Formulación y resolución de problema de planificación de la producción

Informe Final

Febrero de 2014

Matías González Russo – Fernando Islas De Maio

Facultad de Ingeniería – Universidad de la República

Contenido

Índice de Figuras

Índice de Cuadros

Resumen

El presente trabajo aborda la problemática de la planificación de la producción, haciendo foco en el *Problema de Dimensionado de Lotes Capacitados* (Capacitated Lot Sizing Problem, CLSP). Partiendo de un estudio bibliográfico se presentan las distintas variantes del problema, así como las diversas formas de modelado y diferentes estrategias de resolución. La presentación inicial se complementa con un análisis detallado para un problema de planificación de producción particular. Dicho análisis incluye la formulación algebraica del problema, resolución mediante metodología exacta, diseño e implementación de una heurística computacional, y estudio experimental del comportamiento de las mismas frente a distintos escenarios de prueba. A partir del estudio experimental, se identifican los principales aspectos que repercuten tanto en el desempeño de la heurística implementada, como en la metodología de resolución exacta. Se concluye el trabajo presentando los resultados del estudio experimental, así como las principales conclusiones y algunas posibles líneas de trabajos a futuro.

Palabras clave: planificación de producción, heurística.

Capítulo 1

Introducción

La planificación de la producción trata de la organización de los recursos y las actividades necesarias para transformar insumos y derivados en productos finales para atender su demanda de forma eficiente. El proceso productivo implica la gestión de actividades y equipamiento compartido. Dicha producción tiene como destino la demanda o el almacenamiento para determinados períodos en el tiempo. El proceso abarca en general la producción de lotes de diferentes productos mediante distintos niveles de actividades que están restringidas por capacidades de tamaño y tiempo en la disposición de insumos, derivados y en el uso de equipamiento. Normalmente existe un compromiso entre minimizar los costos de los procesos de producción y atender la demanda con cierto nivel de satisfacción; en ambos casos teniendo en cuenta restricciones impuestas por los procesos y por el uso de los distintos recursos.

1.1 Objetivos y alcance del trabajo

El presente trabajo aborda la problemática de la planificación de producción, haciendo foco en el *Problema de Dimensionado de Lotes Capacitados* (Capacitated Lot Sizing Problem, CLSP). El principal objetivo detrás de este estudio consiste en identificar aquellos aspectos del problema con mayor impacto en la resolución del mismo. Para esto se analizan dos enfoques de resolución bien diferenciados. El primero se realiza a partir de una herramienta de resolución exacta, mientras que el segundo se basa en la implementación computacional de una heurística específicamente diseñada para el problema tratado.

Se comienza en este primer capítulo con un estudio detallado de la problemática, analizando bibliografía que trata las diferentes variantes del problema, así como diversas formas de modelado y estrategias para la resolución del mismo.

Luego de introducir las distintas variantes del problema, en el Capítulo 2 se presenta un problema de planificación de producción en particular. Para este problema se elabora un modelo matemático basado en programación entera.

En una primera instancia se intenta resolver el problema anterior aplicando una metodología de resolución exacta. Los detalles de este enfoque se presentan en el Capítulo 3, donde adicionalmente se busca mejorar el desempeño del mismo a partir de la reformulación del modelo presentado originalmente. Para esto se analiza el comportamiento de una herramienta de resolución exacta para cada una de las formulaciones presentadas.

Posteriormente en el Capítulo 4 se aborda la resolución del problema aplicando una metodología diferente. Con base en la bibliografía estudiada, se realiza el diseño e implementación computacional de estructuras y metodología de resolución heurística. En este capítulo se presentan los detalles de la selección y diseño de la solución heurística, así como los distintos aspectos de la implementación computacional.

Inicialmente la heurística es sometida a un conjunto de pruebas sobre diversas instancias del problema con el fin de determinar los valores con mejores cualidades para los diferentes parámetros. Una vez configurada la heurística, se realiza una segunda etapa de pruebas donde se analiza el comportamiento de la misma ante distintos escenarios del problema. Esta es la fase principal del trabajo, donde se busca identificar cuáles son las características del problema con mayor impacto en la resolución del mismo, analizando el desempeño tanto de la heurística implementada como de la herramienta de resolución exacta.

Tanto la fase de configuración como la fase de análisis de escenarios requieren la generación de un conjunto elevado de instancias del problema, con el fin de otorgarle una mayor validez estadística a los resultados obtenidos. Para esto se realiza el desarrollo de una herramienta de generación automática de instancias del problema. Los principales aspectos de la generación de datos, así como los resultados obtenidos a partir del estudio computacional se detallan en el Capítulo 5.

Una vez se evalúan los resultados obtenidos, se presentan las conclusiones del estudio realizado en el Capítulo 6. Se finaliza este informe en el Capítulo 7 analizando algunas posibles líneas de trabajos a futuro que complementan el presente estudio.

1.2 Variantes del problema

En las siguientes secciones se analiza la problemática de la planificación de producción, presentando a partir de un estudio bibliográfico los distintos modelos algebraicos propuestos a lo largo del tiempo. Se comienza con las versiones más simples introducidas a principios del siglo pasado, y se avanza presentando las diferentes alternativas que han buscado modelar de manera más realista esta temática.

1.2.1 Primeros modelos

Como se mencionó anteriormente, la planificación de la producción incluye la organización de los recursos y las actividades necesarias para transformar insumos atendiendo cierta demanda de forma eficiente. En general existe un compromiso entre minimizar los costos asociados a los procesos y atender la demanda con cierto nivel de satisfacción, teniendo en cuenta las diferentes restricciones impuestas por los procesos y el uso de los recursos.

Los primeros trabajos desarrollados que abordan la optimización de la producción y manejo de inventario datan de comienzos del siglo XX. En estos casos el modelo se reduce a una ecuación matemática vinculando la demanda y los distintos costos asociados a la producción para el cumplimiento de la misma.

El modelo de la *Cantidad Económica de Pedido* (Economic Order Quantity, EOQ) fue desarrollado por Ford W. Harris en 1913 [16], y es una de las primeras aproximaciones conocidas a la planificación de producción que incluye además la gestión de inventarios. En el mismo se maneja una escala de tiempo continua, demanda constante y un horizonte de tiempo infinito. Dicho modelo busca determinar el tamaño óptimo de pedido Q, que permita atender la demanda D minimizando los costos totales de compra y de inventario z. La siguiente expresión vincula estos tres elementos:

$$z(Q) = pD + \frac{sD}{Q} + \frac{hQ}{2}$$

Donde p representa el costo de cada unidad comprada, s es costo de cada orden realizada, y h es el costo unitario de inventario. La expresión D/Q representa el total de pedidos realizados, mientras que $Q/2$ es la cantidad de inventario promedio.

Se quiere hallar el valor de Q que minimiza la expresión anterior. El mismo se ubica donde se anula la derivada parcial respecto a Q.

Por lo tanto se quiere determinar el tamaño de pedido Q tal que:

$$\frac{dz(Q)}{dQ} = -\left(\frac{sD}{Q^2}\right) + \frac{h}{2} = 0$$

Finalmente podemos establecer la siguiente expresión para determinar el valor del tamaño de lote económico *Q*:

$$Q = \sqrt{\frac{2sD}{h}}$$

El modelo anterior fue extendido primero por E. W. Taft en 1918 [27], en lo que se conoce como el *Lote Económico de Producción* (Economic Production Quantity, EPQ), y posteriormente por Jack Rogers en 1958 [25], en el *Problema de Planificación de Lote Económico* (Economic Lot Scheduling Problem, ELSP). Esta última extensión modela la producción de varios ítems compartiendo una misma máquina, y por lo tanto, el hecho de decidir qué producir y cuánto producir en cada período.

Si bien los tres modelos mencionados son diferentes, todos ellos comparten un supuesto significativo: demanda constante. Este supuesto incluido en las versiones más primitivas no es aceptable para el modelado de problemas más realistas.

El primer modelo que contempla variación de la demanda a lo largo del tiempo fue presentado por H.M. Wagner y T.H. Whitin en 1958 [31] y se conoce como el modelo de *Dimensionado de Lote Dinámico* (Dynamic Lot-Size Model). Al igual que para los modelos anteriores, se busca determinar el punto de equilibrio entre los costos de producción *pc* y costos de inventario *hc* asumiendo capacidad ilimitada (problema no capacitado). Para esto Wagner y Whitin definen una función f_{jt} representando el costo total de producir en el período *j* para satisfacer la demanda *d* hasta el período *t*, suponiendo inventarios nulos al comienzo de *j* y al final de *t*:

$$f_{jt} = s_j + pc_j \sum_{i=j}^{t} d_i + \sum_{k=j}^{t-1} \sum_{h=k+1}^{t} hc_k d_h$$

A partir de esta expresión, el problema consiste en encontrar la secuencia de períodos $w_1 w_2 ... w_m$ donde se active la producción de forma tal que se minimicen los costos totales:

$$z = \min_{1..m} \left\{ \sum_{i=1}^{m} f_{w_{i-1} w_i} \right\}$$

Donde para el conjunto de períodos $T=1,\ldots,|T|$ se debe cumplir que $w_1=1$ y $w_m=|T|$

Para resolver este nuevo problema Wagner y Whitin desarrollan un algoritmo basado en programación dinámica. Este algoritmo es conocido como el método de Wagner-Whitin y su vigencia continúa hasta el día de hoy, siendo utilizado habitualmente en la generación de cotas de forma eficiente a complejos problemas de producción. Una descripción más detallada del mismo se realiza en la sección 4.3.3.

En las siguientes secciones se presentan distintos modelos matemáticos que abordan la problemática del *Dimensionado de Lote Dinámico* incorporando los conceptos industriales más relevantes. Para cada modelo presentado se detallan sus principales componentes, y se analizan diferentes extensiones que han sido propuestas a lo largo del tiempo.

1.2.2 Problema de Dimensionado de Lotes No Capacitado

Una de las versiones más simples del problema de *Dimensionado de Lote Dinámico*, es la versión no capacitada para un único ítem (Single Item Uncapacitated Lot-Sizing Problem):

$$\min \sum_{t \in T} vc_t\, x_t + sc_t\, y_t + hc_t\, s_t \qquad (1)$$

$$s.a. \quad s_{t-1} + x_t = d_t + s_t, \qquad \forall t \in T \qquad (2)$$

$$x_t \leq D_t\, y_t, \qquad \forall t \in T \qquad (3)$$

$$x_t, s_t \geq 0,\ y_t \in \{0;1\}, \qquad \forall t \in T \qquad (4)$$

Donde tenemos tres variables para cada período *t*: la cantidad x_t de producción del ítem, denominado lote, la activación de producción y_t, y el total de inventario al final de período s_t. Para cada una de estas variables existe un costo asociado: vc_t, sc_t y hc_t representan respectivamente los costos de producción, activación o configuración (setup) y de inventario. Los últimos dos parámetros del modelo son la demanda para cada período d_t, y la demanda acumulada desde *t* hasta el último período D_t.

El objetivo del problema es minimizar el costo total de producción, configuración e inventario (1). Al igual que para el problema de *Cantidad Económica de Pedido*, existe un compromiso entre los costos de setup y los costos de inventario. La demanda puede ser atendida con producción del período, o inventario de períodos anteriores (2). Cualquier excedente en la producción es arrastrado al siguiente período como inventario. Como no se requiere inventario al final del horizonte de tiempo, la producción en un período se ve limitada por la demanda restante acumulada D_t. Esto permite obtener una cota para la restricción de activación de producción (3). Por último, tanto la cantidad de producción como de inventario al final de cada período deben ser positivas, y la activación de producción por definición es binaria (4).

Como se mencionó en el punto anterior, bajo ciertas hipótesis este problema puede resolverse de forma eficiente con el algoritmo desarrollado por Wagner-Whitin. La idea detrás de este algoritmo es analizar el ratio entre los costos de setup y los costos de inventario, activando la producción sólo cada cierta cantidad de períodos para satisfacer la demanda de los mismos. Dicha cantidad de períodos se conoce como el tiempo entre órdenes (time between orders, TBO) y determina la forma de la solución de esta clase de problemas.

1.2.3 Problema de Dimensionado de Lotes Capacitado

Los supuestos del modelo anterior no se corresponden en general con la realidad de una empresa que cuenta con una capacidad limitada y produce más de un producto.

Contemplar estas dos características permite separar los distintos modelos en dos grandes clasificaciones introducidas por Raf Jans y Zeger Degraeve [7]:

- Modelos *large bucket*: donde varios ítems pueden ser producidos en una misma máquina en un mismo período,
- Modelos *small bucket*: para los cuales sólo se permite producir un único ítem por período.

El *Problema de Dimensionado de Lotes Capacitado* (Capacited Lot Sizing Problem, CLSP) es un ejemplo de problema large bucket, y puede formularse algebraicamente de la siguiente manera:

$$\min \sum_{i \in P} \sum_{t \in T} sc_t^i y_t^i + hc_t^i s_t^i \tag{1}$$

$$s.a. \quad s_{t-1}^i + x_t^i = d_t^i + s_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{2}$$

$$\sum_{i \in P} et^i x_t^i \leq cape_t, \qquad \forall t \in T \tag{3}$$

$$x_t^i \leq D_t^i y_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{4}$$

$$x_t^i, s_t^i \geq 0, \; y_t^i \in \{0;1\}, \qquad \forall i \in P, \quad \forall t \in T \tag{5}$$

Podemos observar que a grandes rasgos la formulación es similar a la presentada para el problema no capacitado en la sección anterior, donde para reflejar la producción de distintos ítems, se extienden los parámetros y las variables agregando un nuevo índice *i* que identifica los mismos (1) (2) (4) (5).

Contemplar una capacidad de producción limitada implica ajustes más significativos. En primer lugar se define un nuevo parámetro $cape_t$ para representar la capacidad máxima de producción para cada período, donde cada unidad producida del ítem *i* consume et^i unidades de capacidad. De esta manera, el total de producción para un período determinado no puede superar la capacidad máxima disponible (3).

Para estos problemas donde todos los ítems pueden ser producidos en una misma máquina en un mismo período, ésta es además la única restricción que vincula los diferentes ítems. Como se verá en el siguiente capítulo, una estrategia de resolución muy común para esta clase de problemas consiste en relajar la restricción de capacidad, transformando el mismo en $|P|$ problemas no capacitados independientes.

Existen sin embargo algunos escenarios donde no es posible producir más de un ítem en cada período. Para estos casos se puede ajustar el modelo anterior agregando una restricción adicional:

$$\sum_{i \in P} y_t^i \leq 1, \qquad \forall t \in T$$

De esta manera, para cada período *t* podrá activarse la producción, a lo sumo de un único ítem *i*.

Es importante destacar que para cualquiera de los dos escenarios presentados, contemplar una capacidad limitada convierte a estos problemas de dimensionado en NP-duros [13].

1.2.4 Costos de producción unitarios

El modelo presentado en el punto anterior contempla únicamente dos tipos de costos: costos de activación de producción y costos de inventario. Los primeros normalmente son independientes de la cantidad de ítems producidos.

Algunos modelos contemplan además los costos unitarios asociados a la producción de los distintos ítems, es decir, aquellos costos en los que se incurre por cada unidad producida. Dentro de estos costos normalmente se incluyen los asociados a la adquisición y preparación de materias primas, elaboración y envasado de los productos finales.

Para las versiones más sencillas se define un único costo *pc* asociado a cada unidad producida. Este costo puede o no variar para los distintos ítems y/o períodos. De esta manera se extiende la función objetivo presentada en la sección anterior:

$$\min \sum_{i \in P} \sum_{t \in T} pc_t^i x_t^i + sc_t^i y_t^i + hc_t^i s_t^i$$

Versiones más complejas pueden incluir además costos de producción variables en función de la cantidad de ítems producidos. Este punto es muy común cuando se trabaja con materias primas, las cuales pueden ser adquiridas al por mayor con su correspondiente reducción de costos. También son consideraros costos variables cuando la cantidad de ítems producidos excede la capacidad de la empresa y se debe recurrir a un tercero para cumplir con la demanda. En estos casos los costos unitarios asociados a la producción tercerizada superan generalmente los costos unitarios de producción.

Existen aún más alternativas a la hora de contemplar costos unitarios de producción. Sin embargo una gran variedad de problemas, como el estudiado por Wagner-Whitin, no los tienen en cuenta. Este último punto es mencionado por Maes y Wassenhove [20], quienes destacan que la forma de la solución a estos problemas no varía si estos costos son constantes, pudiendo haber mínimos cambios si los mismos son variables. Estas afirmaciones fueron verificadas experimentalmente dentro del análisis de escenarios que se presenta en el Capítulo 5.

1.2.5 Tiempos de configuración

La utilización de maquinaria para la producción implica muchas veces incurrir en tiempos de preparación, calibración, limpieza, inspección, y otro tipo de actividades que forman parte de la configuración (setup) de las mismas. Al igual que los tiempos de producción, los tiempos de setup consumen parte de la capacidad de producción en cada período.

Existen algunas realidades donde estos tiempos son insignificantes en relación con el tiempo total de producción, y por lo tanto no son incluidos en los modelos matemáticos. Sin embargo en muchas ocasiones los mismos representan un porcentaje no despreciable. En estos casos, no incluir los tiempos de configuración implica no modelar el problema de forma realista [29].

El CLSP se vuelve sensiblemente más complejo de resolver cuando son considerados los tiempos de setup. Por ejemplo si estos tiempos no son considerados, la factibilidad del problema puede determinarse a partir del análisis de la demanda y capacidad acumuladas. Basta verificar que en cada período la primera no supera a la segunda para asegurar la existencia de alguna solución factible. Cuando los tiempos de setup se incluyen en la formulación, el problema de factibilidad se vuelve NP-completo [29]. Por lo tanto no se cuenta con un algoritmo eficiente para determinar la existencia o no de una solución factible, menos aún hallar una solución óptima.

Para hacer énfasis en este último punto, analicemos el problema de planificación de producción con dos ítems y tres períodos que se presenta en el Cuadro 1.1. En el mismo se detalla además una posible solución al problema donde el tamaño de lote para cada ítem en cada período coincide con su demanda. Se observa además que para esa solución la capacidad utilizada en el tercer período supera la capacidad disponible, por lo tanto la solución propuesta es no factible.

Item	**Tiempo configuración (s)**	**Tiempo producción unitario (s)**	**Demanda por período (u)**		
			1	**2**	**3**
1	10	1	10	0	11
2	4	1	0	6	0
		Capacidad utilizada (s)	20	10	21
		Capacidad disponible (s)	20	20	20

Cuadro 1.1 – Problema de planificación de la producción: Ejemplo I

El problema anterior parece sencillo en primera instancia: la capacidad utilizada acumulada (51) no supera en ningún período la capacidad disponible acumulada (60); la tasa de uso de capacidad promedio (51/60) es del 85%. Más aún, se tiene una única unidad de exceso de capacidad a eliminar para obtener una solución factible.

Sin embargo no es posible hallar una solución factible: el período 1 no puede aceptar mayor producción, y la capacidad disponible en el período 2 no permite la activación de producción del ítem 1. De esta manera, el problema anterior no tiene soluciones factibles.

Antes de concluir nuestra discusión sobre los tiempos de setup, mencionamos otra diferencia importante en comparación con los problemas que no los contemplan. Para esto previamente definimos el esquema de producción *lote-por-lote* como aquel donde el tamaño de lote para cada ítem coincide con la demanda en cada período. Cuando se consideran tiempos de setup es posible encontrar soluciones factibles aún cuando la capacidad utilizada *lote-por-lote* supera la capacidad acumulada disponible.

A continuación se presenta otro ejemplo para hacer énfasis en este punto, el cual se discute más en detalle en el Capítulo 5. Analicemos el problema de planificación de producción con dos ítems y tres períodos que se presenta en el Cuadro 1.2.

Item	**Tiempo configuración (s)**	**Tiempo producción unitario (s)**	**Demanda por período (u)**		
			1	**2**	**3**
1	50	1	10	10	10
2	60	1	10	10	10
		Capacidad utilizada (s)	130	130	130
		Capacidad disponible (s)	150	80	130

Cuadro 1.2 – Problema de planificación de la producción: Ejemplo II

Se observa que para el problema anterior la capacidad utilizada con un esquema de *lote-por-lote* (390) supera la capacidad acumulada disponible (360).

Sin embargo, desplazando el total de producción del período 2 hacia el período 1, se obtiene una solución factible. La eliminación de excesos en la capacidad total acumulada a través del desplazamiento de lotes enteros no es posible para problemas donde no se consideran los tiempos de setup.

Trigeiro identifica ciertas propiedades que deben cumplir los problemas con ajustadas restricciones de capacidad para ser resolubles [29]. En estos casos las restricciones de demanda deben ser menos exigentes en los primeros períodos, permitiendo la generación de inventario para satisfacer la demanda de los últimos períodos. De esta manera es posible encontrar esquemas de producción factibles aún cuando la demanda requerida hacia el final del horizonte de tiempo supere la capacidad disponible.

A partir de los dos ejemplos anteriores, podemos concluir que los métodos tradicionales para verificar la factibilidad (o no factibilidad) del problema sin considerar los tiempos de setup, no son aplicables para esta nueva clase de problemas.

1.2.6 Inventario mínimo

En algunos casos no es suficiente satisfacer la demanda, también es necesario contar con una cierta cantidad de inventario mínimo *ms* para cada ítem *i* al final de cada período *t*:

$$s_t^i \geq ms_t^i, \qquad \forall i \in P, \quad \forall t \in T$$

Manejar esta clase de restricciones permite tener una cierta holgura en caso que la demanda real de algún período supere la demanda previamente estimada. Este punto es especialmente importante para aquellos escenarios con gran incertidumbre en los valores de la demanda.

Si bien la restricción de inventarios mínimos es fácil de modelar, el impacto de la misma sobre las soluciones del problema no es menor. Consideremos por ejemplo el problema de determinar el total de producción necesario para satisfacer la demanda. Cuando no se consideran inventarios mínimos es fácil demostrar que el total de producción necesario coincide con el total de la demanda acumulada. El Cuadro 1.3 ilustra esto para un problema sencillo con un único ítem, seis períodos y un esquema de producción *lote-por-lote*.

Período	**1**	**2**	**3**	**4**	**5**	**6**	**Total**
Demanda	10	15	20	10	35	10	100
Producción	10	15	20	10	35	10	100

Cuadro 1.3 – Problema de planificación de la producción sin inventario mínimo: Ejemplo I

El total de producción necesario se mantiene incluso si se aplica algún esquema de producción que haga uso de inventarios al final de los períodos. Esto se detalla en el Cuadro 1.4 para el mismo problema anterior.

Período	**1**	**2**	**3**	**4**	**5**	**6**	**Total**
Demanda	10	15	20	10	35	10	100
Producción	20	10	20	15	30	5	100
Inventario final	10	5	5	10	5	0	

Cuadro 1.4 – Problema de planificación de la producción sin inventario mínimo: Ejemplo II

Sin embargo, cuando se extiende el problema anterior agregando restricciones de inventarios mínimos, no es posible de antemano determinar el total de producción necesario a partir de la demanda acumulada. Este se ve reflejado en el Cuadro 1.5, donde se observa que el inventario mínimo requerido al final del quinto período excede la demanda total acumulada hasta el último período. De esta manera, el total de producción necesario (105) supera la demanda total acumulada (100).

Período	1	2	3	4	5	6	Total
Demanda	10	15	20	10	35	10	100
Inventario mínimo	2	7	6	4	15	3	37
Producción	12	20	19	8	46	0	105
Inventario final	2	7	6	4	15	5	

Cuadro 1.5 – Problema de planificación de la producción con inventario mínimo.

Teniendo en cuenta los escenarios anteriores, podemos encontrar una expresión para hallar el total de producción necesario bajo la siguiente hipótesis:

- Para cada ítem, la cantidad de inventario mínimo al final de cada período es menor a la demanda total acumulada hasta el último período.

Asumiendo la hipótesis anterior, tenemos que el total de producción (mínimo) necesario para satisfacer tanto la demanda como la restricción de inventario mínimo es igual a la demanda total acumulada más el inventario mínimo requerido al final del último período.

Concluimos este punto mencionando que no fue posible encontrar bibliografía que trate la problemática del manejo de inventarios mínimos. Se estudiaron distintos trabajos donde se incluyen restricciones de cantidad de producción mínima y de inventario máximo, pero ninguno que contemple la restricción de este punto.

1.2.7 Traslado de configuración

En algunos casos es posible reutilizar parte del trabajo realizado en períodos anteriores. Por ejemplo la configuración de maquinaria puede ser evitada en un período si ya fue ajustada en el período anterior. De esta manera la configuración previamente realizada se traslada (carry over) al período siguiente evitando incurrir nuevamente en costos de setup.

Al incluir carry over entre distintos períodos, se permite además que los tiempos de setup para un ítem puedan ser reutilizados, reduciendo de esta manera la capacidad utilizada.

Se presentan dos escenarios de trabajo bien diferenciados cuando se considera el carry over:

- El carry over entre períodos es independiente para cada ítem: por ejemplo cuando cada ítem se produce en una máquina distinta. En estos casos pueden trasladarse tantas configuraciones como ítems.
- Sólo se permite carry over para un único ítem por período: esto sucede en general cuando todos los ítems se producen utilizando la misma maquinaria. En este caso sólo se podrá arrastrar la configuración del último ítem producido.

El primer escenario se modela sin mayores cambios, agregando una nueva variable binaria *z* para indicar la existencia de carry over para el ítem *i* en el período *t*. La activación de *z* queda definida a partir de la variable de activación de producción *y* como se describe en la siguiente restricción:

$$2z_t^i \leq y_t^i + y_{t-1}^i, \qquad \forall i \in P, \quad \forall t \in T$$

Observamos que *z* se debe anular a menos que se activen ambos términos del lado derecho. De esta manera, únicamente se trasladará la configuración cuando se active la producción del período actual *t*, y la del período anterior *t*-1.

Finalmente se refleja el traslado de configuración entre períodos en la función objetivo de la siguiente manera:

$$\min \sum_{i \in P} \sum_{t \in T} sc_t^i (y_t^i - z_t^i) + hc_t^i s_t^i$$

Donde los costos de configuración en el período *t* se anulan cuando existe carry over desde el período *t*-1.

El segundo escenario implica mayores desafíos. Para este caso debemos ahora considerar un nuevo tipo de decisión: la del ordenamiento de la producción dentro de un período, ya que importará qué ítem se produzca primero y qué ítem se produzca último.

Para la mayoría de los modelos presentados el carry over se lleva a cabo cuando el último ítem producido en el período *t* coincide con el primer ítem producido en el período *t*+1. Gopalakrishnan presenta un algoritmo basado en una búsqueda tabú para esta clase de problemas [15].

Existen sin embargo algunos modelos que contemplan el traslado de configuración en otras circunstancias. Por ejemplo si existe suficiente capacidad disponible entre la producción del último ítem de un período *t* y el final de dicho período, se puede hacer uso de este excedente para adelantar el setup del primer ítem a ser producido en $t+1$. Otra alternativa, si no se activa la producción en un determinado período *t*, implica utilizar esta capacidad para hacer el setup del próximo ítem a producir como en el punto anterior, o se puede mantener el estado de la máquina desde el período anterior $t-1$ hasta el período en $t+1$.

Algunos modelos van un paso más allá y no se limitan a determinar el primer y último ítem en ser producidos, sino que contemplan el ordenamiento completo de la producción dentro de cada período. La complejidad de estos modelos crece sensiblemente debido al gran número de variables binarias que se deben incluir para contemplar las decisiones de secuenciado para todos los ítems en cada período.

1.2.8 Demanda atrasada

En algunas situaciones es posible satisfacer la demanda de un período con un cierto atraso (backlog). Para estos modelos la demanda puede cumplirse a partir de producción del período actual, de períodos anteriores (inventario), y/o de períodos futuros. En este último caso se incurre generalmente en una penalidad dependiendo del nivel de atraso. La variante de backlogging en el estudio de los problemas de planificación de producción fue introducida por Zangwill en 1966 [35].

Debemos entonces extender el modelo para establecer desde qué período se satisface determinada demanda. Este último punto normalmente se implementa reemplazando la variable de lote x_t^i por una nueva variable w_{tj}^i donde: *i* continúa representado el ítem producido, *t* el período donde se produce, y *j* el período donde se utiliza dicha producción para satisfacer la demanda.

Asumiendo una penalidad bc_t^i por backlog, que en este caso no depende de los períodos de atraso, podemos ajustar la función objetivo de la siguiente manera:

$$\min \sum_{i \in P} \sum_{t \in T} pc_t^i \left(\sum_{j \in T} w_{tj}^i \right) + bc_t^i \left(\sum_{\substack{j \in T \\ j > t}} w_{jt}^i \right) + sc_t^i y_t^i + hc_t^i s_t^i$$

Observamos que el primer término de la ecuación se corresponde con el costo de producción del período actual, ya sea para satisfacer demanda anterior, actual o futura. El segundo término representa la penalidad en que se incurre por satisfacer la demanda desde un período *j* posterior. Los últimos dos términos coinciden con los modelos anteriores y representan los costos de activación de producción y de inventario.

Esta nueva formulación no sólo permite determinar el nivel óptimo de producción de cada período, sino que también detalla el período en que se utiliza la misma. Este último punto es fundamental si existen restricciones asociadas con la cantidad de períodos en que un lote determinado puede permanecer como inventario, algo muy común cuando en los problemas se contempla la caducidad de los ítems.

Como consecuencia de esta formulación extendida, el tamaño del espacio de soluciones factibles aumenta sensiblemente. De esta manera se facilita la generación de las mismas, pero se dificulta la obtención de óptimos globales. Pochet y Wolsey proponen para esta clase de problemas un enfoque general basado en reformulaciones sencillas y aplicación de planos de cortes genéricos [23]. Este tipo de enfoque general favorece la flexibilidad de la solución, evitando incurrir en costos para la adaptación de algoritmos específicos. Por su parte Federgruen presenta un algoritmo que extiende el método de Wagner-Whitin para esta clase de problemas [11].

1.2.9 Otras variantes al problema de planificación de producción

Terminando esta sección se presentan brevemente algunas variantes menos comunes al problema de planificación de producción.

1.2.9.1 Dimensionado discreto

Existen algunas realidades donde no es posible graduar el nivel de producción. En estos casos, si se activa la producción debe ser al tope de la capacidad disponible.

Este tipo de extensiones son contempladas en el *Problema de Dimensionado Discreto* y *Planificación de Lotes* (Discrete Lot Sizing and Scheduling Problem, DLSP). La formulación general de este problema fue presentada por Fleischmann en 1990 [12], y tiene una estructura similar al CLSP detallado anteriormente. La principal diferencia se presenta en la restricción de activación de producción, la cual en este caso se convierte en la siguiente igualdad:

$$et^i x_t^i = cap_t y_t^i, \qquad \forall i \in P, \quad \forall t \in T$$

Notar que en estos casos la variable de tamaño de lote *x* puede ser reemplazada por la variable de activación de producción, simplificando la formulación del problema.

De manera similar, en algunos casos el nivel de producción puede graduarse pero sólo a tamaños de lotes discretos. Esto es algo muy común en la industria química o de alimentos cuando la producción está limitada por ejemplo por el tamaño de un tanque. En estos casos el tamaño del lote necesariamente debe ser un múltiplo de la capacidad del tanque.

1.2.9.2 Desempeño y gestión de los recursos

Casi la totalidad de los modelos de dimensionado de lotes asumen que la producción se realiza con maquinaria completamente confiable. Existen sin embargo algunas formulaciones que analizan los aspectos asociados a la confiabilidad de los diferentes recursos.

Las versiones más simples abordan estos temas a través de la configuración de las máquinas. Por ejemplo Tzur presenta un modelo genérico en el cual los tiempos y/o los costos de setup varían en función de la utilización de las máquinas [30]. Kuhn introduce otra alternativa contemplando costos y tiempos de mantenimiento, y presentando un análisis para el problema no capacitado [18].

Modelos más complejos pueden contemplar incluso la producción de un mismo producto en diferentes máquinas, cada una de éstas con distintos desempeños, costos y tiempos asociados. Algunos autores anexan a la gestión de maquinaria el manejo de otros recursos. Tal es el caso de Akturk y Onen, quienes extienden el problema de dimensionado de lotes con el manejo de herramientas a partir de la inclusión de restricciones de disponibilidad y compatibilidad de los distintos recursos [5].

1.2.9.3 Niveles múltiples y dependencia entre productos

Como fue mencionado anteriormente, los problemas de dimensionado de lotes abordan la problemática de planificación de producción básicamente respondiendo a dos preguntas: ¿cuándo y cuánto producir?

Como consecuencia, esta clase de problemas se enfocan en un único tipo de decisión, aquellas asociadas a los niveles de producción, y debido a esto se encuadran dentro de la categoría de *problemas de nivel único* (single level problems).

Existe otra clase de problemas de producción donde la toma de decisiones no se limita a un único tipo de decisión. Estos son llamados *problemas de nivel múltiple* (multiple level problems), donde las decisiones tomadas en un nivel pueden servir como insumo para decisiones en otros niveles.

Dentro de esta clase de problemas, los más comunes son aquellos que combinan la optimización de producción con la distribución de los diferentes ítems. En estos casos las decisiones asociadas a los niveles de producción deben complementarse con decisiones de distribución como por ejemplo: ¿dónde producir? y ¿desde dónde atender la demanda?

Problemas de múltiples niveles pueden presentarse incluso cuando únicamente se contemplan decisiones de producción pero a distinto orden. Este es el caso, por ejemplo, cuando la producción se realiza a partir de diversos insumos que se transforman en diferentes derivados intermedios. Estos derivados intermedios posteriormente se combinan entre ellos o con otras materias primas para formar los productos finales. En estos casos se debe decidir no sólo cuánto producir, sino cómo combinar de forma eficiente los distintos insumos y derivados intermedios.

Este tipo de producción en varios niveles es muy común en la producción de petróleo, y trae aparejado además nuevas restricciones a considerar. Las más comunes son las *restricciones de balance*, las cuales establecen por ejemplo que la cantidad de producción saliente de un nivel debe coincidir con la entrante al siguiente nivel. De manera similar se establecen restricciones que determinan las proporciones entre insumos a combinar para la producción de algún producto.

Por último mencionamos otra clase de problemas de nivel múltiple que se establecen cuando se consideran diferentes horizontes de tiempos. En estos casos normalmente se maneja un modelo operacional para decisiones a corto plazo, y un modelo táctico para decisiones a largo plazo.

Este tipo de problemas son muy comunes en la industria de celulosa donde se deben combinar decisiones a corto plazo vinculadas con el transporte y acopio de madera, con decisiones a largo plazo asociadas con el manejo y planificación de forestación. Mientras que las primeras normalmente determinan la planificación semanal o mensual, las segundas implican decisiones que se extienden durante varios años e incluso décadas.

1.3 Estrategias de resolución

Concluimos este capítulo discutiendo los diferentes enfoques de resolución para la problemática de la planificación de producción, prestando especial atención en los problemas de dimensionado de lotes capacitados.

Las distintas estrategias de resolución pueden agruparse en dos grandes categorías: métodos de resolución exacta y métodos de resolución heurística. Los primeros, en caso de existir, garantizan la obtención de una solución óptima. Para una gran cantidad de problemas esto requiere una capacidad de cómputo que habitualmente excede la disponible. Las heurísticas por otro lado requieren una capacidad de cómputo sensiblemente menor a las estrategias de resolución exacta. Por contrapartida, estos métodos de resolución habitualmente no dan garantías de encontrar soluciones factibles, menos aún asegurar la calidad de las mismas.

Como se ha mencionado en la sección anterior, aún las formulaciones más básicas del CLSP son problemas NP-duros. El problema es considerado complejo incluso en un sentido práctico, ya que la mayoría de los métodos de resolución exacta sólo han logrado encontrar óptimos en tiempos razonables para versiones pequeñas del problema [20]. Por lo tanto, la mayoría de los métodos de resolución utilizados para problemas de esta realidad son por definición heurísticas.

Como siempre para este tipo de métodos, existe un compromiso entre los costos de cómputo para obtener soluciones, y la calidad de estas soluciones con respecto al óptimo del problema.

Maes y Wassenhove dividen las distintas heurísticas que abordan esta problemática en dos grandes categorías: heurísticas de un único recurso (*single-resource heuristics*), y heurísticas basadas en programación matemática (*mathematical-programming-based heuristics*) [20].

En las siguientes secciones se describen las principales características, ventajas y desventajas de cada uno de estos enfoques, presentándose además heurísticas específicas de cada clase.

1.3.1 Heurísticas de un único recurso

Estas heurísticas son llamadas “de sentido común”, en el entendido que se rigen por reglas prácticas y no por la utilización de técnicas matemáticas en la búsqueda de óptimos. Las mismas intentan explotar aspectos operativos específicos del problema, como pueden ser propiedades de los lotes, de la demanda o de los inventarios. Como consecuencia la utilización de las mismas se reduce a un conjunto reducido de instancias del problema.

Este tipo de heurísticas a su vez se puede dividir en dos clases: heurísticas que construyen la solución período a período y heurísticas que parten de una solución buscando mejorar la misma paso a paso con un enfoque glotón (greedy).

1.3.1.1 Heurísticas de período por período

Este tipo de heurísticas recorren el problema desde el primer período hasta el último, construyendo una solución normalmente en una única recorrida (pudiendo eventualmente realizarse alguna iteración adicional para asegurar la factibilidad de la solución obtenida).

Para cada período de tiempo *t*, el procedimiento de construcción de la solución es el siguiente: para cumplir con la demanda de cada ítem *i* en dicho período, se deberá producir $\max\{0; d_t^i - s_{t-1}^i\}$, donde s_{t-1}^i representa el inventario del ítem *i* al final del período *t*-1. Si existe capacidad extra, ésta puede ser utilizada para producir demanda futura. De esta manera se incurre en nuevos costos de inventario, pero se genera una potencial reducción de los costos de setup en períodos futuros.

Para decidir cómo se utilizará esta capacidad ociosa, una alternativa es utilizar índices de prioridad. Dados un período y un ítem determinado, habrá un índice de prioridad para dicho ítem en cada uno de los futuros períodos.

Este índice permite estimar para cada período futuro cuál es el potencial ahorro de adelantar la producción hasta el período actual *t*, evaluando la diferencia entre la potencial reducción en costos de setup y la aparición de nuevos costos de inventario. Diversos trabajos han sido presentados para estimar dichos índices, donde nuevamente se establece el compromiso entre la calidad del índice obtenido y los cálculos requeridos para estimarlo.

Una vez hallados los índices de prioridad para cada ítem y para cada período futuro, se destina la capacidad ociosa en el período actual a la producción del ítem con mayor valor de índice, hasta que no quede más capacidad ociosa o hasta que no queden índices que reduzcan costos. Este procedimiento se repite avanzando en el horizonte de tiempo hasta que no queden más períodos por evaluar.

Es importante destacar que no existen garantías de que se encontrará una solución factible utilizando este método. Si sólo se consideran los índices que reducen costos, es posible avanzar hasta algún período donde la capacidad disponible no permita satisfacer la demanda. Existen diversos enfoques para intentar solucionar este problema:

- Un enfoque posible es utilizar algún mecanismo de feedback. En el momento en que no se puede satisfacer la demanda en un período *t*, la producción necesaria para satisfacerla se desplaza hacia períodos anteriores.
- Otro enfoque calcula previamente cuál es el valor de la demanda acumulada hasta cada período *t*. De esta manera se pueden considerar los índices con aumento de costos hasta asegurar la factibilidad de la solución en el período *t*+1.

El enfoque constructivo de este tipo de heurísticas tiene dos ventajas principales. En primer lugar los tiempos de resolución son cortos incluso para instancias del problema de tamaño considerable. Esto se debe a que normalmente se realiza una única recorrida para construir la solución. En segundo lugar, la construcción de la solución para los primeros períodos es poco influida por la demanda de los últimos. Esta es una propiedad deseable para los escenarios donde la estimación de la demanda futura tiene altos niveles de incertidumbre, y donde sólo las decisiones de los primeros períodos del modelo son consideradas para llevar a cabo.

Dentro de esta clase de heurísticas Eisenhut presenta un algoritmo que permite obtener soluciones similares a las del método de Wagner-Whitin pero con menores requerimientos de cómputo [10]. Este algoritmo fue posteriormente extendido por Lambrecht y Vanderveken [19], y luego por Dixon y Silver quienes desarrollan una heurística que garantiza, en caso de existir, la obtención de alguna solución factible [9].

1.3.1.2 Heurísticas de mejora de solución

Estas heurísticas parten de una solución que puede o no ser factible (este último caso generalmente debido a la relajación de alguna restricción), y luego intentan mejorar esa solución con un enfoque glotón (greedy). De este modo se analizan distintos movimientos en la vecindad de la solución actual, en busca de la factibilidad o en busca de reducción de costos. Estos movimientos no son más que ajustes sobre el esquema de producción definido en la solución actual, ya sea aumentando, reduciendo o desplazando los lotes para algún subconjunto de ítems y de períodos.

Este tipo de heurísticas comienza generando una solución inicial, normalmente relajando las restricciones de capacidad, y utilizando alguno de los métodos de resolución eficientes para el problema de dimensionado de lotes no capacitado.

Una vez obtenida, los siguientes pasos buscan o bien minimizar el valor de la función objetivo, o bien acercar la solución existente a una factible.

Supongamos por simplicidad costos de setup e inventario constantes en el tiempo. Una posible estrategia para estimar la mejora de los movimientos puede ser la siguiente:

Se tiene: $\delta c = s_i\beta + h_i(t - m)\, x_t^i$, donde:

- δc es la potencial diferencia de costos.
- β vale 1 si se agrega un setup en t, -1 si se quita, y 0 si no hay cambios.
- m es el período destino.
- x_t^i, es la cantidad de producción desplazada desde el período t hacia el período m.
- s_i, h_i son respectivamente los costos de activación de producción y costos de inventario.

La heurística deberá analizar entre los posibles movimientos antes de definir cuál aplicar, debiendo priorizar entre:

- factibilidad: dado que δc representará un incremento de costo, buscar el menor δc.
- mejorar la solución: δc representa ahorros en los costos, por lo que se buscará el mayor δc.

Notar que este análisis se deberá hacer para todos los movimientos, en todas las iteraciones de la heurística. De esta manera, a medida que los problemas a resolver aumentan en tamaño, los tiempos de cómputo pasan a limitar la utilización de este tipo de métodos.

Adicionalmente, a diferencia de las heurísticas que trabajan período a período, esta clase de métodos sí tiene en cuenta los costos y las demandas de todo el horizonte de períodos. Por lo tanto su utilización es más conveniente en escenarios con poca incertidumbre en los costos y en la demanda.

Finalmente se debe observar que los métodos descritos en esta sección no son excluyentes. Una posible heurística pudiera aplicar alguna técnica constructiva período a período, y luego acercar la solución obtenida al óptimo aplicando alguna técnica de mejora.

1.3.2 Heurísticas basadas en programación matemática

A diferencia de los métodos descritos en la sección anterior basados en el "sentido común", estas heurísticas se basan en la búsqueda de óptimos a partir de métodos de programación matemática clásica. De esta manera algunos de los métodos descritos son heurísticas en esencia, mientras que otros son métodos de resolución exacta normalmente con algún criterio de parada específico para minimizar los tiempos de cómputo.

Esta base en la programación matemática implica que los métodos descritos en esta sección sean menos intuitivos que los anteriores. Sin embargo su aplicación puede generalizarse a una gran cantidad de problemas.

1.3.2.1 Heurísticas basadas en relajación

Las heurísticas de este tipo normalmente parten relajando la restricción de capacidad. Como se mencionó anteriormente, para las versiones básicas del CLSP ésta es la única restricción que vincula los $|P|$ ítems. Como resultado se obtienen $|P|$ problemas de dimensionado de lotes no capacitado independientes. Para esta clase de problemas existen métodos de resolución eficientes.

Dentro de este tipo de métodos se encuentra el presentado por Newson [22]. Se comienza con una solución al problema no capacitado generada con el algoritmo de Wagner-Whitin (el cual se describe en detalle en la sección 4.3.3). Posteriormente, para cada período donde se viola la restricción de capacidad, se fuerza a cero la producción para cada uno de los ítems activados en este período, y se vuelve a generar una nueva solución con Wagner-Whitin para estos ítems. De los posibles planes que se generan, se selecciona el que agregue un menor costo a la solución inicial.

La heurística anterior tiene dos importantes desventajas. En primer lugar no garantiza encontrar soluciones factibles. En segundo lugar, únicamente se generan soluciones con la forma de Wagner-Whitin, la cual coincide con la solución óptima sólo bajo ciertas hipótesis.

Este último problema fue abordado por Thizy y Van Wassenhove [28], quienes aplican una relajación de Lagrange para la restricción de capacidad. En su algoritmo, los multiplicadores de Lagrange son actualizados a través de la optimización del sub-gradiente. En cada iteración del método de sub-gradiente se resuelve con Wagner-Whitin una instancia independiente por cada ítem al problema no capacitado, donde los multiplicadores de Lagrange actúan como costos. Estas soluciones determinan una cota inferior al problema original. Adicionalmente los setups establecidos por estas soluciones se fijan y se calcula a partir de los mismos una nueva solución factible, que será cota superior al problema original. Estos pasos se pueden repetir hasta igualar las dos cotas o hasta alcanzar algún criterio de parada.

1.3.2.2 Procedimientos de ramificado y acotamiento (branch-and-bound)

Este es en general el método por excelencia para resolver de forma exacta problemas de programación entera. Para simplificar la descripción que se presenta a continuación, asumiremos que las únicas variables enteras en nuestro problema son las variables de activación de producción.

Se comienza relajando la restricción de integridad de las variables binarias, transformando el problema original en un problema de programación lineal. Este nuevo problema relajado se resuelve a partir de algún método de resolución eficiente para programación lineal (generalmente simplex). Se analiza la solución óptima obtenida al problema relajado; si la misma satisface las restricciones de integridad, es entonces solución óptima al problema original. De lo contrario se selecciona alguna de las variables enteras que violan la restricción de integridad en la solución al problema relajado y se divide (branch) el problema original en dos nuevos sub-problemas: uno donde se fuerza a 1 la variable seleccionada, y otro donde se fuerza a 0. De esta manera se genera en sucesivos pasos la estructura arborescente que da nombre al método.

Se vuelven a repetir los pasos para cada uno de los sub-problemas hasta que se llega a alguno de los siguientes tres criterios de parada: a) la solución óptima del sub-problema relajado cumple con las restricciones de integridad de las variables enteras, por lo tanto es solución óptima al sub-problema original, b) la solución del sub-problema relajado no es mejor que la mejor solución factible obtenida hasta al momento para alguna de las ramas, c) el sub-problema relajado no tiene solución factible.

De esta manera *branch-and-bound* es en esencia un método de resolución exacta. El tamaño del árbol de sub-problemas generado crece exponencialmente a medida que aumenta el tamaño del problema original. Existen diversas estrategias para evitar este crecimiento exponencial en los costos de cómputo. En primer lugar se puede ajustar el segundo (b) criterio de parada presentado, de manera que se descarte una rama si la solución al sub-problema relajado no mejora (por ejemplo) en un 10% a la mejor solución factible encontrada hasta el momento. Otra posible técnica es ponderar aquellas ramas con mejores soluciones al problema relajado, podando con anterioridad aquellas ramas menos promisorias. Por último, se puede acotar la cantidad de ramas activas, obligando al algoritmo a podar ramas de modo de favorecer las búsquedas en profundidad.

Dentro de estas heurísticas se destaca el método de *branch-and-price*, el cual combina la estrategia básica de ramificado y acotamiento, anexando generación de columnas al problema relajado permitiendo encontrar mejores cotas en cada iteración. Zager Degraeve y Raf Jans presentan un estudio detallado de este tipo de heurística [7].

1.3.2.3 Heurísticas basadas en programación lineal

Al igual que los procedimiento de ramificado y acotamiento, estos métodos parten de una solución óptima al problema relajado a programación lineal. En lugar de continuar iterando como los métodos anteriores, estas heurísticas buscan ajustar la solución obtenida a una solución factible al problema original. Esto puede realizarse de distintas formas. Algunas de las más simples realizan el ajuste "redondeando" todas las variables de la solución al entero más cercano. Otros métodos fijan en 1 únicamente los valores no enteros más altos, y vuelven a resolver este nuevo problema relajado a programación lineal. Este procedimiento se repite hasta encontrar alguna solución factible. Por último, algunas heurísticas combinan la relajación a programación lineal con el ramificado y acotamiento. Esto se realiza fijando los valores para todas las variables con valores enteras al problema relajado, y aplicando *branch-and-bound* sobre este nuevo sub-problema. Como siempre, existe un compromiso entre la reducción de tiempos que aporta la relajación a programación lineal, y la calidad en las soluciones obtenidas que generan los métodos de ramificado.

1.3.2.4 Otros métodos basados en programación matemática

Diversos trabajos que abordan la problemática del CLSP desde el punto de visto matemático han sido presentados. Además de los métodos descritos anteriormente, existen otras alternativas basadas en otras propiedades como la generación de columnas y dualidad del problema.

No todos estos métodos son generales como los presentados anteriormente, algunos se basan en características específicas de las instancias del problema y no necesariamente pueden ser aplicadas a cualquier problema de planificación.

Uno de los métodos pioneros sobre el cual se continúan desarrollando trabajos al día de hoy es la descomposición de Dantzig–Wolfe, presentada por George Dantzig y Philip Wolfe en 1960 [6]. La aplicación de este método requiere que el problema cumpla con ciertas propiedades. En particular, es necesario que la matriz de restricciones pueda descomponerse en un conjunto de sub-matrices que liguen de manera disjunta las distintas variables del problema.

Capítulo 2

Caso de estudio

En este capítulo se presenta un problema de planificación de producción particular. A partir de las distintas variantes discutidas en el capítulo anterior se elabora un modelo algebraico basado en programación entera.

2.1. Descripción del problema

El presente problema trata la planificación de un proceso productivo de alimentos por lotes para atender la demanda durante un horizonte de tiempo discretizado en períodos. El proceso productivo consiste de las etapas de *preparación* de ingredientes, *elaboración* de productos a partir de la mezcla de los ingredientes, y *envasado* de los productos. Todas las etapas tienen lugar en todos los períodos y ninguna instancia de una etapa se extiende más allá del período en que tiene lugar.

Para las etapas de preparación y envasado existe equipamiento independiente para cada ingrediente y producto respectivamente. Por otro lado, para la etapa de elaboración se comparte equipamiento que debe restablecerse (limpiarse, calibrarse, etc.) cada vez que se cambie de producto a elaborar.

Mientras que la etapa de preparación es insignificante en el proceso, las etapas de elaboración y envasado son relevantes en el uso del equipamiento; en particular la de elaboración, cuyo equipamiento es compartido por todos los productos y debe restablecerse al intercambiarse la producción de estos. En cada período y para cada producto existe una única instancia de elaboración y envasado.

La utilización del equipamiento se mide en unidades de tiempo y se cuenta con tasas de tiempo consumido por unidad de cada producto elaborado y envasado. Además se cuenta con el tiempo consumido para restablecer y configurar el equipamiento de elaboración para cada producto. Por otra parte, para cada equipo se dispone de un tiempo total de capacidad de operación.

Se deben mantener inventarios mínimos de productos para cada período y se parte de un inventario inicial conocido.

Se cuenta con los costos de elaboración, almacenamiento y envasado de cada producto por período, y con el costo de restablecimiento del equipamiento de elaboración por producto.

El objetivo es minimizar los costos mientras se atiende la demanda y se cumplen las restricciones de disponibilidad de equipamiento.

2.2. Modelo matemático

A partir de la realidad descrita en el punto anterior, podemos comenzar identificando un conjunto *P* de ítems a ser producidos a lo largo de *T* períodos. Los distintos ítems pueden pertenecer a alguna de las *C* clases de productos, donde cada una de estas clases de productos se envasan en una máquina diferente.

Una vez introducidos los principales parámetros del problema, podemos representar el mismo con el siguiente modelo matemático:

$$\min \sum_{i \in P} \sum_{t \in T} (vc_t^i + ec_t^i + pc_t^i) x_t^i + sc_t^i y_t^i + hc_t^i s_t^i \tag{1}$$

$$s.a. \quad s_{t-1}^i + x_t^i = d_t^i + s_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{2}$$

$$x_t^i \leq DS_t^i y_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{3}$$

$$\sum_{i \in P} (y_t^i gt^i + x_t^i et^i) \leq cape_t, \qquad \forall t \in T \tag{4}$$

$$\sum_{i \in j} x_t^i pt^i \leq capp_t^j, \qquad \forall j \in C, \quad \forall t \in T \tag{5}$$

$$s_t^i \geq ms_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{6}$$

$$s_0^i = is^i, \qquad \forall i \in P \tag{7}$$

$$x_t^i, s_t^i \in \mathbb{N}, \; y_t^i \in \{0;1\}, \qquad \forall i \in P, \quad \forall t \in T \tag{8}$$

Al igual que para la versión básica del CLSP se tiene tres variables para cada ítem *i* y para cada período *t*: la cantidad de producción x_t^i o tamaño de lote; la activación de producción y_t^i; y el total de inventario al final del período s_t^i. Para cada una de estas variables existe un costo asociado: $vc_t^i + ec_t^i + pc_t^i$ representa la suma de los costos unitarios de elaboración, producción y envasado; sc_t^i representa el costo de configuración (setup); mientras que hc_t^i el costo unitario de inventario. Nuevamente la demanda para cada ítem y para cada período d_t^i es conocida, así como la demanda más el inventario mínimo acumulados desde *t* hasta el último período DS_t^i. Existe una cierta capacidad máxima de producción $cape_t$ para cada período, donde cada unidad producida y cada activación de producción consumen respectivamente et^i y gt^i unidades de capacidad. De la misma manera, existe una cierta capacidad máxima de envasado $capp_t^j$ para cada período y para cada tipo de producto, donde cada unidad envasada consume pt^i unidades de capacidad. Por último existen cantidades mínimas de inventario requerido ms_t^i para cada ítem al final de cada período, contando además al inicio del horizonte de tiempo con un cierto inventario is^i para cada ítem.

El objetivo del problema es minimizar el costo total de elaboración, producción, envasado, configuración e inventario (1). Al igual que para el problema de *Cantidad Económica de Pedido*, existe un compromiso entre los costos de setup y los costos de inventario. La demanda puede ser cumplida con producción del período actual o con inventario arrastrado desde períodos anteriores (2). Cualquier exceso es llevado al siguiente período como inventario. Eventualmente puede requerirse cierto inventario al final del horizonte de tiempo, por lo que la producción en un período se ve limitada tanto por la demanda como por el inventario mínimo restantes acumulados. Esto permite obtener una cota para la restricción de activación de producción (3). Existe además una capacidad de producción limitada por período (4), y una capacidad de envasado para cada tipo de producto *j* (5). Para cada ítem se requiere una cantidad mínima de inventario al final de cada período (6), partiendo con un cierto inventario inicial para el primer período (7). Por último, tanto la cantidad de producción como el inventario al final de cada período deben ser no negativos, y la activación de producción por definición es binaria (8).

Capítulo 3

Resolución mediante metodología exacta

En una primera instancia se intenta resolver el problema presentado en el capítulo anterior aplicando una metodología de resolución exacta. Para esto se selecciona la herramienta GLPK, comúnmente utilizada en el ámbito académico.

Se comienza este capítulo presentando brevemente la herramienta seleccionada. Se identifican las principales funcionalidades, ventajas y desventajas asociadas a la misma. Posteriormente se analizan algunas alternativas para mejorar el desempeño de esta herramienta en la resolución del problema estudiado. Finalmente se concluye el capítulo presentando brevemente los principales resultados obtenidos del estudio realizado. Estos resultados se complementan con los presentados en el Capítulo 5 como parte del análisis de escenarios.

3.1. Introducción a la herramienta GLPK

El paquete GNU Linear Programming Kit [2] es un conjunto de rutinas desarrolladas en ANSI C y organizadas en forma de librería. Su objetivo principal es resolver problemas de programación lineal (LP) y programación entera mixta (MIP) de gran escala.

Los problemas pueden ser modelados utilizando el lenguaje GNU MathProg, que guarda una gran similitud en sintaxis con la notación matemática estándar de los problemas de optimización.

Los principales componentes del paquete GLPK son los siguientes:

- Método simplex primal y dual.
- Método del punto interior primal y dual.
- Métodos de ramificado y acotamiento.
- Traductor para GNU MathProg.
- API.
- Motor de resolución LP/MIP (solver).

Programación lineal (LP): GLPK contiene un completo juego de algoritmos para resolver problemas LP, que abarca el simplex dual, el simplex primal, y métodos de punto interior. El solver de simplex tiene su propio pre-procesamiento, el cual permite simplificar el problema LP original:

- Procedimientos para eliminar variables y restricciones redundantes.
- Fijación de variables no básicas y búsqueda de cotas de variable.
- Mejoras en las propiedades numéricas del modelo.

Programación entera (MIP): GLPK implementa un algoritmo de ramificado y acotamiento para resolver problemas con variables enteras. En las últimas versiones se agregan mejoras al optimizador entero, permitiendo resolver modelos complejos y de gran tamaño. Algunas de estas mejoras son:

- Algoritmo de pre-procesamiento MIP, que ayuda a disminuir el tamaño del problema y reducir los tiempos de resolución.
- Utilización de algoritmos de planos de corte.
- Estrategias de selección de nodos y variables.

Salidas de la herramienta: Para cada modelo que se resuelve la herramienta genera un reporte conteniendo el valor de la mejor solución factible, el detalle de dicha solución, así como el detalle de la activación para cada una de las restricciones. El reporte contiene además información acerca del tamaño del problema generado: cantidad de filas, cantidad de columnas, así como cantidad de valores no nulos en la matriz del problema.

La resolución del problema puede insumir varias horas. Para estos casos la herramienta reporta en la terminal información sobre la mejor solución encontrada hasta el momento. Esta información incluye el número de iteración del simplex utilizado en la relajación de cada sub-problema, el valor de la función objetivo para la mejor solución factible encontrada hasta el momento, la mejor cota global para la solución óptima, el gap relativo entre la solución actual y la mejor cota, el número de sub-problemas activos, y el número de sub-problemas que han sido podados.

3.2. Reformulación del modelo

En una primera instancia se implementa en la herramienta de resolución exacta la formulación original del modelo presentado en el capítulo anterior. A partir de esta implementación se estudia el desempeño de la herramienta GLPK para diversas instancias del problema. En particular se analizan los tiempos requeridos por la herramienta para la obtención de soluciones factibles y de soluciones óptimas. Los resultados del estudio se detallan en la Figura 3.1 y en la Figura 3.2.

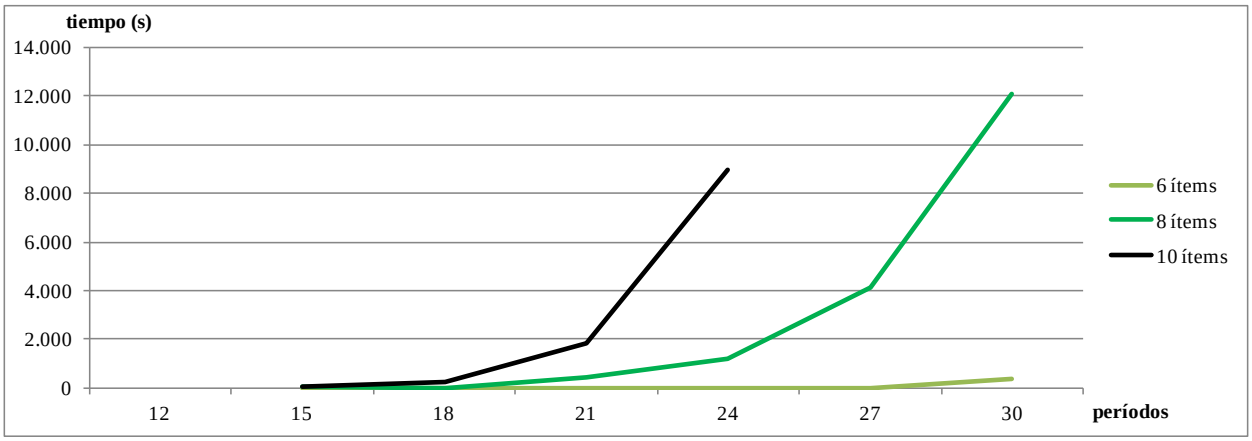

Figura 3.1 – Evolución de tiempos de ejecución insumidos para la obtención de soluciones factibles.

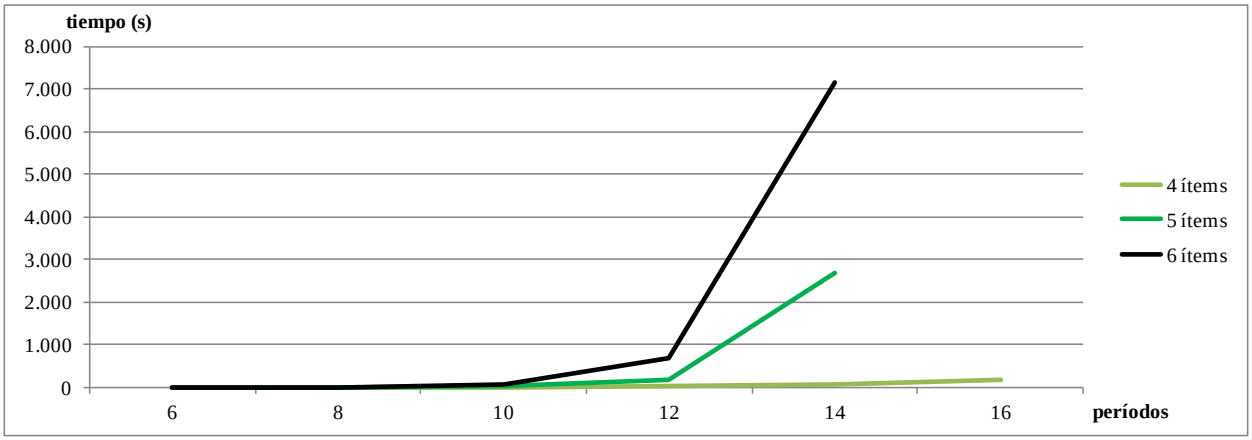

Figura 3.2 – Evolución de tiempos de ejecución insumidos para la obtención de soluciones óptimas.

Se observa que para problemas de tamaño mediano y grande se requiere una gran capacidad de cómputo para poder encontrar soluciones factibles y más aún para obtener soluciones óptimas. Esta dificultad en la obtención de soluciones con la herramienta de resolución exacta para esta clase de problemas implica la necesidad de analizar otras opciones. Para esto se plantea como alternativa ajustar el modelo original del problema en busca de reformulaciones con otras características.

3.2.1. Primer reformulación: aproximación al casco convexo

Basados en la bibliografía estudiada, la primera reformulación presentada busca acercar el espacio de soluciones al casco convexo del problema a través de la incorporación de nuevas restricciones. Con esta aproximación al casco convexo se espera mejorar el desempeño de los métodos de planos de corte implementados por la herramienta GLPK.

Este enfoque es utilizado por diversos autores [17][34] para problemas similares a los presentados en las secciones 1.2.2 y 1.2.3 donde no se cuenta con restricciones de inventario mínimo:

$$\min \sum_{i \in P} \sum_{t \in T} sc_t^i y_t^i + hc_t^i s_t^i \tag{1}$$

$$s.a. \quad s_{t-1}^i + x_t^i = d_t^i + s_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{2}$$

$$\sum_{i \in P} et^i x_t^i \leq cape_t, \qquad \forall t \in T \tag{3}$$

$$x_t^i \leq D_t^i y_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{4}$$

$$x_t^i, s_t^i \in \mathbb{N},\ y_t^i \in \{0;1\}, \qquad \forall i \in P, \quad \forall t \in T \tag{5}$$

Para entender esta estrategia se debe prestar atención a la restricción de activación de producción (4). Como fue mencionado anteriormente, para este tipo de problemas el tamaño del lote en un período determinado está acotado por la demanda acumulada hasta el último período (D_t^i). Esta cota es especialmente holgada para los primeros períodos, siendo habitualmente redundante si se consideran además restricciones de capacidad.

En busca de ajustar dicha cota, se comienza redefiniendo la variable de lote x_t^i, por una nueva variable w_{jt}^i representando la cantidad de producción del ítem *i* en el período *j* para satisfacer parcial o totalmente la demanda del período *t*. Esta redefinición de la variable de lote permite especificar más detalladamente el esquema de producción óptimo indicando no sólo dónde se produce sino también cuándo será consumida dicha producción. Sabiendo dónde será consumida la producción de determinado período, se puede ajustar la restricción de activación de producción de la siguiente manera:

$$w_{jt}^i \leq d_t^i y_j^i, \qquad \forall i \in P, \quad \forall j,t \in T$$

Donde la producción utilizada para satisfacer la demanda de un cierto período *t* está acotada únicamente por la demanda en *t* y no por la demanda acumulada hasta el último período.

Debido a la ausencia de restricciones de inventarios mínimos se cumple que toda producción es consumida en alguno de los períodos. Esto permite prescindir de la variable de inventario s_t^i y reformular el problema con el siguiente modelo matemático:

$$\min \sum_{i \in P} \sum_{t \in T} sc_t^i y_t^i + hc_t^i \sum_{\substack{j \in T \\ j \le t}} \sum_{\substack{k \in T \\ k > t}} w_{jk}^i \tag{1}$$

$$s.a. \quad \sum_{\substack{j \in T \\ j \le t}} w_{jt}^i = d_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{2}$$

$$\sum_{i \in P} \sum_{\substack{j \in T \\ j \ge t}} et^i w_{tj}^i \le cape_t, \quad \forall t \in T \tag{3}$$

$$w_{jt}^i \le d_t^i y_j^i, \qquad \forall i \in P, \quad \forall j,t \in T \tag{4}$$

$$w_{jt}^i \in \mathbb{N}, \; y_t^i \in \{0;1\}, \quad \forall i \in P, \quad \forall j,t \in T \tag{5}$$

El inconveniente de la holgura en la restricción de activación de producción se agudiza si se consideran restricciones de inventarios mínimos. Para estos casos el tamaño de lote para un período determinado se ve acotado por la suma de la demanda más el inventario mínimo acumulados hasta el último período (DS_t^i).

Como fue mencionado en la sección 1.2.6, no es posible establecer a priori una expresión simple para estimar la cantidad de producción necesaria para satisfacer la restricción de inventario mínimo. Por lo tanto no es de esperarse conseguir mejoras significativas de la cota definida anteriormente.

La inclusión de restricciones de inventarios mínimos presenta un nuevo desafío en la reformulación, ya que a diferencia del problema anterior, no toda la producción será consumida en algún período. Para abordar este tema, una primera alternativa consiste en ajustar la definición de la variable w_{jt}^i permitiendo representar la producción en *j* destinada a satisfacer la demanda o la restricción de inventario mínimo en el período *t*. Esta nueva definición supone un número mayor de inconvenientes que no serán abordados en el presente trabajo. A modo de ejemplo, se debe considerar que la producción no consumida es arrastrada a períodos posteriores, por lo tanto aquella fracción de w_{jt}^i destinada a satisfacer la restricción de inventario mínimo en *t* también lo hará en los períodos intermedios. En estos casos, si no se consideran las restricciones de balance entre períodos, el problema puede converger en soluciones donde la producción destinada a satisfacer el inventario mínimo de cierto período se descarta luego para minimizar los costos. Este último punto, si bien se presenta en la práctica, no es contemplado por el caso de estudio y es una de las posibles líneas de trabajos a futuro que se discuten en el Capítulo 7.

Debido a lo anterior, la reformulación presentada en este trabajo no prescinde de la variable de inventario ni de las restricciones de balance (2):

$$\min \sum_{i\in P}\sum_{t\in T} sc_t^i y_t^i + \left(vc_t^i + ec_t^i + pc_t^i\right)\sum_{\substack{j\in T\\ j\geq t}} w_{tj}^i + hc_t^i s_t^i \qquad (1)$$

$$s.a. \quad s_{t-1}^i + \sum_{\substack{j\in T\\ j\geq t}} w_{tj}^i = d_t^i + s_t^i, \qquad \forall i \in P, \quad \forall t \in T \qquad (2)$$

$$w_{jt}^i \leq (d_t^i + ms_t^i) y_j^i, \qquad \forall i \in P, \quad \forall j,t \in T \qquad (3)$$

$$\sum_{i\in P}(y_t^i gt^i + \sum_{j\in T} w_{tj}^i et^i) \leq cape_t, \quad \forall t \in T \qquad (4)$$

$$\sum_{i\in j}\sum_{k\in T} w_{tk}^i pt^i \leq capp_t^j, \qquad \forall j \in C, \quad \forall t \in T \qquad (5)$$

$$s_t^i \geq ms_t^i, \qquad \forall i \in P, \quad \forall t \in T \qquad (6)$$

$$\sum_{j\in T} w_{0j}^i = is^i, \qquad \forall i \in P \qquad (7)$$

$$w_{jt}^i, s_t^i \in \mathbb{N},\ y_t^i \in \{0;1\}, \qquad \forall i \in P, \quad \forall j,t \in T \qquad (8)$$

Donde la definición de los distintos parámetros (tiempos, costos, demanda, tasas) se mantiene con respecto al modelo original presentado en la sección 2.2.

Como se observa, se debe generar una nueva expresión a partir de w_{tj}^i para obtener una formulación equivalente del tamaño del lote en cada período *t* de manera de reemplazar la variable x_t^i:

$$x_t^i = \sum_{\substack{j\in T\\ j\geq t}} w_{tj}^i \qquad \forall i \in P, \quad \forall t \in T$$

El tamaño de los distintos lotes a ser consumidos en el período *t* está acotado por la demanda de dicho período (3). El resto de las restricciones son análogas a la formulación original (4 - 8).

3.2.2. Segunda reformulación: reducción del tamaño del problema

La segunda reformulación realizada es conceptualmente más sencilla. La misma busca reducir el tamaño del problema prescindiendo de la variable s_t^i de inventario al final de cada período. Si bien el modelo resultante es menos legible, se espera que esta reducción del número de variables disminuya los tiempos de ejecución de los métodos de ramificado y acotamiento implementados por la herramienta GLPK.

Al desechar la variable s_t^i, debemos obtener una expresión equivalente para representar el inventario al final de cada período:

$$s_t^i = is^i + \sum_{\substack{j \in T \\ j \leq t}} (x_j^i - d_j^i), \ \forall i \in P, \quad \forall t \in T$$

Donde para cada ítem, el inventario al final de cada período es igual a la producción acumulada menos la demanda acumulada.

Como resultado la cantidad de variables disminuye desde $3|P||T|$ del modelo original, a $2|P||T|$ del nuevo modelo:

$$\min \sum_{i \in P} \sum_{t \in T} \left(sc_t^i y_t^i + \left(vc_t^i + ec_t^i + pc_t^i \right) x_t^i + hc_t^i \left(is^i + \sum_{\substack{j \in T \\ j \leq t}} (x_j^i - d_j^i) \right) \right) \tag{1}$$

$$s.a. \quad x_t^i \leq DS_t^i y_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{2}$$

$$\sum_{i \in P} (y_t^i gt^i + x_t^i et^i) \leq cape_t, \qquad \forall t \in T \tag{3}$$

$$\sum_{i \in j} x_t^i pt^i \leq capp_t^j, \qquad \forall j \in C, \quad \forall t \in T \tag{4}$$

$$is^i + \sum_{\substack{j \in T \\ j \leq t}} (x_j^i - d_j^i) \geq ms_t^i, \qquad \forall i \in P, \quad \forall t \in T \tag{5}$$

$$x_t^i \in \mathbb{N}, \quad y_t^i \in \{0;1\}, \qquad \forall i \in P, \quad \forall t \in T \tag{6}$$

Donde la definición de los distintos parámetros (tiempos, costos, demanda, tasas) se mantiene con respecto al modelo original presentado en la sección 2.2.

Se debe notar cómo la producción es explícitamente consumida en la función objetivo (1) y en la restricción de inventarios mínimos (5). Esto permite además prescindir de la restricción de balance.

3.3. Implementación computacional

La implementación en GLPK tanto de la formulación original presentada en el punto 2.2, como de las reformulaciones del modelo incluidas en la sección 3.2 se detallan en los anexos A, B y C.

Es importante destacar que las mismas se realizaron para instancias del problema con una cantidad fija de tipos de ítems. Se debe tener en cuenta que contemplar una cantidad variable de tipos de ítems implica una cantidad variable de restricciones en la capacidad de envasado. Incluir este punto dificulta sensiblemente la implementación tanto del modelo en GLPK como la de la heurística desarrollada que se presenta en el siguiente capítulo. En particular para la implementación de esta última, se deben desarrollar y mantener estructuras de datos más elaboradas, lo cual agrega complejidad adicional a la heurística.

Adicionalmente a los aspectos asociados a la implementación del modelo, contemplar una cantidad variable de tipos de productos dificulta la obtención de datos necesaria para el estudio de las diferentes estrategias de resolución. Como se verá más adelante, ninguno de los modelos incluidos en la bibliografía estudiada contemplan esta característica. Por tal motivo las instancias del problema sobre las cuales se realiza el estudio computacional deben ser generadas especialmente para el mismo.

3.4. Comparación de formulaciones

Fue analizado el desempeño de la herramienta de resolución exacta para diferentes instancias del problema para cada uno de los modelos presentados anteriormente. Para las instancias del problema donde se obtuvieron soluciones óptimas, el análisis del desempeño se focaliza en los tiempos de ejecución insumidos. En estos escenarios se descarta rápidamente la primer reformulación del modelo, para la cual los tiempos de ejecución duplican y en algunos casos triplican los tiempos requeridos por las otras formulaciones. De esta manera, se observa cómo los costos de cómputo incurridos superan las mejoras producidas por intentar acercar el espacio de soluciones hacia el casco convexo del problema. Esta diferencia se vio acentuada a medida que aumenta el tamaño del problema.

La diferencia en los tiempos de ejecución son menores entre el modelo original y la segunda reformulación. De todas formas luego de experimentar con casi 5.000 instancias del problema, se obtuvo una reducción de aproximadamente un 20% en los tiempos de ejecución con la segunda reformulación sobre el problema original.

Por su parte para las instancias de mayor tamaño no se pudo obtener una solución óptima a través de la herramienta de resolución exacta. Para estos casos se fija el tiempo de ejecución y se presta especial atención a la mejor solución factible (cota superior), a la mejor solución al problema relajado (cota inferior), así como a la cantidad de instrucciones ejecutadas.

Nuevamente la primer reformulación del modelo fue rápidamente descartada. Para esta reformulación se observaron importantes dificultados en la obtención de soluciones factibles dentro de los tiempos fijados. No se encontraron diferencias significativas entre las cotas obtenidas por la formulación original y la segunda reformulación. Si bien tanto la cota superior (mejor solución factible) como la cota inferior

(mejor solución al problema relajado) obtenidas por la segunda reformulación son mejores que las obtenidas con la formulación original, esta diferencia no supera el 3%. La principal diferencia entre estas dos formulaciones se encuentra a la hora de analizar la cantidad de instrucciones ejecutadas. Como era de esperar, un modelo más compacto (menor número de variables) implica menores costos de cómputo. De esta manera la cantidad de instrucciones ejecutadas para la formulación original, en promedio duplica las ejecutadas para la segunda reformulación. Similares son los resultados si se observa la memoria consumida por la herramienta dentro de los tiempos fijados.

A partir de los resultados anteriores, se selecciona la segunda reformulación al modelo original como base para la obtención de resultados a partir de una herramienta de resolución exacta.

Capítulo 4

Resolución mediante metodología heurística

Luego de observar las ventajas y limitaciones asociadas a una herramienta de resolución exacta, en este capítulo se aborda el problema de planificación con una metodología de resolución heurística. En el mismo se presentan los detalles de la selección y diseño de una aproximación heurística, así como los distintos aspectos de la implementación computacional.

4.1. Selección de la heurística

Como se mencionó en el Capítulo 1, se estudiaron diferentes trabajos que tratan diversas heurísticas para resolver la problemática del CLSP. Estas heurísticas pueden dividirse en dos grandes categorías: heurísticas matemáticas y heurísticas computacionales.

Ninguna de las heurísticas incluidas en la bibliografía relevada contempla todas las particularidades del problema presentado en la sección anterior. En particular existen tres puntos ausentes en todos los trabajos estudiados. El primero de estos puntos es la restricción de inventario mínimo. Si bien esta restricción es fácil de modelar, tiene un importante impacto en los mecanismos de resolución heurística, en especial para las heurísticas matemáticas. El segundo punto corresponde al manejo de múltiples restricciones de capacidad, como son en el problema tratado las restricciones de capacidad de producción y de envasado. Esta restricción incluye el último de los tres puntos, el cual refiere a contemplar distintas clases de productos.

Nos encontramos entonces ante dos posibles escenarios: realizar el desarrollo de una nueva heurística específica para resolver el problema presentado en este trabajo o ajustar alguna de las heurísticas estudiadas para contemplar las características específicas del problema. En este contexto se opta por la segunda alternativa en busca de aplicar métodos y estrategias ya estudiadas con conocimiento de sus ventajas y limitaciones. Se debe tener en cuenta además que como objetivo de este trabajo se planteó el análisis de distintos escenarios y su impacto en los métodos de resolución del problema, y no la implementación de un procedimiento de resolución innovador.

Se deben entonces analizar las heurísticas estudiadas y realizar los ajustes necesarios para incluir aquellos puntos no contemplados. La dificultad de realizar estos ajustes para las heurísticas matemáticas dirige la búsqueda hacia una heurística computacional. Como se comentó en puntos anteriores, estos métodos son significativamente más intuitivos, lo cual simplifica el análisis del impacto de cualquier ajuste que se realice a los mismos.

Siguiendo esta línea se opta por seleccionar una meta-heurística: método de aplicación general no acoplado a ningún tipo de problema particular. De esta manera el presente trabajo se basa en la heurística propuesta por Gopalkrishnan en el artículo "*A tabu-search heuristic for the capacitated lot-sizing problem with set-up carryover*" publicado en 2001 [15].

Como indica el nombre del artículo, originalmente la heurística fue diseñada para resolver problemas de dimensionado de lotes capacitado con traslado de configuración entre períodos (setup carry over). Debido a esto, no solamente se deben considerar los puntos no contemplados por la heurística propuesta, sino que también es necesario desestimar algunos puntos que sí contempla.

A modo de resumen, los ajustes a realizar sobre la heurística presentada por Gopalkrishnan son los siguientes:

No contemplar traslado de la configuración entre períodos. Se deben ignorar las decisiones de secuenciado propias del carry over, lo cual significa principalmente ajustar algunos de los movimientos presentados por Gopalkrishnan.

Considerar múltiples restricciones de capacidad. Además de las restricciones de capacidad de producción, se deben contemplar restricciones de capacidad de envasado, las que deben definirse para los distintos tipos de ítems.

Considerar restricciones de inventarios mínimos. Dado que los métodos utilizados generalmente arrojan soluciones que violan esta restricción, la principal dificultad se encuentra al momento de generar y gestionar las soluciones por las que avanza la heurística.

Considerar inventario inicial. Este ajuste no implica mayores desafíos. En el Capítulo 5 se presenta un estudio específico del impacto de este valor en los problemas con restricciones de capacidad altamente ajustadas.

Además de los ajustes anteriores, los cuales son necesarios para contemplar la realidad del problema estudiado, se evalúa la posibilidad de realizar algunas modificaciones adicionales.

Movimientos implementados. Se analiza la posibilidad de implementar nuevos movimientos para aplicar sobre las soluciones intermedias. Este estudio se detalla en la sección 4.4.

Determinación de parámetros. Debido a los ajustes efectuados sobre la heurística para contemplar las particularidades del problema, se realizaron nuevamente los estudios necesarios para definir los valores de los distintos parámetros con mejor desempeño. Este estudio se detalla en las secciones 4.5.2, 4.5.4, 4.5.5 y 4.6.1.

4.2. Descripción de la heurística

En las siguientes secciones se analizan en detalle los principales componentes de la heurística implementada (CLSP-TS), los cuales se ilustran en la Figura 4.1. Estos componentes son:

Generador de soluciones iniciales: el cual provee seis diferentes soluciones al problema no capacitado buscando cubrir de forma eficiente el espacio de soluciones al problema.

Procedimiento de búsqueda tabú básica: que mejora las soluciones iniciales a partir de la aplicación de los diferentes movimientos definidos.

Procedimiento de diversificación e intensificación: donde se trabaja sobre las soluciones mejoradas a partir de la búsqueda tabú básica. Esta optimización se realiza combinando en forma probabilística las soluciones encontradas y mejorando las mismas a partir de nuevas búsquedas tabú.

Figura 4.1 – Principales componentes de heurística implementada (CLSP-TS).

4.3. Soluciones iniciales

La heurística implementada parte de seis soluciones iniciales y avanza a través de la ejecución de los movimientos definidos en la sección 4.4. Dichas soluciones buscan "cubrir" de la mejor manera posible el espacio de soluciones factibles al problema relajado (no capacitado), aplicándose en cada una un esquema de producción distinto.

Estas soluciones iniciales se mejoran luego a partir de una búsqueda tabú básica para posteriormente combinarse probabilísticamente en la fase de diversificación e intensificación. Este procedimiento permite maximizar de forma eficiente el cubrimiento del espacio de soluciones factibles.

Los esquemas de producción a partir de los cuales se generan las soluciones iniciales son los siguientes:

- Lote por lote
- División del horizonte de planificación en dos sub-horizontes
- División del horizonte de planificación en cuatro sub-horizontes
- División del horizonte de planificación activando la producción cada dos períodos
- División del horizonte de planificación activando la producción cada cuatro períodos
- Método de Wagner-Whitin

La forma de construcción de cada solución inicial puede resumirse en cuatros grandes pasos:

- Se relaja la restricción de capacidad.
- Se satisface totalmente la demanda para cada período de acuerdo a los criterios de producción descritos anteriormente.
- Se ajusta la producción del primer período a partir de la utilización del inventario inicial.
- Se ajusta la producción para cumplir con las restricciones de inventario mínimo al final de cada período. Para esto se avanza desde el período inicial, incrementando la producción de cada período que viola la restricción de inventario mínimo. Esta producción adicional, o bien se mantiene como inventario hasta el final, o bien puede ser utilizada para satisfacer la demanda en algún período posterior. De cumplirse esto último es posible reducir la producción calculada originalmente y satisfacer la demanda a partir de este nuevo inventario generado.

Debido a la forma en que son construidas, las seis soluciones iniciales listadas anteriormente comparten una importante característica: la cantidad total de producción coincide para todas las soluciones iniciales. Para probar esto olvidemos por un momento la restricción de inventario mínimo. En este contexto es fácil demostrar que el total de producción óptimo coincide con la demanda acumulada de cada ítem (de lo contrario, o bien no estaríamos cumpliendo con la demanda, o bien estaríamos produciendo en exceso). Resta analizar el ajuste para cumplir con la restricción de inventario mínimo. Como se mencionó anteriormente debemos tener en cuenta que cada unidad extra producida, o bien se mantiene como inventario hasta el último período, o bien se utiliza para satisfacer la demanda en un período posterior (permitiendo reducir la producción original para satisfacer la demanda en este nuevo período). De esta manera el total de producción para cada ítem es el mismo para todas las soluciones iniciales, y coincide con la demanda acumulada más el inventario mínimo del último período.

Si tenemos en cuenta que las instancias del problema estudiado cumplen con la hipótesis planteada en la sección 1.2.6, el total anterior es además el total óptimo de producción, es decir, coincide con el total de producción de la solución óptima. Esto se demuestra fácilmente por absurdo. En primer lugar sabemos que el total de producción hallado es el mínimo necesario para cumplir con las restricciones de demanda e inventario mínimo (cualquier reducción del mismo implica no cumplir con la demanda en algún período, o no cumplir con el inventario mínimo del último período). De manera similar cualquier solución donde la cantidad de producción supere la cantidad óptima, puede mejorarse eliminando el excedente de producción. Esto último se debe a que todos los costos son positivos, por lo tanto cualquier solución con producción mayor al óptimo no es solución óptima.

A partir de los puntos anteriores podemos concluir que:

- Existe un total de producción óptimo, el cual coincide con la demanda acumulada más el inventario mínimo del último período
- Las soluciones iniciales generadas como se describió anteriormente cumplen con la propiedad de cantidad de producción óptima

Como se verá en la sección 4.4, es en esta propiedad de las soluciones iniciales en la que se basan los movimientos que forman parte de la búsqueda tabú implementada.

En las siguientes secciones se detallan las diferentes estrategias utilizadas en la generación de las seis soluciones iniciales. Como se mencionó anteriormente las mismas buscan cubrir de manera eficiente el espacio de soluciones factibles al problema no capacitado. De esta manera, para una instancia del problema dada, algunas de las soluciones presentarán características similares a la solución óptima mientras que otras no. Para ejemplificar este punto, consideremos un escenario donde los costos de inventario superan largamente los costos de activación de la producción. En este caso la solución óptima del problema será similar a un esquema de producción *lote-por-lote*, minimizando la cantidad de producción arrastrada como inventario. Por otro lado, para un escenario donde los costos de inventario son mínimos, la solución óptima será similar a aquella donde la producción se active en el menor número de períodos, minimizando de esta forma los costos de configuración.

4.3.1. Lote por lote

Para esta clase de esquema de producción, el tamaño de lote para cada ítem en cada período coincide con su demanda. Como consecuencia, al aplicar este esquema de producción no se incurre en costos de inventario. Se debe notar además que este tipo de soluciones no cumplen con la restricción de inventario mínimo cuando éste es mayor a cero. En estos casos las mismas deben ser ajustadas incrementando la producción en cada período que viole dicha restricción.

4.3.2. División del horizonte de planificación

Esta estrategia consiste en la partición del horizonte total de planificación en un conjunto de sub-horizontes, donde para cada sub-horizonte el total de producción requerida se produce en el primer período. De esta manera se definen cuatro tipos de particiones diferentes:

- División del horizonte de tiempo a la mitad; se generan dos sub-horizontes de tamaño $|T|/2$.
- División del horizonte de tiempo en cuatro partes iguales; se generan cuatro sub-horizontes de tamaño $|T|/4$.
- División del horizonte de tiempo en sub-horizontes de dos períodos; se generan $|T|/2$ sub-horizontes de tamaño 2.
- División del horizonte de tiempo en sub-horizontes de cuatro períodos; se generan $|T|/4$ sub-horizontes de tamaño 4.

De la misma manera que para el esquema de producción *lote-por-lote*, el total de inventario al final de cada sub-horizonte es cero. Como consecuencia puede ser necesario ajustar la producción al inicio de cada sub-horizonte para satisfacer la restricción de inventario mínimo al final del mismo.

4.3.3. Método Wagner-Whitin

El algoritmo de Wagner-Whitin, desarrollado por Harvey Wagner y Thomson Whitin en 1958 [31], es una técnica matemática para dimensionar lotes con demanda determinística discreta que utiliza programación dinámica para la minimización de costos. Fue demostrado que la obtención de soluciones con este método consume en el peor caso un tiempo de orden $o(|T|^2)$ [24].

Como fue mencionado en el Capítulo 1, este método fue desarrollado para problemas con las siguientes características:

- Se debe considerar el problema en el que se produce únicamente un ítem.
- No se tienen restricciones de capacidad.
- No se tienen restricciones de inventarios mínimos.

Bajo las siguientes condiciones, las soluciones obtenidas con el algoritmo son óptimas:

- Costos de producción constantes, y por lo tanto despreciables.
- Costos de inventario constantes.
- Costos de configuración constantes.

Si en lugar de considerar costos de producción constantes, los costos varían, el algoritmo permite la obtención de soluciones aproximadas. Qian Qian y Jones demostraron que en estas circunstancias, para problemas de gran escala el algoritmo produce soluciones cercanas a las óptimas [24].

Las soluciones halladas por este método tienen la propiedad de que solamente se producirá en un período cuando el nivel de inventario llega a cero [14]. Por su construcción, las soluciones tienen la característica de que en cada período o bien se produce el total de la demanda para cierta cantidad de períodos siguientes, o bien no se produce. Teniendo esto en cuenta, a continuación se explican los fundamentos de programación dinámica en las que se basa el algoritmo.

Consideremos una función f_{jt} que representa el costo de realizar una orden en un período j para cubrir la demanda requerida entre los períodos $j,\ldots,t$, suponiendo inventarios nulos al principio del período j y al final del período t. Este valor estará dado por el costo de configuración en el período j, el costo de producir en el período t toda la demanda comprendida entre los períodos j y t, y finalmente, el costo de mantener como inventario el excedente de producción durante cada uno de los períodos intermedios. La expresión que se obtiene es la siguiente:

$$f_{jt} = s_j + pc_j \sum_{i=j}^{t} d_i + \sum_{k=j}^{t-1} \sum_{h=k+1}^{t} hc_k d_h$$

A partir de esta expresión, el problema consiste en encontrar la secuencia de períodos $w_1 w_2 \ldots w_m$ donde se active la producción de forma tal que se minimicen los costos totales:

$$z = \min_{1..m} \left\{ \sum_{i=1}^{m} f_{w_{i-1} w_i} \right\}$$

Donde para el conjunto de períodos $T=1,\ldots,|T|$ se debe cumplir que $w_1=1$ y $w_m=|T|$.

Para resolver este nuevo problema Wagner y Whitin desarrollan un algoritmo basado en programación dinámica. Suponiendo inventarios nulos al final de cada período t, la recursión detrás del algoritmo está dada por la función F_t:

$$\begin{cases} F_0 = 0 \\ F_t = \min_{j=1..t} \{F_{j-1} + f_{jt}\} \end{cases}$$

Esta recursión permite obtener el costo mínimo de la solución de Wagner-Whitin, el cual coincide con el valor de F para el último período ($F_{|T|}$). A partir de este costo el método permite determinar además la secuencia de períodos $w_1 w_2 \ldots w_m$ donde se debe activar la producción.

Hasta aquí se tienen las herramientas necesarias para la obtención de planificaciones de un único ítem. Dado que el problema tratado considera la producción de múltiples ítems, la solución utiliza el algoritmo de Wagner-Whitin en la determinación de la planificación de cada ítem por separado, y luego unifica los esquemas de producción obtenidos para generar una solución final.

4.4. Movimientos

Como fue descrito en la sección anterior, la construcción de las soluciones iniciales no garantiza que las mismas sean factibles al problema capacitado. Incluso cuando éstas cumplen con la propiedad de factibilidad, normalmente difieren de las soluciones óptimas. Debido a esto es necesario ajustar las mismas para aproximarlas a las soluciones óptimas del problema. Este ajuste se realiza a partir de la aplicación de movimientos sobre los esquemas de producción incluidos en las soluciones.

Se debe recordar que la cantidad total de producción de las soluciones iniciales es óptima, es decir, coincide con la cantidad de producción de la mejor solución factible. A los efectos de mantener esta propiedad en las soluciones intermedias, ninguno de los movimientos generados incrementa o reduce el total producido.

Adicionalmente, cada movimiento deberá preservar la factibilidad al problema relajado (no capacitado), es decir, satisfacer las restricciones de demanda y de inventario mínimo al final de cada período. El motivo de este punto se verá más adelante cuando se detalle la manera en que avanza la búsqueda tabú hacia soluciones factibles.

Con base en el trabajo de Gopalakrishnan [15] los movimientos implementados son los siguientes:

BackwardMove. Este movimiento consiste en desplazar cierta cantidad de producción de un ítem *i*, desde un período determinado hacia algún período anterior (back). La cantidad de producción a mover dependerá de la capacidad ociosa en el período destino. En caso de existir cierta holgura en la capacidad del período destino, se intentará mover tanta producción desde el período origen como sea posible hasta cubrir esta holgura. Este movimiento busca reducir los costos eliminando posibles penalidades por exceder la capacidad en el período origen.

Otra alternativa es intentar desplazar la totalidad de la producción desde el período origen hacia algún período anterior. Para este caso se deberá analizar las ventajas de eliminar un costo de setup en el período origen, contra los costos incurridos por mantener como inventario esta producción en los períodos intermedios, y la posible penalidad por exceder la capacidad en el período destino.

Si partimos de una solución factible (del problema relajado no capacitado), es fácil demostrar que la solución resultante de aplicar este tipo de movimiento preservará la factibilidad. Esto se debe a que la producción desplazada hacia un período anterior será arrastrada como inventario hasta el período actual. De esta manera, este tipo de movimientos matiene la factibilidad al problema no capacitado.

ForwardMove. Este movimiento es análogo al anterior pero desplazando una cierta cantidad de producción de un ítem desde determinado período hacia un período posterior (forward). Las consideraciones sobre la cantidad de producción desplazada son las mismas que para el *BackwardMove*, pero a diferencia del movimiento anterior, desplazar producción hacia un período posterior puede implicar violar las restricciones de demanda y/o de inventario mínimo. Debido a esto, para preservar la factibilidad del problema no capacitado, se deberá analizar la producción y la demanda en cada período intermedio previo a determinar qué cantidad de producción mover y hacia cuál período posterior dirigirla.

SwapMove. Este movimiento es una combinación de los movimientos anteriores. En primer lugar se desplaza cierta cantidad de producción de un ítem *i* desde un período *t* hacia un período anterior *d*. De esta manera se genera una cierta holgura en la capacidad disponible en el período *t*. Esta holgura es posteriormente ocupada por una cierta cantidad de producción del ítem *j* (distinto de *i*) a ser desplazada desde *d* hacia *t*. Nuevamente, previo a desplazar parte de la producción hacia un período posterior, se deberá analizar la cantidad a mover de manera de mantener la factibilidad al problema no capacitado.

Adicionalmente a los movimientos anteriores, fue analizada la posibilidad de implementar otras clases de movimientos:

FixedBatchMove: la cantidad de producción desplazada es fija. De esta manera se minimizan los cálculos para determinar el mejor tamaño de lote a desplazar. Se definieron dos tamaños de lote: uno pequeño para desplazarse hacia soluciones cercanas, y otro de mayor tamaño para desplazarse hacia regiones distantes en busca de la factibilidad de la solución.

RandomMove: el período destino del movimiento se obtiene de manera aleatoria, reduciendo la cantidad de movimientos generados y los cálculos para determinar los mismos. La cantidad de producción a desplazar se determina de acuerdo a la factibilidad de la solución. Para soluciones factibles, de la misma manera que para el movimiento anterior, se selecciona un lote fijo de tamaño pequeño, mientras que para soluciones no factibles se desplaza el excedente de producción del período origen.

Se estudió el comportamiento de la heurística contemplando los movimientos anteriores para distintas instancias del problema, y en ninguno de los casos se obtuvieron resultados favorables. La reducción en el tiempo de ejecución fue mínima, y la calidad de las soluciones encontradas fue sensiblemente inferior a las encontradas con los otros movimientos, por lo que no fueron incluidos en la versión definitiva de la heurística implementada.

4.5. Búsqueda tabú básica

En esta sección se describen los principales componentes de la búsqueda tabú implementada. El objetivo de la misma es encontrar soluciones factibles al problema original con buenas propiedades, las cuales posteriormente serán combinadas de forma probabilística con el fin de obtener soluciones más cercanas a los óptimos del problema.

4.5.1. Función objetivo

Como fue mencionado en secciones anteriores, la resolución del problema de factibilidad para un CLSP con tiempos de configuración es de tipo NP-completo. Es necesario entonces trabajar con una relajación del problema original. Debido a esto se permite que las soluciones sobre las que avanza la heurística violen la restricción de capacidad. Se debe agregar por lo tanto algún mecanismo que permita distinguir entre soluciones factibles y no factibles al problema capacitado. Esto se logra penalizando aquellas soluciones que violan las restricciones de capacidad. Los detalles de esta penalidad son descritos en la siguiente sección.

Se extiende entonces la función objetivo presentada junto al modelo matemático en la sección 2.2 agregando una penalidad *p* por los excesos en la utilización de capacidad:

$$\min \sum_{i \in P} \sum_{t \in T} sc_t^i y_t^i + (vc_t^i + ec_t^i + pc_t^i) x_t^i + hc_t^i s_t^i + p\varepsilon$$

Donde ε representa el excedente en la capacidad utilizada y se calcula a partir de la siguiente expresión:

$$\varepsilon = \sum_{t \in T} \max \left\{ \sum_{i \in P} (y_t^i gt^i + x_t^i et^i) - cape_t + \sum_{h \in C} \sum_{j \in h} x_t^j pt^j - capp_t^j ; 0 \right\}$$

De la expresión anterior se desprende que la penalidad se activa para aquellas soluciones que exceden la capacidad disponible, y se anula para las restantes.

4.5.2. Penalidad

Dado que tanto las soluciones iniciales como los movimientos implementados generan soluciones factibles al problema no capacitado (no necesariamente factible al problema capacitado), la heurística no sólo deberá avanzar hacia soluciones con menor costo, sino que también deberá avanzar hacia soluciones factibles al problema capacitado.

Esto último se logra penalizando la violación de la restricción de capacidad para las soluciones no factibles al problema capacitado. Para esto se define un coeficiente de penalidad que se aplica a cada unidad producida que exceda la capacidad disponible.

Cuanto mayor sea el valor de *p* mayor será la penalidad en la que se incurra. De esta manera la heurística convergerá más rápido hacia una solución factible evitando cualquier exceso en la capacidad. Por contrapartida esto generará que la heurística evite cualquier solución con exceso de capacidad (aún cuando este exceso sea mínimo) y que la solución desechada se acerque a la solución óptima. De manera similar, un valor pequeño de *p* implicará que la heurística analice de forma más detalla la vecindad de la solución actual. Esto último es algo deseable cuando la solución actual se acerca al óptimo, sin embargo genera problemas cuando la solución actual no es factible, ya que prolonga el proceso de convergencia hacia una solución factible.

Estas propiedades fueron verificadas a través de distintos experimentos, ejecutando la heurística con valores de penalidad variando entre 5 y 50.000. Para problemas con restricción de capacidad ajustada, donde se parte de soluciones iniciales no factibles al problema capacitado, valores bajos de *p* impiden encontrar en la mayoría de los casos soluciones factibles. Este problema no se presenta en dichos problemas cuando el valor de *p* es alto. Sin embargo, cuando la holgura entre la capacidad necesaria y la capacidad disponible aumenta, el valor de la mejor solución encontrada cuando la penalidad es baja llega a ser en promedio un 20% menor al encontrado cuando la penalidad es alta.

Para poder aprovechar a la vez las ventajas de un valor bajo y de un valor alto de penalidad, nuevamente nos basamos en el trabajo de Gopalakrishnan [15] sobre penalidad auto-ajustable. Este establece que si la región que se está analizando genera buenas soluciones, se reduce el valor de *p* para intensificar la búsqueda de soluciones en la misma. Por otra parte, si la región que se está analizando no es promisoria, generando mayormente soluciones no factibles, se incrementa el valor de *p* para avanzar más rápidamente hacia una región con mejores propiedades.

Para estimar la bondad de la región actual se analiza la factibilidad de las últimas soluciones encontradas. Si la mayor parte de estas soluciones son factibles, nos encontramos en una región con buenas propiedades. Si por el contrario la mayor parte de las soluciones encontradas recientemente son no factibles, podemos afirmar que la región actual no es promisoria. Gopalakrishnan implementa este punto llevando un registro de factibilidad de las últimas diez soluciones encontradas. Dicho registro debe ser actualizado en cada iteración de la heurística.

La heurística desarrollada en este trabajo resuelve este punto de otra manera, buscando evitar los problemas asociados a implementar y mantener en cada iteración una estructura adicional donde se

registren las últimas soluciones halladas. La solución desarrollada se basa en la implementación del protocolo de control de transporte de redes (TCP), en particular en el mecanismo de estimación del tiempo de ida y vuelta (round trip time). La idea es sencilla y busca llevar registro del porcentaje de soluciones no factibles encontradas hasta el momento, ponderando aquellas halladas recientemente. De esta manera se busca que el porcentaje de soluciones no factibles hallado en cada paso se vea poco influenciado por las soluciones más antiguas. La expresión resultante es la siguiente:

$$\delta_{h+1} = \delta_h (1 - \alpha) + \omega\alpha$$

donde δ_h es el porcentaje de soluciones no factibles encontradas hasta el momento; α es un coeficiente de ponderación para priorizar las soluciones recientes por sobre las soluciones más antiguas; ω es igual a 1 si la solución actual es no factible y 0 en caso contrario. A partir de los parámetros anteriores se calcula en cada iteración el nuevo porcentaje de soluciones no factibles δ_{h+1}.

Para determinar el mejor valor de α se analiza la evolución de δ_h, buscando aquella que mejor se ajusta a la definición de Gopalakrishnan. Este estudio se detalla en las Figuras 4.2 a 4.5 donde se compara el valor de la expresión $\delta_{h+1} = \delta_h (1 - \alpha) + \omega\alpha$, con el porcentaje de soluciones no factibles de las últimas diez soluciones encontradas a lo largo de cien iteraciones de la búsqueda tabú.

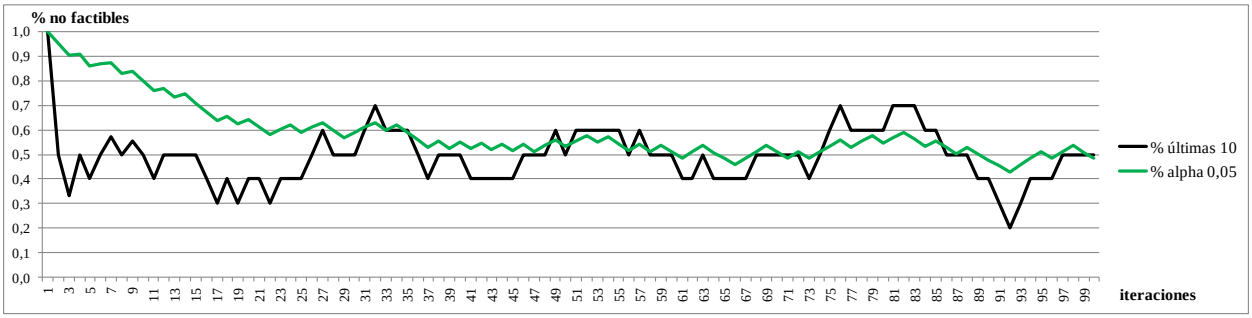

Figura 4.2 – Evolución del porcentaje de soluciones no factibles: α = 0,05.

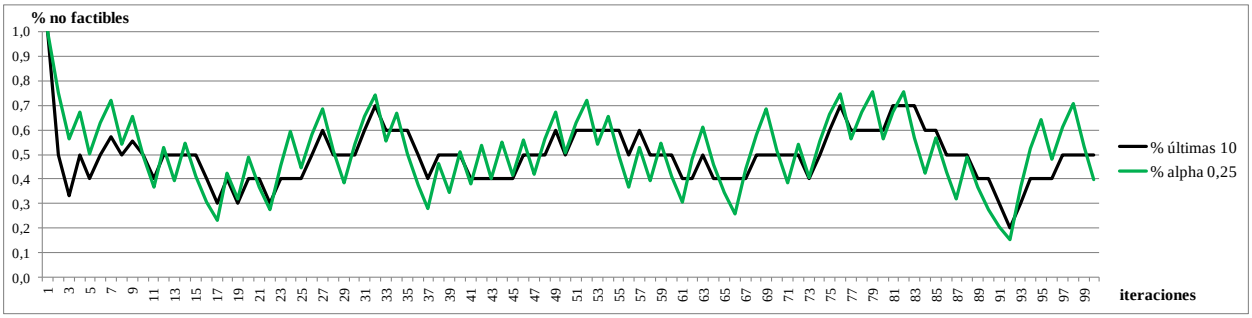

Figura 4.3 – Evolución del porcentaje de soluciones no factibles: α = 0,25.

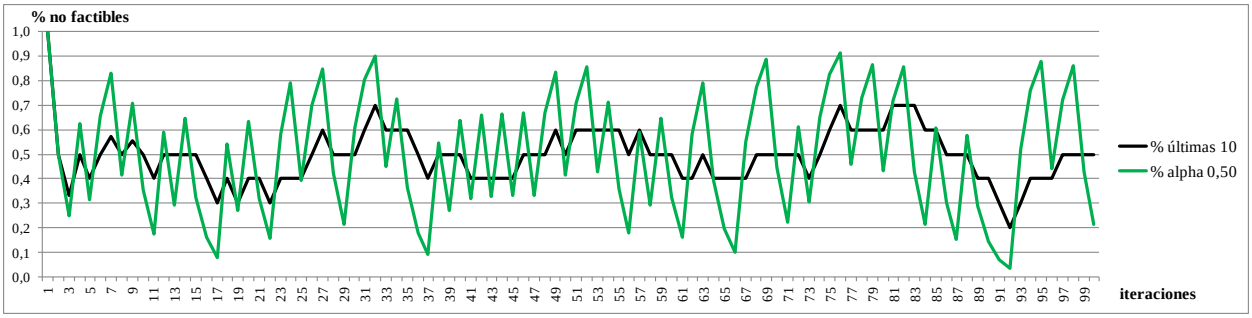

Figura 4.4 – Evolución del porcentaje de soluciones no factibles: α = 0,50.

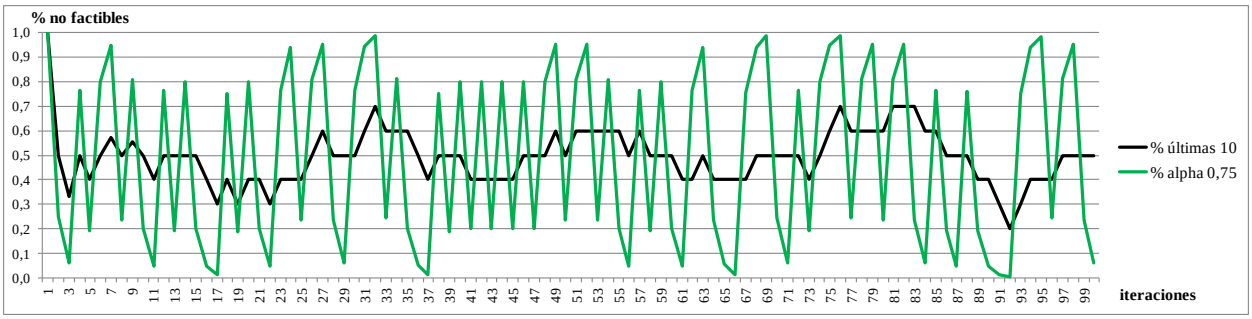

Figura 4.5 – Evolución del porcentaje de soluciones no factibles: α = 0,75.

Se observa que para α = 0,25 la evolución del porcentaje de soluciones no factibles hallado es el que más se ajusta al definido por Gopalakrishnan en la versión original de la heurística. Se selecciona entonces este valor para caracterizar la bondad de la región actual sobre la que avanza la heurística implementada.

Finalmente se realizaron pruebas con distintos valores iniciales de *p*, variando entre 5 y 50.000 para determinar los valores de penalidad con mejores resultados en la heurística. A partir de los primeros experimentos, inmediatamente se descartaron los valores cercanos a los extremos. No se encontraron mayores diferencias en los resultados obtenidos para penalidades entre 50 y 150. Debido a esto se decide trabajar con una penalidad inicial igual a 50, de la misma manera que la versión original de la heurística.

4.5.3. Vecindad

Dada una solución durante la ejecución de la heurística, su vecindad está dada por los posibles movimientos a soluciones vecinas que pueden ser aplicados sobre la misma. Como fue comentado en la sección 4.4, estos movimientos cumplen con dos propiedades fundamentales: la primera garantiza la factibilidad de las soluciones generadas al problema no capacitado; la segunda preserva la cantidad total de producción para cada ítem. Por lo tanto estas dos propiedades se cumplen para todas las soluciones de la vecindad.

En cada iteración la heurística analiza todas las soluciones que componen la vecindad, seleccionando aquella que genere mayores reducciones a los costos de la función objetivo. Recordar que esto se puede lograr reduciendo los costos de producción o reduciendo los excesos en la capacidad utilizada. En caso de que ninguna de las soluciones de la vecindad mejore la solución actual, la heurística seleccionará aquella con menor valor. En estas condiciones es de esperarse que la heurística seleccione un movimiento de tipo *SwapMove*. Se debe recordar que los movimientos de este tipo pueden verse como la combinación de un *BackwardMove* y un *ForwardMove*, por lo tanto en cada iteración la cantidad de movimientos de este tipo será mayor a la de los restantes.

Este último punto fue verificado experimentalmente monitoreando los movimientos generados en cada iteración de la heurística. Los resultados se grafican en la Figura 4.6 donde se detalla la distribución promedio de los movimientos generados en cada paso.

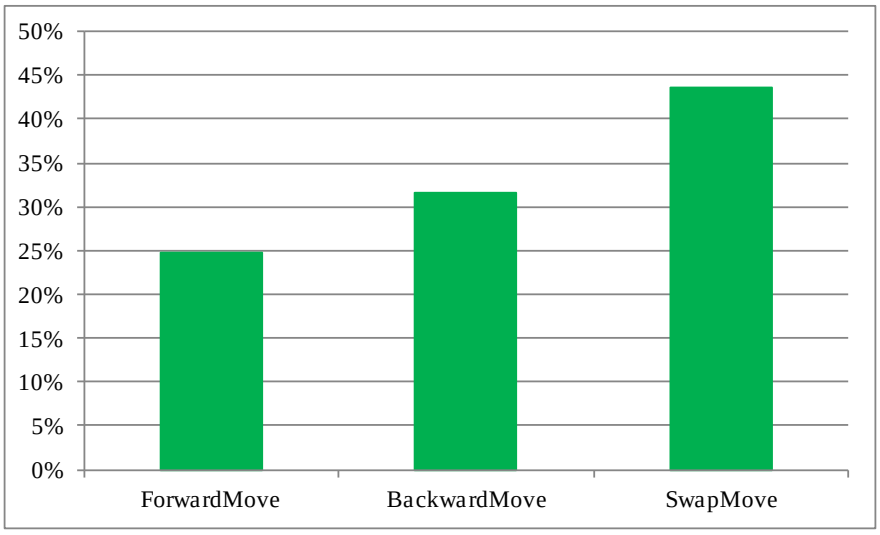

Figura 4.6 – Distribución de movimientos generados en cada iteración.

Adicionalmente se debe tener en cuenta que esta clase de movimientos no tiene mayor impacto en la utilización de la capacidad de los períodos involucrados, ya que la capacidad liberada por el primero de los ítems es posteriormente utilizada por el segundo. Esta característica hace que este tipo de movimientos sea especialmente ineficiente cuando la heurística avanza por regiones con mayoría de soluciones no factibles al problema capacitado.

Para minimizar el impacto de los puntos anteriores se aplica un mecanismo similar al implementado por Gopalakrishnan en la versión original de la heurística. El mismo consiste en habilitar la generación de movimientos del tipo *SwapMove* cada cierto número de iteraciones. Gopalakrishnan define esta cantidad de iteraciones en tres.

Fue analizado el comportamiento de la heurística implementada habilitando esta clase de movimientos cada cuatro y hasta cinco períodos. Para ninguno de los casos se observan cambios significativos en la calidad de las soluciones halladas. Debido a esto se decide habilitar los movimientos de tipo *SwapMove* cada cinco iteraciones. De esta manera además de mejorar la eficiencia de la heurística para abandonar regiones con mayoría de soluciones no factibles, se reducen los tiempos insumidos para la generación y evaluación de la vecindad.

4.5.4. Lista tabú

Para evitar que estos procedimientos se sesguen analizando una misma región, se debe tomar nota de las últimas soluciones encontradas o de los últimos movimientos aplicados. Esta información se registra en una lista tabú que da nombre al método.

Antes de presentar los detalles de la lista tabú, recordemos la definición de los posibles movimientos que se aplican a las soluciones durante la ejecución de la heurística. Tanto los *ForwardMove* como los *BackwardMove* desplazan una cierta cantidad de producción de un ítem hacia un período destino ocupando la mayor cantidad de capacidad disponible. Por lo tanto la heurística debiera evitar desplazar dos veces el mismo ítem hacia el mismo período con esta clase de movimientos. Los *SwapMove* por su parte liberan cierta capacidad en un período y la ocupan con producción desplazada desde otro período. De esta manera se debería evitar aplicar dos veces este tipo de movimientos sobre los mismos períodos, ya que por su definición, la aplicación del segundo anularía el efecto del primero.

Para evitar las dos situaciones descritas, la heurística lleva registro del ítem y período destino involucrados en los *ForwardMove* y en los *BackwardMove* y de los períodos que componen los *SwapMove*. Esta es entonces la información que se registra en la lista tabú. Notar que al registrar los atributos de los movimientos en lugar de las soluciones por las que avanza la heurística, se minimiza el procesamiento requerido para identificar un posible tabú. Esto se debe a que para comparar la igualdad entre dos soluciones se requiere analizar el tamaño de lote para cada ítem y cada período.

Como se verá en la sección de implementación, un movimiento está caracterizado dentro de la lista tabú por un código (hash) que registra los atributos anteriores. Dicho código se compone de cinco dígitos de la forma ***TSSDD*** donde*:*

- ***T*** distingue al tipo de movimiento (1 para los movimientos *ForwardMove* y *BackwardMove*; y 2 para los movimientos *SwapMove*).
- ***SS*** identifica al ítem involucrado para los *ForwardMove* o *BackwardMove*; y al período origen en los *SwapMove*.
- ***DD*** identifica al período destino en todos los movimientos.

Para facilitar la generación y comparación de los códigos, éstos se implementan como valores enteros de la misma manera para los movimientos *ForwardMove* y *BackwardMove*:

$$h_B = h_F = 10000 + 100SS + DD$$

De manera similar se calcula para los movimientos *SwapMove*:

$$h_S = 20000 + 100SS + DD$$

Al igual que para la penalidad, sería deseable que la cantidad de movimientos registrados en la lista tabú se ajustara favoreciendo explorar aquellas regiones promisorias. Para esto la heurística implementada se basa en el trabajo de Dell'Amico y Trubian [8] sobre tamaños de listas auto-ajustables:

- Si la solución obtenida al aplicar el movimiento es mejor que la anterior, se disminuye el largo de la lista en 1 para intensificar la búsqueda de soluciones en la región actual.

- Si la solución obtenida al aplicar el movimiento no es mejor, se aumenta el largo en 1 para evitar los movimientos anteriormente aplicados y avanzar hacia alguna otra región con mejores propiedades.

Adicionalmente se deben definir cotas para evitar que el tamaño de lista aumente o disminuya desproporcionadamente. El valor de estas cotas debe depender del tamaño del problema, permitiendo registrar en la lista tabú una mayor cantidad de movimientos para los problemas de mayor tamaño.

Se define entonces la cota inferior $L_{\min}$ y la cota superior $L_{\max}$ de la siguiente forma:

- $L_{\min} = \alpha_{\min}(|P|+|T|)$
- $L_{\max} = \alpha_{\max}(|P|+|T|)$

Para determinar el valor de $\alpha_{\min}$ y $\alpha_{\max}$ se analiza el comportamiento de la heurística para distintas instancias del problema variando estos parámetros. No se encontraron mayores diferencias en el desempeño de la heurística a partir de la variación de estos valores, por lo que se decide mantener los definidos por Gopalakrishnan en la versión original [15].

Concluimos esta sección comentando que se estudiaron distintos tamaños de lista tabú, así como diferentes parámetros de ajuste, y para ninguno de los casos se encontraron diferencias significativas.

4.5.5. Criterios de parada

Para evitar que la heurística itere de forma indeterminada se debe definir algún criterio de parada. Dicho criterio de parada no debe ir en detrimento de la calidad de las soluciones encontradas. De esta manera se establece un compromiso entre la calidad de las soluciones encontradas y los tiempos de ejecución de la heurística.

Existen distintas estrategias para definir estos criterios: cantidad de iteraciones, tiempo de ejecución, diferencia entre la mejor solución y la mejor cota encontradas (GAP), o alguna combinación de los criterios anteriores.

Para aplicar el criterio de GAP entre soluciones, se necesita una cota para la solución óptima. Un posible método es relajar la restricción de integridad de las variables enteras, transformando el problema en uno de programación lineal y aplicando algún método de resolución eficiente (por ejemplo simplex). Para estos casos no existe ninguna garantía de que la cota hallada se aproxime a la solución óptima al problema original. Diversos autores comprobaron el punto anterior de forma empírica [6][29][15]. Otra alternativa posible es la generación de cotas a partir de algún método más elaborado, aunque esto agrega complejidad potencialmente innecesaria a la heurística.

Quizás la estrategia más simple es aplicar un criterio de parada por cantidad de iteraciones. Si bien esta estrategia es fácil de implementar, no brinda ninguna garantía de calidad de la solución hallada. Adicionalmente existe la posibilidad cierta de detener la ejecución de la heurística cuando ésta aún puede continuar mejorando la mejor solución hallada.

Una manera de evitar este último problema es no detener la ejecución de la heurística mientras ésta se encuentre avanzando en alguna región promisoria. En otras palabras, no se debiera aplicar el criterio de parada mientras exista cierta probabilidad de que la heurística mejore la mejor solución hallada hasta el momento. A partir de lo anterior se define un criterio de parada por *Cantidad de iteraciones sin mejora*, donde la heurística se detiene luego de un cierto número de pasos donde no se ha podido reducir el valor de la mejor solución hallada hasta el momento.

Para poder aplicar este criterio debemos primero definir el número de iteraciones k_{max} luego de las cuales debiera detenerse la ejecución de la heurística. Se debe tener en cuenta que un número pequeño permite mejorar los tiempos de ejecución, pero incrementa la posibilidad de dejar regiones del espacio de soluciones sin inspeccionar. Por contrapartida, aumentar esta cantidad de iteraciones permite potencialmente a la heurística cubrir un mayor espectro del espacio de soluciones, incrementando los tiempos de ejecución.

¿Cómo seleccionar entonces a priori esta cantidad de iteraciones?

Para contestar la pregunta anterior se realiza un estudio experimental del comportamiento de la heurística para distintas instancias del problema, haciendo foco en la evolución de la mejor solución hallada durante el avance de la misma. Las Figuras 4.7 a 4.10 resumen los resultados obtenidos en la primera instancia de este estudio, donde cada curva representa la evolución promedio partiendo desde cada una de las soluciones iniciales.

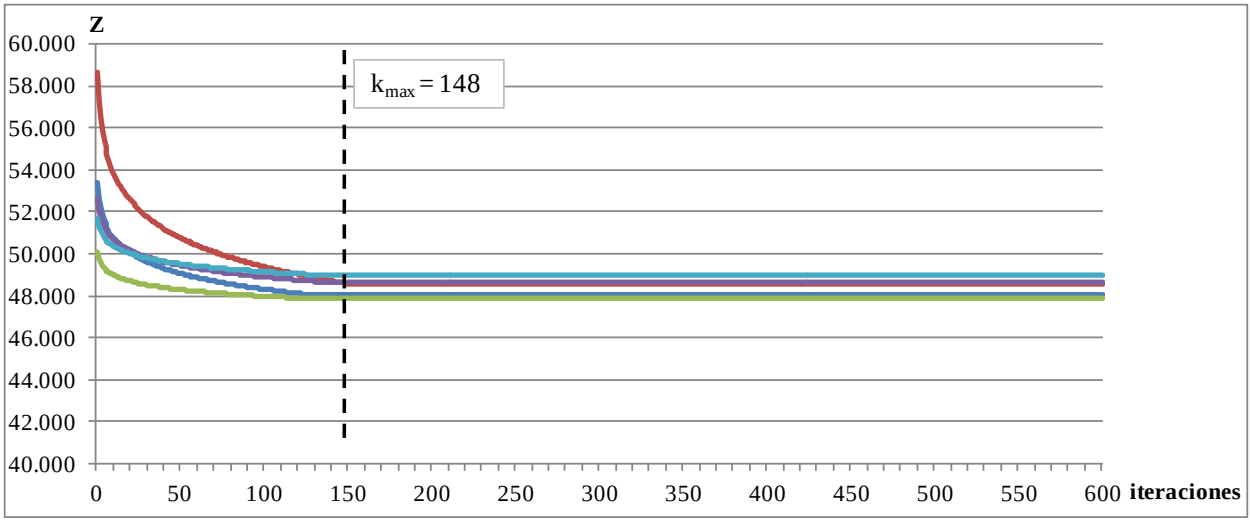

Figura 4.7 – Evolución del valor de la mejor solución hallada: 6 ítems y 9 períodos.

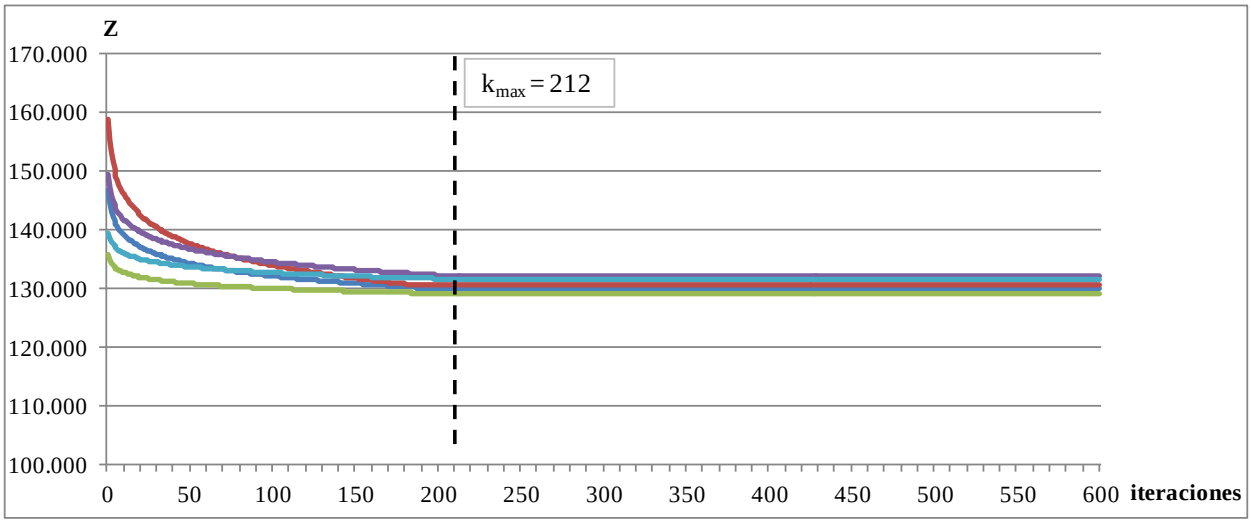

Figura 4.8 – Evolución del valor de la mejor solución hallada: 10 ítems y 15 períodos.

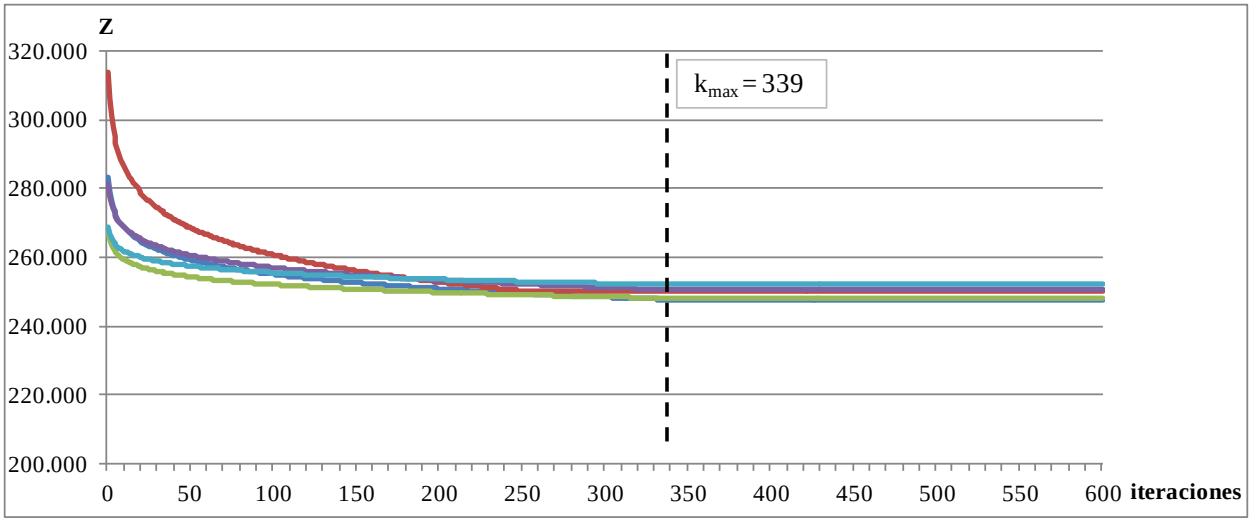

Figura 4.9 – Evolución del valor de la mejor solución hallada: 15 ítems y 20 períodos.

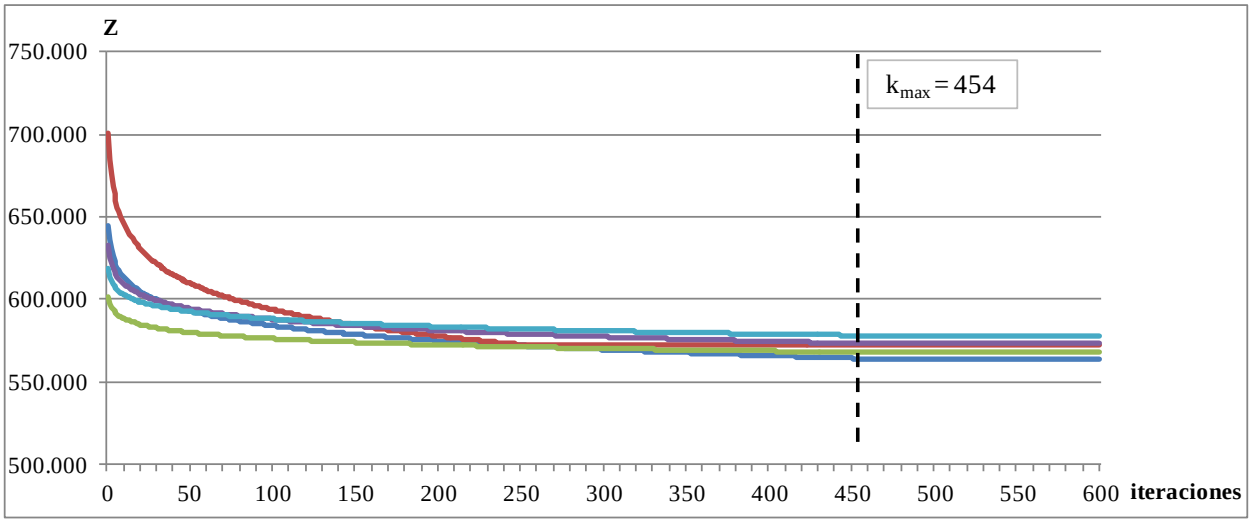

Figura 4.10 – Evolución del valor de la mejor solución hallada: 24 ítems y 30 períodos.

Como era de esperar, el número de iteraciones k_{max} a partir de las cuales la heurística converge en una solución, depende de la cantidad de ítems y la cantidad de períodos. Sería deseable poder obtener una expresión para la cantidad de iteraciones a partir de alguno de estos dos parámetros.

Para obtener esta expresión se realizó una segunda etapa de pruebas. En esta etapa nuevamente se ejecutaron distintas instancias del problema para una mayor cantidad de tamaños (cantidad de ítems y períodos) del mismo. En el Cuadro 4.1 se detalla el número de iteraciones promedio a partir de las cuales la heurística converge en una solución factible.

Ítems	Períodos	Iteraciones
3	**6**	118
6	**9**	148
8	**12**	184
10	**15**	212
12	**18**	301
15	**20**	339
20	**24**	402
24	**30**	454
30	**30**	526

Cuadro 4.1 – Cantidad de iteraciones promedio requerida para converger en una solución factible.

Los resultados anteriores fueron aproximados a una función lineal aplicando mínimos cuadrados. Este método permite obtener una expresión para estimar el número de iteraciones a partir de la cantidad de ítems y de la cantidad de períodos. Los resultados obtenidos se grafican en la Figura 4.11 y Figura 4.12.

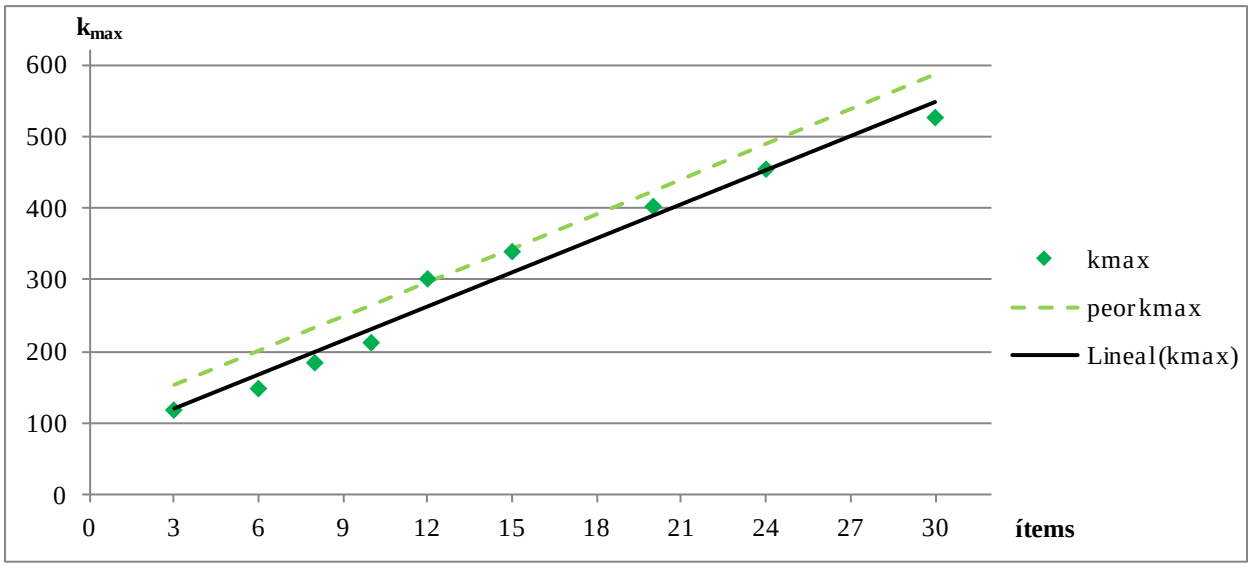

Figura 4.11 – Aproximación lineal de k_{max} en función de la cantidad de ítems.

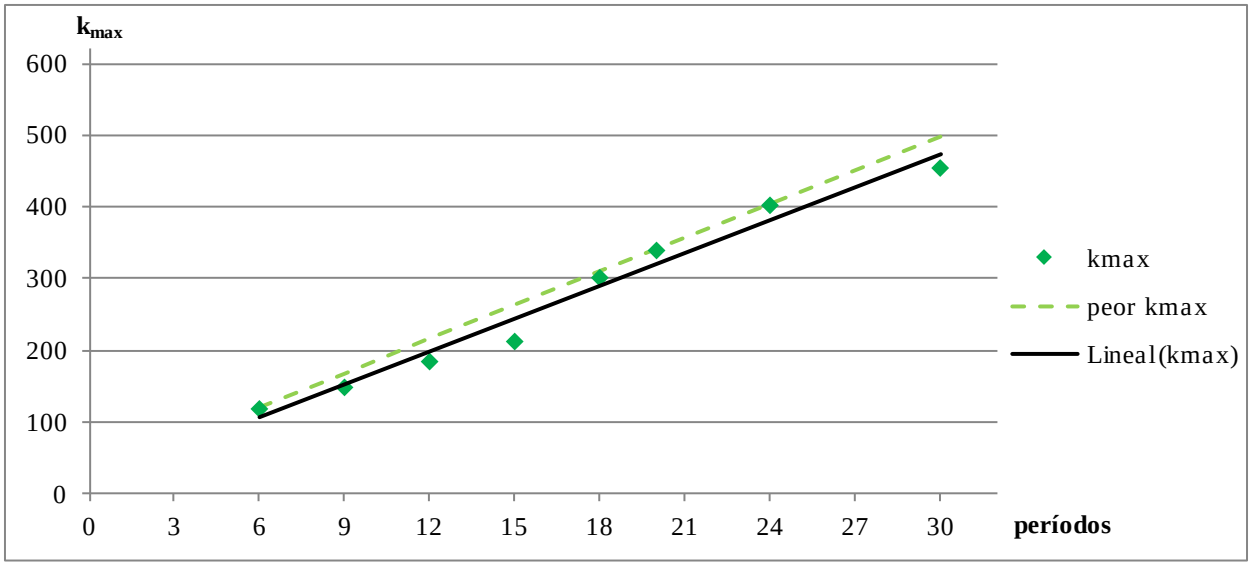

Figura 4.12 – Aproximación lineal de k_{max} en función de la cantidad de períodos.

A partir de las gráficas se observa una dependencia similar tanto para la cantidad de ítems ($R_{|P|}$) como para la cantidad de períodos ($R_{|T|}$):

$R_{|P|}$: 15,9016$|P|$ + 72,0659

$R_{|T|}$: 15,6023$|T|$ + 8,4120

Estas similitudes permiten seleccionar como base una de las expresiones anteriores. Buscando un criterio de parada más conservador, la expresión base se ajusta de forma de incluir todos los puntos hallados en la fase experimental. Este ajuste se ilustra con una la línea punteada en las figuras anteriores.

Finalmente se define el número de iteraciones k_{max} necesario para que la heurística converja en una solución:

$$k_{max} = 16|P| + 100$$

Esta es la cantidad de iteraciones que fue seleccionada como criterio de parada de la heurística implementada. La Figura 4.13 permite comparar el resultado anterior con la expresión obtenida por Gopalakrishnan para la versión original de la heurística: k_{max} = 1,5(300 + 10$|P|$). La misma ilustra la evolución de las dos expresiones para el universo de ítems incluido en el estudio computacional.

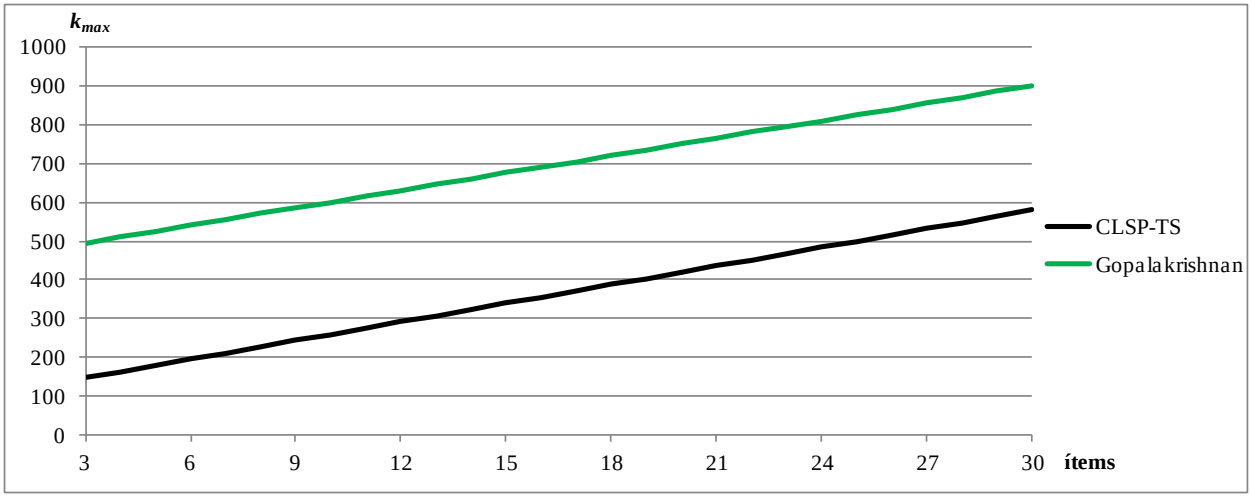

Figura 4.13 – Comparación entre el k_{max} hallado y el definido originalmente por Gopalakrishnan.

En la figura anterior se puede observar que si bien la cantidad de iteraciones halladas en este trabajo es menor a las de la versión original, ambas expresiones comparten similitudes. En ambos casos se obtuvo una relación lineal entre la cantidad de iteraciones y la cantidad de ítems con una pendiente casi idéntica.

Concluimos esta sección estudiando el impacto de k_{max} en las soluciones obtenidas aplicando el criterio de parada anterior. Para esto analizamos la diferencia entre las soluciones obtenidas con el mismo y las obtenidas iterando una cantidad de pasos arbitraria mayor. Los resultados de este análisis figuran en el Cuadro 4.2.

Items	**Períodos**	**$16\cdot\|P\| + 100$**	**Mejor solución factible**		**GAP**
			$k_{max} = 16\cdot\|P\| + 100$	**$k_{max} = 5.000$**	
3	**6**	148	16.325	16.217	0,01%
8	**12**	228	83.305	82.183	0,01%
12	**18**	292	183.506	179.874	0,02%
20	**24**	420	386.389	378.602	0,02%
30	**30**	580	727.943	707.452	0,03%

Cuadro 4.2 – GAP entre mejor solución hallada con $k_{max} = 16\cdot|P|+100$ y $k_{max} = 5.000$.

A partir de los resultados anteriores podemos concluir que el criterio de parada utilizado permite reducir sensiblemente la cantidad de iteraciones de la heurística, con un impacto menor en la calidad de las soluciones encontradas.

4.5.6. Pseudocódigo

En este punto presentamos una descripción paso por paso de la búsqueda tabú básica utilizada para mejorar una solución dada. Este procedimiento recibe una función objetivo conocida, y una solución inicial factible al problema no capacitado.

Paso 0: Inicialización

Definir Soluciones:
    solucion_actual := solucion_inicial;
    mejor_solucion := solucion_inicial;
    mejor_z := F(solucion_inicial);
    z_anterior := ∞;
    SI (solucion_inicial.esFactible())
        mejor_solucion_factible := solucion_inicial;
        mejor_z_factible := F(solucion_inicial);
    SINO
        mejor_z_factible := ∞;
    FIN SINO

Definir Lista tabú:
    L_{min} := 0,25*(|P|+|T|);
    L_{max} := 1,75*(|P|+|T|);
    lista_tabu := new listaTabu();

Definir Penalidad (p):
    p := 50;
    p_{min} := 0,1;
    p_{max} := 1000;
    porc_sol_no_factibles := 0;
    alpha := 0,25;

Definir Criterio de parada:
    k_{max} := 16*|P| +100;
    iteraciones_sin_mejora := 0;

MIENTRAS (k_{max} > iteraciones_sin_mejora)

Paso 1: Evaluación y Selección del Mejor movimiento

CALL BackwardMove(solución_actual);
CALL FowardMove(solución_actual);
SI (iteraciones_sin_mejora MOD 5 == 0)
    CALL SwapMove(solución_actual);
FIN SI

Generar vecindad a partir de los movimientos anteriores;
Ignorar aquellos movimientos incluidos en lista_tabu y que no mejoran mejor_solucion;

solucion_actual := obtener el vecino que tiene menor valor de función objetivo;
z_actual := F(solucion_actual);

Paso 2: Actualizar Lista Tabú

SI (z_actual < mejor_z)
    iteraciones_sin_mejora := 0;
    Remover todos los movimientos de lista_tabu;
SINO
    iteraciones_sin_mejora++;

    SI (z_actual <= z_anterior)
        SI (lista_tabu.length() > L_{min})
            Remover los dos elementos más antiguos de lista_tabu;
        SINO SI (lista_tabu.length() == L_{min})
            Remover el elemento más antiguo de lista_tabu;
        FIN SINO
    SINO SI (z_actual > z_anterior && lista_tabu.length() == L_{max})
            Remover el elemento más antiguo de lista_tabu;
    FIN SINO
FIN SINO
Agregar a lista_tabu el movimiento elegido;

Paso 3: Actualizar Solución

SI (solucion_actual.esFactible() && mejor_z_factible > z_actual)
    mejor_solucion_factible := solucion_actual;
    mejor_z_factible := z_actual;
FIN SI
SI (mejor_z > z_actual)
    mejor_solucion := solucion_actual;
    mejor_z := z_actual;
FIN SI
z_anterior = z_actual;

Paso 4: Actualizar p

SI (solucion_actual.esFactble())
    porc_sol_no_factibles := porc_sol_no_factibles*(1 - alpha);
SINO
    porc_sol_no_factibles := porc_sol_no_factibles*(1 - alpha) + alpha;
FIN SINO

SI (porc_sol_no_factibles > 0,5)
    p := p * 2 * porc_sol_no_factibles;
    p := MIN(p, p_{max});
SINO
    p := p * (porc_sol_no_factibles + 0,5);
    p := MAX(p, p_{min});
FIN SINO
FIN MIENTRAS

4.6. Diversificación e intensificación

En líneas generales este procedimiento trabaja con la meta heurística de búsqueda tabú, tomando como entrada las soluciones mejoradas a partir de la búsqueda tabú básica, y combinándolas de forma aleatoria para procurar obtener mejores soluciones probabilísticamente. Estas nuevas soluciones a su vez son utilizadas como entrada para nuevas búsquedas tabú.

La combinación de soluciones se realiza a nivel de planificaciones para cada ítem individualmente. De esta manera definimos un schedule como la planificación de producción para un ítem determinado, que incluye el tamaño de lote, activación de producción e inventario a lo largo del horizonte de tiempo para dicho ítem. A su vez a cada planificación se le asigna un valor funcional que coincide con el valor de la solución original de la que proviene.

Para operar sobre los schedules se hace uso de una estructura auxiliar, una memoria adaptativa de planificaciones individuales que llamaremos *D*. Cada vez que se obtiene una salida del tabú básico, se descompone la solución obtenida en schedules y se insertan los mismos en la memoria en orden ascendente según su valor funcional. Basados en el trabajo de Gopalakrishnan [15], el tamaño de la memoria adaptativa se inicializa en $8|P|$ schedules de modo de evitar un crecimiento desmesurado a medida que se avanza en esta fase.

Para la generación de una nueva solución que sirva de entrada al tabú básico, se selecciona alguno de los schedule en la memoria adaptativa para cada ítem. Esta selección se realiza probabilísticamente otorgándole una mejor probabilidad a los schedules con menor valor funcional, es decir, aquellos asociados a las mejores soluciones halladas hasta el momento.

Durante la construcción de una nueva solución, se permite la violación de las restricciones de capacidad por un cierto porcentaje con el fin de promover la combinación entre las distintas soluciones halladas hasta el momento. De esta manera, cada vez que es seleccionado un schedule se valida que no se viole la restricción de capacidad por este porcentaje, eliminando aquellos que no cumplen con la misma. Si la restricción se cumple, se agrega el schedule a la nueva solución y se ignoran los restantes schedules correspondientes al ítem seleccionado. Naturalmente este criterio de selección no garantiza la inclusión de todos los ítems en la solución construida. De darse este caso, se completa la solución generada agregando un schedule correspondiente a un esquema de producción *lote-por-lote* para cada ítem no incluido.

A medida que avanza la fase de diversificación e intensificación, la búsqueda tiende a concentrarse en zonas prometedoras. Esto se debe a que el proceso de selección de los distintos schedules favorece aquellos con mejor valor de función objetivo, mientras que aquellos con mayores valores se desechan a medida que el tamaño de la memoria adaptativa crece.

Como consecuencia, esta fase de la heurística lentamente evoluciona de un proceso de diversificación de soluciones a un proceso de intensificación de las mismas.

4.6.1. Criterios de parada

Al igual que para la búsqueda tabú básica se debe definir algún criterio de parada para evitar que se itere indeterminadamente durante la fase de diversificación e intensificación de las soluciones. El razonamiento es análogo al descrito en la sección 4.5.5. Nuevamente se define un criterio de parada por cantidad de iteraciones sin mejorar la mejor solución encontrada hasta el momento.

Para determinar la cantidad de iteraciones sin mejora a partir de las cuales se debiera detener la fase de diversificación e intensificación aplicamos el mismo análisis que para la búsqueda tabú básica. Las Figuras 4.14 a 4.17 resumen los resultados obtenidos en la primera instancia de este nuevo estudio para la fase de diversificación e intensificación (D&I).

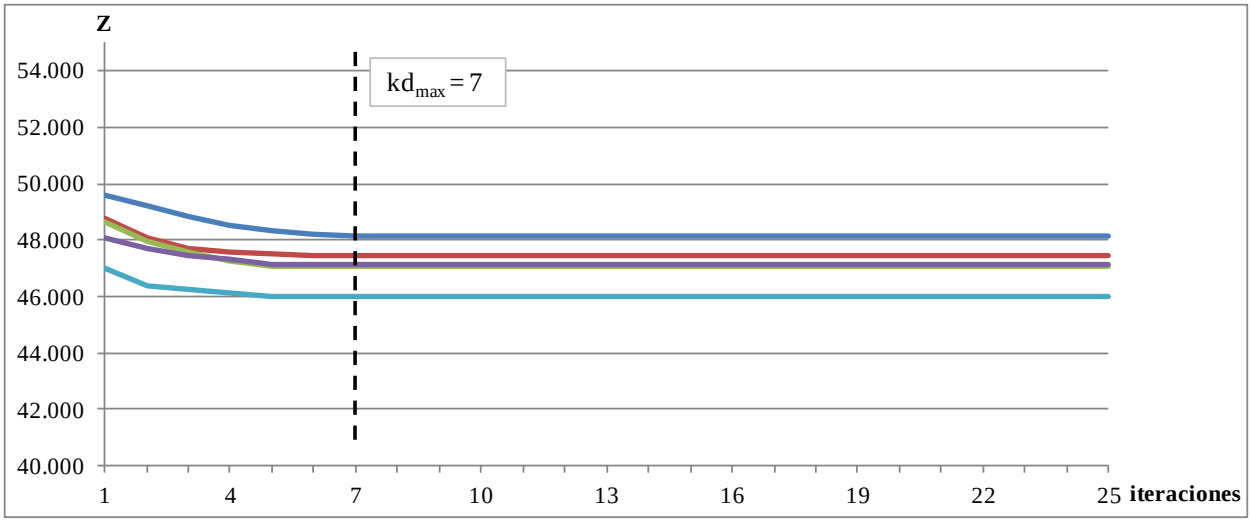

Figura 4.14 – Evolución de la mejor solución durante la fase D&I: 6 ítems y 9 períodos.

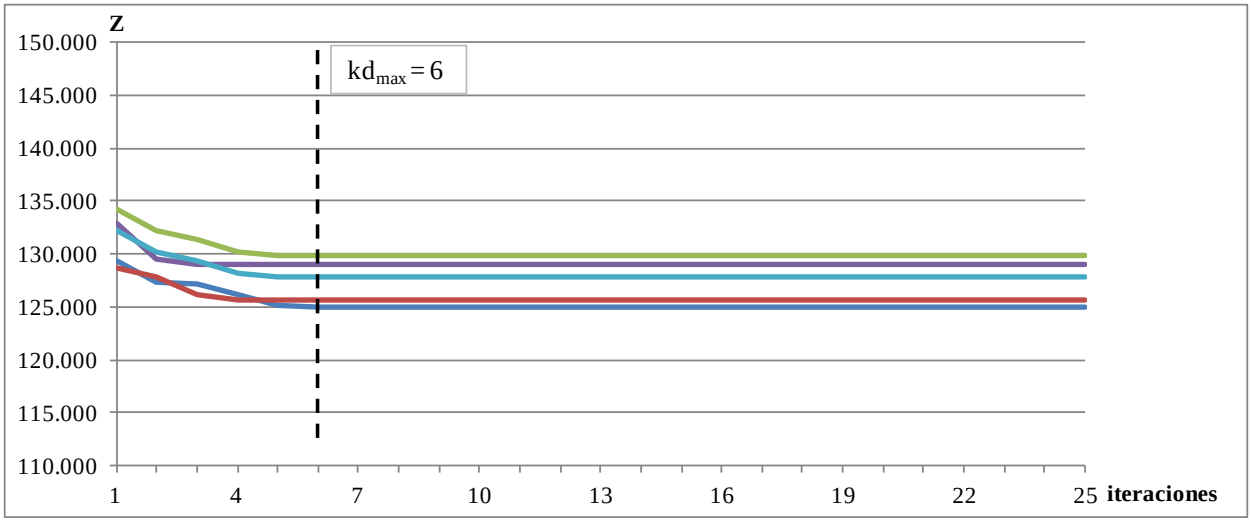

Figura 4.15 – Evolución de la mejor solución durante la fase D&I: 10 ítems y 15 períodos.

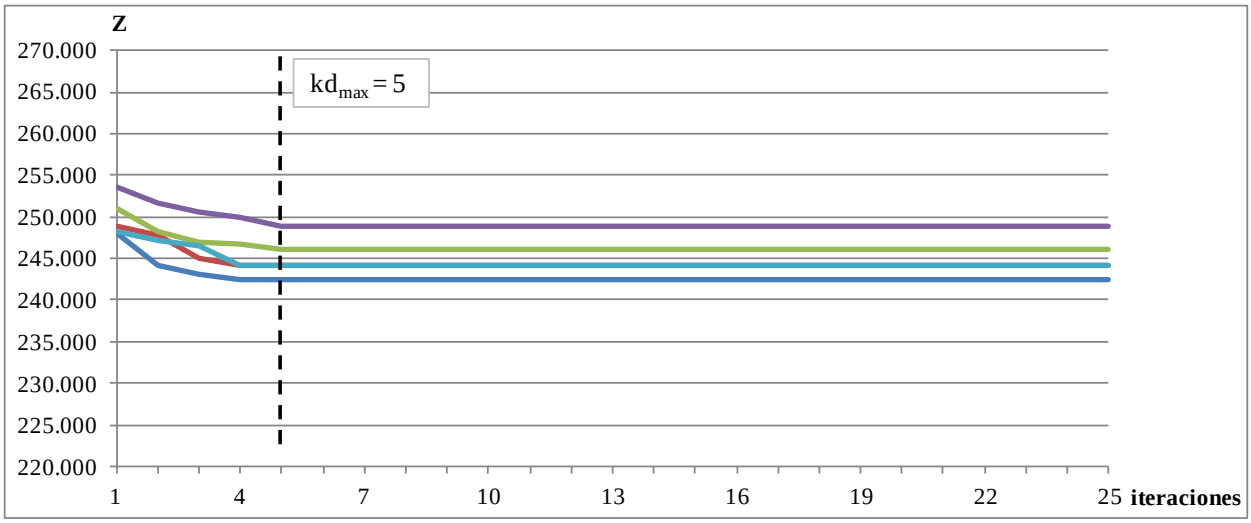

Figura 4.16 – Evolución de la mejor solución durante la fase D&I: 15 ítems y 20 períodos.

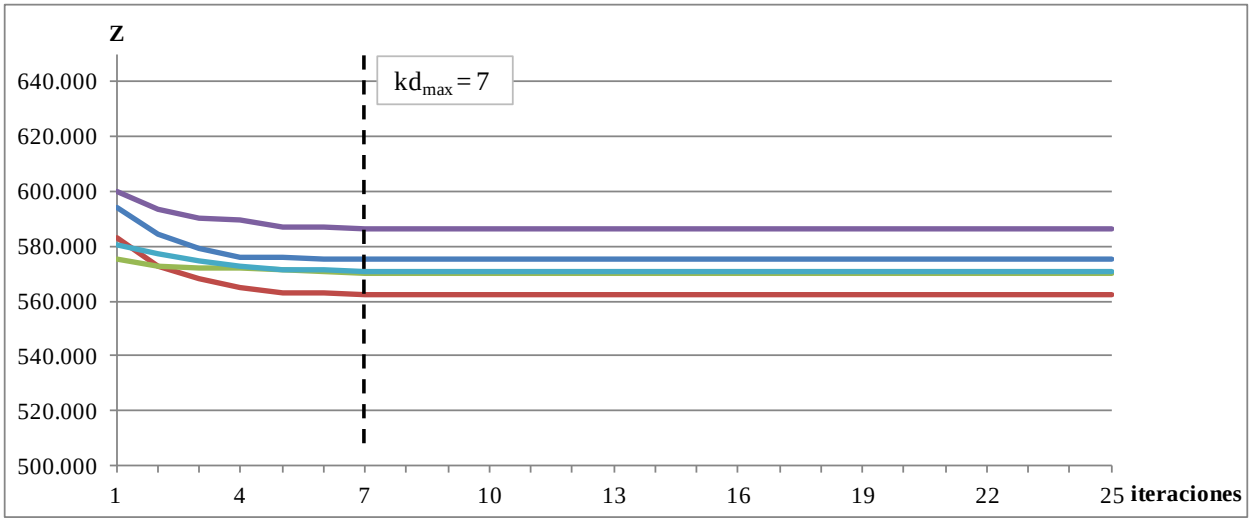

Figura 4.17 – Evolución de la mejor solución durante la fase D&I: 24 ítems y 30 períodos.

Se observa que a diferencia de la búsqueda tabú básica, el número de iteraciones a partir de las cuales la fase de diversificación e intensificación converge en una solución no parece depender de la cantidad de ítems y la cantidad de períodos. Esto se justifica si tenemos en cuenta que cada iteración de esta fase se encuentra precedida por una búsqueda tabú básica cuya convergencia sí depende del tamaño del problema.

A partir de los resultados anteriores podemos concluir que la cantidad de iteraciones buscada se encuentra entre 5 y 7. Al igual que para k_{max} se aplica un criterio conservador y se define el número de iteraciones kd_{max} necesario para que la fase de diversificación e intensificación converja en una solución:

$$\boldsymbol{kd_{max} = 7}$$

Nuevamente se debe analizar la calidad de las soluciones obtenidas aplicando el criterio de parada anterior. Estudiamos la diferencia entre las soluciones obtenidas con el mismo, y las obtenidas iterando una cantidad de pasos sensiblemente mayor. Los resultados de este análisis figuran en el Cuadro 4.3.

Items	**Períodos**	**Mejor solución factible**		**GAP**
		kd_{max} = 7	**kd_{max} = 50**	
3	**6**	16.191	16.012	0,01%
8	**12**	82.827	80.453	0,03%
12	**18**	183.506	171.296	0,07%
20	**24**	378.697	337.184	0,09%
30	**30**	718.758	702.578	0,02%

Cuadro 4.3 – GAP entre mejor solución hallada con kd_{max} = 7 y kd_{max} = 50.

4.6.2. Pseudocódigo

En este punto se presenta una descripción general de cómo se combina la búsqueda tabú básica descrita en la sección 4.5 con la fase de diversificación e intensificación introducida en la sección anterior. Estos son los dos principales componentes que dan forma a la heurística implementada.

1 - Etapa de Inicialización

1.1 Generar las seis soluciones iniciales descritas en la sección 4.3 y llamar para cada una de ellas al método tabú básico, buscando obtener soluciones *X'* mejoradas. Actualizar la mejor solución factible y la mejor solución al problema no capacitado, de manera análoga a la búsqueda tabú básica.
1.2 Descomponer *X'* en |*P*| schedules (uno para cada ítem). De esta forma, inicialmente se tendrán 6|*P*| schedules. Insertar los schedules en la memoria *D* en orden descendente según el valor de la función objetivo.

2 - Etapa de Diversificación e Intensificación

Mientras no se llegó al criterio de parada (cantidad de iteraciones sin mejora < kd_{max})

2.1 *D'* = copiar(*D*) y crear una nueva solución vacía nuevaSol.
2.2 Mientras *D'* no está vacío y resta agregar a nuevaSol algún schedules para algún ítem
   a. Sortear un schedule de *D'* con probabilidad $2i/|D'|*(|D'|+1)$ de seleccionar el *i*-ésimo peor schedule, esto es, se obtiene el schedule contando desde la peor posición de *D'*, *i* lugares hacia arriba.
   b. Obtener elemento *i* sorteado y eliminarlo de la lista.
   c. Agregar el schedule a nuevaSol.
   d. Si no se viola un porcentaje alfa el nivel de capacidad agregando este schedule, eliminar todos los schedules de ítems que correspondan al ítem agregado.
   e. Si se viola la restricción de capacidad por un porcentaje alfa, quitar el schedule agregado a la nuevaSol.
2.3 Si existen ítems que no han sido cubiertos por los schedules agregados a nuevaSol, rellenar con schedules lot for lot.
2.4 Realizar una búsqueda Tabú que tome como solución inicial a nuevaSol.
2.5 Actualizar la memoria *D* al igual que en el paso 1.2 de la etapa de inicialización. Actualizar la mejor solución factible y la mejor solución a la relajación según corresponda.

4.7. Implementación computacional

En este capítulo se describen los distintos aspectos asociados a la implementación computacional de la heurística descrita en el capítulo anterior.

Se analizaron diferentes opciones para la implementación de la heurística. Se estudiaron diversos lenguajes y herramientas, y finalmente se optó por trabajar con OpenTS [4] framework Java de código abierto.

4.7.1. Framework Java OpenTS

OpenTS es un framework Java utilizado como base para la implementación de la metaheurística de búsqueda tabú a través de un diseño orientado a objetos [4]. El mismo está implementado en unas pocas clases y define una organización básica de los diferentes componentes que forman parte de la solución.

El principal motivo para trabajar con esta herramienta es su simplicidad y la posibilidad de acceder al código fuente de la misma. Se pudo analizar en detalle todos los fuentes que forman parte de la solución, y de esta manera mantener bajo control la ejecución de la heurística. Adicionalmente, al tratarse de una metahurística en lugar de una heurística específica, la flexibilidad de la herramienta facilita adaptar la solución desarrollada por Gopalakrishnan a la realidad del problema presentado en este trabajo.

4.7.2. Funcionamiento del framework

En líneas generales el funcionamiento del framework puede resumirse en cinco grandes pasos a partir de los cuales itera la metaheurística, los cuales se describen someramente en la Figura 4.18.

Figura 4.18 – Esquema de ejecución de OpenTS

Crear Nueva Solución: al comienzo se parte de las soluciones iniciales que fueron descritas en la sección 4.3 y se continúa explorando las vecindades de acuerdo a los movimientos generados. Se debe recordar que todas las soluciones iniciales son factibles al problema no capacitado.

Generar Movimientos: en esta etapa se parte de una solución que puede o no cumplir con la restricción de capacidad y se analiza cuáles son las posibilidades para desplazar parte de la producción entre diferentes períodos. Es importante recordar que los movimientos generados deberán preservar la factibilidad al problema no capacitado, ya que esto afecta sensiblemente el desempeño de la heurística, en especial para aquellos movimientos que desplazan producción hacia períodos posteriores.

Evaluar los movimientos con la Función Objetivo: dada la solución actual y un posible movimiento generado en el paso anterior, este paso consiste en evaluar la función objetivo para la solución resultante de aplicar el movimiento.

Elegir mejor movimiento no Tabú: una vez evaluados todos los movimientos, la heurística avanza sobre el que produzca mejores resultados, es decir, aquel con menor costo en la función objetivo. En caso que este movimiento sea tabú, se analiza si el mismo mejora la solución actual. Si se produce una mejora se avanza con este movimiento, de lo contrario se descarta el mismo y se analiza el siguiente.

Aplicar movimiento seleccionado sobre la solución: en este paso se desplaza parte de la producción desde un período a otro a partir de la definición de cada tipo de movimiento generando una nueva solución actual. Una vez aplicado el movimiento se vuelve al primer paso y se continúa iterando hasta llegar a alguno de los criterios de parada.

4.7.3. Principales clases implementadas

En esta sección se presentan las principales clases implementadas para la solución computacional de la heurística.

`Main.java`

Esta clase implementa la heurística desarrollada (CLSP-TS) de acuerdo al pseudocódigo presentado en la sección 4.6.2. La misma incluye la generación de soluciones iniciales, búsqueda tabú básica y fase de diversificación e intensificación de acuerdo a lo descrito en las secciones 4.3, 4.5 y 4.6 respectivamente.

`CLSPTabuSearch.java`

Esta clase implementa la búsqueda tabú básica de acuerdo al pseudocódigo presentado en la sección 4.5.6. La misma incluye una función objetivo, una lista tabú, y una solución sobre la cual se itera. La misma es además la encargada de gestionar la lista tabú, así como el valor de la penalidad a lo largo de la ejecución de la heurística, como se describe en los distintos puntos de la sección 4.5.

`CLSPObjectiveFunction.java`

Esta clase implementa la función objetivo descrita en la sección 4.5.1 e incluye la información asociada a las restricciones del problema. Cada instancia de esta clase se construye a partir de un archivo conteniendo los diferentes parámetros que definen una función objetivo. Adicionalmente implementa dos funciones fundamentales: la primera permite obtener el resultado de evaluar una solución dada, es decir, los costos asociados a la misma. La segunda permite determinar la factibilidad de una solución dada al problema capacitado.

`CLSPSolution.java`

Permite representar una solución del problema. La misma incluye para cada ítem el tamaño de lote, activación de producción e inventario en cada período. Adicionalmente se incluye el valor de la función objetivo asociado a la misma, para evitar realizar cálculos cada vez que se deban comparar las mismas.

`CLSPTabuList.java`

Esta clase permite definir una lista tabú. La misma se implementa como un vector circular, el cual tiene un tamaño variable. Dicho tamaño se acota entre un mínimo y un máximo, y se gestiona de acuerdo a la calidad de las soluciones encontradas durante la ejecución de la heurística, como fue descrito en la sección 4.5.4.

`CLSPMoveManager.java`

Esta clase implementa el manejador de movimientos, el cual, como se describió anteriormente, es el encargado de generar todos los posibles movimientos a aplicarse sobre una solución dada. Estos posibles movimientos son los que definen la vecindad actual.

`CLSPMove.java`

Esta interfaz define los distintos procedimientos a ser implementados por los movimientos particulares. Para la versión original de la heurística no se definió ningún procedimiento a ser implementado. De todas formas, la misma fue extendida para realizar un estudio detallado de los distintos movimientos que son aplicados durante su ejecución.

`CLSPBackwardMove.java`

Esta clase implementa el tipo de movimiento *BackwardMove* descrito en la sección 4.4. El mismo está compuesto por un ítem y un período al cual se desplaza una cierta cantidad de producción desde algún período posterior.

`CLSPForwardMove.java`

Esta clase implementa el tipo de movimiento *FowardMove* descrito en la sección 4.4. El mismo está compuesto por un ítem y un período al cual se desplaza una cierta cantidad de producción ítem desde algún período anterior.

`CLSPSwapMove.java`

Esta clase implementa el tipo de movimiento *SwapMove* descrito en la sección 4.4. El mismo está compuesto por dos ítems y dos períodos entre los cuales se intercambia una cierta cantidad de producción.

`CLSPSchedule.java`

Esta clase permite representar un esquema de producción (schedule) para un determinado ítem. Como se describió anteriormente, un schedule representa la planificación de producción a lo largo del tiempo para un ítem determinado.

`CLSPComparator.java`

Esta es una clase auxiliar utilizada para comparar los distintos esquemas de producción (schedules) sobre los que avanza la fase de diversificación e intensificación. Dicha comparación se basa en el valor de la función objetivo asociado a cada esquema.

4.7.4. Modificaciones al framework

La interfaz original de OpenTS para el manejador de movimientos (*MoveManager*) únicamente recibe como parámetro una solución al problema. De esta manera no se contemplan las restricciones del problema para la generación de la vecindad de la solución actual.

Como fue mencionado en secciones anteriores, los movimientos sobre los que avanza la heurística deben preservar la factibilidad de las soluciones al problema no capacitado. Se debe implementar entonces algún mecanismo que garantice esta factibilidad.

Una primera alternativa consiste en mantener la definición de la interfaz original y analizar los movimientos generados por el *MoveManager* descartando aquellos que no preserven la factibilidad al problema no capacitado. Esta opción implica un impacto importante en el desempeño de la heurística, debiéndose verificar una a una la factibilidad para todas las soluciones en la vecindad. Adicionalmente, al no tener que preservar esta factibilidad el tamaño de la vecindad generada aumenta significativamente.

Debido a los problemas descritos se descarta la opción anterior y se decide implementar una alternativa más elaborada. La misma consiste en extender la interfaz definida originalmente para invocar al manejador de movimientos con la función objetivo. Como fue mencionado en la sección anterior, esta última clase incluye además la información asociada a las restricciones del problema. De esta manera a partir de la solución actual y de las restricciones del problema, el manejador de movimientos puede verificar el cumplimiento de la factibilidad de las soluciones al problema no capacitado a medida que se construye la vecindad.

Capítulo 5

Estudio computacional

En este capítulo se analiza experimentalmente el desempeño tanto de la heurística implementada como de la herramienta de resolución exacta, estudiando el comportamiento de las mismas ante diferentes escenarios del problema planteado. El objetivo de este estudio consiste en identificar aquellos aspectos con mayor impacto en la resolución de este tipo de problemas.

5.1. Definición de parámetros

Para poder analizar el desempeño de las diferentes estrategias de resolución, es necesario el estudio detallado de las mismas ante diferentes instancias del problema. En este sentido se destaca la publicación de dos repositorios de datos. El primero publicado por la Universidad Técnica de Darmstadt contiene una serie de instancias al problema capacitado sin backlogging [1]. El segundo, publicado por Wolsey, incluye un completo conjunto de instancias de diferentes problemas de planificación de producción que van más allá del problema de dimensionado de lotes capacitados [3]. Ambos repositorios incluyen entre otros datos las instancias del problema incluidas en el trabajo de Trigeiro de 1989 [29].

Ninguna de las fuentes de datos anteriores se adapta por completo a la realidad del problema presentado en este trabajo. En particular no se encontraron datos disponibles que permitieran simular diferentes tipos de capacidades por período, restricciones de inventario inicial o de inventario mínimo. Esto no sólo dificulta la obtención de datos para estudiar la heurística, sino que también imposibilita comparar cuantitativamente los resultados de la misma contra los obtenidos a partir de otras heurísticas estudiadas.

Debido a lo anterior los distintos experimentos se realizaron a partir de datos generados aleatoriamente. La generación de datos para la mayoría de los parámetros del problema se basa en el trabajo de Trigeiro [29] como se detalla en la siguiente sección.

5.1.1. Parámetros incluidos en el trabajo de Trigeiro

Demanda. En todos los casos se trabaja con una demanda promedio de 100 unidades para cada ítem y período generada a partir de una variable aleatoria uniformemente distribuida en el intervalo [90; 110]. Notar como de esta manera la variación media de la demanda es del 10%. Adicionalmente se generaron escenarios específicos para estudiar el impacto de la variación la misma (baja, media y alta variación de demanda) ante diferentes instancias del problema, los cuales se detallan en los cuadros de experimentos de la sección 5.3.3.

Tiempo de configuración. Para la mayoría de los escenarios se trabaja con tiempo de configuración (setup) constante igual a 30 unidades de tiempo. También en este caso se generaron escenarios específicos para analizar el impacto de la media y de la variación de estos tiempos. En estos escenarios, para cada ítem, el tiempo de setup se determina a partir de una variable aleatoria uniformemente distribuida en diferentes intervalos de acuerdo al valor medio y a la variación de los mismos.

Tiempo de producción y envasado. El mismo se fija en 1 unidad de tiempo para todos los escenarios estudiados. Este valor constante no sólo coincide con el definido por Trigeiro en su trabajo, sino que también se ajusta al resto de los autores estudiados.

Costo de producción y envasado. Al igual que con los tiempos de producción y envasado, este costo se fija en 1 unidad para la mayoría de los escenarios. Se generaron conjuntos de datos para estudiar el comportamiento de la heurística ante una alta variación del costo de producción y envasado. En este caso se sortearon aleatoriamente los tiempos a partir de una variable aleatoria uniformemente distribuida en el intervalo [0,5; 1,5]. Como fue comentado anteriormente, para un mismo escenario el total de ítems producidos es el mismo para todas las soluciones generadas. De esta manera cuando este costo es constante para todos los períodos, el mismo se vuelve irrelevante.

Costo inventario. En todos los casos este parámetro se calcula como un porcentaje de los costos de producción y de envasado. A diferencia de estos últimos, este costo sí impacta en las soluciones aunque el mismo no varíe a lo largo del tiempo. Costos de inventario elevados redundarán en soluciones donde la producción en cada período genere excedentes mínimos. Por contrapartida costos de inventario reducidos implican soluciones donde se produce hasta ocupar toda la capacidad, buscando reducir la cantidad de períodos donde se activa la producción, minimizando de esta manera los costos de setup.

Costo de configuración/Costo de inventario. A partir de la demanda media de 100 unidades, esta relación determina el tamaño de lote económico y a su vez permite establecer el tiempo entre órdenes (TBO) esperado para un problema no capacitado. Definimos entonces este ratio a partir de los costos de inventario y del TBO. Los costos de inventario ya fueron descritos en el punto anterior, mientras que para el TBO se fija el valor en 2 para la mayoría de los escenarios. Adicionalmente se elaboraron escenarios para analizar el desempeño de la heurística a partir de un TBO bajo, medio y alto. El impacto de este valor en la solución es análogo al que se describe en el punto anterior para el costo de inventario.

Capacidad de producción. Este parámetro se calcula a partir de la utilización promedio de capacidad más una cierta holgura. La utilización de capacidad promedio se estima a partir de un esquema de producción de *lote-por-lote*. Para la mayoría de los escenarios se trabaja con una holgura del 15% de la capacidad disponible. Por otro lado se diseñaron escenarios específicos variando la holgura entre 5% y 25%, teniendo en cuenta que a menor holgura más ajustada será la restricción de capacidad. Adicionalmente se estudió el comportamiento de la heurística cuando la capacidad disponible es menor a la utilización de capacidad promedio. Recordar que la presencia de los tiempos de configuración permite la aparición de soluciones factibles en estos casos [29].

Tamaño de problemas. Para la mayoría de los escenarios se realizaron pruebas sobre instancias del problema de tamaño mediano/pequeño (5 ítems y 12 períodos). Para los escenarios específicos de análisis del impacto del tamaño del problema, se generaron instancias variando tanto la cantidad de ítems como de períodos dentro del intervalo [3; 30].

5.1.2. Parámetros no incluidos en el trabajo de Trigeiro

Resta definir los valores para los parámetros presentes en el problema y no incluidos en el trabajo de Trigeiro. Estos son: inventario inicial, inventario mínimo y capacidad de envasado.

Capacidad de envasado. Para todos los escenarios este parámetro se determina siguiendo la misma lógica que para la capacidad de producción, con la diferencia de que en este caso se debe determinar una capacidad de envasado para cada tipo de producto.

Inventario mínimo. Este parámetro se define como un porcentaje de la demanda promedio. Para todos los escenarios se fijó el inventario mínimo como un 20% de la demanda promedio. Al igual que la demanda, el inventario mínimo varía para cada ítem en cada período.

Inventario inicial. El inventario inicial también se define como un porcentaje de la demanda promedio. Para la mayoría de los escenarios este porcentaje se fijó en un 20%. Adicionalmente se realizó un estudio específico para analizar el impacto de los mismos cuando las restricciones de capacidad se encuentran altamente ajustadas.

5.1.3. Factibilidad del problema

Como fue mencionado anteriormente determinar la factibilidad de un CLSP es en sí mismo un problema NP-completo. Para los problemas de menor tamaño se verifica la factibilidad de los problemas a través de la herramienta GLPK y las soluciones obtenidas con la misma. Por otra parte, para los problemas de mediano y gran tamaño se decidió descartar los experimentos para los cuales no se encontraron soluciones factibles, generando nuevos experimentos para el mismo escenario. Se debe tener en cuenta que esto ocurrió para menos del 1% de las instancias del problema, en aquellos escenarios para los cuales la restricción de capacidad se encuentra fuertemente ajustada.

5.2. Generación automática de datos

Para que el análisis de escenarios tenga validez estadística se debe evaluar la heurística bajo un número suficiente de experimentos. Además, es necesario generar un número adicional de instancias del problema para la determinación de valores de parámetros, así como para el análisis de las distintas reformulaciones al problema original presentadas en la sección 3.2.

Todos estos estudios requieren de un número significativo de instancias del problema, cuya generación manual no es viable. Para ilustrar esto último se debe tener en cuenta que para una única instancia del problema con 20 ítems y 30 períodos, se requiere la generación de 4370 valores para la definición de todos los parámetros involucrados (costos, demandas, capacidades, etc.).

Para facilitar esta tarea, se desarrolló una herramienta automática de generación de instancias del problema. Esta herramienta fue implementada a través de una planilla y de la programación de macros en Microsoft Excel y permite la generación de datos tanto para la heurística como para la herramienta GLPK.

La elección de Microsoft Excel como herramienta de automatización se basa principalmente en dos puntos. En primer lugar la programación de macros se realiza en base a procedimientos sencillos implementados en Visual Basic, sin necesidad de definición de clases o elaboradas estructuras de datos. En segundo lugar una planilla Excel facilita la manipulación y visualización de la información a generar sin necesidad de la generación de todos los datos. Esto permite detectar cualquier inconsistencia en la información en forma temprana.

Previo a la utilización de la herramienta, se debió verificar el desempeño del generador de números aleatorios de Microsoft Excel, así como los tiempos incurridos por la misma en la generación de datos para las instancias del problema de mayor tamaño.

Para el primer punto fue analizada la implementación de la función RANDOM utilizada para la generación de números aleatorios en Microsoft Excel. La misma se basa en un algoritmo de generación de números pseudoaleatorios desarrollado por B.A. Wichman [33][32]. Este algoritmo, presente también en otros paquetes de generación de número pseudoaleatorios, supera con éxito las pruebas DIEHARD [26]. Estas pruebas, desarrolladas por George Marsaglia y publicadas en 1995 [21], son la principal referencia en la evaluación de la calidad de los generadores de números pseudoaleatorios.

Para el segundo punto se analizaron los tiempos de generación de datos en función del tamaño del problema y se compararon los mismos con los tiempos de resolución tanto de la heurística como de la herramienta de resolución exacta. Para ambos casos se observa que los tiempos de generación de datos son despreciables en comparación con los tiempos de resolución y crecen linealmente a medida que aumenta el tamaño del problema. Esto último se grafica en la Figura 5.1, donde se observa la evolución de los tiempos de generación de datos para un problema con 30 períodos.

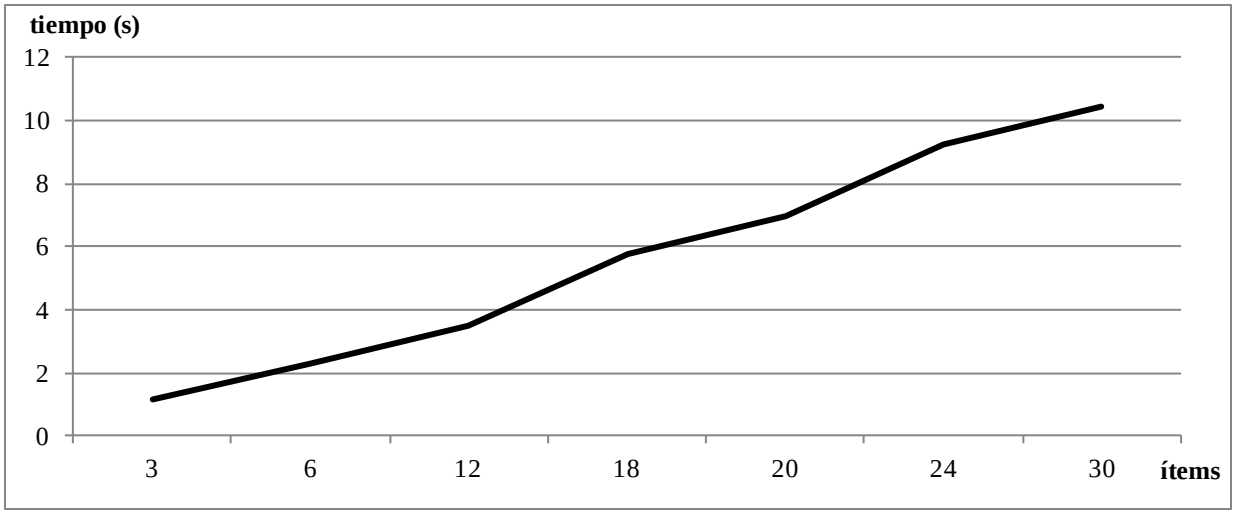

Figura 5.1 – Evolución de los tiempos de generación de datos.

Si tenemos en cuenta además que los tiempos de ejecución crecen de forma exponencial para la herramienta GLPK a medida que aumenta el tamaño del problema, podemos concluir que los tiempos de generación de datos son cada vez menos significativos. Esto se grafica en la Figura 5.2 donde se observa la evolución del ratio (GAP) entre el tiempo de generación de datos y el tiempo total de ejecución.

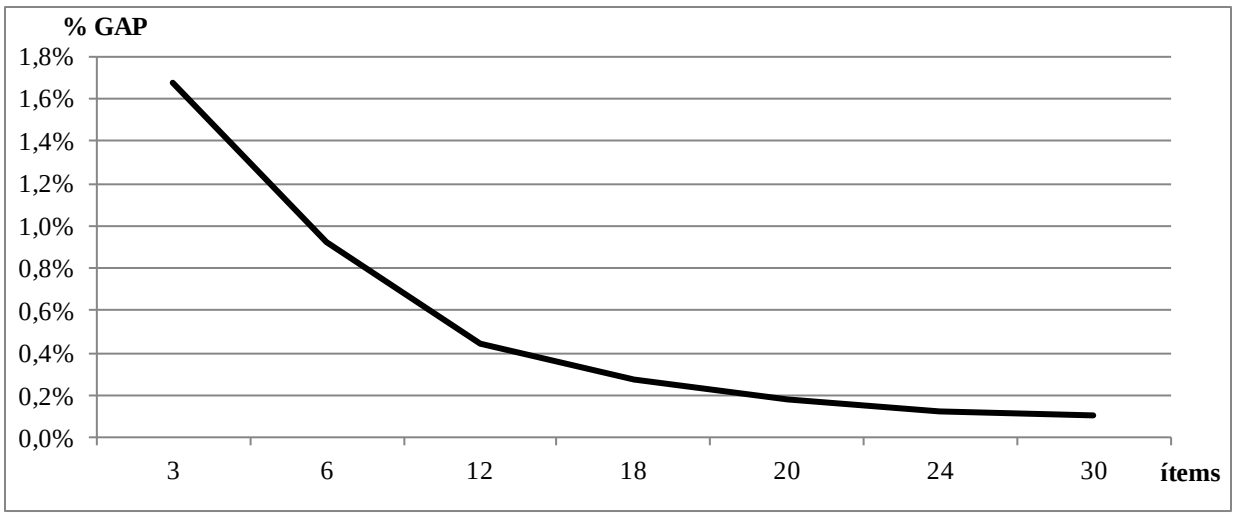

Figura 5.2 – Ratio entre los tiempos de generación de datos y tiempos de ejecución.

5.3. Resultados obtenidos

En esta sección se presentan los resultados obtenidos para los distintos escenarios analizados. Dicho análisis comprende la ejecución de pruebas para estudiar el desempeño de la heurística ante la resolución de diversas instancias al problema planteado.

Las características de cada uno de los escenarios fueron someramente presentadas en la sección anterior y se detallan en las siguientes secciones. Para cada uno de estos escenarios se ejecutaron pruebas sobre no menos de 80 instancias del problema, tanto con la heurística implementada como con la herramienta de resolución exacta GLPK.

El análisis de desempeño se centra principalmente en dos puntos: calidad de las soluciones obtenidas y tiempos de ejecución requeridos. Para cada escenario se detalla la brecha (GAP) entre la mejor solución obtenida por la heurística ($Z_{CLSP\text{-}TS}$) y la mejor cota encontrada por la herramienta de resolución exacta GLPK (Z_{GLPK}). Se debe tener presente que con excepción de los problemas de mayor tamaño, esta cota coincide con la solución óptima al problema. Adicionalmente, para las soluciones encontradas por la heurística implementada se incluye la mejora porcentual ($\%_{Diver}$) sobre las soluciones de la búsqueda tabú básica (Z_{BTS}) obtenida luego de la fase de diversificación e intensificación:

$$\%_{Diver} = (Z_{BTS} - Z_{CLSP\text{-}TS}) / Z_{BTS}$$

Los experimentos fueron llevados a cabo en un equipo Intel® Pentium® Dual Core de 64 bits con procesadores de 2.20 GHz de velocidad; sistema operativo Windows 7 de 64 bits Home Premium y 3,00 GB de RAM.

Como fue mencionado anteriormente, aquellas instancias del problema donde no se hallaron soluciones factibles con la herramienta de resolución exacta fueron descartadas y reemplazadas por una nueva instancia con similares características.

5.3.1. Análisis I: Holgura en las restricciones de capacidad

En esta sección se busca analizar el desempeño de la heurística en función de las restricciones de capacidad; más específicamente en función de la holgura entre la capacidad disponible y la capacidad necesaria para satisfacer la restricción de demanda. Para esto se plantean tres escenarios: holgura baja, holgura media y holgura alta.

En primer lugar, a partir del trabajo de Trigeiro [29] se define la holgura media en 15%. Es decir, la capacidad total disponible es un 15% mayor a la capacidad total necesaria para satisfacer la restricción de demanda a partir de un esquema de producción *lote-por-lote*. Una vez definida la holgura media definimos la holgura baja y alta en 5% y 25% respectivamente.

Como es de esperarse, la dificultad en la resolución del problema aumenta conforme disminuye la holgura en la restricción de capacidad. Esto efectivamente se constata con la herramienta de resolución exacta, donde la cantidad de problemas para los cuales no existen soluciones factibles aumenta a medida que las restricciones de capacidad se encuentran más ajustadas.

Es importante destacar que para establecer la capacidad disponible no se contemplan las restricciones de inventario mínimo, lo cual implica mayores desafíos a la hora de encontrar alguna solución factible. Por ejemplo si al 5% de holgura en la capacidad le sumamos la restricción de inventario mínimo, tenemos escenarios donde la capacidad requerida roza (pudiendo incluso superar) el total de la capacidad disponible. Como describe Trigeiro este tipo de escenarios pueden resolverse bajo ciertas condiciones, cuando las restricciones de capacidad son menos ajustadas en los períodos iniciales. Para simular esta característica en las instancias generadas, Trigeiro descarta el 25% de la demanda de los cuatro primeros períodos, desplazándola hacia los últimos cuatro. En lugar de replicar la estrategia de Trigeiro, en este trabajo se analiza el impacto del inventario inicial para este tipo de escenarios. Esto permite suavizar las restricciones de capacidad en los períodos iniciales sin necesidad de ajustar la demanda requerida.

Para reflejar las ventajas de la alternativa seleccionada se realiza el siguiente análisis comparando las ventajas entre el aumento del inventario inicial y el aumento de la holgura en la capacidad. Dicho análisis se centra en estudiar la factibilidad del problema para un conjunto determinado de instancias. En primer lugar se fija en 5% la holgura en las restricciones de capacidad y se aumenta de forma incremental el inventario inicial partiendo del 20% original. Luego se fija el inventario inicial en 20%, y se incrementa para todos los períodos la holgura en la capacidad desde el 5%. Los resultados obtenidos se ilustran en la Figura 5.3 y en la Figura 5.4.

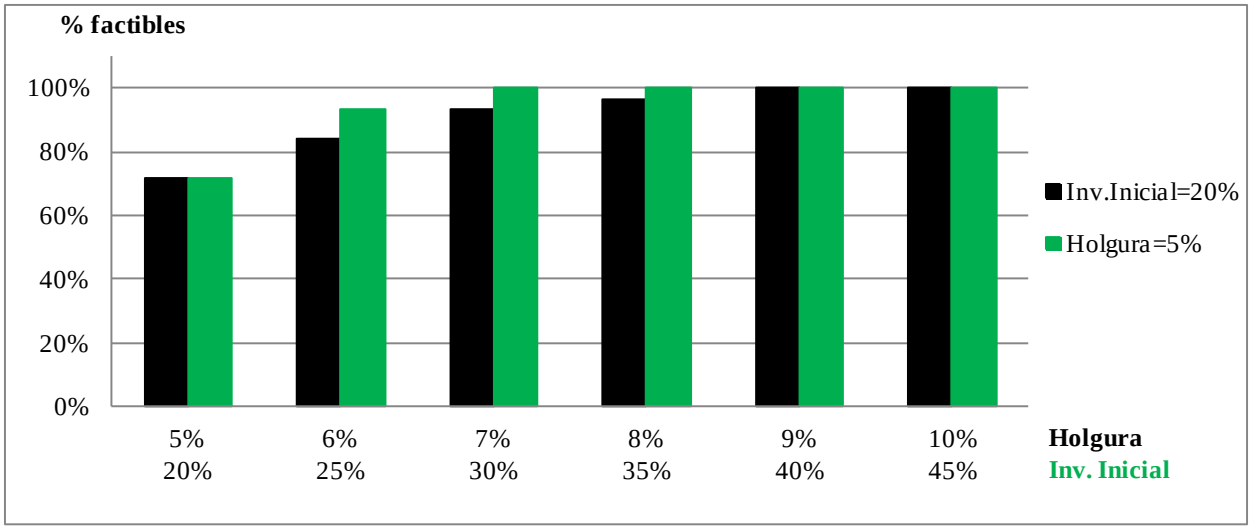

Figura 5.3 – Factibilidad del problema en función de la holgura de capacidad y el inventario inicial.

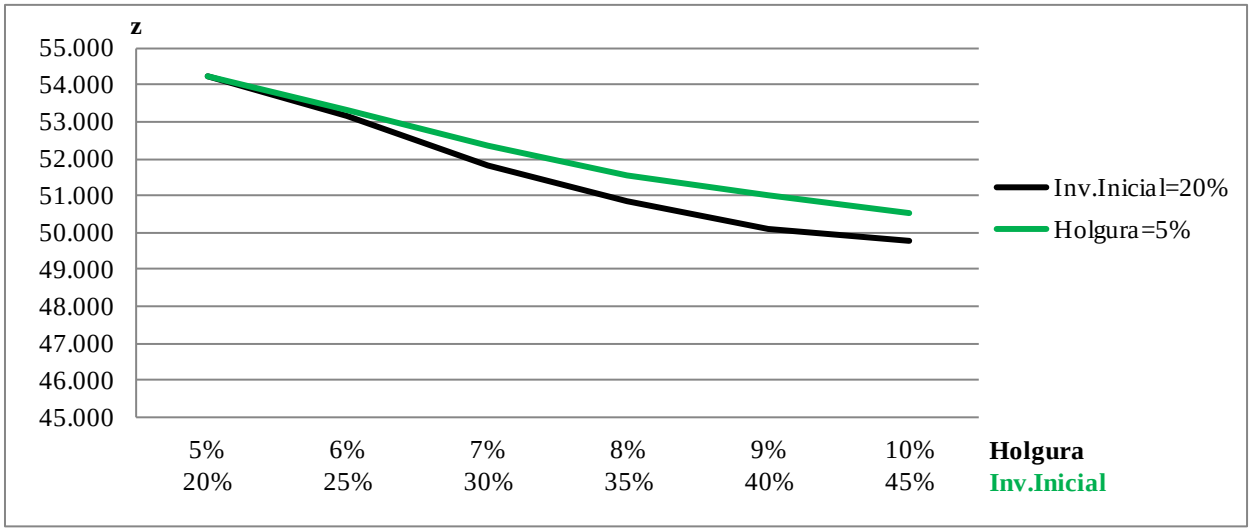

Figura 5.4 – Valor de la solución óptima en función de la capacidad y el inventario inicial.

A partir de la Figura 5.3 se observa cómo para un inventario inicial igual al 30% de la demanda promedio, se encontraron soluciones factibles para la totalidad de las instancias ejecutadas. Para obtener resultados similares se debe aumentar la holgura en la restricción de capacidad hasta casi un 10%. Los valores de la mejor solución encontrada (z) para estos dos escenarios, como puede observarse en la Figura 5.4 son similares.

Resulta entonces más eficiente incrementar el inventario inicial de 20% a 30%, en lugar de incrementar la holgura del 5% al 10%. Tener en cuenta que esta última alternativa implica cambios y potenciales costos durante todo el horizonte de tiempo, mientras que la primera únicamente afecta al primer período.

A partir de los resultados anteriores se reafirman las conclusiones de Trigeiro en su trabajo. Se pudo comprobar que menores exigencias de demanda en los períodos iniciales permiten la obtención de soluciones factibles para problemas con capacidad total utilizada mayor a la capacidad promedio disponible determinada a partir de un esquema de producción de *lote-por-lote*.

Otro punto a tener en cuenta para estas pruebas son los tiempos de ejecución. A partir de los escenarios analizados se observa una correlación marcada entre la holgura del problema y los tiempos de ejecución para la herramienta de resolución exacta. Esta dependencia se ilustra en la Figura 5.5 donde además se detalla la evolución de los tiempos de ejecución de la heurística (CLSP-TS) en función de la holgura en las restricciones de capacidad.

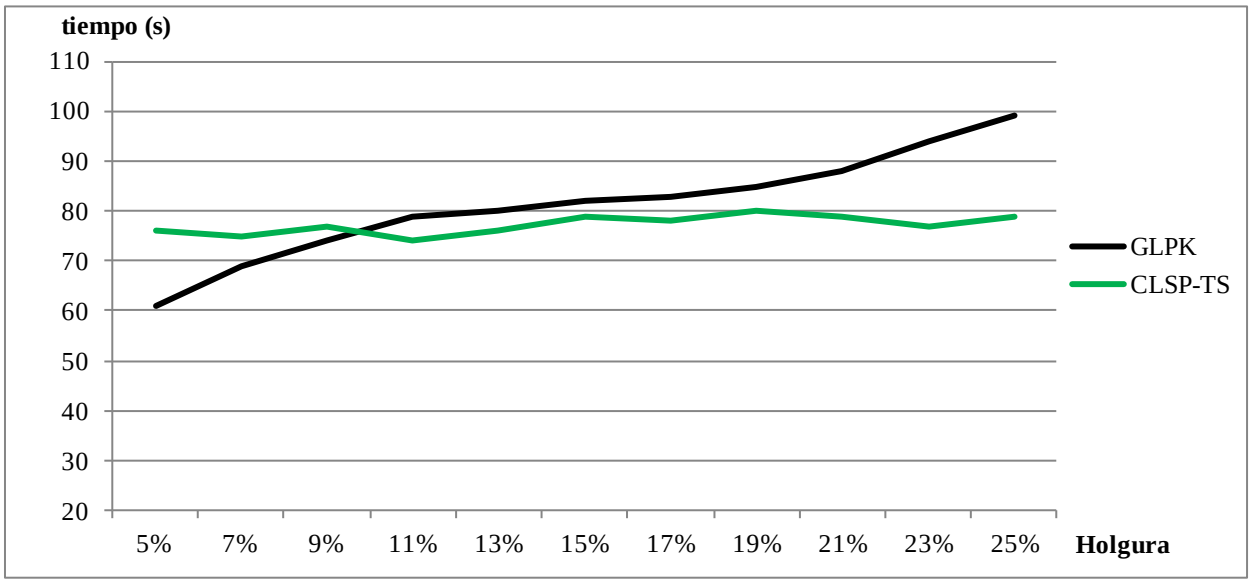

Figura 5.5 – Tiempos de ejecución en función de la holgura en la capacidad.

En la Figura 5.5 se observa cómo a medida que crece la holgura aumentan los tiempos de ejecución de la herramienta de resolución GLPK. Este incremento en los tiempos de ejecución se explica a partir del tamaño de la región factible. A medida que crece la holgura aumenta también la cantidad de soluciones factibles y por lo tanto mayores son los tiempos requeridos por la herramienta GLPK para cubrir la totalidad de esta región.

Este impacto en los tiempos no se observa para la heurística implementada (CLSP-TS), para la cual los tiempos de ejecución requeridos no varían significativamente según aumenta la holgura en las restricciones de capacidad. La explicación se debe en gran medida a que el tamaño de los lotes desplazados en los movimientos ejecutados por la heurística se ajusta a la capacidad disponible. Al aumentar la holgura en la restricción de capacidad se incrementa la capacidad ociosa de cada período. Este aumento en la capacidad disponible implica un incremento en el tamaño de los lotes a trasladar y no un aumento en la cantidad de lotes desplazados.

Finalmente es interesante destacar que el GAP entre la mejor solución hallada con la heurística y la solución óptima para los escenarios menos exigentes, no varía significativamente respecto al escenario de holgura mínima. Estos resultados se resumen en el Cuadro 5.1 y pueden explicarse a partir de las características de las soluciones óptimas. Una holgura reducida limita fuertemente los excesos de producción en cada período, favoreciendo los esquemas de producción de *lote-por-lote*. Este tipo de esquema se encuentra contemplado dentro de las soluciones iniciales a partir de las cuales avanza la heurística.

Holgura	**Inv. Inicial**	$\mathbf{Z_{CLSP\text{-}TS}}$	$\mathbf{Z_{GLPK}}$	**GAP**	$\mathbf{\%_{Diver}}$
25%	20%	48.424	45.753	5,52%	3,37%
15%	20%	52.591	47.784	9,14%	0,52%
5%	30%	55.478	51.989	6,29%	0,06%

Cuadro 5.1 – Impacto de la holgura en las restricciones de capacidad.

5.3.2. Análisis II: Tamaño del problema

El análisis presentado en esta sección busca estimar el impacto del tamaño del problema en el desempeño tanto de la heurística implementada como en la herramienta de resolución exacta. En particular en esta sección prestamos especial atención a los tiempos y recursos computacionales insumidos para la resolución de los problemas planteados.

Debemos recordar que el tamaño del problema se establece a partir de la cantidad de ítems y cantidad de períodos del mismo. A partir de la bibliografía relevada la categorización de un problema a partir de su tamaño es similar para los diferentes autores estudiados y se detalla en el Cuadro 5.2. Se debe destacar además que ninguno de los trabajos analizados contempla instancias del problema con más de 30 ítems o 30 períodos.

Items (*i*)	**Períodos (*t*)**	***i* · *t***	**Tamaño**
3 – 5	3 – 10	<60	Pequeño
5 – 15	10 – 20	<300	Medio
15 – 30	20 – 30	>300	Grande

Cuadro 5.2 – Tamaño del problema en función de la cantidad de ítems y períodos.

Comenzamos estudiando el desempeño de la herramienta de resolución exacta para diferentes configuraciones. Este análisis incluye el estudio de los tiempos de ejecución para la configuración por defecto de la herramienta, así como para configuraciones específicas para problemas de programación entera mixta (MIP).

Dentro de las configuraciones específicas de la herramienta GLPK para MIP se estudian la opción *gomory* y la opción *cuts*. La primera añade restricciones de planos de cortes enteros mixtos de Gomory al algoritmo de ramificado y acotamiento del motor de resolución del GLPK. La segunda agrega a la opción anterior una serie de planos de corte adicionales, como ser cortes enteros mixtos con redondeo, cortes de cobertura mixtos y cortes de clique.

Este estudio parte de instancias del problema con 3 ítems y se analizan los tiempos de ejecución a medida que aumenta la cantidad de períodos. En total se estudia el comportamiento de la herramienta para casi 4.000 instancias del problema. Los resultados obtenidos se presentan en el Cuadro 5.3.

Periodos	Cantidad pruebas	Tiempo total de ejecución (s)			Tiempo promedio por instancia (s)		
		T_{gomory}	T_{cuts}	$T_{default}$	t_{gomory}	t_{cuts}	$t_{default}$
6	1.000	12,14	13,00	10,77	0,01	0,01	0,01
8	500	14,25	15,97	10,32	0,03	0,04	0,02
10	500	43,49	50,11	38,43	0,09	0,10	0,08
12	500	114,4	132,9	187,2	0,23	0,27	0,37
14	300	175,0	202,1	533,7	0,58	0,67	1,78
16	300	521,7	601,1	3.633	1,74	2,00	12,11
18	200	908,0	1.052	-	4,54	5,26	-
20	200	2.466	2.742	-	12,3	13,7	-
22	200	6.649	7.720	-	33,3	38,6	-
24	100	13.800	15.634	-	138,0	156,0	-

Cuadro 5.3 – Tiempos de ejecución de GLPK según configuraciones de la herramienta.

No se observan mayores diferencias entre las tres configuraciones para problemas de tamaño pequeño. Sin embargo a media que el tamaño del problema aumenta, los tiempos de ejecución de la configuración por defecto empiezan a ser sensiblemente mayores al de las otras configuraciones. Esto último puede apreciarse en la Figura 5.6.

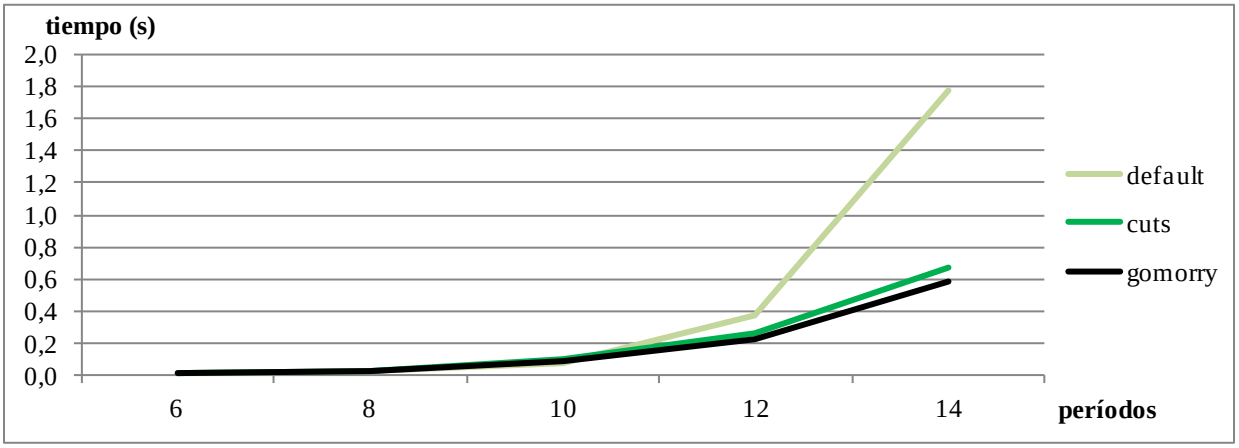

Figura 5.6 – Tiempos de ejecución según configuraciones de GLPK: problemas pequeños.

Debido al punto anterior se descarta en primer lugar la configuración por defecto del motor de resolución.

La Figura 5.7 describe el desempeño de las opciones *gomory* y *cuts* para problemas de mayor tamaño. Si bien no se observan diferencias significativas, los tiempos de ejecución con la opción *gomory* son menores a los tiempos con la opción *cuts*. Debido a esto se selecciona la primera de las opciones para el estudio comparativo con la heurística implementada.

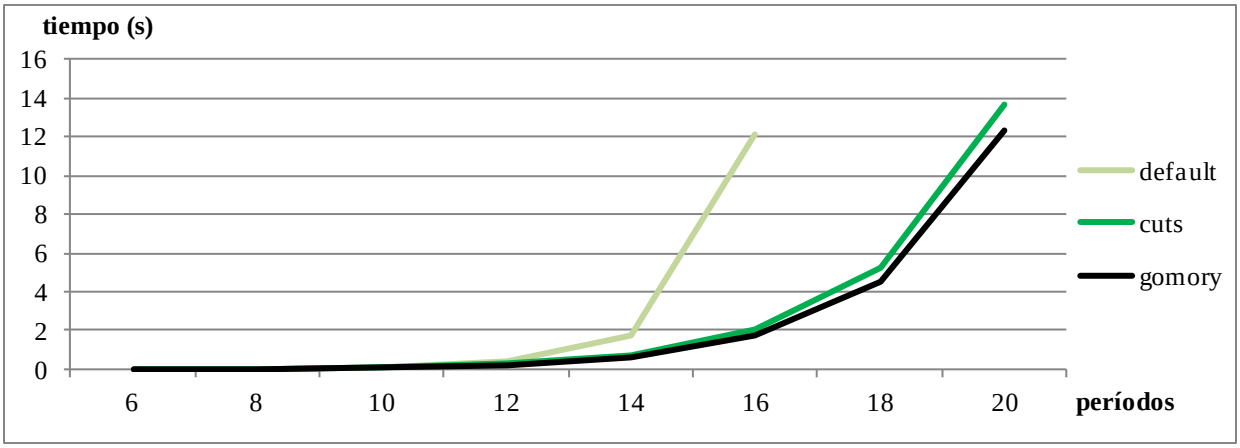

Figura 5.7 – Tiempos de ejecución según configuraciones de GLPK: problemas medianos.

Una vez determinada la configuración de la herramienta de resolución exacta, se procede a comparar los tiempos de ejecución de ésta ($Tiempo_{GLPK}$) con respecto a los tiempos de la heurística implementada ($Tiempo_{CLSP\text{-}TS}$). Los resultados de este estudio se detallan en el Cuadro 5.4 y se ilustran gráficamente en la Figura 5.8.

Items	Períodos	$Tiempo_{GLPK}$ (s)	$Tiempo_{CLSP\text{-}TS}$ (s)
3	6	0,01	19
4	9	0,5	33
5	10	12	50
5	12	42	89
6	12	125	133
7	12	281	200
7	15	590	315

Cuadro 5.4 – Comparación entre tiempos de ejecución de CLSP-TS y de la herramienta GLPK.

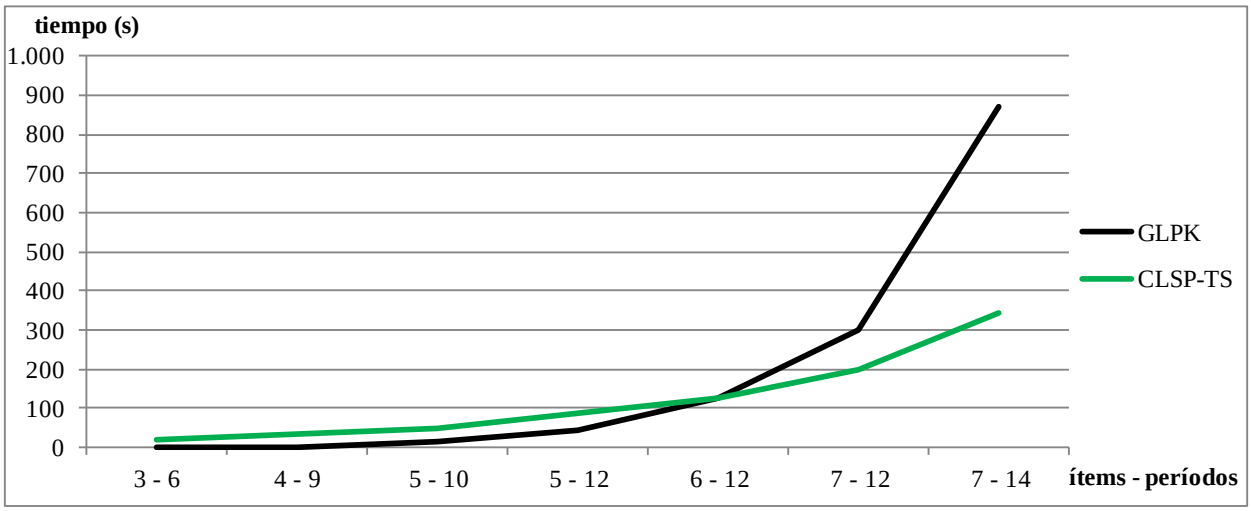

Figura 5.8 – Evolución de los tiempos de ejecución para CLSP-TS y herramienta GLPK.

Para problemas de tamaño pequeño se evidencia un mejor desempeño de la herramienta de resolución exacta. Sin embargo a medida que el tamaño del problema aumenta estas diferencias se minimizan, siendo el desempeño de la heurística muy superior a la herramienta de resolución exacta para las instancias del problema de tamaño mediano y grande.

Para instancias del problema con más de 20 ítems y más de 20 períodos, no se pudieron encontrar soluciones factibles dentro de las primeras 36 horas de ejecución. En estos casos la herramienta retorna las mejores cotas encontradas. Se debe tener en cuenta que para problemas de programación entera estas cotas pueden distar significativamente de la mejor solución factible [17].

Para evitar este problema se plantea como alternativa aumentar el tiempo de ejecución de manera de obtener alguna solución factible con la herramienta de resolución exacta. Esta opción fue descartada luego de observar la evolución de la ejecución de la herramienta GLPK. La Figura 5.9 ilustra la evolución de la mejor cota encontrada hasta el momento, mientras que en la Figura 5.10 se detalla la memoria utilizada durante la ejecución.

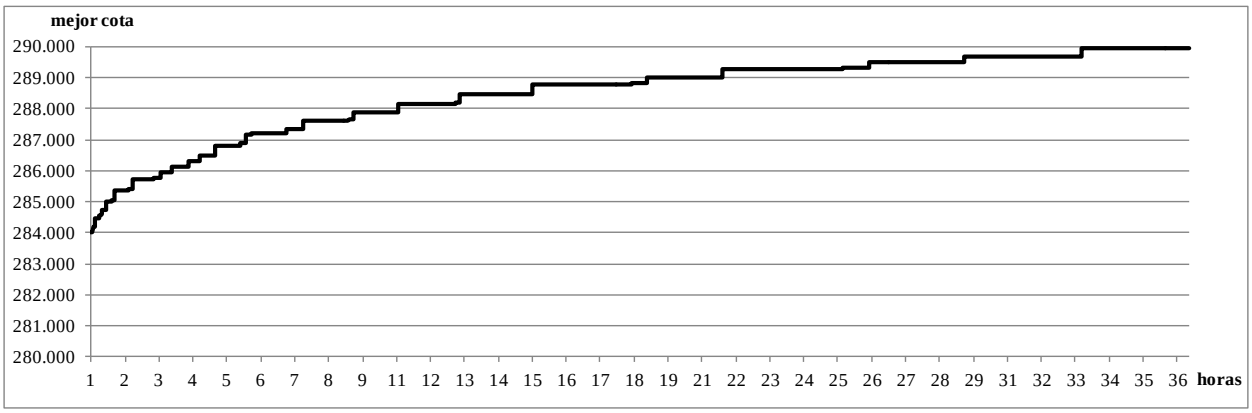

Figura 5.9 – Evolución de la mejor cota hallada por herramienta GLPK: 20 ítems y 30 períodos.

A medida que la ejecución avanza, se observa un incremento de orden logarítmico en la mejor cota encontrada. Recordar que esta cota corresponde a soluciones no factibles al problema capacitado, pudiendo distar significativamente de la solución óptima del problema. Durante el mismo horizonte de tiempo analizado se observa que la memoria consumida por la herramienta GLPK aumenta de forma lineal. Por lo tanto si se cuenta con recursos de cómputo limitados, aumentar los tiempos de ejecución no es una estrategia viable para encontrar cotas cercanas al óptimo.

Figura 5.10 – Evolución de la memoria utilizada por herramienta GLPK: 20 ítems y 30 períodos

Los problemas mencionados anteriormente se continúan presentando incluso para problemas de tamaño medio como se ilustra en la Figura 5.11. La misma detalla para la herramienta de resolución exacta la evolución del GAP entre la mejor solución factible hallada hasta el momento y la mejor cota durante la primera hora de ejecución. Las pruebas fueron realizadas para una serie de instancias del problema con 10 ítems y 15 períodos.

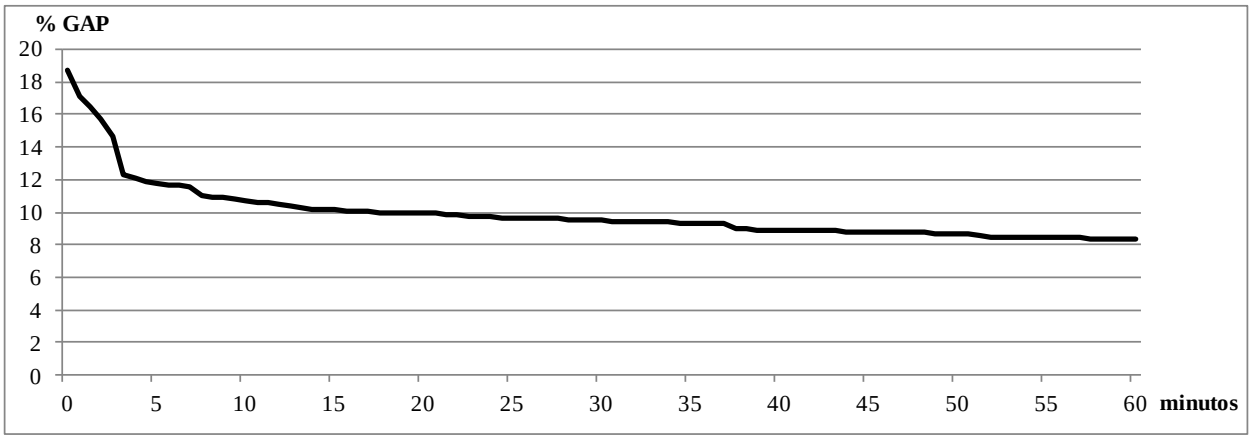

Figura 5.11 – Evolución de la mejor solución hallada por GLPK: 1 hora de ejecución.

A media que la herramienta itera, las mejoras en las soluciones encontradas son cada vez más espaciadas. La Figura 5.12 detalla la evolución del GAP anterior extendido durante el primer día de ejecución.

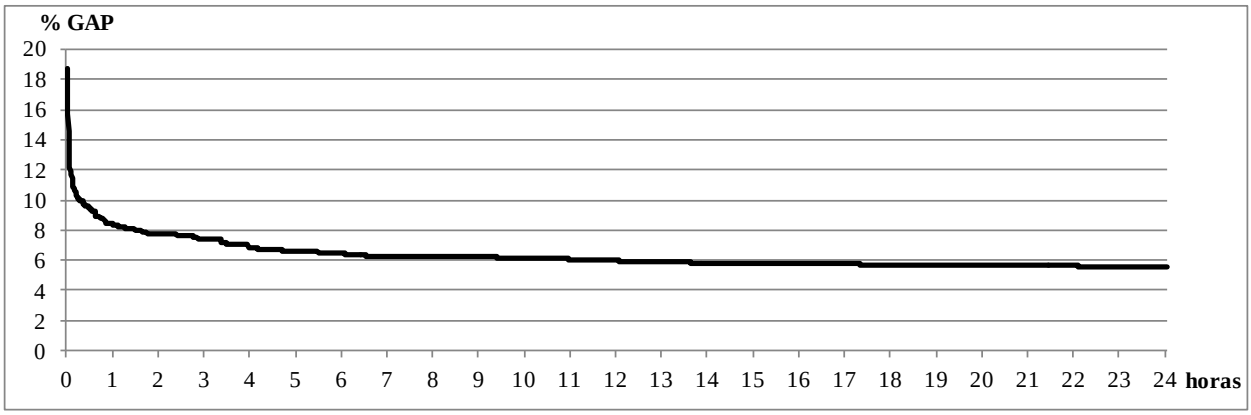

Figura 5.12 – Evolución de la mejor solución hallada por GLPK: 24 horas de ejecución.

Notar cómo luego de las primeras 12 horas de ejecución la mejora del GAP es de tan solo 0,2%. A medida que el tamaño del problema aumenta las dificultades para la resolución que presenta la herramienta GLPK se incrementan.

Es importante destacar que en estos mismos escenarios la heurística converge en una solución factible con menos de una hora de ejecución.

Concluimos esta sección analizando la calidad de las soluciones encontradas con la heurística, las cuales se resumen para los tres escenarios estudiados en el Cuadro 5.5.

Para los problemas de tamaño pequeño podemos comparar las soluciones halladas con las soluciones óptimas del problema encontradas con la herramienta de resolución exacta GLPK. En estos casos se observa cómo las soluciones que se obtienen con la heurística se acercan a la solución óptima del problema (GAP 0,93%).

Esta comparación con las soluciones óptimas del problema no es posible para problemas de tamaño medio y grande. En estos casos se debe comparar las soluciones halladas por la heurística con las mejores cotas encontradas por la herramienta GPLK. En el caso de los problemas de tamaño medio estas cotas se corresponden con la mejor solución factible hallada, no necesariamente solución óptima al problema capacitado (*). Por otro lado para problemas de tamaño grande estas cotas se corresponden a la mejor cota hallada por la herramienta GLPK, no necesariamente solución factible al problema capacitado (**).

Para estos últimos se observan importantes diferencias que llegan al entorno del 40% de GAP para los problemas de mayor tamaño. Como fue mencionado anteriormente, las cotas encontradas pueden distar significativamente de la solución óptima. Debido a lo anterior no es posible extraer conclusiones definitivas respecto a la calidad de las soluciones encontradas por la heurística para las instancias de mayor tamaño del problema.

Si bien resulta negativo no poder determinar la calidad de las soluciones encontradas para problemas de tamaño grande, se debe tener en cuenta que bajo ningún contexto real parece razonable fijar un plan de trabajo a largo plazo sin realizar ajustes del mismo a lo largo del tiempo. En estos casos los esquemas de producción hallados normalmente marcan la definición de lotes para los primeros períodos. Esta producción a su vez sirve como insumo para la determinación de lotes de los siguientes períodos.

Items	**Períodos**	$\mathbf{Z_{CLSP\text{-}TS}}$	$\mathbf{Z_{GLPK}}$	**GAP**	$\mathbf{\%_{Diver}}$
3	6	16.062	15.912	0,93%	2,20%
10	15	127.999	115.921 (*)	9,44%	3,98%
20	30	480.317	286.019 (**)	40,45%	2,04%

Cuadro 5.5 – Impacto del tamaño del problema.

5.3.3. Análisis III: Variación de la demanda

En esta sección se estudia el impacto que tiene la variación de la demanda en el rendimiento de la heurística. Para ello se analizan tres escenarios: variación media, variación alta y variación muy alta de la demanda, descartándose el análisis de escenarios con baja variación de la demanda. Esto se debe a que para escenarios de demanda constante existen métodos de resolución eficientes para los problemas de determinación de lotes capacitados. Definimos entonces la variación media de la demanda en 10% y las variaciones alta y muy alta en 20% y 30% respectivamente.

Una propiedad importante respecto a la variación de la demanda es la dependencia que existe o no entre los diferentes ítems. Si la variación de la demanda para cada ítem es independiente, es de esperarse que la variación de la demanda acumulada en cada período sea menor. De esta manera los cambios en la demanda se suavizan y se ajustan a la capacidad disponible. Por el contrario en algunos casos puede existir dependencia en la variación de la demanda para los diferentes ítems. Para estos casos se debe tener en cuenta que alta variación en la demanda implica la sucesión de períodos con alta exigencia de producción, seguidos de períodos con baja exigencia de la misma. Esta variación en la demanda normalmente no se ve acompañada con la variación en las capacidades de producción y envasado. Como consecuencia el horizonte de producción alterna entre etapas con capacidad ociosa, y períodos de capacidad altamente ajustada pudiendo en muchos casos exceder la capacidad disponible. Como se mencionó anteriormente cuando estos excesos en la demanda se presentan en los primeros períodos, no es esperable encontrar soluciones factibles al problema capacitado. Por otra parte, cuando la demanda es menor en los períodos iniciales dejando lugar a una mayor holgura en la capacidad, aumenta significativamente la probabilidad de hallar soluciones factibles.

Para los escenarios analizados se asume independencia en la variación de la demanda de los diferentes ítems. Los resultados obtenidos se resumen en el Cuadro 5.6, donde no se observan mayores diferencias entre las soluciones óptimas encontradas con la herramienta de resolución exacta. Podemos afirmar que la variación de la demanda no es una característica con alto impacto en el problema de dimensionado de lotes capacitados incluido en este trabajo. De manera similar sí se observan mejoras en las soluciones halladas por la heurística implementada, aunque ninguna de estas mejoras parecen ser significativas.

Var(D)	$Z_{CLSP\text{-}TS}$	Z_{GLPK}	GAP	$\%_{Diver}$
10%	52.643	47.818	9,16%	0,32%
20%	51.386	47.651	7,27%	1,31%
30%	49.606	47.121	5,01%	0,08%

Cuadro 5.6 – Impacto de la variación de la demanda.

5.3.4. Análisis IV: Variación de los costos unitarios de producción y envasado

En esta sección se estudia el impacto de la variación de los costos unitarios de producción y envasado, a partir del análisis de dos escenarios: variación baja y variación alta en los costos unitarios de producción y envasado. El primero se corresponde con costos constantes mientras que para el segundo se trabaja con variaciones de un 50% respecto a la media.

Como se observa en el Cuadro 5.7 no se aprecian diferencias significativas entre los resultados obtenidos en los escenarios anteriores. Debido a esto se decide descartar el análisis para un escenario de variación de costos intermedia.

Como fue mencionado en secciones anteriores a partir de ciertas hipótesis que cumple el problema estudiado en este trabajo, existe una cantidad de producción óptima que coincide con la demanda acumulada más el inventario mínimo del último período. Debido a esto para los casos donde los costos de producción no varían se tiene una cantidad de producción fija multiplicada por un valor constante. Podemos concluir que en estos casos los costos unitarios de producción y envasado no impactan en el esquema de producción de las soluciones óptimas.

La afirmación anterior no se aplica para los escenarios donde los costos de producción y envasado varían. Para estos casos como es de esperarse los esquemas de producción óptimos favorecen la producción en los períodos con menores costos. Si existe independencia en la variación de costos entre los distintos ítems, no es esperable que los períodos con menores costos de producción coincidan para todos los ítems. Para estos casos existe la posibilidad de absorber la producción de los ítems con menores costos de producción en un mismo período y desplazar las de mayor costo hacia otros períodos. Esto facilita la obtención de soluciones factibles cercanas a los esquemas de producción óptimos. Por contrapartida, de no existir independencia en la variación de costos de producción, es esperable la aparición de períodos con altos costos de producción así como períodos de costos reducidos para la mayoría de los ítems. Para estos últimos la capacidad de producción acotada en estos períodos obligará a desplazar la producción de algunos ítems hacia períodos con mayores costos de producción. De esta manera las mejores soluciones encontradas distarán de los esquemas de producción óptimos.

Para los escenarios analizados se asume independencia en la variación de los costos de producción y envasado entre los diferentes ítems. De esta manera observamos que las mejores soluciones encontradas por la heurística tanto para el escenario de variación alta como de costos constantes son similares.

A partir de estos resultados podemos concluir que los costos de producción y costos de envasado no son parámetros que impacten de forma significativa en el problema de dimensionado de lotes capacitados ni en la heurística implementada.

Var(C)	$Z_{CLSP\text{-}TS}$	Z_{GLPK}	GAP	$\%_{Diver}$
0%	52.833	47.854	9,42%	0,06%
50%	52.350	46.707	10,78%	0,29%

Cuadro 5.7 – Impacto de la variación de los costos de producción y envasado.

5.3.5. Análisis V: Media y varianza de los tiempos de configuración

Como fue mencionado al comienzo de este trabajo contemplar tiempos de configuración tiene importantes consecuencias en los problemas de dimensionado de lotes capacitados. En esta sección se estudia el impacto de los mismos en el desempeño de la heurística, contemplando tanto el valor medio como la variación de estos tiempos.

Nuevamente, basados en el trabajo de Trigeiro definimos un tiempo de configuración bajo y un tiempo de configuración alto en 20 y 40 unidades respectivamente. De manera análoga definimos la variación baja del mismo en 10% y variación alta en 30%.

Los resultados obtenidos figuran en el Cuadro 5.8 donde no se observan cambios significativos en los valores de las soluciones halladas tanto por la heurística como por la herramienta de resolución exacta. Desde este punto de vista podemos concluir que los cambios en los tiempos de configuración no tienen mayor impacto en el problema tratado en este trabajo.

De todas formas si bien no se aprecian cambios significativos en las soluciones halladas, sí se observa el impacto en los tiempos de ejecución. Dicho impacto fue especialmente notable en la herramienta de resolución exacta a medida que los tiempos medios de configuración aumentan. Se debe tener en cuenta que cuando los tiempos de configuración aumentan, no activar la producción de algún ítem en determinado período libera una mayor cantidad de capacidad. Esta mayor capacidad ociosa adicional permite una mayor diversidad en la utilización de la misma para la producción de otros ítems. Como consecuencia el tamaño de la región factible para estos problemas es mayor al de los problemas con menores costos de configuración dificultando la obtención de soluciones óptimas para la herramienta de resolución exacta.

Media	**Var**	**$Z_{CLSP\text{-}TS}$**	**Z_{GLPK}**	**GAP**	**$\%_{Diver}$**
20	10%	52.298	47865	8,48%	0,73%
20	30%	52.193	47.909	8,21%	0,61%
40	10%	52.806	47.928	9,24%	0,21%
40	30%	52.946	47.888	9,55%	0,05%

Cuadro 5.8 – Impacto de la media y varianza de los tiempos de configuración.

5.3.6. Análisis VI: Ratio entre los costos de configuración e inventario

En esta sección se estudia el impacto del ratio entre los costos de configuración y costos de inventario. Como se describió en secciones anteriores este ratio define el tiempo óptimo entre órdenes (TBO). A medida que aumenta el TBO mayores serán los beneficios de mantener la producción como inventario. Por contrapartida a medida que el TBO se reduce se favorece la activación de producción en un mayor número de períodos. En este sentido el estudio analiza los siguientes tres escenarios: TBO bajo, TBO medio y TBO alto.

En primer lugar definimos el valor de TBO bajo como el mínimo valor posible, es decir TBO igual a 1. A partir de este valor definimos al TBO medio y alto en 2 y 3 períodos respectivamente. Los resultados obtenidos se detallan en el Cuadro 5.9.

Como era de esperarse, los mejores resultados obtenidos tanto por la heurística como por la herramienta de resolución exacta se presentan para el escenario de TBO bajo. Notar que para este caso el esquema de producción óptimo se asemeja a un esquema de *lote-por-lote*. Recordando que la búsqueda tabú básica que forma parte de la heurística implementada parte de una solución con esta estructura es esperable encontrar soluciones cercanas al óptimo.

A medida que el TBO aumenta se observa cómo el valor de las soluciones óptimas halladas se incrementan. Esto también es esperable, ya que como fue explicado, un TBO alto favorece la generación de inventario por sobre la activación de producción. Esta generación de inventario para satisfacer la demanda de varios períodos requiere una capacidad de producción y envasado sensiblemente mayor a la necesaria para satisfacer la demanda de un único período. Por lo tanto para los escenarios con TBO medio y alto, y con una holgura media en la capacidad del 15%, no es esperable encontrar soluciones factibles con un esquema de producción similar al TBO óptimo.

De la misma manera que las soluciones factibles se alejan de los esquemas de producción óptimos a medida que el TBO aumenta, también lo hacen las soluciones encontradas por la heurística respecto a las soluciones óptimas del problema. Esto puede entenderse de acuerdo a la forma en que avanza la heurística. Inicialmente la misma avanza en búsqueda de soluciones factibles aquellas que minimizan los costos asociados a la penalidad. Esta busca de factibilidad no garantiza aproximarse a regiones cercanas a la solución óptima. Una vez se obtienen soluciones factibles, los movimientos posteriores buscan mejorar las soluciones basados en mejoras locales. Estas mejoras locales no siempre redundan en la obtención de óptimos globales.

TBO	$Z_{CLSP\text{-}TS}$	Z_{GLPK}	GAP	$\%_{Diver}$
1	32.255	32.236	0,06%	0,00%
2	52.384	47.854	8,65%	0,49%
3	80.176	68.454	14,62%	3,06%

Cuadro 5.9 – Impacto del tiempo entre órdenes (TBO).

Capítulo 6

Conclusiones

A partir de la bibliografía relevada, el estudio experimental del problema y los resultados obtenidos, en este capítulo se resumen las conclusiones a las que se llegaron en este trabajo.

6.1. Problema estudiado

A partir de la bibliografía relevada se ha observado cómo la problemática de la planificación de la producción ha evolucionado a lo largo del tiempo. Desde las primeras versiones del problema introducidas a mediados del siglo pasado, la complejidad de los distintos modelos presentados se ha ido incrementando en busca de modelar esta problemática de forma cada vez más realista.

Esta mayor complejidad de los diferentes modelos trae aparejado un incremento en la dificultad de resolución de los mismos. Para las versiones más básicas del problema fueron presentados métodos de resolución eficientes, como por ejemplo el algoritmo de Wagner-Whitin [31]. A medida que se modelan aspectos más realistas se ve limitada la existencia de métodos de resolución eficiente. En particular, se observa cómo la inclusión de tiempos de configuración transforma esta clase de problemas en NP-duros.

Podemos concluir que no es de esperarse la presencia de métodos de resolución eficientes para aquellas formulaciones que modelan el problema de forma más realista.

Diversos métodos de resolución fueron descritos a partir de diferentes trabajos presentados. Se destacan dos grandes líneas de métodos de resolución heurística: heurísticas computacionales y heurísticas matemáticas. Esta última clase de método fue seleccionada para el estudio de un problema específico de dimensionado de lotes capacitado.

6.2. Reformulación del problema

Fue estudiado el desempeño de una herramienta de resolución exacta para resolver un problema específico de dimensionado de lotes capacitados. Dicho estudio incluye el análisis del comportamiento de la herramienta para reformulaciones del modelo realizadas a partir de diferentes enfoques.

El primer enfoque, presente en la bibliografía relevada, busca aproximar la región factible hacia el casco convexo del problema introduciendo información adicional a partir de la redefinición de la variable de lote. Para este caso, se observa como la inclusión de restricciones de inventarios mínimos no permite la aplicación de este enfoque de la misma manera que lo realizan otros autores.

El segundo enfoque analizado busca minimizar el tamaño del problema a través de la reducción del número de variables.

El análisis de desempeño realizado se basa en dos puntos centrales: tiempos de ejecución y cantidad de recursos computacionales requeridos. Luego de la ejecución de diversos experimentos sobre una gran cantidad de instancias del problema, se observa un mejor desempeño de la herramienta GLPK para la reformulación con menor número de variables.

De esta manera podemos concluir que el tamaño del problema estudiado, definido a partir de la cantidad de variables, tiene mayor impacto en el desempeño de la herramienta GLPK que la estructura del mismo.

6.3. Selección de la heurística

Para la selección de la heurística el principal desafío fue poder contemplar todas las particularidades del problema presentado. Ninguna de las heurísticas estudiadas a través de la bibliografía relevada incluye el manejo de inventarios mínimos, inventario inicial o múltiples restricciones de capacidad. Debido a lo anterior se selecciona como base una heurística computacional presentada por Gopalakrishnan [15], la cual fue ajustada para contemplar todas las particularidades del problema estudiado.

Los principales componentes de la heurística son un generador de soluciones iniciales, procedimiento de búsqueda tabú básico que opera sobre las soluciones iniciales a través de movimientos definidos y finalmente un procedimiento de diversificación e intensificación. Este procedimiento trabaja sobre las mejores soluciones de la búsqueda tabú básica intentando mejorar las mismas probabilísticamente.

En este punto se pudo comprobar cómo las particularidades del problema tratado implican ajustes no menores en la heurística implementada. En particular se destaca el impacto de contemplar las restricciones de inventario mínimo para asegurar la factibilidad de las soluciones al problema no capacitado.

6.4. Implementación de la heurística

La implementación de la heurística no sólo incluye los ajustes necesarios para contemplar las particularidades del problema sino también la definición de los distintos parámetros que componen la misma. En este sentido se realiza un estudio diferenciado para alguno de los principales componentes de la heurística como son los movimientos implementados, criterios de parada, lista tabú y penalidad por incumplimiento de las restricciones de capacidad entre otros.

6.4.1. Movimientos implementados

Respecto a los movimientos desarrollados, además de los presentados por Gopalakrishnan en su trabajo, se analiza la viabilidad de implementar nuevos movimientos con el fin de atenuar las principales desventajas de los presentados por Gopalakrishnan. Estas desventajas se asocian a los procesamientos requeridos para determinar los períodos y el tamaño de los lotes a desplazar. Para abordar el primer problema se implementa un movimiento (*RandomMove*) que prescinde de cálculos para determinar los períodos a ajustar seleccionando los mismos de forma aleatoria. Para el segundo problema se implementa un movimiento que evita realizar cálculos para determinar el tamaño del lote a desplazar, desplazando cantidades fijas de producción (*FixedBatchMove*). Se analizó el desempeño de la heurística para diversos tamaños de lotes, y diferentes criterios de selección aleatoria de períodos.

Para ninguno de los casos se observaron mejoras significativas en los tiempos de ejecución, sin embargo, sí se pudo constatar una reducción en la calidad de las soluciones halladas. Se puede concluir que no es posible prescindir de los procesamientos requeridos por los movimientos presentados por Gopalakrishnan sin una reducción sensible en la calidad de las soluciones encontradas.

6.4.2. Criterios de parada

Para evitar que la heurística itere indeterminadamente se definieron dos criterios de parada, uno para la etapa de búsqueda tabú básica y otro para la fase de diversificación e intensificación. Ambos criterios se basan en la cantidad de iteraciones sin que se encuentren mejoras sobre la mejor solución hallada hasta el momento. La idea detrás de este criterio es acotar la ejecución de la heurística evitando detener la misma mientras se continúe mejorando la solución hallada. De esta manera se respeta el compromiso entre los tiempos de ejecución y la calidad de las soluciones encontradas.

Para la etapa de búsqueda tabú básica se observa que la cantidad de iteraciones necesarias para converger en una solución factible depende directamente del tamaño del problema. Este resultado es más que esperable si se tiene en cuenta que en cada paso la heurística aplica un único movimiento, y que éstos a lo sumo desplazan dos lotes de producción. Por lo tanto a media que aumenta la cantidad de ítems y la cantidad de períodos, aumenta también la cantidad de lotes a desplazar.

Por otro lado, para la fase de diversificación e intensificación se observa que la cantidad de iteraciones necesarias para que esta etapa converja en una solución factible no depende del tamaño del problema. Este resultado se entiende si se tiene en cuenta que cada iteración de diversificación e intensificación es precedida por una búsqueda tabú básica. La convergencia de esta búsqueda tabú básica sí depende del tamaño del problema, y es en esta fase de la heurística donde se absorbe el impacto del tamaño del problema.

6.4.3. Penalidad

La heurística implementada avanza sobre soluciones factibles al problema no capacitado, no necesariamente factibles al problema capacitado. Para lograr que el método converja hacia soluciones factibles se deben penalizar aquellas soluciones que violen las restricciones de capacidad.

Se pudo comprobar cómo valores pequeños de penalidad permiten intensificar la búsqueda de soluciones en una región particular. Como consecuencia, este tipo de valores son especialmente ineficientes para abandonar regiones poco prometedoras, en especial cuando las mismas implican la aparición de soluciones no factibles. Esta situación es inversa para valores altos de penalidad, los cuales permiten converger rápidamente en soluciones factibles, pero no necesariamente próximas a los óptimos.

Para aprovechar las ventajas de ambos tipos de valores fue presentada una estrategia de penalidad auto-ajustable. La misma difiere a la presentada por Gopalakrishnan en la versión original y se basa en el protocolo de control de transporte (TCP) para minimizar la generación y administración de estructuras de datos adicionales. Experimentalmente se puedo comprobar cómo este tipo de penalidad dinámica consigue mejores resultados que penalidades estáticas.

6.4.4. Lista tabú

A partir de la definición de los distintos movimientos fueron identificadas situaciones a ser evitadas por la heurística a medida que ésta avanza. En particular se presentaron los problemas asociados a desplazar lotes hacia (o entre) los mismos períodos. Debido a esto el método implementado mantiene un registro de los movimientos aplicados en lugar de las soluciones por las que se ha avanzado.

Adicionalmente, con base en el trabajo de Dell'Amico y Trubian [8], se implementa un lista de tamaño autoajustable. Este tipo de lista permite intensificar la búsqueda de soluciones en las regiones más promisorias, facilitando abandonar las mismas cuando las soluciones encontradas son en su mayoría no factibles.

Tanto el estudio para determinar los diferentes parámetros asociados a la lista tabú como para identificar el impacto de la misma en el desempeño de la heurística arrojaron resultados poco significativos.

6.5. Generación de datos

El análisis del rendimiento de la heurística implementada requiere un estudio minucioso de la misma ante un gran número de instancias del problema. Para la generación de estas instancias se analiza la disponibilidad de datos, destacándose el repositorio publicado por la Universidad Técnica de Darmstadt [1] y en especial el publicado por Laurence Wolsey [3]. Ambos repositorios incluyen entre otros datos las instancias del problema incluidas en el trabajo de Trigeiro [29].

Lamentablemente ninguno de los repositorios mencionados anteriormente se adapta por completo a la realidad del problema que trata este trabajo. En particular no se encontraron datos disponibles que contemplaran múltiples restricciones de capacidad por período, restricciones de inventario mínimo o de inventario inicial.

Debido a lo anterior, las diferentes instancias del problema sobre las cuales se estudia el desempeño de la heurística debieron ser generadas específicamente para este trabajo. La generación de estos datos fue descrita en las secciones 5.1.2 y 5.1.3 donde se destaca además la implementación de una herramienta de generación automática. El desarrollo de esta herramienta permitió la elaboración de más de 10.000 instancias del problema sobre las que se basan los diferentes estudios incluidos en este trabajo.

6.6. Estudio computacional

A partir del estudio de más de 1.500 instancias del problema fue analizado el desempeño de la heurística implementada ante distintos escenarios de prueba. Dicho análisis de desempeño se centra en dos puntos: tiempos de ejecución insumidos y calidad de las soluciones encontradas.

Como fue mencionado, la falta de trabajos que contemplen todas las particularidades del problema estudiado en este trabajo impide comparar los resultados obtenidos con los alcanzados por otros autores. De esta manera se comparara el desempeño de la heurística en relación con una herramienta de resolución exacta. Esto no sólo permite estudiar la heurística implementada sino también identificar cuáles son los aspectos con mayor impacto en el rendimiento de la herramienta de resolución exacta.

6.6.1. Análisis I: Holgura en las restricciones de capacidad

Este análisis incluye el estudio del impacto de la holgura en las restricciones de capacidad para tres escenarios: holgura baja, holgura media y holgura alta.

Para estos casos no se observaron diferencias significativas en la calidad de las soluciones encontradas por la heurística. El valor de las mismas se encuentra en el entorno del 7% por encima de la solución óptima del problema. Sí se observa un impacto en los tiempos de ejecución para la herramienta de resolución exacta, los cuales se incrementan a medida que la holgura en la restricción de capacidad

aumenta. Este impacto se explica por el crecimiento de la región factible que el incremento en la holgura trae aparejado. Es importante destacar que esta situación no se presenta en la heurística implementada, para los cual los tiempos insumidos en los tres escenarios son similares. Esto se debe a que el tamaño de los lotes desplazados en los movimientos implementados se ajusta a la capacidad disponible. Mayor holgura implica un incremento en la capacidad ociosa y por lo tanto mayor tamaño de los lotes trasladados, pero no implica una mayor cantidad de lotes a desplazar.

Adicionalmente fue realizado un estudio del comportamiento de la heurística y herramienta de resolución exacta cuando la capacidad requerida (estimada a partir de un esquema de *lote-por-lote*) alcanza e incluso supera el total de la capacidad disponible. Para estos escenarios como establece Trigeiro en su trabajo sólo es esperable encontrar soluciones factibles al problema cuando la demanda es menos exigente en los períodos iniciales. Los resultados de Trigeiro fueron comprobados para el problema estudiado en este trabajo, donde además se destaca el beneficio de incrementar la cantidad de inventario inicial en lugar de aumentar la holgura en las restricciones de capacidad. En estos casos se observaron mejores resultados al aumentar el inventario inicial un 50% en lugar de incrementar la holgura en un 100%.

6.6.2. Análisis II: Tamaño del problema

Este análisis incluye el estudio del impacto del tamaño del problema definido a partir de la cantidad de ítems y cantidad de períodos para tres escenarios: tamaño pequeño, mediano y grande.

Como era de esperarse el tamaño del problema es el parámetro con mayor impacto en el desempeño de la heurística, pero especialmente en el de la herramienta de resolución exacta.

Para problemas de tamaño pequeño se observa que la calidad de las soluciones encontradas por la heurística se acerca al óptimo del problema con un GAP del 0,93%. Esta diferencia se hace más evidente a media que el tamaño del problema aumenta, superando el 10% de GAP para problemas de tamaño mediano. Este decremento en la calidad de las soluciones encontradas es menor si se compara con el impacto en los tiempos de ejecución de la herramienta de resolución exacta. Con esta última, para problemas de tamaño mediano, si bien se obtienen soluciones factibles, no se encuentra soluciones óptimas durante las primeras 24 horas de ejecución. Esta falencia se agudiza a medida que aumenta el tamaño del problema, llegando al punto de no encontrarse soluciones factibles para problemas con más de 20 ítems y 20 períodos. Para todos estos casos, sí se encontraron soluciones factibles con la heurística implementada con menos de una hora de ejecución incluso para los problemas de mayor tamaño.

Debido a la ausencia de soluciones factibles por parte de la herramienta de resolución exacta no se puede evaluar de forma fehaciente la calidad de las soluciones encontradas por la heurística. En este sentido nos limitamos a comparar las mismas con las mejores cotas encontradas por la herramienta GLPK. Dichas cotas se corresponden a soluciones factibles al problema no capacitado, la cuales pueden distar sensiblemente de la solución óptima al problema capacitado.

De esta manera podemos concluir que la heurística implementada es una buena alternativa para problemas de tamaño mediano y grande. Para estos casos a partir de soluciones al problema relajado generadas de forma eficiente, es posible construir soluciones factibles al problema capacitado sin grandes requerimientos de cómputo.

6.6.3. Análisis III: Variación de la demanda

Este análisis incluye el estudio del impacto de la variación en la demanda para tres escenarios: variación baja, media y alta.

Para estos casos, si bien se observan algunas diferencias entre las soluciones óptimas encontradas con la herramienta de resolución exacta, éstas no parecen significativas. El GAP entre la mejor solución hallada para el escenario de variación baja y el escenario de variación muy alta apena supera al 5%.

De manera similar se encontraron mejoras en las soluciones halladas por la heurística implementada pero ninguna de éstas es significativa. Para los tres escenarios analizados el GAP porcentual entre las soluciones halladas por la heurística y la herramienta GLPK se encuentra en el entorno del 5% y 9%.

A partir de lo anterior podemos afirmar que la variación de la demanda no es un parámetro con alto impacto en el problema de dimensionado de lotes capacitados, así como tampoco para el funcionamiento de la heurística y de la herramienta de resolución exacta.

6.6.4. Análisis IV: Variación de los costos unitarios de producción y envasado

Este análisis incluye el estudio del impacto de la variación en los costos unitarios de producción y de envasado para dos escenarios: variación baja y variación alta.

Al igual que para la variación de la demanda, podemos afirmar que la variación en los costos unitarios de producción es un parámetro con impacto mínimo en el problema de dimensionado de lotes capacitados. Esto fue constatado tanto para la heurística como para la herramienta de resolución exacta. Para esta última, el valor de la solución óptima hallada varía por debajo del 1%. Adicionalmente tanto para el escenario de variación baja como de variación alta, el GAP porcentual entre las soluciones halladas por la heurística y la herramienta GLPK se encuentra en el entorno del 9%. Debido a estos dos puntos fue descartada la necesidad de estudiar un escenario de variación media en los costos de producción.

De esta manera se verifica lo establecido por Maes y Wassenhove, quienes destacan el mínimo impacto que tienen los costos unitarios de producción para esta clase de problemas [20].

6.6.5. Análisis V: Media y varianza de los tiempos de configuración

Este análisis incluye el estudio del impacto del valor medio y variación de los tiempos de configuración para cuatros escenarios: media de tiempos de configuración baja con variación baja, media de tiempos alta con variación baja, media baja con variación alta, media alta y variación alta.

Como fue mencionado en diferentes secciones de este trabajo y en otros trabajos estudiados, la inclusión de tiempos de configuración transforma esta clase de problemas en NP-duros. Sin embargo una vez contemplados estos tiempos no se observan mayores cambios asociados a la variación de los mismos. Este último punto puede confirmarse observando las soluciones óptimas al problema halladas por la herramienta GLPK, las cuales difieren en el entorno del 1% entre los cuatros escenarios.

Para todos los escenarios estudiados el GAP entre la mejor solución hallada por la heurística y la herramienta de resolución exacta se encuentra en el entorno del 8%. Esta diferencia coincide con el GAP promedio de los escenarios estándar, por lo tanto podemos concluir que tanto la media como la varianza en los tiempos de configuración no tienen mayor impacto en el problema ni en el rendimiento heurística implementada.

6.6.6. Análisis VI: Ratio entre los costos de configuración e inventario

Este análisis incluye el estudio del impacto de la variación del ratio entre los costos de configuración e inventario, el cual define el tiempo óptimo entre órdenes (TBO). Dicho estudio se realiza para tres escenarios: TBO bajo, medio y alto.

Para este caso los mejores resultados obtenidos por la heurística se presentan para el escenario de TBO bajo, donde el esquema de producción óptimo se asemeja a un esquema de *lote-por-lote*. La generación de una solución inicial con este esquema de producción permite aproximar a la heurística hacia una región cercana a la solución óptima.

A medida que el TBO se incrementa se aprecia cómo impacta en los valores de las soluciones óptimas del problema. Este punto se evidencia observando los valores de las soluciones halladas con la herramienta de resolución exacta, los cuales se triplican cuando el TBO aumenta de 1 a 3. Como fue mencionado en la sección 5.3.6, esto se debe a la ausencia de capacidad disponible para implementar esquemas de producción que respeten el tiempo entre órdenes óptimo. Este impacto también se observa en las soluciones encontradas por la heurística, para las cuales el GAP porcentual con respecto a las soluciones encontradas con la herramienta GLPK alcanza un 15%.

En este sentido podemos concluir que el TBO es un parámetro con un alto impacto tanto en la heurística implementada como en el problema en general, en especial si se analizan los valores de las soluciones óptimas.

Capítulo 7

Trabajos a futuro

Finalmente en este último capítulo mencionamos algunas de las posibles líneas de trabajo para extender o complementar el estudio presentado en este informe. Estas líneas incluyen aspectos de modelado del problema, diseño e implementación de la solución, y análisis de los resultados obtenidos.

7.1. Extensión del modelo

El modelo presentado en este trabajo incluye algunos de los aspectos más generales asociados a los problemas de planificación de producción. Entre estos aspectos se destaca el manejo de costos de producción, costos de inventario, costos de configuración y demandas variables, así como tiempos de configuración y restricciones de inventarios mínimos.

Existen sin embargo algunas variantes del problema dentro de las presentadas en la sección 1.2 que pueden extender el modelo presentado. En primer lugar mencionamos el traslado de configuración (carry over) de un período a otro, el cual es contemplado por la heurística original presentada por Gopalakrishnan [15]. Otro aspecto a incluir es la posibilidad de permitir que se cumpla con la demanda con un cierto atraso (backlog). Como fue mencionado esto aumenta la cantidad de soluciones factibles permitiendo resolver el problema capacitado con menores exigencias de capacidad. Adicionalmente, para todos los modelos presentados se asume que el total de la producción se destina a satisfacer la demanda o la restricción de inventario mínimo de algún período. Este supuesto no siempre se cumple en la práctica, en especial, en aquellas situaciones donde los costos de mantener inventario superan a los costos de desechar lo producido y volver a generar lo desechado.

Tanto los parámetros incluidos en el modelo como en las posibles extensiones mencionadas en el párrafo anterior comparten una propiedad: todos son conocidos al momento de la resolución del problema. Esta premisa puede no ser menor para problemas con gran incertidumbre en los datos. Sin dudas el valor con más incertidumbre para esta clase de problemas es la demanda, aunque también pueden incluirse en esta lista los costos y en menor medida la capacidad disponible. Todas estas cuestiones pueden ser contempladas con un modelo enfocado en programación estocástica.

7.2. Desarrollo de heurística matemática

Debido a la complejidad para contemplar las particularidades del problema presentado en este trabajo, la heurística implementada es una heurística computacional. Como fue mencionado en la sección 1.3.1, este tipo de métodos de "sentido común", si bien son más fáciles de comprender y ajustar, no necesariamente son los más eficientes para resolver esta clase de problemas.

En este sentido se presenta la posibilidad de rediseñar el algoritmo implementado apuntando más hacia una heurística matemática. La inclusión de planos de corte, generación de columnas y relajación de alguna de las restricciones no asociadas con la capacidad del problema son algunas de las alternativas. Otras alternativas combinan estas técnicas con la reformulación del problema. Quizás la más popular de acuerdo a la bibliografía relevada es la reformulación presentada por Dantzig-Wolfe [6] la cual es ampliamente estudiada y extendida por otros autores.

7.3. Implementación de heurística computacional

En esta sección se presentan algunos aspectos técnicos para complementar el trabajo. Los mismos incluyen cuestiones que van desde el entorno de programación hasta la implementación específica de algunos fragmentos de la heurística.

7.3.1. Lenguaje de programación

Debido a su simplicidad y potencialidad, el lenguaje seleccionado para la implementación de la heurística fue Java. Esta elección deja de lado algunas cuestiones asociadas al desempeño computacional de la heurística. La orientación a objetos, manejo de excepciones y algunos otros aspectos relacionados con el framework OpenTS utilizado en el desarrollo implican una sobrecarga a la solución, que bien podría evitarse.

En este sentido, la selección de un lenguaje menos sobrecargado, como puede ser C, o bien prescindir de un framework adicional, puede redundar en un mejor rendimiento por parte de la heurística. Otra alternativa posible es promover la concurrencia en la heurística utilizando múltiples hilos de ejecución. A modo de ejemplo, la etapa inicial del algoritmo busca mejorar seis soluciones iniciales a partir de seis búsquedas tabú independientes. Estas búsquedas pueden realizarse en paralelo mejorando de esta forma el rendimiento de la heurística.

7.3.2. Estructuras de datos

De la misma manera que para las herramientas de implementación, la selección de estructuras de datos prioriza la simplicidad por sobre la eficiencia. De esta manera las estructuras de datos seleccionadas se basan principalmente en vectores lineales, los cuales no necesariamente son la forma más eficiente de almacenar la información.

Estructuras más elaboradas pueden llegar a minimizar costos asociados al registro de información o a la búsqueda de algún elemento. Estas dos operaciones están presentes por ejemplo al momento de generar la lista de movimientos a aplicar en una iteración determinada o al momento de verificar si un movimiento particular se encuentra en la lista tabú.

Otras alternativas podrían contemplar el ordenamiento de la información a partir de algún criterio específico de acuerdo a los movimientos implementados. A modo de ejemplo si los movimientos priorizan aquellos períodos con capacidad ociosa, un ordenamiento de los diferentes períodos de acuerdo a la capacidad utilizada puede ser provechoso.

7.3.3. Movimientos implementados

Si bien se fue analizado el desempeño de algunos movimientos no incluidos en el trabajo de Gopalakrishnan, este es un punto que puede ser abordado con mayor profundidad en futuros trabajos. Los movimientos estudiados y descartados, *FixedBatchMove* y *RandomMove*, buscan abordar algunas de las desventajas presentadas por los movimientos implementados en la heurística original. Otros enfoques pueden dar a luz una nueva serie de movimientos posibles a ser estudiados.

En este sentido un análisis de la frecuencia de ejecución de los movimientos, así como las características de los mismos, tamaños de lotes a desplazar y períodos involucrados, pueden permitir mejores conclusiones.

7.4. Estudio computacional

Mencionamos en esta sección algunas cuestiones asociadas al estudio computacional para la problemática de planificación de producción en general y para el problema presentado en este trabajo en particular.

7.4.1. Problema de factibilidad

El problema incluido en este trabajo busca determinar los esquemas de producción más eficientes para satisfacer cierta demanda minimizando los costos en los que se incurre. Cuando para esta clase de problemas se contemplan aspectos como los tiempos de configuración el mismo se transforma en uno altamente complejo.

En estos escenarios, problemas en principio más sencillos, como la determinación de existencia de soluciones factibles se transforman en NP-completos. La resolución del problema de factibilidad puede determinar de forma temprana si corresponde o no avanzar en busca de esquemas de producción óptimos.

7.4.2. Escenarios estudiados

El análisis presentado en la sección 5.2 incluye una variedad de casos para los cuales se estudia el desempeño tanto de la heurística como de una herramienta de resolución exacta. Existen sin embargo algunos escenarios no incluidos en este trabajo sobre los cuales se podría extender el estudio.

Dentro de estos casos se podrían destacar los siguientes:

- Análisis de la variación en los tiempos de producción.
- Análisis de la variación de los costos de inventario.
- Análisis de la variación de la demanda cuando la misma no es independiente para cada ítem.
- Análisis de la variación de los costos de producción cuando ésta no es independiente para cada ítem.
- Análisis del desempeño de la heurística para instancias de problema con datos generados aleatoriamente.

7.4.3. Comparación de resultados

Como ya fue mencionado, uno de los principales desafíos del problema estudiado en este trabajo fue la falta de bibliografía que contemplara todas las particularidades del mismo. Esto no sólo implica la necesidad de ajustar las heurísticas estudiadas, sino que también impide la comparación de los resultados obtenidos con los presentados en otros trabajos.

En busca de comparar los resultados numéricos con los presentados por otros autores, una primera alternativa es ajustar el modelo para contemplar los problemas incluidos en otros trabajos. Esto implica la eliminación de restricciones de inventario mínimo, de inventario inicial y eliminación de alguna de las restricciones de capacidad. En estas condiciones los resultados obtenidos serían comparables por ejemplo con los presentados por Gopalakrishnan [15] y los presentados por Trigeiro =[29]=[29] .

Debido a lo anterior se optó por comparar los resultados hallados por la heurística con los obtenidos por una herramienta de resolución exacta. Para esto fue necesario entre otras cosas modelar el problema con la herramienta GLPK. Este último punto fue especialmente complejo al intentar contemplar una cantidad variable de tipos de productos y como consecuencia, una cantidad variable de restricciones de capacidad de envasado. Debido a esto la comparación de soluciones contra la herramienta de resolución exacta se basa en instancias del problema con dos tipos de productos diferentes. Extender el modelo para contemplar una cantidad variable de tipos de productos es otro desafío para resolver en trabajos futuros.

Por último otra alternativa si bien más compleja, también más realista, es ajustar el modelo a partir de información relevada desde alguna empresa particular. A partir de datos reales, los resultados obtenidos por la heurística pueden ser comparados con los resultados obtenidos empíricamente.

Referencias

[1] Description of CLSPL Test Instances, http://www.vwl.tu-darmstadt.de/bwl1/forschung/ti_clspl/clspl.html.

[2] GLPK (GNU Linear Programming Kit), http://www.gnu.org/software/glpk/.

[3] Lot-Sizing Problems: A Library of Models and Matrices, http://www.core.ucl.ac.be/~wolsey/lotsizel.htm.

[4] OpenTS - Java Tabu Search, Framework para la implementación de la metaheurística de Búsqueda Tabú en Java, http://www.coin-or.org/.

[5] Akturk, M. y Onen, S. Dynamic lot sizing and tool management in automated manufacturing systems. *Comput. Oper. Res.*, 29(8):1059–1079, Julio 2002.

[6] Dantzig, G. y Wolfe, P. Decomposition Principle for Linear Programs. *Operations Research,* 8:101-111, 1960.

[7] Degraeve, Z. y Jans, R. A New Dantzig-Wolfe Reformulation and Branch-and-Price Algorithm for the Capacitated Lot-Sizing Problem with Setup Times. *Operations Research*, 55(5):909–920, Setiembre/Octubre 2007.

[8] Dell'Amico, M. y Trubian, M. Applying tabu search to the job-shop scheduling problem. *Annals of Operations Research*, 41(3):231–252, Setiembre 1993.

[9] Dixon, P. y Silver, E. A heuristic solution procedure for the multi-item, single-level, limited capacity, lot-sizing problem. *Journal of Operations Management*, 2(1):23 – 39, 1981.

[10] Eisenhut, P. A Dynamic Lot Sizing Algorithm with Capacity Constraints. *A I I E Transactions*, 7(2):170–176, 1975.

[11] Federgruen, A. y Tzur, M. The dynamic lot-sizing model with backlogging: A simple o(n log n) algorithm and minimal forecast horizon procedure. *Naval Research Logistics (NRL)*, 40(4):459–478, 1993.

[12] Fleischmann, B. The discrete lot-sizing and scheduling problem. *European Journal of Operational Research*, 44(3):337–348, Febrero 1990.

[13] Florian, M., Lenstra, J., y Rinnooy Kan, A. Deterministic Production Planning: Algorithms and Complexity. *Management Science*, 26(7):669–679, 1980.

[14] Gonçalves, J. *Gestao de aprovisionamentos*. Publindústria, 2000.

[15] Gopalakrishnan, M., Ding, K., Bourjolly, J., y Mohan, S. A Tabu-Search Heuristic for the Capacitated Lot-Sizing Problem with Set-up Carryover. *Management Science*, 47(6):851–863, 2001.

[16] Harris, F. How Many Parts to Make at Once. *Operations Research*, 38(6):947–950, 1990 (Reimpresión de 1913).

[17] Jans, R. y Degraeve, Z. Modeling industrial lot sizing problems: a review. *International Journal of Production Research*, 46(6):1619–1643, Noviembre 2005.

[18] Kuhn, H. "A dynamic lot sizing model with exponential machine breakdowns". *European Journal of Operational Research*, 100(3):514–536, 1997.

[19] Lambrecht, R. y Vanderveken, H. *Heuristic Procedures for the Single Operation, Multi-item Loading Problem*. Departement voor Toegepaste Economische Wetenschappen van de Katholieke Universiteit Leuven, 1979.

[20] Maes, J. y Wassenhove, L. Multi-Item Single-Level Capacitated Dynamic Lot-Sizing Heuristics: A General Review. *The Journal of the Operational Research Society*, 39(11):pp. 991–1004, 1988.

[21] Marsaglia, G. The Marsaglia Random Number CDROM, http://stat.fsu.edu/pub/diehard/.

[22] Newson, E. *Lower Bounding the Capacitated Lot Size Problem*. BiblioBazaar, 2011.

[23] Pochet, Y. y Wolsey, L. Lot-size models with backlogging: Strong reformulations and cutting planes. *Mathematical Programming*, 40(1):317–335, Enero 1988.

[24] Qian, T., Philip, C., y Ye, Y. Worst Case Analysis of Forward Wagner Whitin Algorithm with Rolling Horizons, 1996.

[25] Rogers, J. A Computational Approach to the Economic Lot Scheduling Problem. *Management Science*, 4(3):264–291, 1958.

[26] Rotz, W., Falk, E., y Mulrow, J. A Comparison of Random Number Generators Used in Business. *Joint Statistical Meetings, Atlanta, GA*, 2001.

[27] Taft, E. The Most Economical Production Lot. *The Iron Age*, (101):1410–1412, 1918.

[28] Thizy, J. y Van Wassenhove, L. Lagrangean relaxation for the multi-item capacitated lot-sizing problem: A heuristic implementation. *IIE Transactions (Institute of Industrial Engineers)*, 17(4):308–313, 1985.

[29] Trigeiro, W., Thomas, L., y McClain, J. Capacitated Lot Sizing with Setup Times. *Management Science*, 35(3):353–366, 1989.

[30] Tzur, M. Learning in Setups: Analysis, Minimal Forecast Horizons, and Algorithms. *Management Science*, 42(12):1732–1743, 1996.

[31] Wagner, H. y Whitin, T. Dynamic Version of the Economic Lot Size Model. *Management Science*, 5(1):89–96, 1958.

[32] Wichmann, B. y Hill, D. Building a random-number generator. *BYTE*, 12(3):127–128, Marzo 1987.

[33] Wichmann, B. y Hill, I. Algorithm AS 183: An Efficient and Portable Pseudo-Random Number Generator. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 31(2):pp. 188–190, 1982.

[34] Wolsey, L. A. *Integer programming*. Wiley-Interscience, New York, NY, USA, 1998.

[35] Zangwill, W. A Deterministic Multi-Period Production Scheduling Model with Backlogging. *Management Science*, 13(1):105–119, Setiembre 1966.

Anexo A: Modelo original GLPK

```
/* --------CONJUNTOS-------- */

set P;
/* productos */

set C1;
set C2;
/* tipos de productos */

set T;
/* periodos */

/* --------PARÁMETROS: Costos-------- */

param sc{i in P, t in T};
/* costo de producción */

param vc{i in P, t in T};
/* costo por unidad preparada */

param ec{i in P, t in T};
/* costo por unidad elaborada */

param pc{i in P, t in T};
/* costo por unidad envasada */

param hc{i in P, t in T};
/* costo por unidad de inventario */

/* --------PARÁMETROS: Capacidades, inventario y demanda-------- */

param cape{t in T};
/* capacidad (en horas) del equipamiento para la elaboración de los productos
*/

param capp1{t in T};
param capp2{t in T};
/* capacidad (en horas) del equipamiento para el envasado de productos tipo 1
y 2 */

param ms{i in P, t in T};
/* inventario mínimo del producto i en el período t */

param is{i in P};
/* inventario inicial del producto i */

param d{i in P, t in T};
/* demanda del producto i en el período t */

param D{i in P, t in T};
/* demanda acumulada del producto i desde el período t hasta el final */
```

```
/* --------PARÁMETROS: tiempos-------- */

param et{i in P};
/* tiempo elaboración por unidad del producto i */

param pt{i in P};
/* tiempo envasado por unidad del producto i */

param gt{i in P};
/* tiempo de restablecimiento de equipamiento para el producto i */

/* --------VARIABLES-------- */

var x{i in P, t in T} >= 0, integer;

var y{i in P, t in T} >= 0, binary;

var s{i in P, t in T union {0}} >= 0, integer;

/* --------FUNCIÓN OBJETIVO-------- */

minimize z: sum{i in P, t in T}
(sc[i,t]*y[i,t] + (vc[i,t] + ec[i,t] + pc[i,t])*x[i,t] + hc[i,t]*s[i,t]) ;

s.t. balance{i in P, t in T}: s[i,t-1] + x[i,t] = d[i,t] + s[i,t];

s.t. activacion_produccion{i in P, t in T}: x[i,t] <= D[i,t] * y[i,t];

s.t. capacidad_elaboracion{t in T}:
sum{i in P} (y[i,t] * gt[i] + x[i,t] * et[i]) <= cape[t];

s.t. capacidad_envasado1{t in T}:
sum{k in P: k in C1} x[k,t] * pt[k] <= capp1[t];

s.t. capacidad_envasado2{t in T}:
sum{k in P: k in C2} x[k,t] * pt[k] <= capp2[t];

s.t. inventario_minimo{i in P, t in T}: s[i,t] >= ms[i,t];

s.t. inventario_inicial{i in P}: s[i,0] = is[i];

end;
```

Anexo B: Primer reformulación GLPK

```
/* --------CONJUNTOS-------- */

set P;
/* productos */

set C1;
set C2;
/* tipos de productos */

set T;
/* periodos */

/* --------PARÁMETROS: Costos-------- */

param sc{i in P, t in T};
/* costo de configuración */

param vc{i in P, t in T};
/* costo por unidad preparada */

param ec{i in P, t in T};
/* costo por unidad elaborada */

param pc{i in P, t in T};
/* costo por unidad envasada */

param hc{i in P, t in T};
/* costo por unidad de inventario */

/* --------PARÁMETROS: Capacidades, inventario y demanda-------- */

param cape{t in T};
/* capacidad (en horas) del equipamiento para la elaboración de los productos
*/

param capp1{t in T};
param capp2{t in T};
/* capacidad (en horas) del equipamiento para el envasado de productos tipo 1
y 2 */

param ms{i in P, t in T};
/* inventario mínimo del producto i en el período t */

param is{i in P};
/* inventario inicial del producto i */

param d{i in P, t in T};
/* demanda del producto i en el período t */
param D{i in P, t in T};
/* demanda acumulada del producto i desde el período t hasta el final */
```

```
/* --------PARÁMETROS: tiempos-------- */

param et{i in P};
/* tiempo elaboración por unidad del producto i */

param pt{i in P};
/* tiempo envasado por unidad del producto i */

param gt{i in P};
/* tiempo de restablecimiento de equipamiento para el producto i */

/* --------VARIABLES-------- */

var w{i in P, j in T, t in T} >= 0;

var y{i in P, t in T} >= 0, binary;

var s{i in P, t in T union {0}} >= 0, integer;

/* --------FUNCIÓN OBJETIVO-------- */
minimize z: sum{i in P, t in T} (sc[i,t]*y[i,t] +
(vc[i,t]+ec[i,t]+pc[i,t])*sum{j in T: j>=t}(w[i,t,j]) + hc[i,t]*s[i,t]);

s.t. balance{i in P, t in T}:
s[i,t-1] + sum{j in T: j>=t}(w[i,t,j]) = d[i,t] + s[i,t];

s.t. activacion_produccion{i in P, j in T, t in T: t>=j}:
w[i,j,t] <= d[i,t]*y[i,j] + ms[i,t];

s.t. capac_elaboracion{t in T}:
sum{i in P}(y[i,t]*gt[i] + sum{j in T: j>=t}(w[i,t,j])*et[i]) <= cape[t];

s.t. capac_envasado1{t in T}:
sum{k in P: k in C1} (sum{j in T: j>=t}(w[k,t,j])*pt[k]) <= capp1[t];

s.t. capac_envasado2{t in T}:
sum{k in P: k in C2} (sum{j in T: j>=t}(w[k,t,j])*pt[k]) <= capp2[t];

s.t. inventario_minimo{i in P, t in T}: s[i,t] >= ms[i,t];

s.t. inventario_inicial{i in P}: s[i,0] = is[i];

end;
```

Anexo C: Segunda reformulación GLPK

```
/* --------CONJUNTOS-------- */

set P;
/* productos */

set C1;
set C2;
/* tipos de productos */

set T;
/* periodos */

/* --------PARÁMETROS: Costos-------- */

param sc{i in P, t in T};
/* costo de producción */

param vc{i in P, t in T};
/* costo por unidad preparada */

param ec{i in P, t in T};
/* costo por unidad elaborada */

param pc{i in P, t in T};
/* costo por unidad envasada */

param hc{i in P, t in T};
/* costo por unidad de inventario */

/* --------PARÁMETROS: Capacidades, inventario y demanda-------- */

param cape{t in T};
/* capacidad (en horas) del equipamiento para la elaboración de los productos
*/

param capp1{t in T};
param capp2{t in T};
/* capacidad (en horas) del equipamiento para el envasado de productos tipo 1
y 2 */

param ms{i in P, t in T};
/* inventario mínimo del producto i en el período t */

param is{i in P};
/* inventario inicial del producto i */

param d{i in P, t in T};
/* demanda del producto i en el período t */

param D{i in P, t in T};
/* demanda acumulada del producto i desde el período t hasta el final */
```

```
/* --------PARÁMETROS: tiempos-------- */

param et{i in P};
/* tiempo elaboración por unidad del producto i */

param pt{i in P};
/* tiempo envasado por unidad del producto i */

param gt{i in P};
/* tiempo de restablecimiento de equipamiento para el producto i */

/* --------VARIABLES-------- */

var x{i in P, t in T} >= 0, integer;

var y{i in P, t in T} >= 0, binary;

/* --------FUNCIÓN OBJETIVO-------- */
minimize z: sum{i in P, t in T}  (sc[i,t] * y[i,t] + (vc[i,t] + ec[i,t] +
pc[i,t])*x[i,t] + hc[i,t]*(is[i] + sum{j in T : j <= t}(x[i,j] - d[i,j])));

s.t. activacion_produccion{i in P, t in T}: x[i,t] <= D[i,t] * y[i,t];

s.t. capacidad_elaboracion{t in T}:
sum{i in P} (y[i,t] * gt[i] + x[i,t] * et[i]) <= cape[t];

s.t. capacidad_envasado1{t in T}:
sum{k in P: k in C1} x[k,t] * pt[k] <= capp1[t];

s.t. capacidad_envasado2{t in T}:
sum{k in P: k in C2} x[k,t] * pt[k] <= capp2[t];

s.t. inventario_minimo{i in P, t in T}:
is[i] + sum{j in T : j <= t}(x[i,j] - d[i,j]) >= ms[i,t];

end;
```