismo ocurre para el caso de sistemas flow shop "sin espera", *no-wait flow shop*). Además de las tres características ya mencionadas, existen otras particularidades: todos los trabajos y máquinas están disponibles para producción al inicio de la planificación; las máquinas pueden estar ociosas durante el horizonte de planificación; cada trabajo puede ser procesado por solo una máquina a la vez; la información del problema es determinística y conocida de ante mano. Esta descripción en sí misma no representa todas las variantes posibles de problemas NPFS, pero sirve como estructura básica del resto de las variantes, las cuales presentan cambios menores, tales como agregando o eliminando alguna condición o restricción.

Para clasificar cada una de las variantes de NPFS adoptaremos la clasificación y la nomenclatura propuesta por Graham et al. (1979) e implementada por Pinedo (2012). Cada una de las variantes será tipificada según 3 datos o campos $\alpha \mid \beta \mid \gamma$. Por medio del dato α, se describe el tipo de entorno de producción o de configuración. En el dato β, detalles específicos de la producción, restricciones especiales, o supuestos específicos, este campo puede tener más de una entrada. El último de los datos clasificatorios es γ y

describe la función objetivo del problema, generalmente lleva una sola entrada (más de una implica un caso multi-objetivo).

Respecto al campo de clasificación α, las posibles entradas pueden ser *Fm* (para flow shop puros con *m* etapas y sólo una máquina por etapa), *FFm* (para flow shop híbridos o flexibles con *m* etapas y con más de una máquina en al menos una de las etapas) o *Jm* (para job shop con *m* máquinas). A pesar de esta potencial variedad de entradas, en nuestro caso nos centraremos solo en flow shop puros, por lo que la única entrada para el campo α es *Fm*.

Para el campo β, son posibles múltiples entradas, enumerando restricciones o supuestos específicos del caso analizado. Una consideración sobre este campo es que la aparición de una entrada especifica implica que esa entrada aplica para el caso, su no inclusión implica que no aplica. Las posibles entradas son:

- r_j: indica que los trabajos no pueden comenzar a procesarse antes de la fecha de liberación de la orden. Si r_j no está presente los trabajos pueden comenzar en cualquier momento. En contraste con las fechas de liberación, las fechas de entrega no se indican en este campo. El campo de la función objetivo proveerá la suficiente información sobre las fechas de entrega.
- *prmp*: significa que los trabajos pueden ser interrumpidos durante su procesamiento, mientras que la ausencia de esta entrada indica que no pueden ser interrumpidos.
- $s_{j,k}$: denota que el tiempo de preparación del trabajo *j* depende del tipo de trabajo procesado previamente *k*. A su vez, si este tiempo depende de la máquina, se incorpora otro subíndice *i*, i.e. $s_{j,k,i}$. Si no hay $s_{j,k}$ se supone que todos los tiempos de preparación son independientes de la secuencia (incluidos dentro de los tiempos de procesamiento) o nulos.
- *prmu*: indica que los ordenamientos de los trabajos tienen que ser el mismo para todos los trabajos.
- *Block*: implica que la capacidad de los almacenamientos intermedios es limitada. Los trabajos deben esperar ocupando la máquina de la etapa previa hasta que se libere el suficiente espacio como para pasar al almacenamiento. Esta condición no es suficiente como para negar los ordenamientos NPFS, ya que puede haber suficiente espacio para reordenar al menos uno de los trabajos.

- *unavail*: indica que las maquinas no están disponibles en ciertos momentos.
- $p_j=p$: significa que todos los tiempos de procesamientos son iguales.

En caso de existir parámetros estocásticos, los indicaremos con la misma notación presentada, pero utilizando letras mayúsculas. Por ejemplo, si el parámetro fecha de liberación de órdenes es un parámetro incierto, la entrada del campo β será R_j, mientras que el parámetro estándar no-estocástico es r_j. Esta notación fue adoptada por Pinedo (2012). Otras posibles entras del campo β, pero que no aplica en esta tesis, son *no-wait* (es infactible para NPFS) y las de precedencias *prec* (para el caso de flow shop puros es redundante). En cualquier otro caso, si aparece otra entrada posible, su notación será suficientemente clara para entender su significado.

Suponga que $C_{i,j}$ representa el tiempo de finalización del trabajo j en la máquina i, y que C_j es el tiempo de finalización en la última máquina (es decir cuando j termina su procesamiento). El tiempo de flujo de ese trabajo j se representa por medio de F_j , e indica el tiempo que el trabajo j transcurrió dentro del sistema de producción, que puede ser calculado como: $F_j = C_j - r_j$. El retraso del trabajo j se define como L_j, y se calcula como $L_j = C_j - d_j$. Expresado de esa forma el retraso puede ser negativo. La tardanza del trabajo j es $T_j = max\{ C_j - d_j, 0\}$ y la entrega temprana es $E_j = max\{ d_j - C_j , 0\}$. Ambos son positivos por definición. Si existiera una penalización por cada trabajo entregado con tardanza, se utilizará una penalidad unitaria, U_j, que es 1 si $C_j > d_j$ y 0 en cualquier otro caso. En muchos casos las funciones objetivo asocian pesos a cada trabajo, w_j. Estos pesos representan la importancia relativa de esos trabajos respecto a los otros trabajos, representando diferentes costos, volúmenes, prioridades u otra cualidad especial que tiene que ser considerada por el planificador. Los elementos como C_j, F_j, etc., son usados para representar las diferentes funciones objetivo que deben ser optimizadas en el problema de secuenciamiento. La Tabla 2.1 muestra la lista de las más comunes. La entrada del campo γ será alguna de ellas.

Para ilustrar cómo se utiliza esta notación, considere la versión estándar de NPFS presentadas al inicio de este capítulo. Ese caso sería descripto como $Fm \mid\mid C_{max}$ (nótese que la entrada β está vacía). Esto significa que la configuración productiva es un sistema flow shop con m etapas y el criterio de optimización es el tiempo máximo de finalización (makespan). Otro ejemplo podría ser, considerando que el número de máquinas se limita en 10, que los trabajos poseen fecha de liberación de orden, que los tiempos de

preparación son dependientes de la secuencia y que el objetivo de optimización es la tardanza máxima. Este problema se representaría como $F_{10} \mid r_j, s_{j,k} \mid T_{max}$.

Tabla 2.1. Funciones objetivos para problemas flow shop.

Notación	Significado
C_{max}	Tiempo máximo de finalización ($max_j\ C_j$)
$\Sigma\ C_j$	Tiempo total de finalización
$\Sigma w_j C_j$	Tiempo total ponderado de finalización
T_{max}	Tardanza máxima ($max_j\ T_j$)
$\Sigma\ T_j$	Tardanza total
$\Sigma\ w_j T_j$	Tardanza total ponderada
E_{max}	Entrega temprana máxima ($max_j\ E_j$)
$\Sigma\ E_j$	Total de entrega temprana
$\Sigma\ w_j E_j$	Total ponderado de entrega temprana
F_{max}	Tiempo máximo de flujo ($max_j\ F_j$)
$\Sigma\ F_j$	Tiempo total de flujo
$\Sigma\ w_j F_j$	Tiempo total ponderado de flujo
$\Sigma\ U_j$	Número de trabajos con retrasos
$\Sigma\ w_j U_j$	Número ponderado de trabajos con retrasos
L_{max}	Retraso máximo ($max_j\ L_j$)

2.2. Revisión de la literatura NPFS

La notación presentada en la sección 2.1. se utilizará para caracterizar los problemas abordados en 67 artículos de la literatura internacional. La información resultante se presenta en la Tabla 2.2, en donde la primera columna indica el año de la publicación, la segunda la referencia, la tercera la caracterización del problema estudiado en ese artículo y la última columna proporciona algunos comentarios sobre la contribución, tales como el método de optimización utilizado y otros aspectos. El formato de esa tabla fue tomado de Ruiz y Vázquez-Rodríguez (2010).

Con el fin de organizar la revisión, dividiremos los artículos según el tipo de objetivo analizado. Los objetivos fueron catalogados como relativos al tiempo de finalización, a las fechas de entrega y a los costos. A su vez, dedicamos una categoría particular al makespan por ser por lejos el más frecuente de todos. Al final, se incluyen

dos casos especiales, uno los enfoques multi-objetivo y, el otro, estudios mono-objetivo contemplando numerosos objetivos por vez.

2.2.1. Objetivos basados en tiempos de finalización

A continuación, repasaremos los artículos que consideraron objetivos relacionados a tiempos de finalización.

2.2.1.1. El Makespan o tiempo máximo de finalización.

Makespan es la función objetivo más frecuentemente estudiada en la literatura. Alrededor del 55% de los artículos revisados considera el makespan como objetivo. Por lo tanto, hemos separado este caso del resto de los objetivos basados en tiempos de finalización. El primer trabajo que abordó un problema NPFS con makespan como objetivo fue Janiak (1988). En ese artículo, la duración de cada operación depende linealmente de la fracción de un recurso limitado almacenado en cada máquina (por ejemplo, combustible), y la decisión a tomar por el planificador es doble, teniendo que determinar tanto la secuencia de los trabajos a procesar como la asignación de los recursos limitados a cada máquina. Para resolver el problema aplica un procedimiento de Branch & Bound. Potts et al. (1991) fueron los primeros en cuantificar el impacto de restringir la solución sólo a ordenamientos permutativos. Ellos encontraron un conjunto de instancias para las que, en el peor de lo casos, el makespan de PFS es $1/2\sqrt{m}$ veces el makespan de NPFS. Tandon et al. (1991) compararon empíricamente las soluciones PFS y NPFS. Para instancias chicas, adoptaron un procedimiento enumerativo mientras que para las más grandes usaron recocido simulado (SA, Simulated Annealing). Ellos mostraron que, para mayor amplitud del rango de los tiempos de procesamiento y más grandes las instancias, NPFS se torna más ventajoso que PFS. Strusevich y Zwaneveld (1994) trataron el caso de dos máquinas considerando tiempos de preparación, procesamiento y de remoción de los trabajos separados entre sí. En este caso, el enfoque PFS no es suficiente para asegurar la solución óptima, y en el peor de los casos, el makespan obtenido con PFS es 3/2 del makespan obtenido con NPFS. Ellos también analizaron el caso de dos máquinas y almacenamiento intermedio finito. Nuevamente PFS no logra asegurar el óptimo del problema. Ambos casos son NP-duros. Grau et al. (1996) estudiaron el secuenciamiento en plantas multipropósito de tipo batch con política de espera finita entre etapas (después de que una máquina termina de procesar un trabajo, el tiempo que puede esperar al siguiente trabajo es limitado). Para afrontar este problema NPFS, ellos implementaron

métodos recursivos. Koulamas (1998) presentó una heurística (HFC) capaz de generar ordenamientos no-permutativos cuando parezca apropiado. Esta heurística tiene un desempeño similar a la heurística NEH (Nawaz et al. 1989), con la ventaja de que permite encontrar soluciones no-permutativas mientras que NEH no. Jain y Meeran (2002), propusieron una meta-heurística híbrida multi-nivel que permite interactuar entre distintas estrategias de intensificación y de diversificación de búsquedas, basada en los algoritmos Scatter Search (SS) y Path Relinking (PR). Liu y Ong (2002) propusieron tres meta-heurísticas para problemas PFS y NPFS basadas en distintas estructuras de inserción en el vecindario. La meta-heurística para NPFS utiliza una estructura de vecindario basada en caminos críticos. Pugazhendi et al. (2003) consideraron problemas NPFS asumiendo que algunos de los trabajos saltean etapas de la producción (es decir, no todos los trabajos necesitan pasar por todas las máquinas). Estos autores proponen una heurística, llamada NPS, que inserta un trabajo en la secuencia cada vez que esta inserción mejora el makespan. Brucker et al (2003) investigaron el problema NPFS con almacenamientos intermedios limitados que eventualmente puedan generar bloqueos en el sistema (cuando el almacenamiento intermedio está completo, los trabajos tienen que esperar ocupando la máquina después de su procesamiento). Para resolver este problema utilizaron un algoritmo Tabu Search (TS). Aggoune (2004) inquirió sobre el problema NPFS considerando restricciones de disponibilidad en las maquinarias sujetas a actividades de mantenimiento. Dos tipos de actividades fueron consideradas en forma separada, un tipo implicaba actividades fijas, mientras que el otro, contemplaba ventanas de tiempo. En el caso de las actividades fijas, las mismas deben realizarse en un momento fijo, mientras que en el caso de ventanas de tiempo, existe un intervalo de tiempo durante el cual pueden realizarse. La solución la obtuvieron usando una combinación de algoritmos genéticos (GA, genetic algorithms) con TS. Pugazhendi et al. (2004) estudiaron problemas NPFS con trabajos que saltean operación y tiempos de preparación dependientes de la secuencia. Para optimizar el problema plantearon un método recursivo que genera una solución PFS de buena calidad, para luego aplicar la heurística NPS (Pugazhendi et al. 2003) que mejora la solución permutativa buscando soluciones no-permutativas. En este artículo también se trabajó con la función objetivo de minimizar el tiempo ponderado total de flujo. Rebaine (2005) estudia el desempeño del ratio del peor caso entre soluciones NPFS y PFS considerando tiempos de demora. Para el caso de dos máquinas, la solución PFS no asegura el óptimo, teniendo un ratio de 2 para el peor de

los casos. Si las operaciones son restringidas a tener tiempos de procesamientos unitarios, el ratio de los makespan disminuye a 2-(3/*n*+2). Para el caso de *m*-máquinas, el ratio de los makespan está acotado por encima en *m*. Haq et al. (2007) abordaron el problema NPFS utilizando un algoritmo SS. El algoritmo se basa en unir soluciones y explotar la memoria adaptativa para evitar generar o incorporar soluciones duplicadas en varios niveles del problema. Ying y Lin (2007) presentaron una multi-heurística basada en sistemas de colonias de hormigas con atracción para problemas NPFS. Ellos mostraron los beneficios de los sistemas ACO para optimizar problemas NPFS. Ying (2008) propuso una heurística codiciosa iterativa para problemas NPFS. Esta heurística es comparada con otras heurísticas constructivas simples y meta-heurísticas del estado del arte. Como conclusión, el autor indica el gran potencial de los métodos codiciosos para problemas NPFS. Rayward-Smith y Rebaine (2008) presentaron dos heurísticas para el caso de dos máquinas, con tiempos de procesamientos unitarios y con tiempos de demora. Estas heurísticas se basan en ordenar los trabajos en orden no creciente de tiempos de demora. Sadjadi et al. (2008a) analizaron tres problemas NPFS, dos de ellos tienen el makespan como función objetivo y el otro tiene tardanza total ponderada. En los casos de makespan, consideraron distintas características específicas. Uno considera desfases de tiempos mientras que el otro asume tiempos de preparación dependientes de la secuencia. Ambos casos consideran trabajos que saltean operaciones. Para estos casos los autores proponen formulaciones matemáticas mixto enteras. Sadjadi et al. (2008b), consideraron dos problemas distintos, uno con makespan como objetivo y el otro con tiempo total de flujo. Para resolver este problema, implementaron un procedimiento de dos pasos. Inicialmente, utilizaron un algoritmo ACO para obtener una buena solución permutativa. Luego, esta solución es mejorada con un algoritmo de búsqueda local que incluye soluciones no-permutativas a la búsqueda. Lin y Ying (2009) presentaron un algoritmo híbrido de SA y TS para problemas NPFS. Nagarajan y Sviridenko (2009) presentaron el límite al ratio entre los makespan obtenidos por los enfoques PFS y NPFS para el caso general, demostrando que para cualquier instancia la solución óptima bajo el enfoque PFS es cuanto mucho $2\sqrt{\min\{m,n\}}$ veces el makespan óptimo de NPFS. Zheng y Yamashiro (2010) propusieron un algoritmo cuántico diferencial evolutivo (QDEA) para problemas NPFS. El algoritmo está basado en la resolución de operaciones diferenciales y búsqueda local a través de una representación conocida como Q-bit. Färber et al. (2010) investigaron un problema de secuenciamiento en donde el reordenamiento de los trabajos

es permitido cuando las estaciones de trabajo tienen acceso a espacios de almacenamiento intermedios. Estos accesos están limitados por el número de almacenes y por el tamaño físico de los trabajos. Para resolver el problema, los autores aplicaron un enfoque hibrido basado en Constraint programming. Brucker y Shakhlevich (2011) estudiaron la versión inversa del problema flow shop, i.e. primero se obtiene una secuencia de trabajos, y luego para asegurar que ese ordenamiento sea óptimo, se restringen los tiempos de procesamiento dentro de ciertas cotas. Ellos dedujeron las condiciones necesarias y suficientes para que ambos enfoques PFS y NPFS sean óptimos. Ramezanian et al. (2011) estudiaron los problemas NPFS con trabajos que saltean operaciones, resolviendo con GA y TS. Rudek (2011) probó que en el caso de dos máquinas con efectos aprendizajes, el sistema PFS no asegura calidad óptima de solución, y que ambos enfoques (PFS y NPFS) son NP-duros, aun si el efecto aprendizaje es considerado para una sola de las máquinas (la curva de aprendizaje debe seguir una forma escalonada). Cheng et al. (2012) analizaron el proceso de derrumbar y reconstruir edificios como un flow shop de dos etapas con recursos restringidos. Los autores proveyeron formulaciones matemáticas mixto-enteras y comentaron acerca de la complejidad, desarrollando algoritmos polinomiales para casos especiales. Rossi y Lanzetta (2013) investigaron el problema NPFS con algoritmos ACO, estableciendo que la capacidad mínima de los almacenamientos intermedios para evitar bloqueos debe ser de (*n-2*). Rossi y Lanzetta (2013) trabajaron con el mismo problema. Una condición particular del algoritmo ACO utilizado en este caso es que desde el principio de la búsqueda el algoritmo trabaja con soluciones no-permutativas. En este trabajo los autores utilizaron para evaluar su algoritmo las instancias propuestas en Taillard (1993), mientras que en Rossi y Lanzetta (2014) usaron las instancias propuestas en Demirkol et al (1998). En este último caso, el algoritmo ACO propuesto supera en desempeño a otros algoritmos ACO propuestos para problemas NPFS. Shen et al. (2014) abordaron el problema NPFS con procesamientos por lotes (*batching*), con tiempos de procesamientos dependientes de la familia de productos. Estos autores desarrollaron un algoritmo TS que incluye doble listas tabu y diversificación multi-nivel. El supuesto de tecnología de grupos es relajado, permitiendo que trabajos de la misma familia sean separados. Gharbi et al. (2014) presentaron varios procedimientos de cotas superiores e inferiores para el caso de una sola máquina. Moukrim et al. (2014) introdujeron un algoritmo Branch & Bound para el problema descripto por Rebaine (2005). Ellos presentaron nuevos procedimientos de cotas para este

Branch & Bound como también reglas de dominancia. Benavides et al. (2014) afrontaron el problema NPFS heterogéneos lidiando con dos cuestiones en simultaneo: asignar los trabajadores a las estaciones de trabajos y secuenciar los trabajos a ser procesados. La principal motivación proviene de casos en donde los trabajadores son personas con discapacidad, y, por lo tanto, el nivel de habilidad no es homogéneo. Para optimizar este problema utilizaron un algoritmo de SS y PR (Path Relinking). Nikjo y Zarook (2014) analizaron el problema es NPFS en el contexto de células de manufacturas con fechas acordadas de liberación de órdenes y tiempos de preparación dependientes de la secuencia de partes relacionadas con los productos. Con GA y TS obtuvieron la solución. Zhang et al. (2014) estudió el problema NPFS con actividades de mantenimiento periódicas. El método usado para resolver es un hibrido entre algoritmos genéticos y una heurística basada en la teoría de la heurística NEH. Rossit et al. (2016) trabajaron con problemas NPFS utilizando una estrategia de lotes de transferencia (*lot streaming*, del inglés). Benavides y Ritt (2016) propusieron una heurística de búsqueda local iterativa para los problemas NPFS. El algoritmo se basa en la observación de que las soluciones permutativas y no-permutativas son bastantes similares, lo que permite suponer que partiendo de una buena solución permutativa se puede llegar a una buena solución no-permutativa. Cui et al. (2016) investigaron problemas NPFS con restricciones de disponibilidad. Las restricciones de disponibilidad de las máquinas dependen de dos tipos de tareas extra productivas, unas involucran tareas fijas mientras que, las otras, intervalos de tiempo flexibles mientras que el tiempo de trabajo continuo en las máquinas no supere cierto límite. La optimización de este problema la hicieron a través de la hibridación de algoritmos genéticos incrementales combinando refinamientos locales y esquema de supervisión de la diversificación de la población.

2.2.1.2. Otros objetivos basados en tiempo de finalización

En esta sección se revisará la literatura de los problemas de NPFS con objetivos basados en tiempo de finalización, distintos del makespan. Particularmente, nos centraremos en tiempo total de finalización, tiempo total ponderado de finalización, tiempo total de flujo y tiempo total ponderado de flujo.

Rajendran y Ziegler (2001) estudiaron el problema NPFS con trabajos que saltean operaciones minimizando el objetivo de tiempo total de flujo. Los autores resolvieron usando reglas de secuenciación combinadas con reglas heurísticas. Pugazhendhi et al

(2004) investigaron dos problemas NPFS con trabajos que saltean operaciones distintas, el primero minimiza el tiempo total de flujo y el segundo minimiza el tiempo total ponderado de flujo. Ellos presentaron una heurística (NPS-set) que trabaja mejorando una solución permutativa. Färber y Coves Moreno (2006) propusieron un algoritmo genético para problemas NPFS con almacenamientos intermedios no disponibles para todas las máquinas, los cuales a su vez eran limitados. Färber et al. (2007) abordaron el problema NPFS con demanda semi-dinámica y donde el re-secuenciamiento es limitado (parecido a Färber y Coves Moreno (2006)). El objetivo utilizado fue minimizar el tiempo total ponderado de finalización. Los autores resolvieron utilizando dos metodologías, una basada en programación con restricciones (*Constraint programming*) y la otra con algoritmos genéticos. Li et al. (2010) trabajaron con un problema NPFS de dos máquinas con implementaciones robóticas. El objetivo analizado fue el tiempo total ponderado de finalización. Los robots son los que se encargar de cargar, descargar y trasladar los trabajos desde una estación a otra. Estos robots pueden manipular un solo trabajo a la vez. La solución fue obtenida mediante algoritmos genéticos. Vahedi-Nouri et al. (2013) abordaron el problema NPFS con efectos aprendizaje y restricciones de disponibilidad de máquinas, minimizando el tiempo total de flujo. Los autores presentaron una formulación matemática mixta entera y propusieron una heurística de mejora. En Isenberg y Scholz-Reiter (2013) se trató un problema con procesamiento por lotes NPFS, donde los lotes son confeccionados en cada etapa. Los autores lo trataron con tres funciones objetivo distintas: tiempo total de flujo, tiempo total de finalización y makespan. Vahdei-Nouri et al. (2014) presentaron una heurística y un algoritmo SA para problemas NPFS considerando efectos aprendizaje, restricciones de disponibilidad y fechas de liberación de órdenes. El objetivo optimizado fue tiempo total de flujo. Benavides y Ritt (2015) estudiaron las ventajas de los ordenamientos NPFS sobre los PFS. Para esto utilizaron una heurística de dos etapas. En la primera etapa una búsqueda local iterativa identifica una buena solución permutativa, y en la segunda, un algoritmo de inserción de vecindario mejora la solución explorando ordenamientos no-permutativos. El objetivo analizado fue tiempo total de finalización. Henneberg y Neufeld (2016) estudiaron el problema NPFS con trabajos que saltean operaciones con el objetivo de minimizar el tiempo total de finalización. Ellos resolvieron modificando la hueristica NPS-set presentada por Pughazhendi et al. (2004), basada en un algoritmo SA de dos etapas.

2.2.2. Objetivos basados en fechas de entrega

En esta sección se abordarán los artículos en los cuales sus funciones objetivos estén basados en fechas de entrega. Los objetivos contemplados en esta sección son: tardanza máxima, tardanza total y tardanza total ponderada.

Swaminathan et al. (2007) estudiaron el impacto de forzar la condición de ordenamientos permutativos en el caso del flow shop general (no-permutativos). El objetivo analizado es minimizar la tardanza total ponderada. Para obtener la solución usaron tres técnicas distintas: permutación pura, basados en turnos y reglas de secuenciación puras. La última, es la única capaz de generar ordenamientos no-permutativos. Sus resultados mostraron que PFS provee una solución ineficiente a este problema. Swaminathan et al. (2004) estudiaron el mismo problema en una versión simplificada. Liao y Huang (2010) estudiaron el problema NPFS teniendo como objetivo la tardanza total, presentando y evaluando tres formulaciones MIP distintas. Luego, presentaron dos algoritmos tabu search. De la comparación de NPFS y PFS indicaron que NPFS permite obtener soluciones muy superiores para este tipo de problemas. Ziaee (2013) abordó el problema NPFS con tiempos de preparación dependientes de la secuencia minimizando la tardanza total ponderada. El autor propone una heurística de dos etapas. Esencialmente, la primera etapa explora soluciones permutativas hasta encontrar una de buena calidad, y luego en la segunda etapa, se mejora esta solución explorando soluciones no-permutativas con búsqueda local. Xiao et al (2015) analizaron el problema flow shop con aceptación de ordenes minimizando la tardanza total ponderada. Los autores presentaron dos formulaciones distintas al problema. La primera es una formulación MIP, que permite a CPLEX resolver para instancias chicas. La segunda, es una formulación mixta entera no lineal (MINLP) que permite abordar problemas de media y gran escala a través de un algoritmo genético de dos etapas.

2.2.3. Objetivos basados en costos

En esta sección se revisan los artículos que trabajan con funciones objetivos que involucran costos. En los cinco artículos revisados, la especificación de los costos a minimizar varía.

Grau et al. (1995), estudiaron un problema NPFS buscando minimizar los costos incurridos cada vez que se cambia el producto a producir. Los autores desarrollaron un método Branch & Bound para resolver el problema. Mohammadi et al. (2010) abordaron

el problema de secuenciamiento y dimensionamiento de lotes de producción para sistemas NPFS. Ellos desarrollaron una formulación MIP para el problema y presentaron cinco heurísticas basadas en esa formulación, para minimizar los costos de preparación, almacenamiento y producción. Algunas de esas heurísticas pueden manejar solo soluciones permutativas. Vahedi-Nouri et al. (2013) analizaron un problema NPFS con efectos aprendizaje y tareas de mantenimientos flexibles. Los objetivos a minimizar son la tardanza total y el costo de las actividades de mantenimiento. Los autores desarrollaron un algoritmo hibrido entre el algoritmo firefly y simulated annealing para resolver una formulación MIP del problema. Ramezanian y Saidi-Mehrabad (2013) investigaron el problema de secuenciamiento y dimensionamiento de lotes considerando tiempos de preparación dependientes de la secuencia, restricciones de capacidad, tiempos de procesamiento y demanda estocásticos. Presentaron un modelo MIP basado en intervalos de tiempos grandes (*big bucket*, del inglés). Dos heurísticas basadas en la formulación MIP implementando la estrategia de rodar en el tiempo también fueron presentadas. Complementariamente, propusieron una heurística hibrida que combina simulated annealing, el algoritmo firefly y una heurística ad-hoc. Babaei et al. (2014) también analizaron el problema de secuenciamiento y dimensionamiento de lotes, pero con ligeras diferencias, principalmente, en los tiempos de preparación y en la inclusión de la condición mantenimiento de la máquina preparada y en la posibilidad de entregar pedidos con retrasos. Ellos propusieron una formulación MIP que la resolvieron mediante algoritmos genéticos.

2.2.4. Estudios experimentales mono-objetivo

En esta sub sección, se presenta un grupo de artículos que comparan a través de un análisis experimental la calidad de las soluciones PFS y NPFS. Estos trabajos consideran distintas configuraciones productivas mono-objetivo, con el fin de evaluar la necesidad del esfuerzo computacional extra que genera el enfoque NPFS. La validez de la comparación se fundamenta en el hecho de que tanto la parametrización como las instancias son programadas y resueltas con los mismos algoritmos. En este sentido, estos trabajos reportan una contribución experimental significativa a la literatura de sistemas no-permutativos. Los objetivos analizados en todos los casos son los seis más comunes de la literatura de secuenciamiento o *scheduling*: tres basados en tiempos de finalización (makespan, tiempo total de finalización y tiempo total ponderado de finalización), y tres

basados en fechas de entrega (tardanza máxima, tardanza total y tardanza total ponderada).

Liao et al. (2006) fueron pioneros en desarrollar este tipo de investigación. Ellos evaluaron un sistema flow shop clásico considerando las seis funciones objetivo mencionadas. Sus resultados indican, en general, que los ordenamientos NPFS mejoran levemente a los PFS para los objetivos basados en tiempos de finalización. Sin embargo, para los basados en fechas de entrega la mejora es significativa, especialmente cuando las instancias comprenden más de treinta trabajos a secuenciar. Ellos usaron para resolver los problemas GA y TS. Lin et al. (2009) presentaron un estudio similar, evaluando las mismas funciones objetivo pero el sistema analizado era una célula de manufactura con tiempos de preparación dependientes de la familia de productos. Nuevamente, la conclusión para los objetivos basados en tiempos de finalización fue análoga, sin grandes diferencias, pero siendo NPFS un poco mejor. Para los objetivos basados en fechas de entrega, en cambio, los ordenamientos no-permutativos superan claramente a los permutativos. Para la resolución los autores utilizaron algoritmos genéticos, SA y TS. Respecto a estos tres algoritmos los autores concluyen que SA tuvo mejor desempeño. Ying et al (2010) revisitaron los problemas de Lin et al. (2009) experimentando con distintos rangos de tiempos de preparación, y concluyeron que en el caso de que los tiempos de preparación sean mayores NPFS supera a PFS en la mayoría de los casos por amplio margen. A su vez, encontraron que NPFS obtiene mejores soluciones, en general, para los seis objetivos, pero para los basados en fechas de entrega, esta mejora es más amplia. Todos los problemas son resueltos mediante un algoritmo SA.

2.2.5. Estudios experimentales multi-objetivo

Un área de investigación prometedora para el enfoque no-permutativo es el área de los problemas multi-objetivo, debido, principalmente, a que el enfoque no-permutativo incorpora un gran número de soluciones que son excluidas por el enfoque permutativo. Los artículos que estudiaron problemas multi-objetivo son revisados a continuación.

Mehravaran y Logendran (2012) fueron los primeros en estudiar problemas multi-objetivos bajo el enfoque no-permutativo. Ellos consideraron una configuración flow shop con tiempos de preparación dependientes de la secuencia asumiendo restricciones de disponibilidad de máquinas, liberación de órdenes de trabajo y trabajos que saltean operaciones. Ellos usaron una función bi-objetivo que minimiza la suma normalizada de

los tiempos totales ponderados de finalización y la tardanza total ponderada. Los autores presentaron una formulación MIP y un algoritmo TS. Mehravaran y Logendran (2013) abordaron el problema NPFS considerando recursos duales: máquinas y operarios. Los objetivos optimizados fueron tiempo total ponderados de finalización y tardanza total ponderada y nuevamente como en Mehravaran y Logendran (2012), usaron una suma ponderada de objetivos. Las especificaciones del problema incluyen distinto nivel de aptitud en los operarios, tiempos de preparación dependientes de la secuencia, restricciones de disponibilidad de máquinas y fecha de liberación de órdenes. Un procedimiento de dos niveles obtiene la solución. El primer nivel resuelve el problema flow shop tradicional (secuenciar los trabajos), y el segundo nivel, busca una asignación de los operarios a cada operación en concordancia con el secuenciamiento de trabajos establecido. Desarrollaron tres algoritmos de búsqueda distintos. Estos autores fueron a su vez pioneros en el estudio de problemas de flow shop con recursos duales, y enfatizaron en la superioridad de los ordenamientos NPFS sobre los PFS. Rahmani et al. (2014) estudiaron un problema NPFS estocástico. Los tiempos de procesamiento y las fechas de la liberación son parámetros inciertos que siguen una distribución normal. Contemplaron tres objetivos: makespan, tiempo total de flujo y tardanza. Para lidiar con el aspecto estocástico aplicaron un enfoque combinado de programación de oportunidades restringida (*chance constraint programming*) y programación por metas con lógica difusa. También adaptaron un algoritmo genético para abordar instancias grandes. Amirian y Sahraeian (2015) analizaron un problema NPFS que minimiza simultáneamente el makespan, el tiempo de flujo total y la tardanza máxima. El problema incluye fechas de liberación de órdenes, tiempos de preparación dependientes de la secuencia pasada, efecto aprendizaje y restricciones de disponibilidad de máquinas. Los autores utilizaron como método de resolución restricciones ε aumentado (Augmented ε-constraint) y una heurística basada en el mismo método.

Tabla 2.2. Revisión de la literatura NPFS.

Año	Referencia	Problema	Comentarios
1988	Janiak (1988)	$Fm\|resource\ constrained\|C_{max}$	Procedimiento B&B
1991	Potts et al (1991)	$Fm\|\|C_{max}$	Ratio entre NPFS C_{max} y PFS C_{max} para casos especiales
	Tandon et al (1991)	$Fm\|\|C_{max}$	Para instancias chichas método enumerativo y para grandes SA

1994	Strusevich et al (1994)	$F2 \mid s, removal\ times \mid C_{max}$	PFS no es óptimo y el problema es NP-duro
		$F2 \mid block \mid C_{max}$	PFS no es óptimo y el problema es NP-duro
1995	Grau et al (1995)	$Fm \mid batch \mid Costs$	Procedimiento B&B
1996	Grau et al (1996)	$Fm \mid batch, finite\ wait \mid C_{max}$	Procedimiento recursivo específico
1998	Koulamas (1998)	$Fm \mid\mid C_{max}$	Heurística (HFC)
2001	Rajendran y Ziegler (2001)	$Fm \mid skip\ operation \mid \Sigma\ C_j$	Heurístico y reglas de secuenciación
2002	Jain y Meeran (2002)	$Fm \mid\mid C_{max}$	Meta-heursitica basada en Scatter Search y Path Relinking, y Tabu Search
	Liu y Ong (2002)	$Fm \mid\mid C_{max}$	Meta- Heurística
2003	Pugazhendhi et al (2003)	$Fm \mid skip \mid C_{max}$	Heurística
	Brucker et al. (2003)	$Fm \mid block \mid C_{max}$	TS
2004	Aggoune (2004)	$Fm \mid avail \mid C_{max}$	GA y TS
	Pugazhendhi et al (2004)	$Fm \mid skip \mid \Sigma\ w_j F_j$	Heurística: NPS set
		$Fm \mid skip \mid \Sigma\ F_j$	
	Pugazhendhi et al (2004)	$Fm \mid skip, S(j,k) \mid C_{max}$	Heurística específica y NPS-set
		$Fm \mid skip, s(j,k) \mid \Sigma\ w_j F_j$	
	Swaminathan et al (2004)	$Fm \mid stochastic \mid Costs$	GA y Heurística ATC
2005	Rebaine (2005)	$Fm \mid time\ delays \mid C_{max}$	PFS no es óptimo y el problema es NP-duro para 2 máquinas
2006	Liao et al. (2006)	$Fm \mid\mid C_{max}$	TS y GA, compara las seis funciones objetivos
		$Fm \mid\mid \Sigma\ C_j$	
		$Fm \mid\mid \Sigma w_j C_j$	
		$Fm \mid\mid Tmax$	
		$Fm \mid\mid \Sigma\ T_j$	
		$Fm \mid\mid \Sigma\ w_j T_j$	
	Färber y Moreno (2006)	$Fm \mid block \mid \Sigma w_j C_j$	GA

2007	Haq et al (2007)	$Fm \mid\mid C_{max}$	SS
	Ying y Lin (2007)	$Fm \mid\mid C_{max}$	ACO
	Färber et al (2007)	$Fm \mid block \mid \Sigma w_j C_j$	GA y Constraint Logic Programming (CLP)
	Swaminathan et al (2007)	$Fm \mid\mid \Sigma\, w_j T_j$	Heurística ATC y GA
2008	Ying (2008)	$Fm \mid\mid C_{max}$	IG
	Rayward-Smith y Rebaine (2008)	$F2 \mid p(j)=p,\ time\ delays \mid C_{max}$	Heurística - (uet: unit execution time)
	Sadjadi et al (2008a)	$Fm \mid\mid \Sigma\, w_j T_j$	PM
		$Fm \mid time\ lags \mid C_{max}$	
		$Fm \mid s(j,k) \mid C_{max}$	
	Sadjadi et al (2008b)	$Fm \mid \mid C_{max}$	ACO y búsqueda local
		$Fm \mid \mid \Sigma\, F_j$	
2009	Lin et al (2009)	$Fm \mid fmls,\ s(j,k) \mid C_{max}$	SA, TS y GA
		$Fm \mid fmls,\ s(j,k) \mid \Sigma\, C_j$	
		$Fm \mid fmls,\ s(j,k) \mid \Sigma w_j C_j$	
		$Fm \mid fmls,\ s(j,k) \mid Tmax$	
		$Fm \mid fmls,\ s(j,k) \mid \Sigma\, T_j$	
		$Fm \mid fmls,\ s(j,k) \mid \Sigma\, w_j T_j$	
	Lin y Ying (2009)	$Fm \mid \mid C_{max}$	SA y TS
	Nagarajan y Sviridenko (2009)	$Fm \mid \mid C_{max}$	Comparación de los makespan PFS y NPFS para el caso general
2010	Ying et al (2010)	$Fm \mid fmls,\ s(j,k) \mid C_{max}$	SA, los tiempos de preparación dependen de la familia
		$Fm \mid fmls,\ s(j,k) \mid \Sigma\, C_j$	
		$Fm \mid fmls,\ s(j,k) \mid \Sigma w_j C_j$	
		$Fm \mid fmls,\ s(j,k) \mid T_{max}$	
		$Fm \mid fmls,\ s(j,k) \mid \Sigma\, T_j$	

		$Fm \mid fmls, s(j,k) \mid \mid \Sigma w_j T_j$	
	Liao et al (2010)	$Fm \mid \mid \Sigma T_j$	TS
	Li et al (2010)	$F2 \mid block \mid \Sigma w_j C_j$	GA
	Mohammadi et al (2010)	$Fm \mid s(j,k) \mid Costs$	Heurística basada en PM
	Zheng y Yamashiro (2010)	$Fm \mid \mid C_{max}$	Algoritmo evolutivo cuántico diferencial (QDEA)
	Farber et al (2010)	$Fm \mid \mid C_{max}$	Híbrido entre CLP y GA
2011	Brucker y Shakhlevich (2011)	*inverse scheduling* - C_{max}	Condiciones suficientes para solución óptima
	Ramezanian et al (2011)	$Fm \mid skip \mid C_{max}$	GA y TS
	Rudek (2011)	$F2 \mid learning \mid C_{max}$	Heurística basada en NEH
2012	Mehravaran y Logendran (2012)	$Fm \mid \mid \Sigma w_j C_j$ & $\Sigma w_j T_j$	TS con perturbación progresiva
	Cheng et al (2012)	$F2 \mid rp \mid C_{max}$ *(rp= relocation)*	Análisis de complejidad, es NP-duro
2013	Vahedi-Nouri et al (2013)	$Fm \mid learning, avail \mid \Sigma F_j$	Heurística: VFR
	Vahedi-Nouri et al (2013)	$Fm \mid learning, avail \mid Costs$	hybrid firefly-SA
	Ziaee (2013)	$Fm \mid s(j,k) \mid \Sigma w_j T_j$	Heurística de búsqueda local
	Isenberg y Scholz-Reiter (2013)	$Fm \mid batch,fmls, rj \mid \Sigma F_j$	PM
		$Fm \mid batch,fmls, rj \mid \Sigma C_j$	
		$Fm \mid batch,fmls, rj \mid C_{max}$	
	Mehravaran y Logendran (2013)	$Fm \mid skip, dual\ resources, s(j,k), avail, rj \mid \Sigma w_j C_j$ & $\Sigma w_j T_j$	meta- Heurística
	Rossi y Lanzetta (2013)	$Fm \mid \mid C_{max}$	ACO
	Rossi y Lanzetta (2013)	$Fm \mid Bi= n-2 \mid C_{max}$	ACO
	Ramezanian y Saidi-Mehrabad (2013)	$Fm \mid s(j,k), P(i,j) \mid Costs$	PM- Heurística y a meta-Heurística, demanda estocástica
2014	Shen et al (2014)	$Fm \mid batch,s(j,k) \mid C_{max}$	TS
	Gharbi et al (2014)	$Fm \mid \mid C_{max}$	Procedimientos de acotamiento

	Moukrim et al (2014)	$F2 \mid uet, time\ delays \mid C_{max}$	B&B
	Rossi y Lanzetta (2014)	$Fm \mid \mid C_{max}$	ACO
	Benavides et al (2014)	$Fm \mid heterogeneous\ resources \mid C_{max}$	Heurística: SS y PR
	Nikjo y Zarook (2014)	$Fm \mid S(j,k), rj \mid C_{max}$	GA y TS
	Vahedi-Nouri et al (2014)	$Fm \mid learning, avail, rj \mid \Sigma F_j$	Heurística y SA
	Babaei et al (2014)	$Fm \mid backlog \mid Costs$	GA
	Zhang et al (2014)	$Fm \mid s(j,k), avail \mid C_{max}$	ACO
	Rahmani et al (2014)	$Fm \mid Rj, P(j,k) \mid C_{max}\ \&\ \Sigma F_j\ \&\ \Sigma T_j$	Chance Constrained Programming (CCP) y Programación por metas difusa
2015	Xiao et al (2015)	$Fm \mid OA = order\ acceptance \mid \Sigma w_j T_j$	TS-GA
	Amirian y Sahraeian (2015)	$Fm \mid learning, s(j,k) \mid C_{max}\ \&\ \Sigma F_j\ \&\ T_{max}$	Augmented ε-constraint, Heurística
	Benavides y Ritt (2015)	$Fm \mid \mid \Sigma C_j$	Algoritmo iterativo codicioso (IGA)
2016	Benavides y Ritt (2016)	$Fm \mid \mid C_{max}$	meta- Heurística
	Cui et al (2016)	$Fm \mid avail \mid C_{max}$	meta- Heurística
	Henneberg y Neufeld (2016)	$Fm \mid skip \mid \Sigma F_j$	SA
	Rossit et al (2016)	$Fm \mid lot\text{-}streaming \mid C_{max}$	PM

2.3. Un análisis cuantitativo de la literatura

Esta revisión comprende más de 67 artículos, representando, hasta lo que sabemos, el total de la literatura NPFS (sin incluir las variantes de flow shop híbrido o flexibles). El análisis desarrollado fue realizado siguiendo la guía de otras revisiones del estado del arte en temas relacionados a flow shop, como el caso de Yenisey y Yagmahan (2014) para flow shop multi-objetivo y Ruiz y Vázquez-Rodríguez (2010) para flow shop híbrido.

Un aspecto destacable de la literatura revisada es que más del 70% de los artículos fueron publicados después del 2007. Esto puede verse en la Figura 2.1, en donde los artículos fueron agrupados según su publicación en periodos de 5 años. Mostrando la

clara tendencia creciente en el número de publicaciones, mientras que el número de artículos por periodo sigue siendo bajo si se compara a otros tópicos de secuenciamiento. Por lo que podemos inferir que NPFS es un área con mucho potencial para futuros desarrollos.

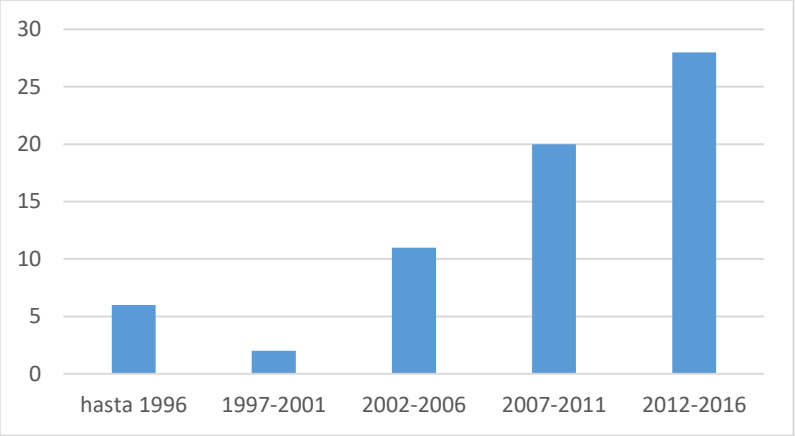


Figura 2-1. Número de artículos NPFS publicados en periodos de 5 años.

La Figura 2.2 muestra los distintos problemas NPFS analizados en la literatura, indicando la proporción de artículos dedicados a cada tipo de función objetivo. Como ya se ha mencionado, los objetivos basados en tiempos de finalización son los más estudiados: el 74% de los artículos se centra en ellos. Un caso especial de éstos es el makespan, cubriendo el 55% de los artículos. El resto de los objetivos basados en tiempos de finalización alcanzan el 19% de las publicaciones. Esto no es algo nuevo dada la supremacía del makespan sobre el resto de los objetivos en la literatura de secuenciamiento, como fue puntualizado por ejemplo Ruiz y Vázquez-Rodríguez (2010). El resto de las funciones objetivo se reparten el 26% restante. De ellos los objetivos basados en fechas de entrega son los que siguen en importancia, agrupando el 9% de las publicaciones. Esto demuestra que este tipo de objetivos no ha sido estudiado aun con profundidad.

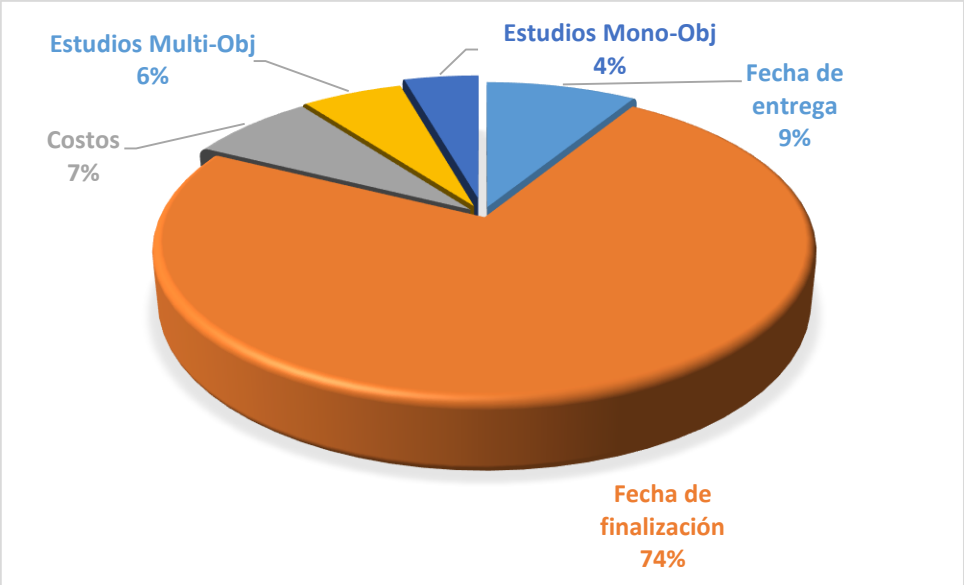


Figura 2-2. Distribución de las funciones objetivos consideradas en la literatura.

La distribución de las distintas técnicas de optimización utilizadas es presentada en la Figura 2.3. Esta muestra que en general los métodos exactos (programación matemática y Branch & Bound) no son frecuentemente aplicados, representando solo el 14% de los casos. En contraste, las heurísticas se utilizan en el 28% de las publicaciones. Casos particulares de meta-heurísticas, como simulated annealing, tabu search, algoritmos genéticos y ACO son las más utilizadas para resolver problemas NPFS.

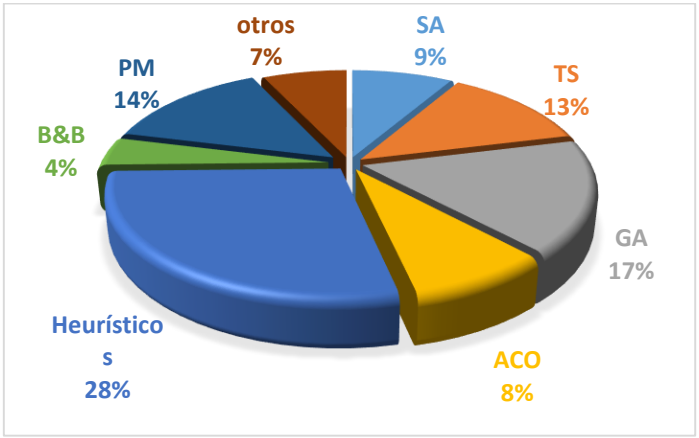


Figura 2-3. Distribución de los métodos de optimización utilizados.
Referencia: ACO: Optimización por Colonias de hormigas, GA: Algoritmos Genéticos, TS: Tabu Search, SA: Simulated Annealing, PM: Programación Matemática, B&B: Branch and Bound

2.3.1. Análisis bibliométrico

Otro aspecto que resulta de interés es la información bibliométrica respecto a la literatura NPFS. Para esto seguimos los trabajos realizados en otras revisiones del estado

del arte, como el caso de Aguezzoul (2014) y Gorman (2016), quienes mostraron información bibliométrica en sus revisiones con el fin de evaluar la investigación realizada en un tema nuevo. Esta información incluye el listado de las revistas donde suelen publicarse los artículos sobre el tema y el impacto generado por esta investigación. Respecto a esto último, Gorman (2016) centra su atención en el número de citas que reporta el buscador científico de Google al momento de realizar la revisión, en nuestro caso agosto 2016.

Tabla 2.3. Lista de las revistas que han publicado dos o más artículos sobre NPFS.
Nota: el porcentaje está calculado sobre el total de artículos revisados.

Nombre de la publicación	No. de artículos	Porcentaje
International Journal of Production Research	8	12%
Inter. Jour. of Advanced Manufacturing Technology	7	10%
Computers & Operations Research	6	9%
Proceedings	6	9%
European Journal of Operational Research	4	6%
Computers & Industrial Engineering	3	4%
Journal of Scheduling	3	4%
Applied Mathematics and Computation	2	3%
Expert Systems with Applications	2	3%
Journal of Applied Sciences	2	3%

La Tabla 2.3 presenta el listado de todas las revistas científicas que publicaron dos o más artículos NPFS. En ella podemos observar que la "International Journal of Production Research" es la que más publicaciones sobre NPFS ha tenido, agrupando el 12% de las mismas. Luego le siguen la "International Journal of Advanced Manufacturing Technology" y la "Computers & Operations Research", cada una publicó 7 y 6 artículos respectivamente. Respecto a los anales y proceedings de congresos, solo consideramos los que están indexados en SCOPUS y Google Académico, y tienen al menos su resumen escrito en inglés. Las revistas listadas en la Tabla 2.3 agrupan el 68% del total de las publicaciones NPFS.

Otras revistas que no aparecen en la Tabla 2.3 pero que también han publicado artículos sobre NPFS son " Information Sciences", "Journal of Manufacturing Systems", " International Journal of Production Economics" y " Applied Mathematical Modelling".

El impacto de los trabajos de NPFS puede evaluarse a través del número de citas reportado por Google Académico. La Tabla 2.4 presenta esta información, donde puede observarse el alto impacto de la investigación sobre NPFS, en conjunto reúnen más de 1200 citas. Esto significa que, en promedio, cada artículo de NPFS recibe 19 citas, mientras que el articulo más citado es Koulamas (1998) con 138 citas. También es de interés remarcar que más de la mitad de los artículos poseen 10 citas o más.

Tabla 2.4. Citas de los artículos NPFS tomadas de Google Scholar, agosto 2016.

Análisis bibliométrico	
Número total de citas de artículos NPFS	1264
Número promedio de citas por artículo NPFS	19
Artículo más citado (Koulamas 1998)	138
Artículos con 10 citas o más	35(>50%)

2.3.2. Casos Especiales

En esta sección se presentan algunos resultados relevantes de la revisión realizada. La primera aclaración que debe hacerse es que el enfoque NPFS debe generar soluciones mejores o iguales que el enfoque PFS para el mismo problema e instancia, ya que NPFS incluye todas las soluciones PFS e incorpora otras. Por otro lado, el esfuerzo computacional que requiere resolver problemas NPFS es mucho mayor que el que requieren los problemas PFS. En este sentido, toma importancia el poder establecer cuando es necesario resolver en formato NPFS o si con el PFS es suficiente. El primer resultado en esta línea fue presentado por Conway et al. (1967), del cual se deduce que para los casos de $F_3||C_{max}$ el enfoque PFS es óptimo. Por lo tanto, el cálculo del enfoque NPFS es redituable para sistemas de más de tres máquinas. Desde entonces se han obtenido y publicado nuevos resultados sobre esta línea. En la Tabla 2.5, destacamos algunos de ellos. La primera fila de la Tabla 2.5, presenta la cota para el peor de los casos si el problema se resuelve bajo el enfoque PFS. Las otras filas, presentan casos especiales en donde PFS no es suficiente para asegurar la solución óptima, incluso para instancias de dos máquinas.

Tabla 2.5. Resultados especiales NPFS, considerando makespan como objetivo.

Problema	**Comentarios**
$F_m \mid\mid C_{max}$ vs $F_m \mid prmu \mid C_{max}$	Peor caso del makespan PFS es: $2\sqrt{min\{m,n\}}$ veces el makespan NPFS. (Nagarajan & Sviridenko 2009)

$F_2 \mid removal\ times \mid C_{max}$	Enfoque PFS no asegura óptimo. Peor caso del makespan PFS es 3/2 veces el NPFS makespan.(Strusevich & Zwaneveld 1994)
$F_2 \mid block \mid C_{max}$	Enfoque PFS no asegura óptimo. (Strusevich & Zwaneveld 1994)
$F_2 \mid time\ delays \mid C_{max}$	Enfoque PFS no asegura óptimo. Peor caso del makespan PFS es 2 veces el NPFS makespan. (Rebaine 2005)
$F_2 \mid learning\ effect \mid C_{max}$	Enfoque PFS no asegura óptimo. (Rudek 2011)

Otros resultados experimentales importantes descriptos recientemente son:

- A mayor amplitud en el rango de los tiempos de procesamientos, mayores las posibilidades de que NPFS supere a PFS. Tando et al. (1991)
- En general, entornos productivos en que el objetivo está basado en fechas de entregas obtienen beneficios mayores del enfoque NPFS, que los que tienen objetivos basados en tiempos de finalización. Liao et al. (2006), Lin et al. (2009), Ying et al. (2010).
- A mayor amplitud en el rango de los tiempos de preparación, mayores las posibilidades que NPFS supere a PFS.
- Para flow shop simples, el makespan suele mejorar no más de un 2-3% en el caso NPFS respecto al PFS.

2.4. Conclusiones del capítulo

En este capítulo se ha revisado la literatura completa de NPFS. Hemos clasificado los artículos según las distintas variantes del problema considerada en ellos, incluyendo supuestos, restricciones, funciones objetivo y métodos de resolución utilizados por los autores. Esta revisión es, hasta lo que los autores conocen, la primera revisión de la literatura sobre NPFS.

Para el análisis de los artículos el estudio se basó en el tipo de función objetivo investigada. De este estudio se concluye que los objetivos basados en fecha de finalización han sido los más ampliamente estudiados, particularmente el caso del makespan, el cual por sí mismo agrupo más de la mitad de las publicaciones. El resto de los objetivos considerados (basados en fecha de entrega y costos) junto con los enfoques multi-objetivo, alcanzan un cuarto del total de las publicaciones. Por lo que puede afirmarse que en estos temas aun corresponde una mayor profundidad en el estudio.

Luego de la revisión completa de la literatura sobre NPFS, se descubre que queda un interrogante abierto en cuanto a calidad de solución y explicación sobre la diferencia de makespan óptimo entre el caso PFS y el NPFS. Según la cota teórica, (Nagarajan y Sviridenko, 2009) el PFS puede arrojar valores de makespan $2\sqrt{\min\{m,n\}}$ veces el makespan de NPFS, es decir que, para instancias chicas de 10 trabajos y 5 máquinas, el PFS puede estar dando un makespan óptimo de prácticamente 4 veces y media el makespan de NPFS, alrededor de un 450% de exceso. Incluso en condiciones diferentes, tales como en Rebaine (2005), que plantea que entre las sucesivas etapas del flow shop existen demoras, para el caso de *m* máquinas el ratio de los makespan óptimos es en el peor de los casos *m*, (en el ejemplo anterior implicaría 500% de diferencia). Todo esto indica que resolver un problema de secuenciamiento bajo el enfoque PFS puede llegar a ser una muy mala opción.

Por otro lado, los resultados empíricos (Tandon et al. 1991; Liao et al 2006; Benavides y Ritt 2016), muestran que el enfoque NPFS sólo logra mejorar como máximo un 2% o 3% el makespan PFS (para el caso de Tandon et al. 1991), mientras que Benavides y Ritt 2016, encuentran diferencias menores al 1%. Estos resultados demuestras que en casos clásicos de flow shop, el makespan óptimo NPFS y el PFS son mucho más cercanos que lo predicho teóricamente. Sin embargo, estos autores no esbozan ningún motivo o especulación que explique o intente explicar el motivo de la diferencia entre las cotas teóricas y las prácticas.

Una consecuencia directa de esta falta de explicaciones es que al momento de resolver un problema concreto no existe mayor información sobre como procesar previamente el problema a fin de mejorar la eficiencia de los algoritmos de búsqueda. A priori lo único que se tiene es la cota teórica, la cual no es motivo suficiente para excluir el enfoque NPFS. Todo esto sin olvidar el hecho de que se está trabajando con problemas NP-duros completos. Por lo que no es trivial pasar del enfoque PFS al enfoque NPFS, siendo un pasaje de *n!* soluciones posible a $n!^m$. Por lo tanto, en la actualidad para asegurar la calidad de una solución es necesario resolver el enfoque NPFS.

Es por eso que en esta tesis se plantea abordar este punto inconcluso de la literatura. Para ello se propone como primera línea de acción abordar los problemas NPFS y PFS modificando la estrategia de planificación de la producción, implementando el método de lotes de transferencias (*lot streaming*). De esta estrategia se pretende obtener

un nuevo escenario de comparación entre los enfoques NPFS y PFS, buscando optimizar no sólo la secuencia sino el tamaño del lote de transferencia.

Luego se pasará a una comparación más teórica e inductiva, basando el análisis en los caminos críticos de las soluciones. Para este estudio se debió desarrollar tanto las herramientas de análisis, como la estructura de la investigación.

Estas dos investigaciones planteadas son contribuciones originales al área de conocimiento NPFS, como también lo es la sistematización de la literatura.

3. Capítulo 3: IMPLEMENTACIÓN DE SUBLOTEO EN PROBLEMAS NPFS

Implementación de subloteo en problemas NPFS

3.1. Lotes de transferencia

Uno de los principales aspectos a considerar en el scheduling es el uso eficiente de los recursos. La metodología de lotes de transferencias o subloteo *(lot streaming*), ha demostrado ser un camino efectivo en esta línea. El subloteo implica dividir los lotes de producción de productos en sublotes más pequeños, lo que permite solapar sublotes del mismo producto en distintas máquinas. En los últimos años, esta técnica a despertado un interés creciente como estrategia para afrontar la optimización de los distintos objetivos de producción, incluyendo el makespan. Más aun, se ha demostrado su eficacia en reducir el volumen de trabajos en proceso como también el tiempo de ciclo de producción (Sarin y Jaiprakash 2007). Desde el primer trabajo en que se concibió esta estrategia, Reiter (1966), los desarrollos y las aplicaciones del subloteo se han incrementado considerablemente, especialmente en sistemas flow shop.

Una de las contribuciones pioneras en este tema fue presentada por Potts y Baker (1989), quienes demostraron que, para instancias de dos y tres etapas y un solo producto, el subloteo puede generar soluciones optimas con sublotes consistentes, es decir, sublotes que tienen el mismo tamaño para todas las etapas de producción. Siguiendo este trabajo, Triesch y Baker (1993) propusieron un esquema de clasificación de los problemas, Vickson (1995) extendió el análisis para múltiples productos y Sriskandarajah y Wagneur (1999) incorporaron condiciones de no-espera al problema. Kumar et al (2000) demostraron que el problema multi producto con tamaños de sublotes continuos puede reducirse al problema del agente viajero. Para afrontar instancias más grandes se utilizaron técnicas heurísticas y meta-heurísticas tales como los casos de Yoon y Ventura (2002) quienes aplicaron algoritmos genéticos para encontrar soluciones, Tseng y Liao (2008) utilizaron algoritmos de optimización por enjambres de partículas y Pan et al (2011) utilizaron un algoritmo de colonia de abejas discretas. Más recientemente, el subloteo fue investigado por Pan y Ruiz (2012) y Davendra et al. (2014). Para una revisión más completa de la literatura sobre subloteo o lotes de transferencia se remite al lector al trabajo de Cheng et al (2013).

Un aspecto común en toda la literatura de subloteo es que en todos los casos se analizaron los problemas únicamente bajo el enfoque PFS. El único artículo que hemos encontrado que trabaja con una estrategia similar al subloteo con ordenamientos no-permutativos es el de Shen et al (2014). Ellos consideraron una célula de manufactura en

donde los trabajos pertenecían a familias de productos y la producción era tipo batch. En este caso el procedimiento de división consiste en dividir las familias de productos en los distintos trabajos, pero no los trabajos en sí mismos. Esta estrategia de división asume que no es necesario procesar todos los trabajos de la misma familia en una sola corrida de producción. Entonces es posible dividir los trabajos pertenecientes a una sola familia en grupos de trabajos más chicos, e ir intercalando la producción de estos grupos de distintas familias. Sin embargo, los autores no dan indicaciones de cómo dividir estas familias en grupos "óptimos" de trabajos, o su equivalente, como generar sublotes óptimos de trabajos.

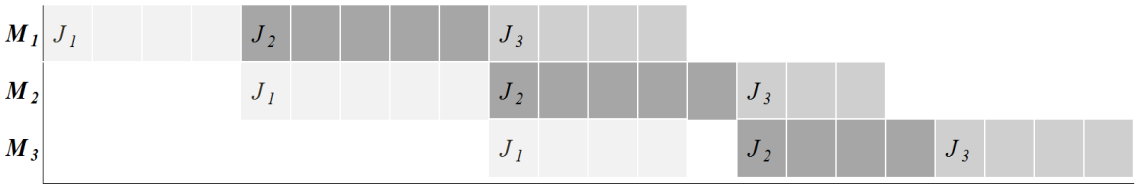
Por lo expuesto es de interés estudiar los sistemas NPFS aplicando subloteo, porque incorpora al problema nuevas variables de decisión, que a su vez, implican modificar la estructura misma de la solución. La solución final resultante será, por un lado el ordenamiento óptimo para procesar los trabajos en cada máquina, y por otro lado el tamaño óptimo de los sublotes a procesarse. De esta forma se tendrán nuevos resultados para poder evaluar el desempeño de los sistemas NPFS y PFS, cómo también ver el comportamiento en términos de estructura de la solución. También es de interés consignar que estos estudios revisten un aporte en sí mismo a la literatura de la temática.

Para ello el resto del capítulo se compone de la siguiente manera: en la siguiente sección 3.2, se introducen conceptos básicos de la metodología subloteo, en la 3.3 se presentan las formulaciones matemáticas, los resultados obtenidos en la sección 3.4 y las conclusiones del capítulo en la sección 3.5.

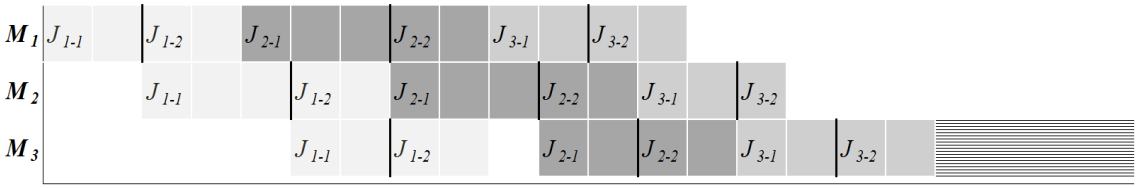
3.2. Metodología de subloteo

La metodología de subloteo aplica a escenarios en donde la producción consiste de lotes de varios ítems discretos e idénticos que tienen que ser procesados por varias máquinas en sistemas productivos flow shop. En lugar de transferir el lote entero luego de que todos sus ítems hayan sido procesados en una máquina, se considera transferir pequeños grupos de ítems del lote, llamados sublotes. Esta técnica de dividir los lotes en sublotes y procesarlos de forma simultánea en distintas máquinas (respetando su secuencia tecnológica dentro del flow shop), es lo que se conoce como subloteo o lot streaming. La Figura 3.1 muestra esquemáticamente la implementación de subloteo a un sistema flow shop de tres productos (J_1, J_2 y J_3) y tres máquinas (M_1, M_2 y M_3). Los

sublotes de los productos en la Figura 3.1.b se indican como $J_{1\text{-}1}$ y $J_{1\text{-}2}$ para referenciar el primer y segundo sublote del producto J_1. En la Figura 3.1.a, se muestra el caso base sin aplicar subloteo, mientras que en la Figura 3.1.b. se muestra la implementación del subloteo y la mejora que ésta provoca. Por ejemplo, en la máquina M_2 en la Figura 3.1.b se comienza a procesar el sublote $J_{1\text{-}1}$ mientras se procesa el sublote $J_{1\text{-}2}$ en la máquina M_1, de esta forma se solapa la producción del producto J_1 en dos máquinas. Esto deriva en una mejora del flujo de trabajo como puede evidenciarse en la región entramada de la Figura 3.1.b, que señala la mejora en el makespan.



a) Ordenamiento de tres trabajos en tres máquinas.



b) Ordenamiento de tres trabajos en tres máquinas aplicando subloteo.

Figura 3-1. Ilustraciones de una solución sin aplicar subloteo a), y de una solución aplicando subloteo b). En la figura b) la zona con trama horizontal es la ganancia en el makespan por aplicar subloteo.

A su vez dentro de la metodología de subloteo es posible considerar distintos tipos de implementación, en las cuales se modelan las distintas cuestiones que involucra el incorporar en el proceso de toma de decisiones el dimensionamiento del tamaño del sublote. Algunas de las más características de la metodología según Feldmann y Biskup (2008) son:

- *Mono producto / múltiple producto*. Se considera un solo producto o múltiples productos.
- *Sublotes consistentes*. Se considera que un sublote es consistente cuando no modifica su tamaño a lo largo del proceso productivo
- *Sublotes iguales*. Todos los sublotes de un mismo producto tienen el mismo tamaño
- *Sublotes variables*. No se aplica ninguna restricción al tipo de sublote.

- *Sublotes sin demoras / con demoras intermitentes*. Para el caso de los sublotes sin demoras todos los sublotes del mismo producto deben procesarse sin demoras en una misma máquina. En cambio, si se aceptan demoras intermitentes, pueden existir demoras en el procesamiento en una misma máquina entre dos sublotes consecutivos de un mismo producto.
- *Ordenamientos sin demoras / con demoras.* Si el ordenamiento no acepta demoras, los sublotes tienen que pasar sin sufrir esperas entre las sucesivas máquinas. Cuando se aceptan demoras, el sublote puede esperar antes de pasar a la siguiente máquina.
- *Tiempos de preparación adjuntos / no adjuntos al sublote.* Si los tiempos de preparación deben ser adjuntos al sublote, no puede comenzar a prepararse la máquina hasta que el sublote no llegue a la máquina. En el caso de que no sean adjuntos, la preparación es independiente de la llegada del sublote.
- *Sublotes discretos / continuos.* Para el caso de sublotes discretos, el número de ítems de un sublote debe ser entero. Para sublotes continuos, no existe esa condición.
- *Sublotes entremezclados / no-entremezclados.* Si en una producción con múltiples productos se permiten sublotes entremezclados, implica que la secuencia de producción de los sublotes de un mismo producto *j* puede ser interrumpidas por la producción de sublotes de otro producto *k*. Si no se permite entremezcla de sublotes, la producción de los sublotes de un mismo producto es ininterrumpida, lo cual se cumple siempre para producciones mono-producto.

En nuestro caso trabajaremos con sistemas de múltiples productos, ya que carecería de sentido abordar problemas mono producto por impedir trabajar el enfoque NPFS. Respecto a los distintos tipos de sublotes: iguales, consistentes y variables, usaremos el tipo de sublote consistente. Los sublotes iguales son los más utilizados en la literatura sobre flow shop con subloteo tal como muestran las revisiones (Chang y Chiu 2005; Cheng et al, 2013). Este tipo de sublotes permite obtener los beneficios propios de la técnica de subloteo, ya que divide los lotes de producción en sublotes. Sin embargo, no reviste mayor interés para nuestro caso porque las decisiones a tomar respecto a los

tamaños de los sublotes quedan bastante restringidas. Al definir el tamaño del sublote, queda implícitamente definido el número de sublotes. Por lo que, al estudiar los sistemas NPFS con ese tipo de sublotes la estructura de la solución puede que resulte bastante similar a la de los PFS. También es válido aclarar que las soluciones de sublotes consistentes engloba a las soluciones de sublotes iguales (no las descarta). Por otro lado, al utilizar sublotes variables implica flexibilizar incluso la restricción de que los sublotes sean iguales a lo largo del proceso productivo, lo que genera un incremento considerable en el esfuerzo computacional. Esto limita el tamaño de la instancia a estudiar. Dado que se pretende estudiar experimentalmente el comportamiento del enfoque NPFS y PFS aplicando subloteo, es necesario fijar algunas condiciones (por no ser un análisis teórico generalizable), especialmente, que los escenarios evaluados sean los mismos y que la calidad de las soluciones sean comparables. Respecto a los escenarios es sencillo fijar el mismo escenario para los dos enfoques. Respecto a la calidad de la solución no es tan simple realizar una comparación. Por eso utilizaremos un solver que permita asegurar la solución óptima, en este caso CPLEX. De esta forma aseguramos que, al comparar la calidad de las soluciones y la distribución de los sublotes, lo estamos haciendo con soluciones óptimas para cada enfoque. El utilizar sublotes variables acotaría demasiado el tamaño de la instancia a experimentar. Además, no es el tipo de sublote más ampliamente investigado.

En cuanto a los tiempos de demora tanto en los sublotes como en los ordenamientos, daremos flexibilidad a estos tiempos de forma de poder abordar el caso más general. De los tiempos de preparación consideraremos que no son adjuntos pero que debe considerarse el tiempo necesario para su preparación. El tipo de sublote considerado es discreto y no se acepta la entremezcla de sublotes de distintos productos. Esto último, lo consideramos de esa manera, por ser poco representativo de una realidad industrial en la que la alternancia en la producción de distintos productos se dé a nivel de sublotes.

3.3. Modelos matemáticos

Los modelos matemáticos presentados en esta sección integran los dos tipos de decisiones que hemos venido tratando: la secuencia de trabajos a procesar y el dimensionamiento de los sublotes. Por lo tanto, la solución del modelo brindará el número de sublotes por trabajo, el tamaño de los sublotes, el ordenamiento de los trabajos y el

correspondiente makespan. En la implementación de subloteo que se presenta a continuación, cada trabajo puede dividirse en sublotes hasta cierto número máximo. Los sublotes son discretos, ya que la producción se cuenta en unidades de producto, y los sublotes que pertenecen al mismo trabajo se procesan de forma consecutiva. El tamaño de los sublotes puede variar entre sí, de esta forma se asegura el óptimo en el dimensionamiento.

Otros supuestos incluidos son:

- Las máquinas no pueden procesar más de un trabajo/sublote a la vez.
- Un sublote no puede ser procesado por más de una máquina a la vez.
- Todas las máquinas están disponibles al inicio del horizonte de planificación.
- Las máquinas pueden estar ociosas durante el horizonte de planificación.

A continuación, presentaremos una formulación mixta entera lineal que represente un problema de secuenciamiento NPFS, para luego pasar a la presentación de NPFS con subloteo como también PFS con subloteo.

3.3.1. Modelo mixto entero NPFS

A continuación la presentación del modelo mixto entero NPFS

Índices

M Número de máquinas; $m = 1,2,\ldots,M$

J Número de trabajos (o productos); $j = 1,2,\ldots,J$

Datos de entradas

U_j Número de unidades del producto j (tamaño del lote de producción de j).

$p_{j,m}$ Tiempo de procesamiento de una unidad de producto j en la máquina m.

$ts_{m,j}$ Tiempo de preparación para producir el producto j en la máquina m.

$tr_{m,j}$ Tiempo de descarga y transferencia asociado a la máquina m después de procesar una o más unidades de producto j ($tr_{m,j}$ no depende de la cantidad de unidades procesadas).

Ω Número positivo lo suficientemente grande.

Variables

$B_{j,m}$ Tiempo de inicio del procesamiento del trabajo j en la máquina m.

$c_{j,m}$ Tiempo de finalización del trabajo j en la máquina m.

$x_{j,j',m}$ Binaria que es 1 si el trabajo j es procesado antes que el trabajo j' en la máquina m y es 0 en cualquier otro caso, j es distinto de j'.

Formulación MILP

Minimizar $z = C_{max}$ (3.1)

Sujeto a:

Restricción 3.2:

$$B_{j,m} + p_{j,m} \cdot U_j + tr_{m,j} + ts_{m,j} \leq B_{j,m+1}\,, \qquad j = 1, \dots, J\,; m = 1, \dots, M-1$$

Restricción 3.3:

$$B_{j,m} + p_{j,m} \cdot U_j + tr_{m,j} + ts_{m,j} \leq B_{j',m} + \Omega \cdot \left(1 - x_{j,j',m}\right), \qquad j \neq j', m = 1, \dots, M$$

Restricción 3.4:

$$B_{j',m} + p_{j',m} \cdot U_{j'} + tr_{m,j'} + ts_{m,j'} \leq B_{j,m} + \Omega \cdot x_{j,j',m}, \qquad j \neq j', m = 1, \dots, M$$

Restricción 3.5:

$$x_{j,j',m} + x_{j',j,m} = 1, \qquad m = 1, \dots, M, j \neq j'$$

Restricción 3.6:

$$B_{j,m} + p_{j,m} \cdot U_j + tr_{m,j} + ts_{m,j} \leq c_{j,m}, \qquad j = 1, \dots, J, m = 1, \dots., M$$

Restricción 3.7:

$$c_{j,m} \leq C_{max}, \qquad j = 1, \dots, J, m = M$$

Restricción 3.8:

$x_{j,j',m}$, $j \neq j', m = 1, \dots, M$ es binaria

Restricción 3.9:

$$B_{j,m}, c_{j,m} \geq 0, \qquad j = 1, .., J, m = 1, \dots, M$$

La función objetivo 3.1 es el makespan de la producción. La restricción 3.2 establece que cada trabajo finalice su operación en la estación dada antes de pasar a la siguiente estación (m+1). Las restricciones (3.3) y (3.4) trabajan de forma conjunta y establecen la producción en cada máquina m. Por ejemplo, si el trabajo j es procesado en

cualquier momento antes que *j'* entonces (3.3) se vuelve activa y (3.4) se vuelve redundante. La condición lógica de precedencia se expresa en (3.5). Si el trabajo *j* precede al trabajo *j'* en la máquina *m*, entonces el trabajo *j'* no puede preceder al trabajo *j* en esa misma máquina. La expresión (3.6) calcula el tiempo de finalización del trabajo *j* en la máquina *m*, mientras que la (3.7) calcula el tiempo máximo de finalización de todos los trabajos. Finalmente (3.8) y (3.9) presentan las condiciones de factibilidad y de variables discretas.

3.3.2. Modelo mixto entero NPFS con subloteo

Para extender el modelo anterior definido por las expresiones (3.1) - (3.9) a NPFS con subloteo, es necesario modificarlo de manera que puedan dividirse los lotes en sublotes. Para ello se requiere un nuevo índice *f* para sublotes y un grupo de nuevas variables.

Índice

F número máximo posible de sublotes por trabajo; $f = 1,2,\ldots,F$

Variables

$A_{f,j,m}$ Tiempo de llegada del sublote *f* del trabajo *j* a la máquina *m.*

$s_{f,j}$ Número de unidades de producto *j* en sublote *f*

$y_{f,j}$ Binaria igual a 1 si sublote *f* del producto *j* no está vacío, 0 cualquier otro caso

La especificación del tiempo de preparación ahora requiere de dos nuevos parámetros. Uno de ellos $tsls_{m,j}$ indica el tiempo de preparación cuando se comienza la producción de un nuevo producto *j* en la máquina *m*, y $tslss_{m,j}$ indica el tiempo de preparación para cada sublote que comienza a producirse en la máquina *m*. Estos tiempos representan, respectivamente, los cambios de productos en las máquinas ($tsls_{m,j}$), y los cambios menores que se requieren en la máquina para producir un nuevo sublote ($tslss_{m,j}$).

Restricción 3.10:

$$\sum_{f=1}^{F} s_{f,j} = U_j \ , \qquad \forall j$$

Restricción 3.11:

$$s_{f,j} \leq \Omega \cdot y_{f,j} \ , \qquad \forall f, j$$

Restricción 3.12:

$$A_{f=1,j,m} + p_{j,m} \cdot s_{f=1,j} + \left(tr_{m,j} + tslss_{m,j}\right) \cdot y_{f=1,j} + tsls_{m,j} \leq A_{f,j,m} \ , \ \forall j, m, \ f > 1$$

Restricción 3.13:

$$A_{f,j,m} + p_{j,m} \cdot s_{f,j} + \left(tr_{m,j} + tslss_{m,j}\right) \cdot y_{f,j} + tsls_{m,j} \leq A_{f+1,j,m} \ , \ \forall j, m, f = 2, \dots, F-1$$

Restricción 3.14:

$$A_{f=1,j,m} + p_{j,m} \cdot s_{f=1,j} + \left(tr_{m,j} + tslss_{m,j}\right) \cdot y_{f=1,j} + tsls_{m,j} \leq A_{f=1,j,m+1} \ , \ \forall j, m < M$$

Restricción 3.15:

$$A_{f,j,m} + p_{j,m} \cdot s_{f,j} + \left(tr_{m,j} + tslss_{m,j}\right) \cdot y_{f,j} + tsls_{m,j} \leq A_{f,j,m+1} \ , \qquad \forall j, m, f > 1$$

Restricción 3.16:

$$A_{f,j,m} + p_{j,m} \cdot s_{f,j} + \left(tr_{m,j} + tslss_{m,j}\right) \cdot y_{f,j} \leq c_{j,m} \ , \qquad \forall \, j, m, f = F$$

Restricción 3.17:

$$A_{f=1,j',m} + \Omega \cdot \left(1 - x_{j,j',m}\right) \leq c_{j,m} \ , \qquad j \neq j', m = 1, \dots, M$$

Restricción 3.18:

$$A_{f=1,j,m} + x_{j,j',m} \leq c_{j',m} \ , \qquad j \neq j', \forall m$$

Restricción 3.19:

$$c_{j,m} \leq C_{max} \ , \qquad j = 1, \dots, J, m = M$$

Restricción 3.20:

$$x_{j,j',m} + x_{j',j,m} = 1, \qquad m = 1, \dots, M, j \neq j'$$

Restricción 3.21:

$x_{j,j',m}, y_{f,j}$, $j \neq j', j = 1, \dots, J, \forall m, f$ son binarias

Restricción 3.22:

$A_{f,j,m} \geq 0$, $s_{f,j,m} \geq 0, \forall f, j, m$ son enteras

La función objetivo presentada en 3.1 sigue siendo el makespan. La restricción 3.10 definen el tamaño de los sublotes $s_{f,j}$ y aseguran de que todas las unidades de producto j, U_j , se asigne a algún sublote. El número máximo de sublotes posibles está dado por F. La restricción 3.11 fuerza a que $y_{f,j}$ sea 1 cuando $s_{f,j}$ sea distinto de cero. En

3.12, el tiempo de inicio del primer sublote del producto *j* considera el tiempo de preparación $tsls_{m,j}$ y lo calcula incluso si el sublote *f* está vacío. Esto asegura que la máquina *m* esté lista para procesar el resto de los sublotes del trabajo *j*. Las restricciones 3.13 determinan la secuencia de sublotes. Antes de que el próximo sublote pueda procesarse, tiene que haber transcurrido el tiempo de preparación, de procesamiento y de transferencia. La restriccion 3.14 regula el tiempo de arribo del primer sublote del trabajo *j* a la próxima máquina *m*+1. El lado izquierdo de la restricción incluye los tiempos de preparación $tsls_{m,j}$ y de procesamiento $p_{j,m}$ en la máquina *m*, y de transferencia $tr_{m,j}$ a la máquina *m*+1. Para el resto de los sublotes, el tiempo de arribo en las máquinas siguientes se rige por la restriccion 3.15. El tiempo de finalización de cada trabajo en cada máquina se calcula por medio de 3.16, considerando como tiempo de finalización el del último sublote de ese trabajo. El secuenciamiento de los trabajos en cada máquina se determina por las restricciones 3.17 y 3.18, que trabajan de forma análoga a las restricciones 3.3 y 3.4 ya descriptas. Si el trabajo *j* es procesado en cualquier momento antes que el trabajo *j'*, entonces 3.17 asegura que el trabajo *j'* se procesará una vez que el trabajo *j* haya terminado, y 3.18 se convierte en redundante. En otro caso, el rol de cada ecuación es invertido. La restricción 3.19 define el makespan como el tiempo de finalización del último sublote en la última máquina. La restricción lógica 3.20 es la misma que la (3.5). Las restricciones 3.21 y 3.22 definen las variables.

3.3.3. Modelo mixto entero PFS con subloteo

En esta sección presentaremos de modo breve el modelo PFS aplicando subloteo. Para ello se utilizará como guía el modelo presentado en la Sección 3.3.2., por las restricciones (3.10) – (3.22). Partiendo de este modelo se mostrarán las principales diferencias entre aplicar subloteo a problemas PFS y NPFS. La primera diferencia a remarcar es que el ordenamiento de los trabajos se mantiene constante para todas las máquinas por lo que es necesario reformular las variables que indican la precedencia de los trabajos. Se presenta la variable binaria $w_{j,j'}$ la cual toma el valor 1 si el trabajo *j* precede al trabajo *j'*, y 0 en cualquier otro caso. Es de interés destacar que esta variable funciona de manera similar a la variable $x_{j,j',m}$, con la salvedad de que no está indizada en el conjunto *m*.

Por lo tanto, el modelo mixto entero que representa al problema PFS con subloteo es idéntico al (3.10) - (3.22) salvo que las restricciones (3.17), (3.18), (3.20) y (3.21) deben ser reemplazadas por otras nuevas presentadas a continuación.

Restricción 3.23:

$$c_{j,m} \leq A_{f=1,j',m} + \Omega \cdot \left(1 - w_{j,j'}\right) , \qquad \forall m, j \neq j$$

Restricción 3.24:

$$c_{j',m} \leq A_{f=1,j,m} + \Omega \cdot w_{j,j'} , \ \forall m, j \neq j$$

Restricción 3.25:

$$w_{j,j'} + w_{j',j} = 1, \qquad j \neq j'$$

Las restricciones 3.23 y 3.24 reemplazan a las restricciones 3.18 y 3.19, respectivamente. El significado de éstas es análogo al de las 3.18 y 3.19, en donde asegura que el trabajo *j'* inicie su procesamiento después de que terminen su procesamiento todos los trabajos que lo preceden. La restricción 3.25, asegura el ordenamiento lógico de los trabajos. Si el trabajo *j* precede al trabajo *j'*, entonces lo inverso no es válido.

3.4. Experimentación Computacional

Los tres modelos presentados anteriormente serán las herramientas utilizadas para evaluar el impacto del subloteo con los problemas NPFS y su comparación respecto a PFS. Se plantearon para esta experimentación instancias de 5 y 10 máquinas, y el número de trabajos fue de 2, 4 y 5. Los parámetros de entrada fueron generados siguiendo la literatura estándar de flow shop (Taillard 1993). En nuestro caso, los tiempos unitarios de procesamiento fueron obtenidos de una distribución uniforme entre [1,5] y los tiempos de transferencia por trabajo/sublote de una distribución uniforme entre [1,4]. Para el número de unidades contenidas en los lotes U_j, se utilizó una distribución uniforme entre [20,50]. Para las instancias sin sublote el tiempo de preparación se modeló a través de una distribución uniforme entre [1,50]; para los caso con subloteo, se muestreó $tsls_{m,j}$ considerando una distribución uniforme entre [1,25] y para $tslss_{m,j}$ entre [1,10].

Para cada instancia, se permitieron distintos números máximos de sublotes (considerando desde el caso sin subdivisión, hasta un máximo de 6 sublotes por trabajo) a fin de evaluar el comportamiento de NPFS bajo diferentes condiciones de subloteo. Para cada conjunto de parámetros, se generaron y evaluaron 5 instancias. Todos los modelos

se resolvieron con CPLEX 12.5.0 usando la configuración estándar y un sistema operativo Windows 10 de 64 bit, con un procesador Intel Core i5 y una memoria RAM de 8 GB.

3.5. Resultados

Los resultados obtenidos tienen varias aristas para ser estudiadas. El primer aspecto analizado es el impacto del subloteo en los sistemas NPFS y PFS. Para esto se estudian los costos-beneficios del subloteo. Los costos computacionales se miden a través del número de restricciones y variables, y los beneficios observando cómo se modifica el makespan. El segundo análisis es la comparación de los sistemas NPFS y PFS.

3.5.1. Costo computacional del subloteo

El incorporar nuevas decisiones al problema de scheduling implica, naturalmente, resolver un problema más grande en términos de variables y restricciones. Para estudiar este aspecto en la implementación de subloteo en sistemas NPFS y PFS presentamos la Tabla 3.1. Esta tabla muestra el tamaño en términos de restricciones y variables, tanto continuas como discretas. También muestra el tiempo promedio de cómputo necesario para resolver cada instancia.

En la Tabla 3.1 puede observarse que, efectivamente, la aplicación de subloteo implica un crecimiento en el tamaño del problema en términos de restricciones y variables, tanto continuas como discretas. A su vez este crecimiento se ve afectado por el máximo número posibles de sublotes en los que puede dividirse cada producto, siendo menor el número de restricciones y variables para los casos en que los números de sublotes posibles es más chico. Las diferencias relativas en el número de restricciones entre los casos en que no hay subloteo y sí lo hay, es mayor cuanto más chica es la instancia. Siendo para el caso NPFS casi 5 veces mayor el número de restricciones para 2 trabajos, 5 máquinas y 6 sublotes (149 restricciones) que cuando no se aplica subloteo (32 restricciones). Mientras, que para el caso de 5 trabajos y 10 máquinas las diferencias relativas son de poco más del doble, 996 restricciones para 6 sublotes frente a 455 restricciones para sin subloteo. Para el caso PFS, las diferencias relativas son más acentuadas, siendo casi de 6 veces para el caso de 2 trabajos y 5 máquinas, y de casi 3 veces para 5 trabajos y 10 máquinas. El número de variables también se ve afectado, aunque en menor medida. En todos los casos se resolvió hasta obtener la solución óptima (gap = 0).

Tabla 3.1. Tamaño de las instancias evaluadas en términos de restricciones, variables y tiempos promedio de cómputo.

Referencia: R: restricciones, VC: variables Continuas, VD: variables discretas, T: tiempo promedio de cómputo medido en segundos.

m	*j*		*Sin subloteo*		*f* = 2		*f* = 3		*f* = 4		*f* = 5		*f* = 6	
			PFS	**NPFS**	**PFS**	**NPFS**	**PFS**	**NPFS**	**PFS**	**NPFS**	**PFS**	**NPFS**	**PFS**	**NPFS**
5	*2*	***R***	24	32	52	61	74	83	96	105	118	127	140	149
		VC	15	23	33	42	47	56	61	70	75	84	89	98
		VD	2	10	10	18	14	22	18	26	22	30	26	34
		T	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1
	4	***R***	96	144	152	202	196	245	240	289	284	333	328	377
		VC	37	85	73	122	101	150	129	178	157	206	185	234
		VD	12	60	28	76	36	84	44	92	52	100	60	108
		T	< 1	< 1	< 1	< 1	< 1	2	2	8	17	58	133	420
	5	***R***	150	230	220	301	275	356	330	411	385	466	440	521
		VC	51	131	96	177	131	212	166	247	201	282	236	317
		VD	20	100	40	120	50	130	60	140	70	150	80	160
		T	< 1	< 1	< 1	3	2	20	26	88	401	1250	4430	14268
10	*2*	***R***	44	62	92	111	134	153	176	195	218	237	260	279
		VC	25	43	53	72	77	96	101	120	125	144	149	168
		VD	2	20	10	28	14	32	18	36	22	40	26	44
		T	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	< 1	3
	4	***R***	176	284	272	381	356	465	440	549	524	633	608	717
		VC	57	165	113	222	161	270	209	318	257	366	305	414
		VD	12	120	28	136	36	144	44	152	52	160	60	168
		T	< 1	< 1	< 1	< 1	< 1	4	3	70	20	752	780	1680
	5	***R***	275	455	395	576	500	681	605	786	710	891	815	996
		VC	76	256	146	327	206	387	266	447	326	507	386	567
		VD	20	200	40	220	50	230	60	240	70	250	80	260
		T	< 1	< 1	< 1	5	3	107	15	890	990	13550	19720	60300

En la Tabla 3.1, puede observarse que para las instancias con mayor número de sublotes es donde los tiempos de cómputo para NPFS se vuelven considerablemente mayores que para PFS. A su vez, los tiempos de cómputo para PFS también aumentan a medida que el número de sublotes aumenta. Esto permite mostrar que la estrategia de subloteo con sublotes consistentes tiene un costo computacional considerable, incrementando el número de variables continuas y discretas. La Figura 3.2 muestra los tiempos de cómputo para los casos NPFS y PFS de 5 trabajos tanto para 5 como para 10 máquinas. En ella puede verse que el caso de 5 máquinas y 5 trabajos verifica claramente el aumento señalado. Para instancias sin subloteo o con número de sublotes igual a menor

a 3, el tiempo de cómputo es de solo algunos segundos, alcanzando los 20 segundos para 3 sublotes. Sin embargo, si el número máximo de sublotes posibles se aumenta hasta 6, los tiempos de cómputo crecen considerablemente, llegando a tardar casi una hora y media (4430 segundos) para el enfoque PFS, y casi 4 horas para el NPFS (14268 segundos). Este incremento en los tiempos de cómputo se da principalmente por el aumento en la cantidad de variables continuas. Este mismo comportamiento, pero de forma más pronunciada, puede observarse en la instancia de 5 trabajos y 10 máquinas, donde los tiempos de cómputo alcanzan más de 16 horas para NPFS (60300 segundos) y más 5 de para PFS (19720 segundos), para el caso de 6 sublotes. También, puede evidenciarse es que el número de restricciones crece en menor medida

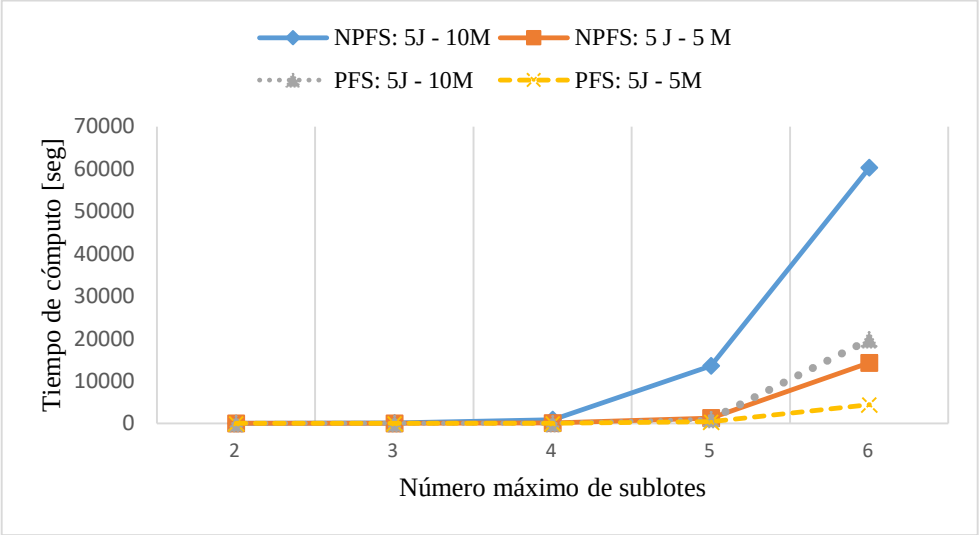


Figura 3-2. Tiempos de cómputo respecto al incremento en el número de sublotes.

3.5.2. Evaluación del makespan aplicando subloteo

Dado que el propósito de aplicar subloteo es obtener mejores soluciones que las que se pueden obtener sin su aplicación, se presenta esta sección. En esta sección se estudia el impacto que tiene sobre la función objetivo estudiada, el makespan, dependiendo de las distintas condiciones evaluadas en las experimentaciones. Respecto al impacto del subloteo en la calidad de la solución obtenida se muestran las Tablas 3.2 y 3.3. En la Tabla 3.2, se presentan los resultados para el enfoque NPFS y en la Tabla 3.3 para el PFS. Los resultados son el promedio de los resultados obtenidos para cada uno de los 5 escenarios. En las tablas se presentan, el valor del makespan y el número promedio de sublotes utilizados en ese makespan óptimo.

Tabla 3.2. Resultados NPFS, valores de makespan y número promedio de sublotes.

				f				
m	***j***		***Sin subloteo***	**2**	**3**	**4**	**5**	**6**
5	**2**	makespan	1187	914	872	863	862	862
		nº sublotes		4	5,8	6,4	6,6	6,6
	4	makespan	1760	1544	1505	1502	1502	1502
		nº sublotes		7,8	9,4	9,8	9,8	9,8
	5	makespan	2028	1762	1728	1726	1726	1726
		nº sublotes		8,4	10,8	11,2	11,2	11,2
10	**2**	makespan	2278	1556	1371	1316	1305	1302
		nº sublotes		4	5,8	7,2	7,6	8
	4	makespan	2950	2170	1996	1950	1941	1940
		nº sublotes		8	10,8	12,4	13,2	13,4
	5	makespan	3112	2366	2220	2188	2185	2185
		nº sublotes		9,8	12,8	14,2	15	15

Tabla 3.3. Resultados PFS, valores de makespan y número promedio de sublotes.

				f				
m	***j***		***Sin subloteo***	**2**	**3**	**4**	**5**	**6**
5	**2**	makespan	1187	914	872	863	862	862
		nº sublotes		4	6	6,4	6,6	6,6
	4	makespan	1760	1544	1505	1502	1502	1502
		nº sublotes		7,8	9,4	9,8	9,8	9,8
	5	makespan	2028	1762	1728	1726	1726	1726
		nº sublotes		8,4	10,8	11,2	11,2	11,2
10	**2**	makespan	2278	1556	1371	1316	1305	1302
		nº sublotes		4	5,8	7,6	7,6	8
	4	makespan	2953	2170	1996	1950	1941	1940
		nº sublotes		8	10,8	12,4	13,2	13
	5	makespan	3117	2366	2223	2188	2185	2185
		nº sublotes		9,8	13,2	14,8	15,6	15,6

En las Tablas 3.2 y 3.3 puede observarse que el subloteo tiene un claro impacto en el makespan tanto bajo el enfoque NPFS como el PFS. Por ejemplo, en el caso NPFS de 2 trabajos y 5 máquinas el makespan pasa de 1187 cuando no hay subdivisión, a 862 cuando el número de sublotes permitido llega hasta 5. Esta reducción representa un 25% de mejora. A su vez esta disminución del makespan guarda relación con el número máximo de sublotes admitidos. En la Tabla 3.3 (PFS), de 2 trabajos y 10 máquinas, el makespan se reduce de 2278 a 1556 (32%), al aplicar subloteo y permitiendo como máximo 2 sublotes por trabajo. Pero si se admiten 3 sublotes por trabajo, el makespan vuelve a reducirse hasta 1371 (40% respecto a sin subloteo y un 12% respecto a admitir

2 sublotes como máximo). Y luego el makespan continúa descendiendo a medida que se aceptan cada vez más sublotes, pero con menor intensidad. Estos fenómenos, tanto la mejora del makespan al aplicar subloteo como la dependencia en el número máximo de sublotes, se observa bajo los dos enfoques, NPFS y PFS.

Para ilustrar el comportamiento del makespan respecto al número de sublotes se presenta la Figura 3.3. En la misma se muestra para el caso NPFS cómo se comporta el makespan para las instancias de 5 trabajos y 10 máquinas, y 2 trabajos y 5 máquinas, respecto al máximo número de sublotes admitidos. En la misma puede verse que el subloteo permite reducir el makespan en relación a cero sublotes (sin subloteo). Aunque hay que destacar que la mejora no es constante, alcanza un cierto límite a partir de donde admitir más sublotes por producto, esto es aumentar el conjunto *F*, no redunda en una mejora del makespan. Esto se aprecia mejor en la Figura 3.4.

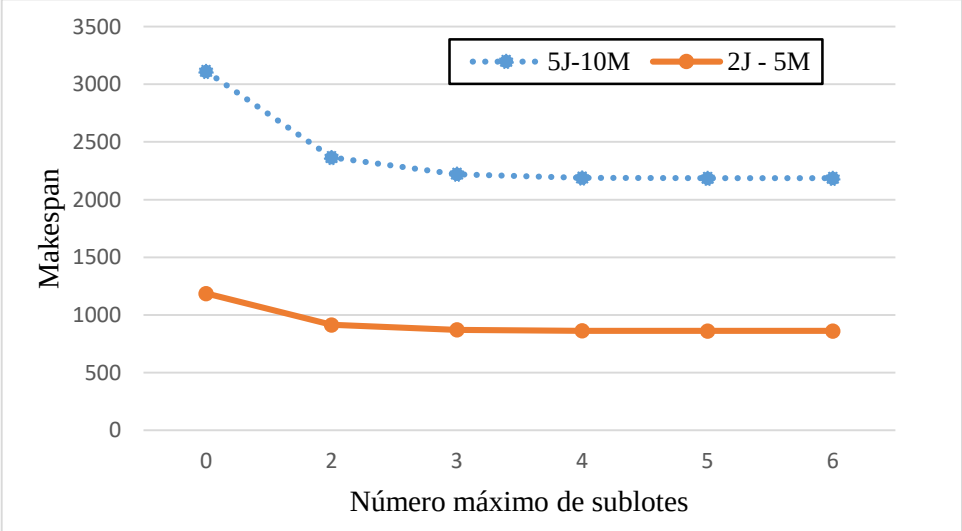


Figura 3-3. Descenso del makespan al incrementar el número máximo de sublotes disponibles

En la Figura 3.4 pueden verse los mismos casos ilustrados en la Figura 3.3, soluciones NPFS para 5 trabajos y 10 máquinas, y 2 trabajos y 5 máquinas. Las curvas de la Figura 3.4 representan el uso promedio de sublotes por producto respecto al número máximo de sublotes admitidos por producto. El uso promedio de sublotes por producto se lo calcula dividiendo el dato reportado en la Tabla 3.2 por el número de productos considerados en el caso. En la misma puede verse cómo a partir de cierto punto el admitir más sublotes por producto (conjunto *F*) no implica que la solución subdivida en más sublotes. En otras palabras, existe un óptimo a partir del cual no incrementa la cantidad de sublotes por productos. Esto demuestra que existe un número óptimo de sublotes.

Admitir más sublotes, no implica que sean incorporados en la solución. Sin embargo, para ver la influencia del número máximo de sublotes admitidos es necesario analizar las Tablas 3.2 y 3.3, y evaluar a partir de qué instancia el número de sublotes totales utilizados en la solución deja de modificarse. Este fue el criterio utilizado para limitar las experimentaciones. Una vez que para un número máximo de sublotes por productos no modifica la solución respecto a la solución del número menor inmediato de sublotes, se consideró que se alcanzó la saturación del sistema. Otro aspecto a notar es que a medida que el sistema crece en número de trabajos y máquinas, el número promedio de sublotes es menor.

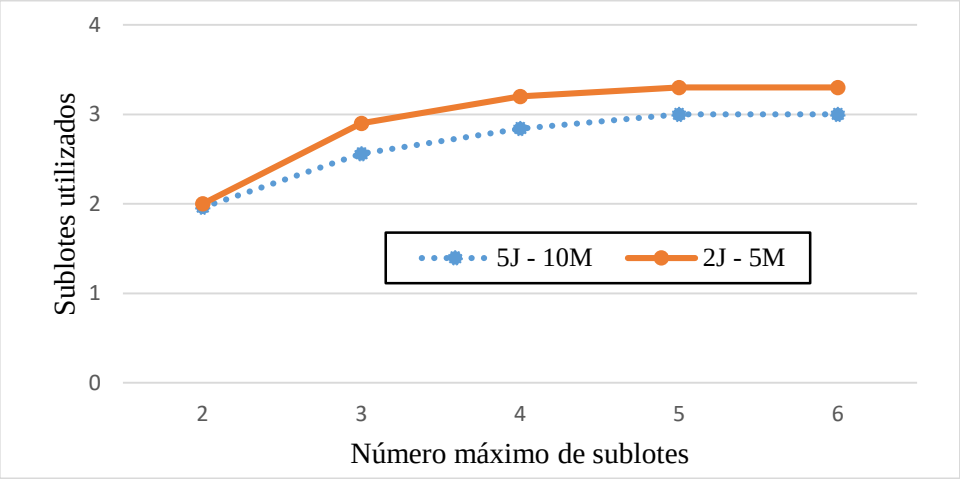


Figura 3-4. Uso promedio de sublotes por producto respecto al número máximo de sublotes admitidos por producto.

Por otro lado, si se comparan los valores de los dos enfoques, NPFS y PFS, entre sí para una misma instancia, se observa que prácticamente las diferencias nulas. Observando los valores del makespan, Tablas 3.2 y 3.3, los únicos que difieren de una Tabla a otra son las instancias sin subloteo de 4 y 5 trabajo y 10 máquinas, y en el caso de admitir hasta 3 sublotes por trabajo para 5 trabajos y 10 máquinas. Cabe destacar que las diferencias entre los makespan son muy reducidas, en todos los casos son diferencias menores al 1% (0,16% es la diferencia máxima), y sólo se dan en las instancias de 10 máquinas. Por otro lado, mirando los valores de sublotes utilizados, hay varios que no son coincidentes. La gran mayoría de estos valores se encuentran para las instancias de 10 máquinas, especialmente en el caso de 5 trabajos. En este último caso, el número de sublotes promedio es distinto para casi todas las condiciones de subloteo, sólo coinciden en el caso de admitir como máximo 2 sublotes por trabajo. Nuevamente, estas diferencias son de magnitud reducida (decimales) y siempre es menor el valor del enfoque NPFS.

Tabla 3.4. Reducción del makespan expresada en porcentajes respecto al caso sin subloteo.
Nota: En caso de que los valores para NPFS y PFS difieren se aclara.

			f			
m	***j***	**2**	**3**	**4**	**5**	**6**
5	2	23,0%	26,6%	27,3%	27,4%	27,4%
	4	12,2%	14,4%	14,6%	14,6%	14,6%
	5	13,1%	14,8%	14,9%	14,9%	14,9%
10	2	31,7%	39,8%	42,2%	42,7%	42,8%
	4	NPFS 26,4%	32,3%	33,9%	34,2%	34,2%
		PFS 26,5%	32,4%	34,0%	34,2%	34,3%
	5	NPFS 24%	28,6%	29,7%	29,8%	29,8%
		PFS 24,1%	28,7%	29,8%	29,9%	29,9%

Para continuar describiendo el efecto del subloteo en el makespan se presenta la Tabla 3.4. En ella se muestra la reducción del makespan respecto al caso sin aplicar subloteo expresado en porcentaje indicando cuándo los enfoques NPFS y PFS difieren en valor. Para las instancias de 5 máquinas, los makespan para NPFS y PFS fueron coincidentes, por lo que los porcentajes de reducción también. Para los casos en que no coincidieron, 10 máquinas y 4 y 5 trabajos, se presentan los porcentajes según cada enfoque. En la tabla puede observarse que las diferencias en las reducciones del makespan para ambos enfoques responden al orden de 0,1%. También es de interés mencionar que el subloteo sí funciona como un método eficaz a la hora de abordar la optimización del makespan, alcanzando reducciones del mismo de más del 40% para el caso de 2 trabajos y 10 máquinas.

Del análisis que se viene desarrollando es posible observar que el número de máquinas y de trabajos son factores de influencia. Para presentar con mayor claridad estas relaciones se presentan las Figuras 3.5 y 3.6. En la Figura 3.5 se ilustra la reducción del makespan respecto al caso sin aplicar subloteo expresada en porcentaje para el caso de 2 trabajos, tanto para 5 como para 10 máquinas, lo que permite poder analizar la influencia del número de máquinas. En la Figura 3.6, la intención del gráfico es similar, pero para analizar el impacto del número de trabajos. Para eso se grafican las curvas para 2 y 5 trabajos, fijando el número de máquinas en 5 para ambas curvas.

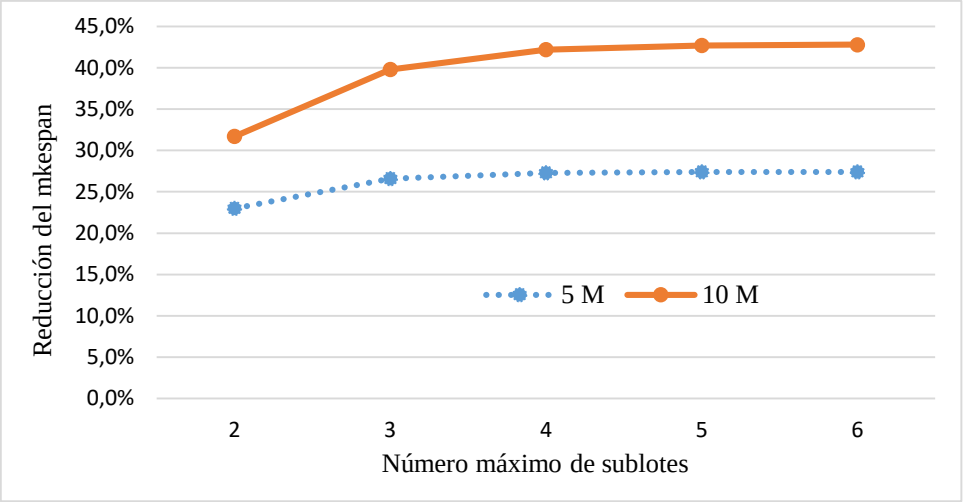


Figura 3-5. Reducción del makespan en relación al máximo número de sublotes admitidos para el caso de 3 productos.

En la Figura 3.5, puede notarse que para un mismo número de productos el incrementar el número de máquinas aumenta el beneficio de aplicar subloteo. Para el caso de 2 productos y 10 máquinas las reducciones del makespan superan el 40%, mientras que para el caso de 5 máquinas, no llegan al 30%. También se evidencia que la reducción alcanza un máximo, y permitir más sublotes no deriva en una mejora del makespan.

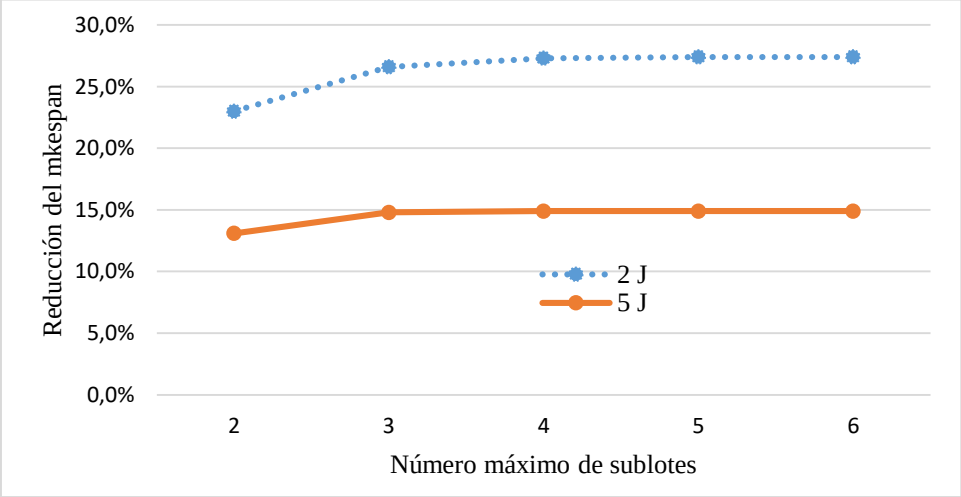


Figura 3-6. Reducción del makespan en relación al máximo número de sublotes admitidos para el caso de 5 máquinas.

En la Figura 3.6, se muestra el impacto del número de trabajos al implementar subloteo. A mayor número de trabajos menores reducciones se alcanzan. Para el caso de 2 trabajos las reducciones superan el 25%, mientras que para 5 trabajos las reducciones aproximan al 15%. También se comprueba que el número de sublotes tiene un impacto acotado. Incluso, cuando el número de trabajos es mayor, se alcanza con menor número

de sublotes la saturación de la reducción del makespan, con 3 sublotes prácticamente se obtuvo toda la mejora posible, mientras que cuando el número de trabajos es menor, para 4 sublotes siguen obteniéndose reducciones.

3.6. Conclusiones del capítulo

De lo expuesto en este capítulo puede concluirse que el subloteo es una metodología que permite reducir significativamente el makespan de un problema NPFS y PFS. También es necesario aclarar que las diferencias del makespan entre las soluciones NPFS y PFS fueron muy estrechas y en pocas ocasiones.

Respecto a la implementación del subloteo se puede afirmar que el número de sublotes a admitir por producto es un aspecto de mucho impacto en el desempeño del sistema. A mayor número de sublotes posibles por producto, mejor valor del makespan se obtiene tanto para NPFS como para PFS. Sin embargo, el número de sublotes tiene una cota, por sobre la cual la solución no reporta mayores beneficios. Esta saturación en la mejora del makespan respecto a los sublotes se comprobó en todas las instancias evaluadas. El número de máquinas del sistema, también tiene influencia sobre la mejora esperada al implementar subloteo, siendo mayores los beneficios que reporta aplicar subloteo en sistemas con mayor número de máquinas. Por otro lado, con el número de trabajos sucede a la inversa. A mayor cantidad de trabajos menor impacto tiene el subloteo.

Comparando los resultados para NPFS y PFS, las únicas diferencias observadas se dan cuando el tamaño de la instancia crece, algo que también condice con la literatura de la temática NPFS sin subloteo (NPFS suele dominar instancias grandes). Por otro lado, en los casos en que el makespan difirió entre los enfoques, también lo hizo el número de sublotes que utilizados para la solución óptima. Esto permitiría entrever que la estructura de la solución puede llegar a ser distinta. Por otro lado, el costo computacional de las soluciones NPFS, pudo observarse que es mucho mayor que el de las soluciones PFS. Así como las soluciones con alto número de sublotes posibles.

4. CAPÍTULO 4: ESTUDIO DE CAMINOS CRÍTICOS DE PROBLEMAS NPFS Y PFS

Estudio de caminos críticos de problemas NPFS y PFS

4.1. Estudio de caminos críticos

A la luz de lo presentado en las conclusiones del capítulo 2 (estado del arte) es evidente la falta de herramientas para un manejo eficiente del problema NPFS, a fin de obtener un cálculo eficiente y mejorar la capacidad de orientación de los algoritmos de búsqueda de soluciones óptimas. En la implementación de las técnicas de subloteo pudo observarse que a modo general el comportamiento de los sistemas PFS y NPFS sigue siendo similar en términos del makespan. Sin embargo, pudo detectarse que en el caso en que NPFS obtiene mejor makespan que el PFS, la estructura de la solución es distinta, ya que el tamaño y el número de sublotes fue distinto.

Por lo tanto, en el presente capítulo se estudia la estructura de soluciones PFS y NPFS. Para ello, fue necesario estudiar los secuenciamientos de trabajos, y cuáles de las operaciones a realizar de los trabajos tenía mayor impacto en el makespan. Para desarrollar este estudio hubo que diseñar y desarrollar algunas herramientas que permitieron estudiar de modo genérico y exhaustivo las soluciones PFS y NPFS. Dada la complejidad del problema abordado y la novedad de las herramientas enfocamos el trabajo en el caso de un flow shop de 2 trabajos. En las siguientes secciones presentamos el desarrollo, en la Sección 4.2 introducimos algunas definiciones y un caso tipo, luego en la Sección 4.3 presentamos las proposiciones y propiedades desarrolladas para el problema. En la Sección 4.4 se presenta el diseño de los experimentos no-paramétricos y los resultados obtenidos que sustentan las proposiciones, luego en la Sección 4.5 se propone una serie de experimentaciones paramétricos que demuestran algunas inferencias de los estudios no-paramétricos y finalmente las conclusiones a las que hemos llegado en la Sección 4.6.

4.2. Definiciones y ejemplo base

Antes de pasar a la presentación de las proposiciones sobre la estructura de las soluciones NPFS y PFS, presentamos algunas definiciones y ejemplos que serán fundamento para el resto del desarrollo.

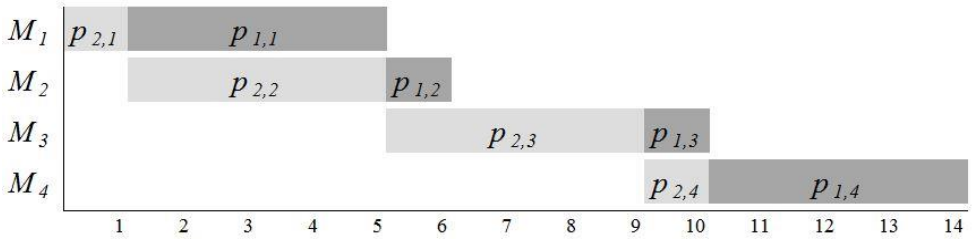
Primero se presenta un ejemplo numérico de 4 máquinas y 2 trabajos, el cual representa la instancia más chica para la cual PFS deja de asegurar la solución óptima. Los datos del ejemplo se presentan en la Tabla 4.1. Este ejemplo se denominará Ejemplo

1, y será retomado a lo largo del texto para poder ilustrar algunas de las proposiciones y resultados presentados en el capítulo.

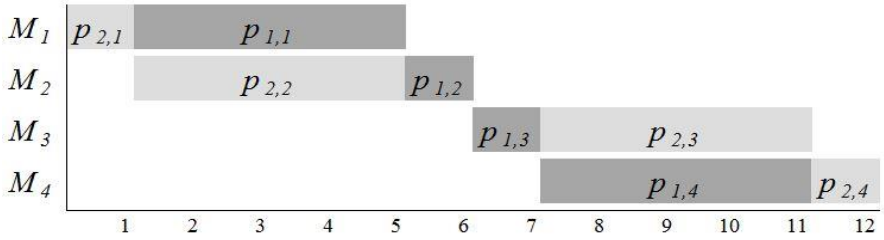
Tabla 4.1. Tiempos de procesamiento del Ejemplo 1.

	M_1	M_2	M_3	M_4
J_1	4	1	1	4
J_2	1	4	4	1

Las soluciones PFS y NPFS óptimas para el Ejemplo 1, están presentadas en la Figura 4.1.a y b. La mejor solución PFS tiene un makespan (14) peor que la mejor solución NPFS (12).



a. Representación de Gantt de la mejor solución PFS para Ejemplo 1.



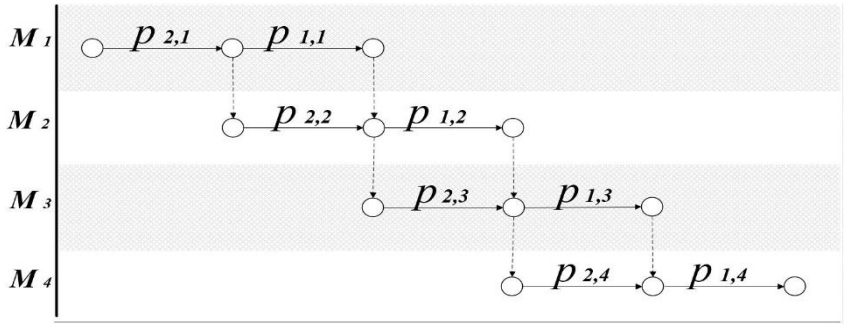
b. Representación de Gantt de la mejor solución NPFS para Ejemplo 1.

Figura 4-1. Representación de Gantt de las soluciones PFS y NPFS para el Ejemplo 1

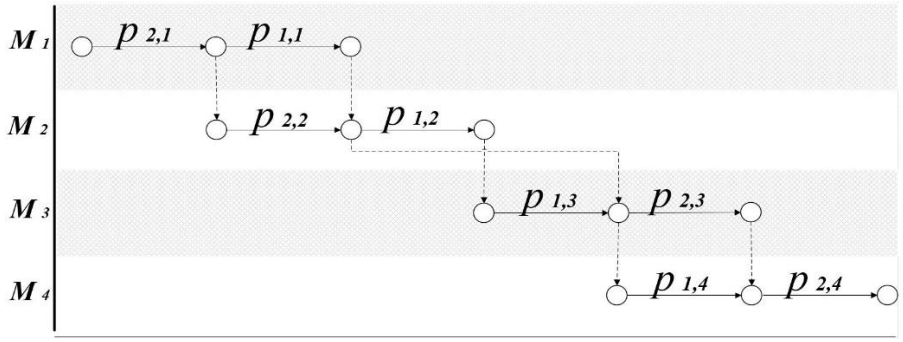
Definición 1 (Representación en grafo). Un grafo $G^{\sigma}(A, V, W)$ es una representación en grafo de un ordenamiento factible σ del problema $F||\mathcal{C}_{max}$ si el conjunto de arcos A representa las relaciones posibles entre las operaciones de los trabajos en σ, el conjunto de vértices V son las operaciones de los trabajos, mientras que el conjunto de ponderaciones W son los tiempos de procesamiento.

La Figura 4.2 ilustra la representación en grafo de las soluciones óptimas PFS y NPFS presentadas en la Figura 4.1. Es válido aclarar que la ponderación de los arcos con orientación vertical es cero (líneas punteadas). Dichos arcos representan las condiciones de precedencia, mientras que los arcos horizontales representan los tiempos de

procesamiento (líneas sólidas). También es de interés consignar que, para un mismo ordenamiento de los trabajos, distintos caminos son posibles, mientras que para un ordenamiento de trabajos distintos, debe modificarse la representación en grafo.



a. Representación en grafo de una solución PFS.

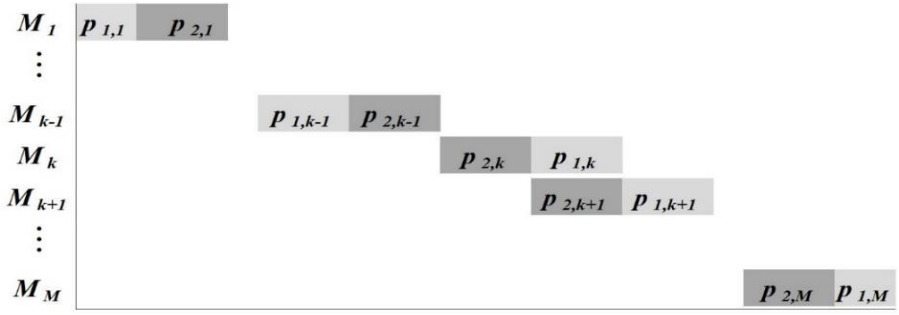


b. Representación en grafo de una solución NPFS.

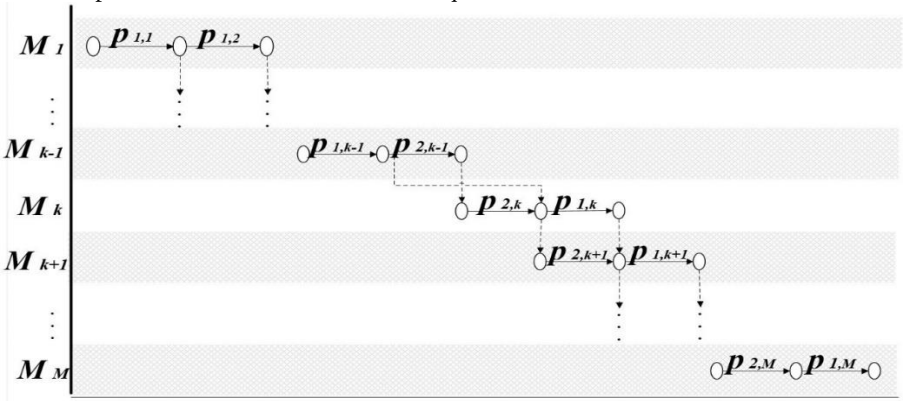
Figura 4-2. Representación en grafo de las soluciones PFS y NPFS para el caso de dos trabajos n=2 y cuatro máquinas m=4 mostrado en la Figura 4.1.

Definición 2 (máquina de mutación). Supongamos que σ es un ordenamiento factible para el problema $F||\mathcal{C}_{max}$. La máquina M_k es la máquina de mutación cuando el ordenamiento de los trabajos de la máquina precedente es invertido.

La Figura 4.3 presenta esta definición, donde M_k es la máquina de mutación. Claramente, la cardinalidad del conjunto de máquinas de mutación, $|S|$, es 0 para cualquier problema $F|prmu|\mathcal{C}_{max}$.



a. Representación de Gantt de una máquina de mutación



b. Representación en grafo de una máquina de mutación

Figura 4-3. Representación en diagrama de Gantt y en grafo para el caso de $n = 2$ trabajos, m máquinas y $|S| = 1$, definiendo a M_k como la máquina de mutación.

Definición 3 (dominancia permutativa). Suponga un ordenamiento factible σ para el problema $F||C_{max}$, con $|S| > 0$ y $F(\sigma)$ es el makespan correspondiente. Considere que S^p es el conjunto de todos los ordenamientos factibles del problema $F|prmu|C_{max}$, es decir permutativos. Un ordenamiento σ es dominado por la permutación si $F(\sigma) \geq min_{\pi \in S^p} F(\pi)$.

4.3. Propiedades estructurales y de dominancia

En esta sección se explorarán propiedades de los problemas NPFS y PFS, asumiendo que el número de trabajos es $n = 2$.

Una noción fundamental sobre la que construiremos nuestros argumentos es la de *camino crítico* en un ordenamiento, recordando que, para un mismo ordenamiento, existen varios caminos que lo describen siendo el crítico el mayor de todos. Entonces, tenemos:

Proposición 1. *El camino crítico de un ordenamiento factible σ, dado como una representación en grafos G^{σ} (A, V, W), resulta en el makespan del problema $F||C_{max}$.*

Prueba. La demostración resulta directamente de la definición de la representación en grafos G^{σ} *(A, V, W)* y considerar el camino más largo. □

Proposición 2. *En un ordenamiento óptimo σ, las máquinas M_1, M_2 y M_k no son máquinas de mutación.*

Prueba. La primera máquina M_1 no es máquina de mutación por definición. La demostración para las otras máquinas responde a los resultados de Conway et al. (1967), que establece que una solución óptima del problema $F||\mathcal{C}_{max}$ tiene el mismo ordenamiento para las dos primeras máquinas y en las dos últimas. □

A fin de dar una presentación más clara de la demostración de esta proposición muy reconocida, tomaremos una demostración sencilla y directa de Gharbi et al. (2014), la cual se basa en el concepto de que los problemas de secuenciamiento de flow shop con makespan como objetivo son simétricos, ya sea PFS o NPFS. Esto significa que el makespan óptimo obtenido por un análisis hacia adelante es el mismo que el obtenido por un análisis hacia atrás (es decir invirtiendo la ruta de los trabajos, empezando por la última operación realizada en la máquina M_M, después la realizada en M_{M-1} y así sucesivamente).

Para ejemplificar esto, considere cualquier solución en donde los ordenamientos de la primer y segunda máquina sean distintos. Considere dos trabajos *a* y *b*, que estén en orden distinto en la primera máquina respecto a la segunda. Estos trabajos pueden invertir su ordenamiento en la primera máquina sin incrementar el comienzo del procesamiento de cualquiera de los dos en la segunda máquina, y por consiguiente tampoco se modificará su tiempo de finalización. Inductivamente, podemos repetir este cambio de a pares hasta que los ordenamientos de la primera máquina sean iguales a los de la segunda. Una consecuencia inmediata de esta Proposición 2, es que $F2||\mathcal{C}_{max}$ y $F2|prmu|\mathcal{C}_{max}$ son equivalentes. Esto es en el caso de un sistema flow shop con dos máquinas, existe un ordenamiento óptimo que es permutativo. Consecuentemente, $F2||\mathcal{C}_{max}$ se puede resolver en tiempo polinomial utilizando la regla de Johnson (Johnson 1954). Más aun, dada la simetría del problema, se puede mostrar que para una solución óptima, las últimas dos máquinas no son máquinas de mutación. Entonces, usando la Proposición 2, se puede afirmar que $F3||\mathcal{C}_{max}$ y $F3|prmu|\mathcal{C}_{max}$ son problemas equivalentes.

El Ejemplo 1 representa el caso más pequeño para el cual $F||\mathcal{C}_{max}$ y $F|prmu|\mathcal{C}_{max}$ dejan de ser equivalentes, mostrando que NPFS da un mejor makespan.

Proposición 3. *Considere S el conjunto de las máquinas de mutación. Su cardinalidad es* $|S| \leq m-3$.

Prueba. Es una consecuencia inmediata de la Proposición 2. □

Para ilustrar esta propiedad, retomamos el Ejemplo 1, donde el sistema flow shop tiene cuatro máquinas. Según la Proposición 3, el conjunto *S* consiste como máximo de un elemento, y de acuerdo a la Proposición 2, puede concluirse que la máquina de mutación es M_3, como está ilustrado en las Figuras 4.1.b y 4.2.b.

Proposición 4. *Considere S como el conjunto de las máquinas de mutación. La cardinalidad del camino crítico representado por* $G^{\sigma}(A, V, W)$ *es como máximo* $m+1+|S|$.

Prueba. Considere dos casos. Un ordenamiento factible σ con $|S| = 0$, el camino crítico en la representación gráfica $G^{\sigma}(A, V, W)$ se define por $m + 1$ arcos horizontales: dos arcos corresponden a los tiempos de procesamientos de la primer y última operación en σ, mientras que el resto $m - 1$ de los arcos representan los tiempos de procesamientos de cada operación definida por el par de máquinas M_k y M_{k+1}, con $k \in \{1, \ldots, m - 1\}$.

Para un ordenamiento factible σ con $|S| \geq 1$, la prueba es por inducción. Para un ordenamiento factible σ con $|S| = 1$ considere la representación $G^{\sigma}(A, V, W)$ y sepárela en dos sub-grafos tales que la última operación de uno de los sub-grafos y la primera operación del otro sub-grafo correspondan al mismo trabajo. La máquina de mutación, M_k, es la que divide a los dos sub-grafos. Entonces, los caminos críticos en el primer y segundo sub-grafo tienen $k + 1$ y $m - k + 1$ operaciones, respectivamente. Por lo tanto, el camino crítico en $G^{\sigma}(A, V, W)$ tiene $m + 2 = m + 1 + |S|$ operaciones.

Ahora considere un ordenamiento factible σ con $|S| = \ell < m - 3$ y su representación gráfica $G^{\sigma}(A, V, W)$ y asuma que la afirmación es cierta para cualquier ordenamiento con $\ell - 1$ máquinas de mutación. Separe el grafo en dos sub-grafos, tales que la última operación de uno de los sub-grafos pertenezca al mismo trabajo que la primera operación del siguiente sub-grafo. A su vez, separe el grafo de forma tal que el segundo sub-grafo no contenga máquinas de mutación. Podemos asumir, sin pérdida de generalidad, que la *k-ésima* máquina, M_k, define la separación de los dos sub-grafos. Dado que la clase de las máquinas de mutación remanentes tiene cardinalidad $\ell - 1$, el camino crítico en el primer sub-grafo tiene $k + 1 + \ell - 1$ arcos mientras que el camino crítico del segundo sub-grafo tiene $m - k + 1$ (recuerde que este sub-grafo no contiene máquinas de mutación). Por lo tanto, la solución del grafo $G^{\sigma}(A, V, W)$ tiene $m + \ell + 1$ arcos. □

La Proposición 4 resume algunos de los principales aportes de este capítulo. El primer concepto importante que incluye la Proposición 4 es que en los sistemas PFS los caminos críticos tienen una longitud de $m + 1$ operaciones ($|S| = 0$). Esta propiedad, puede verificarse en la Figura 4.2.a. Además, si la Figura 4.2.a (representación genérica) se la compara con la Figura 4.1.a (Ejemplo 1), algunas propiedades nuevas pueden ser encontradas:

Lema 1. Cada camino crítico en un PFS de dos trabajos es tal que:

a) Siempre comienza con la primera operación del primer trabajo del ordenamiento
b) Siempre finaliza con la última operación del último trabajo del ordenamiento
c) Incluye secuencialmente operaciones de uno de los trabajos, hasta cierta máquina en donde comienza a incluir operaciones del otro trabajo
d) En la máquina mencionada, incluye operaciones de los dos trabajos.
e) En el resto de las máquinas incluye operaciones de uno solo de los trabajos.

Estas propiedades pueden ser resumidas en la siguiente expresión 4.1:

$$\sum_{i=1}^{i=m'} p_{1,i} + \sum_{i'=m'}^{M} p_{2,i'} \quad , \forall m' \in M \tag{4.1}$$

La expresión 4.1 representa la longitud de un camino crítico para un ordenamiento PFS de dos trabajos, en este caso J_1 y J_2. La expresión suma los tiempos de procesamiento de las operaciones del primer trabajo, en este caso J_1, hasta la máquina $M_{m'}$ donde comienza a sumar los tiempos de procesamientos del segundo trabajo, en este caso J_2. Esta formulación contempla $m + 1$ tiempos de procesamiento, lo que se condice con el número de operaciones esperado para el camino crítico de un PFS indicado por Nagarajan y Sviridenko (2009), $m + n - 1$.

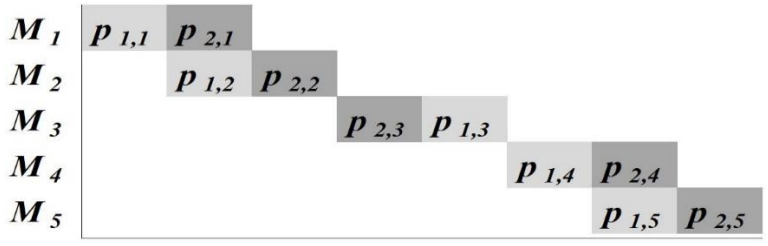
La expresión 4.1 describe la estructura del camino crítico para un ordenamiento PFS dado. Para obtener el conjunto completo de los posibles caminos críticos PFS para un ordenamiento dado, basta con repetir la formulación 4.1 para todos los valores posible de *m'*. Para generar el conjunto completo de caminos críticos para un Sistema PFS de dos trabajos y *m*-máquinas, debe repetirse el proceso para ordenamiento posible.

Por otro lado, la Proposición 4 indica que el número de operaciones incluidas en un camino crítico NPFS es m +1+ $|S|$ y la demostración se basa en la descomposición del grafo $G^{\sigma}(A, V, W)$ en sub-grafos. Esta descomposición está graficada en la Figura 4.2.b, donde hasta la máquina M_2 el ordenamiento es $J_1 - J_2$, y después pasa a ser $J_2 - J_1$. El primer sub-grafo incluye M_1 y M_2, y el segundo sub-grafo incluye M_3 y M_4. La primera operación del segundo sub-grafo corresponde al trabajo J_2, lo mismo que la última operación del primer sub-grado. Entonces, es posible asegurar que la operación de J_2 en la máquina M_3, $O_{2,3}$, no comenzará hasta que no termine la operación del trabajo J_2 en la máquina M_2, $O_{2,2}$. Consecuentemente, la longitud del camino crítico NPFS puede ser calculado como la suma de los caminos críticos de los sub-grafos. Además, ya que los dos sub-grafos representan dos sistemas PFS consecutivos, puede afirmarse que, cuando $|S|=1$, el makespan global del ordenamiento NPFS es la suma de los makespan de dos sistemas PFS consecutivos. La siguiente expresión 4.2, resume estos conceptos representando la longitud del camino crítico para un ordenamiento NPFS de cuatro máquinas y una máquina de mutación.

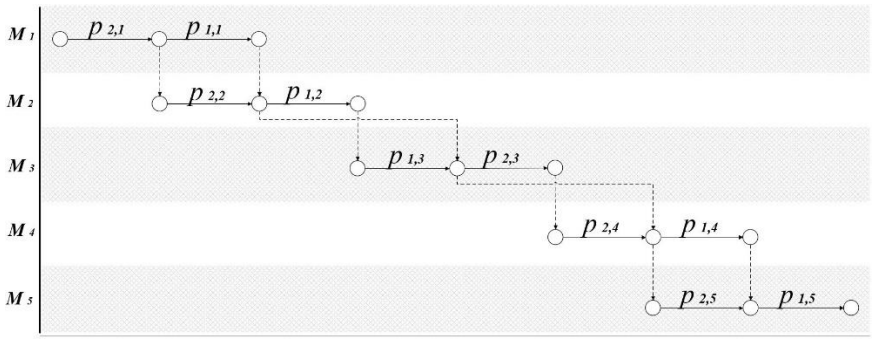
$$\left(\sum_{i=1}^{i=m'} p_{1,i} + \sum_{i'=m'}^{i'=k-1} p_{2,i'}\right) + \left(\sum_{h=k}^{h=m''} p_{2,h} + \sum_{h'=m''}^{M} p_{1,h'}\right), \forall m', m'' \in M | m' < m'' \qquad (4.2)$$

La expresión 4.2 se obtiene como la concatenación de dos sistemas PFS, donde M_k representa la máquina de mutación. Una vez más, para calcular la longitud de cada camino crítico posible para ese ordenamiento dado, basta con repetir la expresión 4.2 para cada valor posible de *m'*, *m''* y M_k (si $|S| > 1$).

Para extender esta caracterización de los caminos críticos NPFS para sistemas más grandes presentamos la figura 4.4 en donde una instancia de cinco máquinas está representada.



a) Gráfico de Gantt de un ordenamiento NPFS de 5 máquinas



b) Representación en Grafo del ordenamiento mostrado en 4.4.a

Figura 4-4. Ilustraciones de la solución de un problema NPFS con dos trabajos $n = 2$, cinco máquinas $m = 5$, y dos máquinas de mutación $S = \{M_3, M_4\}$.

La Figura 4.4 presenta un caso de cinco máquinas ($m = 5$) con dos máquinas de mutación, $S = \{M_3, M_4\}$, cantidad de mutaciones que respeta la Proposición 3. Centrando la atención en la Figura 4.4.b puede verse que el grafo puede descomponerse en tres sub-grafos: el primero incluye las máquinas M_1 y M_2, el segundo M_3, y el tercero, M_4 y M_5. Nuevamente, cada sub-grafo representa una solución a un problema PFS, incluso para el sub-grafo de una sola máquina, M_3. Para cada caso la expresión 4.1 es válida. Por lo que la longitud del camino crítico de la Figura 4.4 satisface la siguiente expresión:

$$\left(\sum_{i=1}^{i=m\prime} p_{1,i} + \sum_{i'=m\prime}^{i'=k^*-1} p_{2,i\prime}\right) + \left(\sum_{h=k^*}^{h=m''} p_{2,h} + \sum_{h'=m''}^{h'=k^{**}-1} p_{1,h\prime}\right) + \left(\sum_{g=k^{**}}^{g=m'''} p_{1,g} + + \sum_{g'=m'''}^{M} p_{2,g\prime}\right), \quad (4.\ 3)$$

$$\forall m', m'', m''' \in M | m' < m'' < m'''$$

Comparando las expresiones 4.2 y 4.3 puede verse que los ordenamientos NPFS pueden representarse como una concatenación de ordenamientos PFS separados por las máquinas de mutación. Esto allana el camino para presentar un procedimiento de descomposición.

Procedimiento de Descomposición. *Dado un ordenamiento factible σ del problema $F||C_{max}$ con dos trabajos, tal que la cardinalidad del conjunto de máquinas de mutación es $|S| \geq 1$, el camino crítico puede descomponerse en $|S|$ ordenamientos $F|prmu|C_{max}$ más pequeños, y el makespan de σ es la suma de los makespans de esos ordenamientos.*

Este procedimiento conduce hacia algunas conclusiones interesantes. Cada vez que haya una máquina de mutación, el procedimiento de descomposición es aplicable.

Entonces, las expresiones 4.1, 4.2 y 4.3 pueden ser concebidas como el núcleo de un *algoritmo generador de caminos críticos para flow shop*.

Proposición 5. *Sea m el número de máquinas, $m \geq 3$. Si m es impar, el ordenamiento con $|S| = m - 3$ es dominado por el permutativo.*

Prueba. Considere un ordenamiento factible σ con $|S| = m - 3$, y dado que se asumen dos trabajos, el universo total de operaciones es $2m$. Según la Proposición 4, en el camino crítico de $G^{\sigma}(A, V, W)$, el número de operaciones incluidas es $m + 1 + m - 3$ (i.e. $2m - 2$). Esto significa que $G^{\sigma}(A, V, W)$ tiene todas las operaciones posibles excepto dos de ellas. Por otro lado, considere un ordenamiento permutativo factible π, de nuevo según la Proposición 4, el camino crítico en $G^{\pi}(A, V, W)$ tiene $m + 1$ operaciones (el conjunto de máquinas de mutación está vacío). Si se comprueba que las dos operaciones no incluidas en el camino crítico de $G^{\sigma}(A, V, W)$ tampoco lo están en el camino crítico de $G^{\pi}(A, V, W)$, entonces sería posible afirmar que el primero incluye todas las actividades del segundo más algunas otras operaciones. Por lo que el makespan del ordenamiento σ será mayor que el del ordenamiento permutativo π.

Para comprobar esto se asume, sin pérdida de generalidad, que la primera operación del ordenamiento σ y del ordenamiento π son las mismas, y como el número de máquinas de mutación, $m - 3$, es par (por lo que el último sub-grafo en la representación en grafo tiene el mismo ordenamiento que el primer sub-grafo), la última operación tiene que ser la misma para los dos ordenamientos. Además, se tiene que los ordenamientos coinciden en las primeras dos y en las últimas dos máquinas según la Proposición 3. Esto implica que, dado que la primer máquina realiza la primer operación en ambos ordenamientos (NPFS y PFS), el camino crítico de $G^{\sigma}(A, V, W)$ tiene las mismas dos primeras operaciones que el camino crítico de $G^{\pi}(A, V, W)$ más otra operación, pero no todas de las cuatro posibles operaciones de las primeras dos máquinas. Entonces, la operación que no está incluida en el camino crítico de $G^{\sigma}(A, V, W)$ (el cual por el argumento previo incluye tres de las cuatro operaciones) tampoco está incluida en el camino crítico de $G^{\pi}(A, V, W)$. Un análisis similar puede hacerse para las últimas dos máquinas. Por lo que puede afirmarse que el camino crítico de $G^{\sigma}(A, V, W)$ incluye las mismas operaciones que el camino crítico de $G^{\pi}(A, V, W)$, mientras que las operaciones descartadas por el camino crítico de $G^{\sigma}(A, V, W)$ también son descartadas por el camino

crítico de $G^{\pi}(A, V, W)$. Finalmente, el camino crítico de $G^{\pi}(A, V, W)$ siempre tendrá un mejor makespan. □

Para ilustrar esta propiedad, se presenta en la Figura 4.5 dos ejemplos de caminos críticos para una instancia de cinco máquinas: uno de un problema PFS y otro NPFS. El camino crítico PFS es el más corto (6 operaciones) mientras que el NPFS es el más largo (8 operaciones). El caso NPFS representa un ordenamiento con $|S| = 2$, (i.e. $m - 3$), y comienza con la misma operación que el camino crítico PFS. Más aún, dado que $|S|$ es un número par, los caminos PFS y NPFS terminan con la misma operación. Una simple inspección muestra que el camino crítico NPFS tiene un makespan mayor que el del PFS, porque incluye todas las operaciones del camino crítico PFS más algunas operaciones extras, básicamente $O_{2,2}$ y $O_{4,1}$, lo que incorpora sus respectivos tiempos de procesamiento. Por lo tanto, el ordenamiento NPFS es dominado por el ordenamiento PFS.

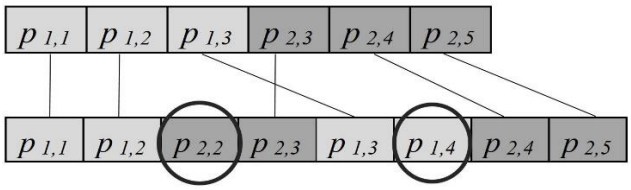


Figura 4-5. Dominancia de caminos críticos.
Aclaración: el camino crítico de arriba corresponde a un ordenamiento PFS, mientras que el de abajo a un ordenamiento NPFS, ambos representan una instancia de cinco máquinas.

Proposición 6. *Sea m el número de máquinas. Considere un ordenamiento no-permutativo σ con $|S| = m - 3$ y un ordenamiento permutativo π. Si m es par, el camino crítico de $G^{\pi}(A, V, W)$ tiene como máximo una operación que difiere del camino crítico de $G^{\sigma}(A, V, W)$.*

Prueba. La demostración es similar a la de la Proposición 5, ya que se parte de la misma situación inicial, sólo dos operaciones no están incluidas en el camino crítico no-permutativo. Considere un ordenamiento factible σ con $|S| = m - 3$, donde el número total de operaciones es $2m$. Según la Proposición 4, el camino crítico de $G^{\sigma}(A, V, W)$ tiene $2m - 2$ operaciones mientras que el de $G^{\pi}(A, V, W)$ incluye $m + 1$. Sin pérdida de generalidad, asumimos que la primera operación del ordenamiento σ y de un ordenamiento permutativo π es la misma, y por la Proposición 3 las primeras dos

máquinas tienen el mismo ordenamiento de los trabajos. Sin embargo, como *m* es par, $|S| = m - 3$ es impar, por lo que, los ordenamientos σ y π no tienen los mismos ordenamientos para las últimas dos máquinas. En éstas últimas, el camino crítico de $G^{\sigma}(A, V, W)$ incluye tres de las cuatro posibles operaciones. La operación restante puede que sea o no incluida en el camino crítico permutativo ya que los ordenamientos para esas máquinas no coinciden. Por lo que el camino crítico $G^{\pi}(A, V, W)$ puede incluir como máximo una operación que no esté incluida en el camino crítico de $G^{\sigma}(A, V, W)$. Para el resto de las máquinas el camino crítico de $G^{\sigma}(A, V, W)$ incluye todas las operaciones posibles. □

Considere nuevamente la Figura 4.5, pero con la diferencia de que ahora *m* se lo asume un número par. Por lo tanto, las soluciones NPFS y PFS comienzan con el mismo ordenamiento, pero como $|S| = m - 3$ es un número impar, los ordenamientos en las últimas dos máquinas son distintos.

Proposition 7. *El número de ordenamientos distitnos que pueden ser óptimos para un problema* $F||C_{max}$ *son como máximo* $2^{\max\{m-3,1\}} + m$.

Prueba. La demostración sigue las Proposiciones 3 y 4. □

Proposición 8. *Sea m el número de máquinas,* $m \geq 4$. *Considere un ordenamiento no-permutativo* σ *con* $|S|=1$. *El camino crítico de la solución gráfica* $G^{\sigma}(A, V, W)$ *tiene como mucho* $\lceil (m+1)/2 \rceil$ *operaciones que no pertenecen al camino crítico de la solución gráfica permutativa.*

Prueba. Considere un ordenamiento factible σ con |S| = 1, donde el total de operaciones es 2*m*. De la proposición 4, el camino crítico de $G^{\sigma}(A, V, W)$ tiene $m + 2$ operaciones. Sea M_k la máquina de mutación, $3 \leq k \leq m - 1$. M_k divide $G^{\sigma}(A, V, W)$ en dos sub-grafos.

Ahora, analizaremos dos casos. En el primer caso, se asume que la primera operación en σ y en el ordenamiento permutativo π son la mismas, y que tienen el mismo ordenamiento de trabajos en las dos primeras máquinas, según la Proposición 2. Esto implica que el camino crítico del primer sub-grafo coincide. Entonces, de las $m + 2$ arcos de camino crítico de $G^{\sigma}(A, V, W)$, *k* serán los mismos que en el camino crítico $G^{\pi}(A, V, W)$. Por lo tanto, como máximo $m + 2 - k$ operaciones serán distintos.

Ahora considere el caso en que la última operación en σ y en el ordenamiento permutativo π´ son la misma, y que tienen el mismo ordenamiento en las últimas dos máquinas, según la Proposición 2. Esto implica que de los $m + 1 - k$ arcos en el camino crítico del segundo sub-grafo sean los mismos en los dos ordenamientos. Entonces, el camino crítico $G^{\sigma}(A, V, W)$ y $G^{\pi'}(A, V, W)$ difieran, como máximo, en $(m + 2) - (m - k + 1) = k + 1$ operaciones.

Luego, dado un camino crítico para $G^{\sigma}(A, V, W)$, podemos encontrar ordenamientos permutativos que coincidan en la primer o última operación, dando como máximo $k + 1$ o $m + 2 - k$ operaciones distintas. Eligiendo el min $\{ k + 1, m + 2 - k \}$para el número de operaciones distintas obtenemos una cota para el número de operaciones distintas posibles entre los dos problemas. El peor caso es cuando $k + 1 = m + 2 - k$, o equivalentemente, $k = (m + 1)/2$. Nótese que $3 \leq \lceil (m + 1)/2 \rceil \leq m - 1$ se cumple para $m \geq 4$ y que el camino crítico de la solución gráfica $G^{\sigma}(A, V, W)$ puede, en el peor de los casos, diferir en $\lceil (m + 1)/2 \rceil$ operaciones del camino crítico de la solución permutativa.

□

Proposición 9. *Sea σ un ordenamiento factible para $F||\mathcal{C}_{max}$ con $|S| > 0$. Sea π un ordenamiento factible para $F|prmu|\mathcal{C}_{max}$. Sea $\mathbb{P}$ el conjunto de operaciones del camino crítico de $G^{\pi}(A, V, W)$ no incluidas en el camino crítico de $G^{\sigma}(A, V, W)$. Sea $\mathbb{Q}$ el conjunto de operaciones incluidas en el camino crítico de $G^{\sigma}(A, V, W)$ ordenadas en orden creciente según sus tiempos de procesamientos. Sea $\mathbb{Q}_{|S|}$ el conjunto de las $|S|$ primeras operaciones de $\mathbb{Q}$. Si,*

$$\sum_{p_{j,i} \in \mathbb{P}} p_{j,i} \leq \sum_{p_{j,i} \in \mathbb{Q}_{|S|}} p_{j,i}$$

el ordenamiento σ es dominado por el permutativo.

Prueba. Considere un ordenamiento factible σ, con |S| > 0, y un ordenamiento permutativo π. De la Proposición 4, se sabe que el camino crítico de σ tiene |S| más operaciones que el camino crítico del ordenamiento π. Si se asume que el tiempo total de procesamiento de las operaciones en $\mathbb{P}$ es mayor que el de las operaciones en $\mathbb{Q}_{|S|}$ se tiene que el ordenamiento π domina al ordenamiento σ. Esto es así porque las operaciones en $\mathbb{P}$ (que no están incluidas en el camino crítico de $G^{\sigma}(A, V, W)$ pero sí lo están en el camino

crítico de $G^{\pi}(A, V, W)$) contribuirán en una cuantía menor a la longitud del camino crítico de $G^{\pi}(A, V, W)$, comparadas con la contribución de las operaciones de $\mathbb{Q}_{|S|}$ al camino crítico de $G^{\sigma}(A, V, W)$. □

Es de interés consignar que el conjunto $\mathbb{Q}_{|S|}$, en realidad, podría contener otras operaciones distintas de las de menor tiempo de procesamiento. Ya que el objetivo es comparar operaciones que no están presentes en ambos caminos críticos, por lo que $\mathbb{Q}_{|S|}$ podría incluir las operaciones del camino crítico de $G^{\sigma}(A, V, W)$ que no están incluidas en el camino crítico $G^{\pi}(A, V, W)$. Pero es claro que si la dominancia se da para el conjunto de los menores tiempos de procesamientos también se dará para otro conjunto de $\mathbb{Q}_{|S|}$ operaciones.

La Proposición 9, es una generalización de las Proposiciones 5 y 6, donde si el número de máquinas de mutación es par (Proposición 5), $|\mathbb{P}| = 0$ y si es impar (Proposición 6), $|\mathbb{P}| \leq 1$. Desde una perspectiva más amplia, la Proposición 9 es la base para desarrollar un procedimiento de acotamiento (*bounding*) en el número de ordenamientos factibles para un problema concreto en el que los tiempos de procesamientos son conocidos. Si el número de operaciones de $G^{\pi}(A, V, W)$ no incluidas en $G^{\sigma}(A, V, W)$ es conocido de antemano ($|\mathbb{P}|$), es posible definir una cota al número de máquinas de mutación ($|S|$). De esta forma se acotan el número de ordenamientos NPFS factibles a ser explorados.

Para ilustrar estas ideas, se retoma el Ejemplo 1, y se presenta la Figura 4.6, la cual describe los dos caminos críticos de las soluciones PFS y NPFS mostrados en la Figura 4.1.

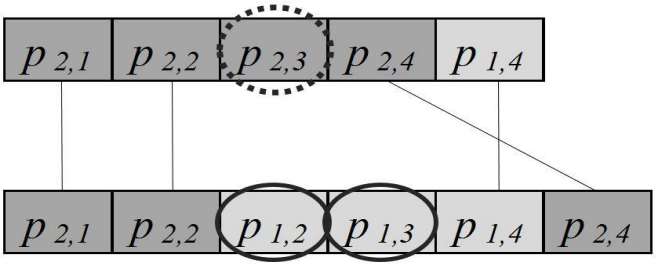


Figura 4-6. Caminos críticos del Ejemplo 1, el camino crítico superior corresponde al ordenamiento PFS (Figura 4.1.a), y de abajo al ordenamiento NPFS (Figura 4.1.b).

El ordenamiento óptimo del Ejemplo 1 está dado por la solución NPFS. De la Figura 4.6, y de la Proposición 9, se puede establecer que el conjunto $\mathbb{P}$ consiste de la operación $O_{2,3}$, y el conjunto $\mathbb{Q}_{|S|}$ de las operaciones $O_{1,2}$ y $O_{1,3}$. La suma de los tiempos

de procesamiento de las operaciones de $\mathbb{P}$ es 4, mientras que la suma de los tiempos de procesamiento de las operaciones de $\mathbb{Q}_{|S|}$ es 2. Esto explica por qué el ordenamiento NPFS tiene un mejor makespan que el ordenamiento PFS, a pesar de tener una operación más en su camino crítico. En el Ejemplo 1, la Proposición 9 no se cumple por lo que el ordenamiento NPFS no está dominado por el permutativo.

Por otra parte, es posible inferir el porqué de las diferencias entre los makespans NPFS y PFS obtenidos en los resultados empíricos de Liao et al (2006) son tan pequeñas, (menores al 3%). La posible razón es que el número máximo de máquinas de mutación que puedan generar ordenamientos candidatos a óptimos esté acotado, en un caso concreto, por los tiempos de procesamientos. En este sentido, Tando et al (1991), habían llegado a una apreciación similar: a mayor dispersión en los tiempos de procesamiento, mayores beneficios reportaba NPFS. Estas ideas se desarrollarán con mayor profundidad luego de los resultados experimentales propios.

4.4. Experimentación no-paramétrica

Los resultados teóricos de las secciones previas pueden ser incorporados en un algoritmo de búsqueda exhaustiva (ExA). Este algoritmo genera el conjunto completo de caminos críticos posibles para problemas PFS y NPFS dados. Luego, compara los caminos críticos y calcula las operaciones que están incluidas en el set $\mathbb{P}$ (de la Proposición 9). La generación de los caminos críticos para un ordenamiento dado está basada en las formulaciones iterativas 4.1 y en el método de descomposición. El pseudocódigo del algoritmo ExA se lo muestra a continuación, en la Tabla 4.2. Básicamente el algoritmo trabaja de la siguiente forma, dados un *m* y *S*, busca un ordenamiento NPFS σ factible. Luego genera el conjunto de posibles caminos críticos para ese ordenamiento σ, a los que los compara con los caminos críticos correspondientes a ordenamientos PFS. Es de interés consignar que para un *m* y *S* dados existen varios ordenamientos NPFS factibles, dependiendo de la ubicación de las máquinas de mutación.

Tabla 4.2. Pseudocódigo del algoritmo exhaustivo ExA.

Algoritmo: Algoritmo Exhaustivo (ExA)

Datos: Instancias con 2 trabajos y m máquinas.

Resultado: Número máximo de operaciones diferentes incluidas en el camino crítico del grafo permutativo

Paso 1: Generar todos los ordenamientos factibles para $F||\mathcal{C}_{max}$ y $F|prmu|\mathcal{C}_{max}$; incluirlos en los conjuntos *PFS* y *NPFS*, respectivamente.

Paso 2: While $|NPFS| > 0$ **do**

1. Seleccionar un ordenamiento $\sigma \in NPFS$.
2. Seleccionar dos ordenamientos $\{\pi,\pi'\} \in PFS$ que correspondan para el número de máquinas del ordenamiento σ.
3. Generar los conjuntos completos de caminos críticos para los ordenamientos σ, π y π'.
4. Comparar los caminos críticos; y registras el conjunto $\mathbb{P}$ y la cardinalidad $|S|$.
5. Eliminar σ del conjunto *NPFS*.

Fin while.

4.4.1. Experimentos

A fin de evaluar los resultados teóricos y el algoritmo exhaustivo es necesario definir el tamaño de las instancias a ser evaluadas. Los papers más citados que proponen instancias experimentales para evaluar algoritmos en problemas flow shop son Taillard (1993) y Demirkol et al. (1998), quienes establecen en 20 el número máximo de máquinas. Siguiendo estos artículos, se ejecutará el ExA en el rango entero de [4,20] máquinas para 2 trabajos, generando de esta forma más de 500.000 ordenamientos, lo que involucra más de 200 millones de comparaciones de caminos críticos. El algoritmo fue programado en el lenguaje abierto R.

4.4.2. Resultados

Los resultados son presentados en la Tabla 4.3, donde las instancias están caracterizadas por el número de máquinas m y la cardinalidad del conjunto $|S|$. Dado que el rango de m es [4:20], el rango de $|S|$ es [1:17]. En las celdas de la Tabla 4.3 se registró el máximo valor del conjunto $\mathbb{P}$ para la correspondiente instancia. Esto es así, ya que existen varios $\mathbb{P}$ que corresponden a esa instancia. Para ejemplificar considere el caso de $m = 8$ y $|S| = 1$. Para esa instancia existen varios ordenamientos NPFS posibles, y por consiguiente ExA reportará varios conjuntos $\mathbb{P}$. Por lo que el máximo de todos ellos es el que está registrado en la Tabla 4.2.

La Tabla 4.3 confirma la Proposición 5, la cual de forma intuitiva indica que, si el número de máquinas es impar, el número máximo de máquinas de mutación es par (respetando la Proposición 3), y el ordenamiento NPFS con esa cantidad de máquinas de mutación es dominado por los ordenamientos PFS. Por otra parte, la Proposición 6 indica que si el número de máquinas es par, entonces el número máximo de máquinas de mutación es impar, y los caminos críticos PFS difieren como máximo en una operación de los caminos críticos NPFS.

Como consideración general, el número de operaciones del camino crítico en $F|prmu|\mathcal{C}_{max}$ no incluidas en los caminos críticos de la versión $F||\mathcal{C}_{max}$ del problema (i.e.$|\mathbb{P}|$), es decreciente en $|S|$. Esto puede explicarse retomando la Proposición 4, la cual indica que el camino crítico NPFS tiene $|S|$ más operaciones que el camino crítico PFS. Por lo tanto, a mayor $|S|$, mayor la cantidad de operaciones incluidas en el camino crítico NPFS. Esto incrementa las posibilidades de que las operaciones del camino crítico PFS se encuentren incluidas en el camino crítico NPFS.

Tabla 4.3. Resultados experimentales no-paramétricos obtenidos con algoritmo ExA.

Nota: Cada celda está definida por el número de máquinas y el número de máquinas de mutación, |S|, de la instancia. Cada entrada muestra el máximo número de operaciones de camino crítico de F|prmu|$\mathcal{C}$max, que no están incluidas en el camino crítico de F| |$\mathcal{C}$max.

	|S|															
m	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	***11***	***12***	***13***	***14***	***15***	***16***	***17***
4	1																
5	1	0															
6	1	1	1														
7	2	1	1	0													
8	2	1	1	1	1												
9	2	2	2	1	1	0											
10	3	2	2	1	1	1	1										
11	3	2	2	2	2	1	1	0									
12	3	3	3	2	2	1	1	1	1								
13	4	3	3	2	2	2	2	1	1	0							
14	4	3	3	3	3	2	2	1	1	1	1						
15	4	4	4	3	3	2	2	2	2	1	1	0					
16	5	4	4	3	3	3	3	2	2	1	1	1	1				
17	5	4	4	4	4	3	3	2	2	2	2	1	1	0			
18	5	5	5	4	4	3	3	3	3	2	2	1	1	1	1		
19	6	5	5	4	4	4	4	3	3	2	2	2	2	1	1	0	
20	6	5	5	5	5	4	4	3	3	3	3	2	2	1	1	1	1

Para ilustrar los conceptos del método de acotamiento propuesto en la Proposición 9, consideraremos resultados de la Tabla 4.3. Por ejemplo, considere un caso de 10

máquinas y 2 trabajos, para el cual los tiempos de procesamientos son conocidos. Considere los ordenamientos NPFS con $|S| \geq 2$. La cota a la cantidad de estos ordenamientos candidatos al óptimos puede obtenerse analizando el siguiente peor caso. De la Tabla 4.3, se sabe que para $|S| \geq 2$, $| \mathbb{P} | \leq 2$. Luego, es necesario ordenar las operaciones en orden creciente según sus tiempos de procesamiento. Una vez hecho esto, se suman las dos últimas operaciones del orden ($|\mathbb{P}| \leq 2$). Para luego comenzar a sumar de a una las operaciones desde el principio del orden, las de menor tiempos de procesamiento, hasta que se cumpla la condición de la Proposición 9. El número mínimo de operaciones para las que se cumple la condición es el número máximo para la cardinalidad de $|S|$. Analizar ordenamientos NPFS con mayor número de máquinas de mutación no será necesario, ya que para ese caso están dominados por los ordenamientos PFS. A la hora de resolver problemas concretos el número de ordenamientos factibles a explorar se reduce drásticamente. Más aún, este ejemplo muestra la utilidad de la cota indicada por la Proposición 8, dado que puede ser calculada de manera sencilla y directa, permitiendo aplicar el método de acotamiento, al menos generando una cota superior al número de actividades.

Para hacer un análisis más profundo del resto de los casos de la Tabla 4.3 dividiremos el conjunto de resultados en 2 casos: i) ordenamientos NPFS "casi" dominados (entradas en la tabla < 3) y ii) ordenamientos con caminos críticos NPFS sustancialmente distintos de los caminos críticos PFS (entradas en la tabla ≥ 3).

El conjunto de instancias con entradas en la tabla menores que 3, representan aquellos ordenamientos NPFS que no contienen a los caminos críticos PFS por 1 o 2 operaciones (como el ejemplo de la Figura 4.6). Estos casos, como ya se comentó, tienden a estar agrupados para instancias en donde $|S|$ es grande. Por lo que, ese tipo de ordenamiento NPFS raramente serán eficientes en la reducción del makespan. No obstante, no es posible establecer que son dominados sin conocer los tiempos de procesamiento.

El otro grupo de instancias con entradas en la tabla ≥ 3, son instancias en las que los caminos críticos PFS y NPFS difieren de forma significativa. Estas diferencias se incrementan a medida que el número de máquinas crece y $|S|$ es bajo. El máximo se da para el caso de 20 máquinas y 1 máquina de mutación, donde hay 6 operaciones del camino crítico PFS que no están incluidas en el camino crítico NPFS. Nótese que el bajo

número de máquinas de mutación implica que la longitud de ambos caminos críticos es prácticamente la misma, siendo esto una diferencia clara con el grupo de instancias con entradas < 3, donde el número de máquinas de mutación tiende a ser alto. Por lo tanto, para los casos en que las entradas de la Tabla 4.3 son grandes, los ordenamientos NPFS permiten obtener o encontrar caminos críticos que los ordenamientos PFS no pueden, y esto sin incrementar significativamente la longitud de los caminos críticos. Por lo que, es esperable que para estas instancias los ordenamientos NPFS obtengan menores (i.e. mejores) valores del makespan que los PFS.

Para poder evaluar estos dos conjuntos de instancias se propone el siguiente análisis paramétrico, es decir evaluando instancias con tiempos de procesamientos dados.

4.5. Experimentación paramétrica

En esta sección evaluaremos de forma paramétrica algunas de las conclusiones obtenidas en la sección 4.4.2. Para ello presentaremos formulaciones mixtas-enteras de los problemas permutativos y no-permutativos. Luego generaremos distintas clases de escenarios para evaluar las formulaciones. Finalmente compararemos los resultados obtenidos.

4.5.1. Formulaciones mixto-enteros

Primeramente, presentaremos la formulación PFS, la cual es similar a la presentada en la Sección 3.3.3 pero adaptada para este caso ya que no hay tiempos de traslados ni de preparación de máquinas.

Modelo PFS

Índices

M Número de máquinas; $m = 1,2,\ldots,M$

J Número de trabajos (o productos); $j = 1,2,\ldots,J$

Datos de entradas

$p_{j,m}$ Tiempo de procesamiento de una unidad de producto j en la máquina m

Ω Número positivo lo suficientemente grande.

Variables

$C_{j,m}$ Tiempo de finalización del trabajo *j* en la máquina *m*

$w_{j,j'}$ binaria que es 1 si el trabajo *j* precede al trabajo *j'*

Formulación MILP

Minimizar $z = C_{max}$ (4.4)

Sujeto a:

Restricción 4.5:
$C_{j,m} \geq C_{j,m-1} + pr_{m,j}\,, \quad \forall j, m > 1$

Restricción 4.6:
$C_{j,m} \geq C_{j',m} + p_{j,m} - \left(1 - w_{j,j'}\right) \cdot \Omega\,, \quad \forall m, j' \neq j$

Restricción 4.7:
$C_{j',m} \geq C_{j,m} + p_{j',m} - w_{j,j'} \cdot \Omega\,, \quad \forall m, j' \neq j$

Restricción 4.8:
$w_{j',j} + w_{j,j'} = 1\,, \quad j \neq j'$

Restricción 4.9:
$C_{max} \geq C_{j,m}\,, \quad \forall j, \forall m$

Restricción 4.10:
$C_{j,m} > 0;\ x_{j,j'}\ \{0,1\}$

La función objetivo 4.4 es el makespan. La restricción 4.5 mantiene la precedencia propia del flow shop, forzando a que un trabajo para pasar a la siguiente máquina debe haber terminado su procesamiento en la máquina previa. Las restricciones 4.6 y 4.7 trabajan de forma conjunta indicando el orden de los trabajos. Si el trabajo *j* es procesado previo al trabajo *j'*, entonces $w_{j,j'}$ es 1, la restricción 4.6 se vuelve activa y la 4.7 redundante. La expresión 4.8 establece el orden lógico, si el trabajo *j* se procesa antes que el trabajo *j'*, entonces la inversa no es válida. El makespan se define por la restricción 4.9 y en 4.10 se presentan las condiciones de factibilidad.

Modelo NPFS

El modelo NPFS es similar al PFS, pero con la diferencia que las condiciones de ordenamientos pueden cambiar. Para ello, utilizaremos la variable binaria *x* definida en

el capítulo 3, $x_{j,j',m}$, la cual es 1 si el trabajo *j* precede al trabajo *j'* en el procesamiento de la máquina *m*. Por lo que sólo modificaremos las ecuaciones en las que esta variable intervenga, es decir ecuaciones 4.6, 4.7 y 4.8, las cuales pasan a ser reemplazadas respectivamente por:

Restricción 4.11:
$$C_{j,m} \geq C_{j',m} + p_{j,m} - \left(1 - x_{j',j,m}\right) \cdot \Omega\,, \qquad \forall m, j' \neq j$$

Restricción 4.12:
$$C_{j',m} \geq C_{j,m} + p_{j',m} - x_{j',j} \cdot \Omega\,, \qquad \forall m, j' \neq j$$

Restricción 4.13:
$$x_{j',j,m} + x_{j,j',m} = 1\,, \qquad j \neq j', \forall m$$

De lo que las restricciones 4.11 y 4.12 son análogas a 4.6 y 4.7, pero el orden de trabajos puede modificarse a lo largo de las máquinas. La restricción 4.13 funciona con la misma lógica que la 4.8, pero evaluada para cada máquina.

4.5.2. Escenarios de prueba

Intentaremos evaluar las hipótesis planteadas a partir de la Tabla 4.3. Para ello generaremos escenarios alternativos, de forma tal que cada escenario difiera de otro en la dispersión de los valores de los tiempos de procesamientos $p_{j,m}$. Los tiempos de procesamientos para cada escenario fueron generados siguiendo la siguiente fórmula:

$$p_{j,m} = 2^{N(0,\delta)}$$

De donde el exponente corresponde a una distribución Gaussiana con media 0 y desvío estándar δ. Por lo tanto, variando el δ se varía el rango de valores $p_{j,m}$. Para generar los valores aleatorios que siguieran esa distribución Gaussiana se utilizó el software Excel.

Las instancias evaluadas consisten de 2 trabajos y el número de máquinas va de 4 a 20 máquinas, el mismo rango para el que se corrió el algoritmo ExA de la Tabla 4.2. El valor del desvío estándar δ utilizado fue de 0.5, 1, 1.5 y 2. Para cada instancia posible se generaron 5 conjuntos de datos y se resolvió cada instancia de forma óptima exacta con CPLEX.

4.5.3. Resultados

Los resultados obtenidos de estas experimentaciones se presentan de forma general en la Tabla 4.4. En la misma se muestra para cada valor de δ y cada número de máquinas si el mejor makespan para esa instancia fue obtenido por el formato PFS o NPFS, y en el caso de que el NPFS fuera el óptimo se registró la cantidad de mutaciones que tenía el ordenamiento óptimo (columna "S") y el gap entre el makespan NPFS y el makespan PFS. Los casos en que el enfoque NPFS era el mejor, están resaltados con sus celdas correspondientes pintadas de gris.

Tabla 4.4. Resultados experimentales paramétricos obtenidos con programación matemática.

		1			2			3			4			5		
	m		**S**	**gap**		**S**	**gap**		**S**	**gap**		**S**	**gap**		**S**	**gap**
$\delta = 0.5$	**4**	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	5	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	6	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	7	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	8	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	9	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	10	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	11	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	12	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	13	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	14	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	15	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	16	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	17	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	18	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	19	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	20	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
$\delta = 1$	**4**	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	5	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	6	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	7	PFS	-	-	PFS	-	-	NPFS	1	5.90%	PFS	-	-	PFS	-	-
	8	PFS	-	-	PFS	-	-	NPFS	1	2.80%	PFS	-	-	PFS	-	-
	9	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	10	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	11	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	0.40%
	12	PFS	-	-	PFS	-	-	NPFS	1	2%	PFS	-	-	NPFS	1	0.38%
	13	PFS	-	-	NPFS	1	2.20%	NPFS	1	3.60%	PFS	-	-	PFS		
	14	PFS	-	-	PFS	-	-	NPFS	1	0.20%	PFS	-	-	NPFS	1	0.39%
	15	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS		

	16	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS		
	17	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	0.30%
	18	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	0.29%
	19	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	0.28%
	20	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	1.80%	NPFS	1	0.27%
δ = 1.5	**4**	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	5	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	6	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	7	PFS	-	-	NPFS	1	0.57%	PFS	-	-	PFS	-	-	PFS	-	-
	8	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	9	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	10	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	11	NPFS	1	4.23%	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	12	NPFS	1	4.12%	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	13	NPFS	1	2.58%	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	14	NPFS	1	2.87%	NPFS	1	0.09%	PFS	-	-	PFS	-	-	NPFS	1	8.46%
	15	PFS	-	-	NPFS	1	0.67%	PFS	-	-	PFS	-	-	NPFS	1	6.76%
	16	NPFS	1	0.77%	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	6.04%
	17	NPFS	1	0.04%	NPFS	1	1.37%	PFS	-	-	PFS	-	-	NPFS	1	1.95%
	18	PFS	-	-	NPFS	1	1.33%	PFS	-	-	PFS	-	-	NPFS	1	2.78%
	19	PFS	-	-	NPFS	1	1.28%	PFS	-	-	PFS	-	-	PFS		
	20	NPFS	1	0.03%	NPFS	1	1.28%	PFS	-	-	PFS	-	-	NPFS	1	0.55%
δ = 2	**4**	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	5	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	6	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	7	PFS	-	-	NPFS	1	2.40%	PFS	-	-	PFS	-	-	PFS	-	-
	8	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	9	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	10	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-	PFS	-	-
	11	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	0.03%	PFS	-	-
	12	PFS	-	-	PFS	-	-	NPFS	1	1.04%	NPFS	1	0.29%	PFS	-	-
	13	PFS	-	-	PFS	-	-	NPFS	1	0.82%	NPFS	1	2.07%	PFS	-	-
	14	NPFS	1	0.10%	PFS	-	-	NPFS	1	0.51%	NPFS	1	2.30%	PFS	-	-
	15	PFS	-	-	PFS	-	-	NPFS	1	0.90%	NPFS	1	1.70%	PFS	-	-
	16	PFS	-	-	PFS	-	-	NPFS	1	0.91%	NPFS	1	0.70%	NPFS	1	2.50%
	17	PFS	-	-	PFS	-	-	PFS	-	-	NPFS	1	0.76%	PFS	-	-
	18	NPFS	1	3.75%	PFS	-	-	PFS	-	-	NPFS	1	0.83%	PFS	-	-
	19	NPFS	1	3.60%	PFS	-	-	PFS	-	-	NPFS	1	0.60%	NPFS	1	4.60%
	20	NPFS	1	3.54%	PFS	-	-	PFS	-	-	NPFS	1	2.64%	NPFS	1	4.38%

Una primera aproximación a la Tabla 4.4 considerando la influencia del parámetro δ, es posible ver que en el caso de que $\delta = 0.5$, en todos los casos el PFS fue mejor que el NPFS. Esto puede explicarse de una forma sencilla y directa. Ya que de los valores posibles de $\delta = 0{,}5$ es el que más acota la dispersión de los $p_{j,m}$, todos ellos resultan

bastante similares. Llevándolo al caso extremo en que todos los $p_{j,m}$ sean iguales, es decir $p_{j,m} = p$, los ordenamientos NPFS que involucren al menos una máquina de mutación nunca van a superar a los ordenamientos PFS. Esto se debe a que los caminos críticos NPFS incorporan más actividades que los caminos críticos PFS por lo que incorporarían más *p's*. Es por eso que cuando la dispersión es muy baja, las actividades extras de los caminos críticos NPFS tienen mucho impacto en el makespan.

Para el resto de los valores de δ la situación no es tan clara. Para estudiar estos casos presentamos una Tabla 4.5 que resume la anterior y permite una mejor comparación. La Tabla 4.5, muestra para cada instancia cuántas veces fue mejor el NPFS que el PFS, y el gap promedio de la mejora obtenida sobre el PFS. El promedio es considerando los 5 escenarios. Las casos en los que 3 o más de los 5 escenarios posibles el NPFS obtuvo mejor makespan que el PFS, se encuentran resaltados en gris. Para estos casos puede decirse que el NPFS obtuvo mejores makespan en general que los PFS (3 de 5 representa más de la mitad de los casos).

Tabla 4.5. Cantidad de instancias para las cuales NPFS es mejor que PFS y la mejora promedio.

m	**δ = 0.5**		**δ = 1**		**δ = 1.5**		**δ = 2**	
	NPFS	GAP	NPFS	GAP	NPFS	GAP	NPFS	GAP
4	0	-	0	-	0	-	0	-
5	0	-	0	-	0	-	0	-
6	0	-	0	-	0	-	0	-
7	0	-	1	2.95%	1	0.29%	1	1.20%
8	0	-	1	1.40%	0	-	0	-
9	0	-	0	-	0	-	0	-
10	0	-	0	-	0	-	0	-
11	0	-	1	0.20%	1	2.12%	1	0.02%
12	0	-	2	0.79%	1	2.06%	2	0.44%
13	0	-	2	1.93%	1	1.29%	2	0.96%
14	0	-	2	0.20%	3	2.86%	3	0.73%
15	0	-	0	-	2	2.48%	2	0.87%
16	0	-	0	-	2	2.27%	3	1.03%
17	0	-	1	0.15%	3	0.84%	1	0.38%
18	0	-	1	0.15%	2	1.37%	2	1.53%
19	0	-	1	0.14%	1	0.64%	3	2.20%
20	0	-	2	0.69%	3	0.47%	3	2.64%

De la Tabla 4.5 puede observarse a grueso modo que la mayor parte de las celdas grises se encuentran para altos valores de δ (>1) y para alto número de máquinas (>13). Esto en cierta forma respalda la inferencia realizada al final de la Sección 4.4.2, en donde

se mencionaba que a medida que el número de máquinas era alto, los ordenamientos NPFS podrían ser mejores que los PFS. Si también se considera la Tabla 4.4, se tiene que el número de máquinas de mutación fue en todos esos casos 1, lo que también sustenta lo mencionado en dicha inferencia, a menor número de máquinas de mutación mayor diferencias de caminos críticos, sin incorporar muchas actividades extras. Un nuevo aspecto que se descubre sobre estos casos, no mencionado en la Sección 4.4.2, es que la dispersión en los tiempos de procesamiento favorece al NPFS. Esto puede sustentarse, considerando que el NPFS permite encontrar un espectro más amplio de caminos críticos que los que permite encontrar el PFS. Entonces, dado que entre las distintas actividades existe un amplio rango de tiempos de procesamiento, el NPFS encuentra caminos críticos que, a pesar de incluir actividades extras, excluyen actividades que pueden tener un alto tiempo de procesamiento. Cabe mencionar que la mejora promedio del makespan para estos casos va desde el 0.47 % al 2.86%, y en todos los casos el número de máquinas de mutación está lejos del máximo posible, limitado por las proposiciones 1 y 2.

Para el resto de los casos con $\delta > 0.5$, puede decirse que los ordenamientos PFS tienden a ser superiores, aunque no de forma absoluta. Esto no se da para el caso en que el número de máquinas es reducido (< 6), en donde en todos los casos el PFS es superior al NPFS. Luego a medida que crece el número de máquinas, comienzan a aparecer casos en los que el NPFS obtuvo mejor makespan.

Considerando las hipótesis realizadas en la sección anterior (Sección 4.4.2) sobre el grupo definido por los valores entradas de la Tabla 4.3 < 3, se pudo comprobar tal como se dijo, que los ordenamientos NPFS con alto número de máquinas de mutación tendían a ser ineficientes, ya que eran pocas las actividades de los caminos críticos PFS que no capturaban, pero muchas las que agregaban de forma adicional por el número de cambios en los ordenamientos ($|S|$). Este fenómeno se observó incluso para los escenarios con alta dispersión en los tiempos de procesamiento, en donde si bien los ordenamientos NPFS dominaron, lo hicieron aquellos ordenamientos con bajo número de máquinas de mutación.

4.6. Conclusiones del capítulo

En este capítulo se consideraron los problemas NPFS y PFS con makespan como objetivo para el caso de dos trabajos. Se presentaron nuevas propiedades estructurales y

de dominancia sobre estos problemas. Estas propiedades han permitido encontrar interesantes resultados nuevos, y una nueva perspectiva sobre estos problemas. Se encontraron nuevas instancias en las que los ordenamientos PFS dominan estrictamente a los NPFS. También fue posible encontrar una cota al número máximo de ordenamientos factibles para el problema $F||C_{max}$ con dos trabajos. Además, se estudió la estructura de los caminos críticos para los problemas PFS, y se pudo representar mediante una fórmula recursiva.

Para el caso de los ordenamientos NPFS, se pudo demostrar que éstos pueden ser descompuestos en ordenamientos PFS más reducidos, y la formulación recursiva para caminos críticos PFS es válida. Asimismo, un procedimiento práctico de acotamiento es presentado. El procedimiento comienza analizando los tiempos de procesamientos de las actividades del camino crítico PFS que no están presentes en el camino crítico NPFS. Esto permite una reducción drástica en el número de ordenamientos candidatos al óptimo.

También se pudo comprobar de forma paramétrica y empírica algunas de las inferencias que derivadas del análisis no-paramétrico. Como comentario general del estudio paramétrico del problema, se pudo comprobar el impacto de la dispersión en los tiempos de procesamientos sobre los ordenamientos NPFS y PFS. Básicamente, a menor dispersión de los tiempos de procesamientos, más difícil que los ordenamientos NPFS obtengan makespan mejores que los PFS. Mientras, a medida que aumenta la dispersión de los mismos y el número de máquinas, los ordenamientos NPFS tienden a obtener mejores resultados que los ordenamientos PFS.

Como comentario general, se puede considerar que para el problema $F||C_{max}$ a mayor número de máquinas de mutación para un mismo ordenamiento, menores posibilidades tiene ese ordenamiento de ser óptimo.

5. CAPÍTULO 5: CONCLUSIONES

Conclusiones

5.1. Conclusiones

Los problemas NPFS son el caso general de los problemas PFS cualquiera sea el entorno, las restricciones o función objetivo que se estudien. Por lo que el estudio de los problemas NPFS tiene la misma o mayor relevancia que los problemas PFS. Sin embargo, dado el gran esfuerzo computacional que requiere su resolución, su tratamiento en la literatura es reciente, principalmente por el aumento de la capacidad de cálculo computacional e impulsado por a la mejora en los algoritmos y herramientas de optimización.

Primeramente, se revisó la literatura existente en la temática de forma completa. Esto permitió conocer el estado actual del conocimiento en el tema, organizando de forma sistemática todos los resultados e investigaciones publicados al respecto. Como primeras conclusiones de esta revisión fue posible detectar que los métodos más utilizados a la hora de abordar problemas NPFS fueron las heurísticas y meta-heurísticas, más del 80% de los artículos las utilizaron, algo esperado por tratarse de problemas NP-duros. Por otro lado, desde el punto de vista bibliométrico, se comprobó que los artículos sobre la temática tienen alto impacto en la comunidad científica dado que, más del 50% tienen 10 citas o más, y que el interés en la temática es creciente, la tendencia de publicaciones es creciente. Respecto a los problemas abordados en la literatura, se evidenció que aquellos que investigan funciones objetivo relativas a fechas de entrega y problemas multi objetivo se encuentra insuficientemente desarrollada. Por otro lado, los estudios sobre funciones objetivos relativas a fechas de finalización dominan ampliamente las publicaciones, siendo el makespan la función objetivo más frecuentemente estudiada. A su vez, se agruparon los casos en los que el enfoque PFS no asegura la solución óptima para problemas con condiciones especiales (efecto aprendizaje, tiempos de demoras, etc.), con makespan como objetivo. Una conclusión destacable del capítulo 2 es que los artículos que trabajan de forma experimental con el makespan como objetivo y comparan las soluciones NPFS y PFS, encuentran que las diferencias en la calidad de la solución nunca supera el 3%. Por su parte los resultados teóricos muestran el peor de los casos, en el que el makespan PFS puede ser varias veces superior al makespan NPFS. Sin embargo, frente a un problema de flow shop concreto, la literatura no presenta herramientas que permitan establecer si ese problema tendrá un makespan PFS parecido al peor caso o a los

resultados experimentales, lo que conlleva tener que resolver los dos problemas PFS y NPFS.

Por lo expuesto en la revisión, se propuso un estudio que modificara la estrategia de planificación de la producción en la que se incorporarán nuevas variables de decisión, puntualmente el subloteo. La implementación de subloteo en problemas NPFS fue abordado por primera vez en el marco de las investigaciones de esta tesis doctoral. Como resultados se obtuvo que el subloteo resultó ser un método eficaz en reducir el makespan del problema. Las conclusiones particulares sobre los distintos aspectos que la implementación de sublote conlleva (número de sublotes, número de máquinas, etc.) se encuentran ampliamente desarrolladas en las conclusiones del capítulo 3. A modo de conclusión general de esta implementación, el comportamiento de sistemas NPFS y PFS implementando subloteo fue similar al resto de los resultados experimentales reportados en la literatura, esto es diferencias del makespan menores al 1%. Con una leve tendencia a incrementarse a medida que el número de máquinas y de trabajos aumenta. Otro punto interesante es que, en este caso, usamos métodos exactos de optimización (CPLEX) a fin de asegurar que las soluciones encontradas eran las óptimas tanto para NPFS como para PFS. Esto genera un soporte más sólido a los resultados de la literatura respecto a las diferencias en el makespan entre NPFS y PFS, ya que en la literatura la gran mayoría de los trabajos usaron métodos heurísticos o meta-heurísticos.

Finalmente, se procedió a investigar las soluciones NPFS y PFS en términos de caminos críticos para el caso de dos trabajos, para lo que se debieron generar herramientas capaces de describir las soluciones. Esto permitió evidenciar algunas propiedades internas del problema, tales como la longitud del camino crítico a partir del tipo de ordenamiento, dependiendo del número de máquinas de mutación. También pudo sintetizarse la estructura de los caminos críticos PFS en una formulación recursiva. Así mismo, se propuso un método que descompone los ordenamientos NPFS en sistemas PFS menores, en los que las formulaciones recursivas son aplicables. Por otro lado, se diseñó y desarrolló un algoritmo capaz de comparar las soluciones NPFS y PFS a nivel de operaciones involucradas en el camino crítico, lo que permite independizarse de los tiempos de procesamiento (estudio no-paramétrico). Esta comparación exhaustiva permitió encontrar soluciones NPFS que están estrictamente dominadas por las soluciones PFS. A su vez, el procedimiento experimental propuesto implementó muchas

de las propiedades definidas. Por otro lado, se desarrolló un análisis de acotamiento que permite restringir el número de ordenamientos NPFS candidatos al óptimo a partir del análisis de los tiempos de procesamiento. Por otro lado, el estudio no-paramétrico permitió obtener una nueva perspectiva sobre aquellos casos en los que la dominancia no era estricta, e inferir posibles explicaciones del porqué de los valores obtenidos de ese estudio. Por otro lado, se plantearon experimentaciones ad hoc para evaluar las inferencias derivadas del estudio no-paramétrico. Éstas permitieron comparar de forma empírica los ordenamientos PFS y NPFS a través de programación matemática. Un resultado interesante es el impacto de la dispersión de los tiempos de procesamiento, donde a medida que la dispersión aumenta los ordenamientos NPFS tienden a ser óptimos y los PFS sub-óptimos. Por otra parte, para los casos de baja dispersión el estudio de ordenamientos NPFS pueden tornarse poco eficientes para reducir el makespan.

Si bien estos últimos resultados fueron obtenidos y demostrados para el caso de dos trabajos, no son generalizables para *n*- trabajos. No obstante, permiten mostrar las herramientas y el tipo de análisis que permitiría trabajar un problema flow shop de forma NPFS pero ajustando el espacio de búsqueda lo suficiente como para que no sea necesario estudiar $n!^m$ ordenamientos, o bien, poder aproximar las diferencias entre el makespan NPFS y PFS.

REFERENCIAS

Aggoune, R. (2004). Minimizing the makespan for the flow shop scheduling problem with availability constraints. *European Journal of Operational Research*, *153*(3), 534-543.

Aguezzoul, A. (2014). Third-party logistics selection problem: A literature review on criteria and methods. *Omega*, *49*, 69-78.

Amirian, H., & Sahraeian, R. (2015). Augmented ε-constraint method in multi-objective flowshop problem with past sequence set-up times and a modified learning effect. *International Journal of Production Research*, *53*(19), 5962-5976.

Babaei, M., Mohammadi, M., & Ghomi, S. F. (2014). A genetic algorithm for the simultaneous lot sizing and scheduling problem in capacitated flow shop with complex setups and backlogging. *The International Journal of Advanced Manufacturing Technology*, *70*(1-4), 125-134.

Benavides, A. J., Ritt, M., & Miralles, C. (2014). Flow shop scheduling with heterogeneous workers. *European Journal of Operational Research*, *237*(2), 713-720.

Benavides, A. J., & Ritt, M. (2015, April). Iterated Local Search Heuristics for Minimizing Total Completion Time in Permutation and Non-permutation Flow Shops. In *ICAPS* (pp. 34-41).

Benavides, A. J., & Ritt, M. (2016). Two simple and effective heuristics for minimizing the makespan in non-permutation flow shops. *Computers & Operations Research*, *66*, 160-169.

BROZ, D. R. G. D. D., & ROSSIT, D. OPTIMIZACIÓN EN UN ENTORNO FLOW-SHOP CON SUB-LOTES VARIABLES Y ENTREMEZCLADOS.

Brucker, P., Heitmann, S., & Hurink, J. (2003). Flow-shop problems with intermediate buffers. *OR Spectrum*, *25*(4), 549-574.

Brucker, P., & Shakhlevich, N. V. (2011). Inverse scheduling: two-machine flow-shop problem. *Journal of Scheduling*, *14*(3), 239-256.

Chang, J. H., & Chiu, H. N. (2005). A comprehensive review of lot streaming. *International Journal of Production Research*, 43(8), 1515-1536.

Cheng, T. E., Lin, B. M., & Huang, H. L. (2012). Resource-constrained flowshop scheduling with separate resource recycling operations. *Computers & Operations Research*, *39*(6), 1206-1212.

Cheng, M., Mukherjee, N. J., & Sarin, S. C. (2013). A review of lot streaming. *International Journal of Production Research*, 51(23-24), 7023-7046.

Conway, R. W., Maxwell, W. L., & Miller, L. W. (1967). *Theory of scheduling*. Courier Corporation.

Cui, W. W., Lu, Z., Zhou, B., Li, C., & Han, X. (2016). A hybrid genetic algorithm for non-permutation flow shop scheduling problems with unavailability constraints. *International Journal of Computer Integrated Manufacturing*, 1-18.

Davendra, D., Senkerik, R., Zelinka, I., Pluhacek, M., & Bialic-Davendra, M. (2014). Utilising the chaos-induced discrete self organising migrating algorithm to solve the lot-streaming flowshop scheduling problem with setup time. *Soft Computing*, *18*(4), 669-681.

Demirkol, E., Mehta, S., & Uzsoy, R. (1998). Benchmarks for shop scheduling problems. *European Journal of Operational Research*, *109*(1), 137-141.

Färber, G., & Moreno, A. M. C. (2006, May). Performance study of a genetic algorithm for sequencing in mixed model non-permutation flowshops using constrained buffers. In *International Conference on Computational Science and Its Applications* (pp. 638-648). Springer Berlin Heidelberg.

Färber, G., Salhi, S., & Moreno, A. M. C. (2007, June). Semi-dynamic Demand in a Non-permutation Flowshop with Constrained Resequencing Buffers. In*International Conference on Large-Scale Scientific Computing* (pp. 536-544). Springer Berlin Heidelberg.

Farber, G., Coves Moreno, A. M., & Salhi, S. (2010). Performance evaluation of hybrid-CLP vs. GA: non-permutation flowshop with constrained resequencing buffers. *International Journal of Manufacturing Technology and Management*,*20*(1-4), 242-258.

Framinan, J. M., Gupta, J. N., & Leisten, R. (2004). A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. *Journal of the Operational Research Society*, *55*(12), 1243-1255.

Framinan, J. M., Leisten, R., & García, R. R. (2014). *Manufacturing scheduling systems*. Berlin: Springer.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, *1*(2), 117-129.

Gharbi, A., Labidi, M., & Louly, M. A. (2014). The Nonpermutation Flowshop Scheduling Problem: Adjustment and Bounding Procedures. *Journal of Applied Mathematics*, *2014*.

Gorman, M. F. (2016). A "Metasurvey" analysis in Operations Research and Management Science: A survey of literature reviews. *Surveys in Operations Research and Management Science*, *21*(1), 18-28.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey.*Annals of discrete mathematics*, *5*, 287-326.

Grau, R., Espuña, A., & Puigjaner, L. (1995). Environmental considerations in batch production scheduling. *Computers & chemical engineering*, *19*, 651-656.

Grau, R., Espuña, A., & Puigjaner, L. (1996). Completion times in multipurpose batch plants with set-up, transfer and clean-up times. *Computers & chemical engineering*, *20*, S1143-S1148.

Haq, A. N., Saravanan, M., Vivekraj, A. R., & Prasad, T. (2007). A scatter search approach for general flowshop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, *31*(7-8), 731-736.

Henneberg, M., & Neufeld, J. S. (2016). A constructive algorithm and a simulated annealing approach for solving flowshop problems with missing operations. *International Journal of Production Research*, *54*(12), 3534-3550.

Isenberg, M. A., & Scholz-Reiter, B. (2013). The Multiple Batch Processing Machine Problem with Stage Specific Incompatible Job Families. In *Dynamics in Logistics* (pp. 113-124). Springer Berlin Heidelberg.

Jain, A. S., & Meeran, S. (2002). A multi-level hybrid framework applied to the general flow-shop scheduling problem. *Computers & Operations Research*,*29*(13), 1873-1901.

Janiak, A. (1988). General flow-shop scheduling with resource constraints.*The International Journal Of Production Research*, *26*(6), 1089-1103.

Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, *1*(1), 61-68.

Koulamas, C. (1998). A new constructive heuristic for the flowshop scheduling problem. *European Journal of Operational Research*, *105*(1), 66-71.

Kumar, S., Bagchi, T. P., & Sriskandarajah, C. (2000). Lot streaming and scheduling heuristics for m-machine no-wait flowshops. *Computers & Industrial Engineering*, *38*(1), 149-172.

Li, J., Zhang, L., ShangGuan, C., & Kise, H. (2010, December). A GA-based heuristic algorithm for non-permutation two-machine robotic flow-shop scheduling problem of minimizing total weighted completion time. In *Industrial Engineering and Engineering Management (IEEM), 2010 IEEE International Conference on* (pp. 1281-1285). IEEE.

Li, J. Q., & Pan, Q. K. (2015). Solving the large-scale hybrid flow shop scheduling problem with limited buffers by a hybrid artificial bee colony algorithm. *Information Sciences*, *316*, 487-502.

Lin, S. W., & Ying, K. C. (2009). Applying a hybrid simulated annealing and tabu search approach to non-permutation flowshop scheduling problems.*International Journal of Production Research*, *47*(5), 1411-1424.

Lin, S. W., Ying, K. C., & Lee, Z. J. (2009). Metaheuristics for scheduling a non-permutation flowline manufacturing cell with sequence dependent family setup times. *Computers & Operations Research*, *36*(4), 1110-1121.

Linn, R., & Zhang, W. (1999). Hybrid flow shop scheduling: a survey. *Computers & industrial engineering*, *37*(1), 57-61.

Liu, S., & Ong, H. L. (2002). A comparative study of algorithms for the flowshop scheduling problem. *Asia-Pacific Journal of Operational Research*, *19*(2), 205.

Liao, C. J., Liao, L. M., & Tseng, C. T. (2006). A performance evaluation of permutation vs. non-permutation schedules in a flowshop. *International Journal of Production Research*, *44*(20), 4297-4309.

Liao, L. M., & Huang, C. J. (2010). Tabu search for non-permutation flowshop scheduling problem with minimizing total tardiness. *Applied Mathematics and Computation*, *217*(2), 557-567.

Mehravaran, Y., & Logendran, R. (2012). Non-permutation flowshop scheduling in a supply chain with sequence-dependent setup times. *International Journal of Production Economics*, *135*(2), 953-963.

Mehravaran, Y., & Logendran, R. (2013). Non-permutation flowshop scheduling with dual resources. *Expert Systems with Applications*, *40*(13), 5061-5076.

Mohammadi, M., Fatemi Ghomi, S. M. T., Karimi, B., & Torabi, S. A. (2010). MIP-based heuristics for lotsizing in capacitated pure flow shop with sequence-dependent setups. *International Journal of Production Research*, *48*(10), 2957-2973.

Moukrim, A., Rebaine, D., & Serairi, M. (2014). A branch and bound algorithm for the two-machine flowshop problem with unit-time operations and time delays. *RAIRO-Operations Research*, *48*(2), 235-254.

Nagarajan, V., & Sviridenko, M. (2009). Tight bounds for permutation flow shop scheduling. *Mathematics of Operations Research*, *34*(2), 417-427.

Nawaz, M., Enscore, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*(1), 91-95.

Nikjo, B., & Zarook, Y. (2014). A Non-Permutation Flow Shop Manufacturing Cell Scheduling Problem with Part's Sequence Dependent Family Setup Times. *International Journal of Applied Metaheuristic Computing (IJAMC)*, *5*(4), 70-86.

Pan, Q. K., Tasgetiren, M. F., Suganthan, P. N., & Chua, T. J. (2011). A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. *Information sciences*, *181*(12), 2455-2468.

Pan, Q. K., & Ruiz, R. (2012). An estimation of distribution algorithm for lot-streaming flow shop problems with setup times. *Omega*, *40*(2), 166-180.

Pan, Q. K., & Ruiz, R. (2013). A comprehensive review and evaluation of permutation flowshop heuristics to minimize flowtime. *Computers & Operations Research*, *40*(1), 117-128.

Pinedo, M. (2012). *Scheduling*. Springer.

Potts, C. N., & Baker, K. R. (1989). Flow shop scheduling with lot streaming. *Operations research letters*, *8*(6), 297-303.

Potts, C. N., Shmoys, D. B., & Williamson, D. P. (1991). Permutation vs. non-permutation flow shop schedules. *Operations Research Letters*, *10*(5), 281-284.

Pugazhendhi, S., Thiagarajan, S., Rajendran, C., & Anantharaman, N. (2003). Performance enhancement by using non-permutation schedules in flowline-based manufacturing systems. *Computers & industrial engineering*, *44*(1), 133-157.

Pugazhendhi, S., Thiagarajan, S., Rajendran, C., & Anantharaman, N. (2004). Generating non-permutation schedules in flowline-based manufacturing sytems with sequence-dependent setup times of jobs: a heuristic approach. *The International Journal of Advanced Manufacturing Technology*, *23*(1-2), 64-78.

Pugazhendhi, S., Thiagarajan, S., Rajendran, C., & Anantharaman, N. (2004) b. Relative performance evaluation of permutation and non-permutation schedules in flowline-based manufacturing systems with flowtime objective. *The International Journal of Advanced Manufacturing Technology*, *23*(11-12), 820-830.

Rahmani, D., Ramezanian, R., & Saidi-Mehrabad, M. (2014). Multi-objective flow shop scheduling problem with stochastic parameters: fuzzy goal programming approach. *International Journal of Operational Research*, *21*(3), 322-340.

Rajendran, C., & Ziegler, H. (2001). A performance analysis of dispatching rules and a heuristic in static flowshops with missing operations of jobs. *European Journal of Operational Research*, *131*(3), 622-634.

Ramezanian, R., Saidi-Mehrabad, M., & Rahmani, D. (2011). Flow Shop Scheduling Problem with Missing Operations: Genetic Algorithm and Tabu Search. *International Journal of Applied Operational Research*, *1*(2), 21-30.

Ramezanian, R., & Saidi-Mehrabad, M. (2013). Hybrid simulated annealing and MIP-based heuristics for stochastic lot-sizing and scheduling problem in capacitated multi-stage production system. *Applied Mathematical Modelling*,*37*(7), 5134-5147.

Rayward-Smith, V. J., & Rebaine, D. (2008). Analysis of heuristics for the UET two-machine flow shop problem with time delays. *Computers & Operations Research*, *35*(10), 3298-3310.

Rebaine, D. (2005). Flow shop vs. permutation shop with time delays.*Computers & Industrial Engineering*, *48*(2), 357-362.

Reiter, S. (1966). A system for managing job shop production. Journal of Business, 34, 371–393.

Ribas, I., Leisten, R., & Framiñan, J. M. (2010). Review: Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers and Operations Research*,*37*(8), 1439-1454.

Rossi, A., & Lanzetta, M. (2013). Nonpermutation flow line scheduling by ant colony optimization. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *27*(04), 349-357.

Rossi, A., & Lanzetta, M. (2013). Scheduling flow lines with buffers by ant colony digraph. *Expert Systems with Applications*, *40*(9), 3328-3340.

Rossi, A., & Lanzetta, M. (2014). Native metaheuristics for non-permutation flowshop scheduling. *Journal of Intelligent Manufacturing*, *25*(6), 1221-1233.

Rossit, D., Frutos, M., Tohmé, F. A., & Broz, D. (2015). Modelo de subloteo considerando el efecto aprendizaje en configuraciones productivas flow-shop. In *IV Argentine Symposium on Industrial Informatics (SII)-JAIIO 44 (Rosario, 2015)*.

Rossit, D., Frutos, M., Tohmé, F., Broz, D., & Rossit, D. MODELOS MIXTO-ENTEROS DE SUBLOTEO PARA PRODUCCIÓN FLOW-SHOP MINIMIZANDO EL MAKESPAN.

Rossit, D., Tohmé, F., Frutos, M., Bard, J., & Broz, D. (2016). A non-permutation flowshop scheduling problem with lot streaming: A Mathematical model. *International Journal of Industrial Engineering Computations*, *7*(3), 507-516.

Rudek, R. (2011). Computational complexity and solution algorithms for flowshop scheduling problems with the learning effect. *Computers & Industrial Engineering*, *61*(1), 20-31.

Ruiz, R., & Maroto, C. (2005). A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research*,*165*(2), 479-494.

Ruiz, R., & Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. *European Journal of Operational Research*, *205*(1), 1-18.

Sadjadi, S. J., Aryanezhad, M. B., & Ziaee, M. (2008) a. The general flowshop scheduling problem: mathematical models. *Journal of Applied Sciences*, *8*(17), 3032-3037.

Sadjadi, S. J., Bouquard, J. L., & Ziaee, M. (2008) b. An ant colony algorithm for the flowshop scheduling problem. *Journal of Applied Sciences*, *8*(21), 3938-3944.

Sarin, S. C., & Jaiprakash, P. (2007). *Flow shop lot streaming*. Springer Science & Business Media.

Shen, L., Gupta, J. N., & Buscher, U. (2014). Flow shop batching and scheduling with sequence-dependent setup times. *Journal of scheduling*, *17*(4), 353-370.

Sriskandarajah, C., & Wagneur, E. (1999). Lot streaming and scheduling multiple products in two-machine no-wait flowshops. *IIE transactions*, *31*(8), 695-707.

Strusevich, V. A., & Zwaneveld, C. M. (1994). On non-permutation solutions to some two machine flow shop scheduling problems. *Zeitschrift für Operations Research*, *39*(3), 305-319.

Swaminathan, R., Fowler, J. W., Pfund, M. E., & Mason, S. J. (2004) Minimizing total weighted tardiness in a dynamic flowshop with variable processing times. In IIE Annual Conference. Proceedings (p. 1). Institute of Industrial Engineers-Publisher.

Swaminathan, R., Pfund, M. E., Fowler, J. W., Mason, S. J., & Keha, A. (2007). Impact of permutation enforcement when minimizing total weighted tardiness in dynamic

flowshops with uncertain processing times. *Computers & Operations Research*, *34*(10), 3055-3068.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *european journal of operational research*, *64*(2), 278-285.

Tandon, M., Cummings, P. T., & LeVan, M. D. (1991). Flowshop sequencing with non-permutation schedules. *Computers & chemical engineering*, *15*(8), 601-607.

Trietsch, D., & Baker, K. R. (1993). Basic techniques for lot streaming.*Operations Research*, *41*(6), 1065-1076.

Tseng, C. T., & Liao, C. J. (2008). A discrete particle swarm optimization for lot-streaming flowshop scheduling problem. *European Journal of Operational Research*, *191*(2), 360-373.

Vahedi Nouri, B., Fattahi, P., & Ramezanian, R. (2013). Hybrid firefly-simulated annealing algorithm for the flow shop problem with learning effects and flexible maintenance activities. *International Journal of Production Research*, *51*(12), 3501-3515.

Vahedi-Nouri, B., Fattahi, P., & Ramezanian, R. (2013) a. Minimizing total flow time for the non-permutation flow shop scheduling problem with learning effects and availability constraints. *Journal of Manufacturing Systems*, *32*(1), 167-173.

Vahedi-Nouri, B., Fattahi, P., Tavakkoli-Moghaddam, R., & Ramezanian, R. (2014). A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints. *The International Journal of Advanced Manufacturing Technology*, *73*(5-8), 601-611.

Vickson, R. G. (1995). Optimal lot streaming for multiple products in a two-machine flow shop. *European Journal of Operational Research*, *85*(3), 556-575.

Vollmann T., William L.. Berry, & Whybark, D. C. (2005). *Manufacturing planning and control systems*. Irwin/McGraw-Hill.

Xiao, Y., Yuan, Y., Zhang, R. Q., & Konak, A. (2015). Non-permutation flow shop scheduling with order acceptance and weighted tardiness. *Applied Mathematics and Computation*, *270*, 312-333.

Yenisey, M. M., & Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega*,*45*, 119-135.

Ying, K. C. (2008). Solving non-permutation flowshop scheduling problems by an effective iterated greedy heuristic. *The International Journal of Advanced Manufacturing Technology*, *38*(3-4), 348-354.

Ying, K. C., Gupta, J. N., Lin, S. W., & Lee, Z. J. (2010). Permutation and non-permutation schedules for the flowline manufacturing cell with sequence dependent family setups. *International Journal of Production Research*, *48*(8), 2169-2184.

Ying, K. C., & Lin, S. W. (2007). Multi-heuristic desirability ant colony system heuristic for non-permutation flowshop scheduling problems. *The International Journal of Advanced Manufacturing Technology*, *33*(7-8), 793-802.

Yoon, S. H., & Ventura, J. A. (2002). An application of genetic algorithms to lot-streaming flow shop scheduling. *IIE Transactions*, *34*(9), 779-787.

Zhang, S.-Y., Lu, Z.-Q., Cui,W.-W (2014). Flow shop scheduling optimization algorithm with periodical maintenance. *Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS.* 20(6), 1379-1387.

Zheng, T., & Yamashiro, M. (2010, September). A novel quantum differential evolutionary algorithm for non-permutation flow shop scheduling problems. In *Electrical Engineering Computing Science and Automatic Control (CCE), 2010 7th International Conference on* (pp. 357-362). IEEE.

Ziaee, M. (2013). General flowshop scheduling problem with the sequence dependent setup times: A heuristic approach. *Information Sciences, 251*, 126-135.