Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID
ESCUELA DE INGENIERIAS INDUSTRIALES

Master de Investigación en Ingeniería de Procesos y Sistemas Industriales

SECUENCIACIÓN EN TIEMPO CONTINUO DE UNA SECCIÓN DE CRISTALIZACION DE LA INDUSTRIA AZUCARERA

Autor:

Alonso Alonso, Arturo

Tutores:

Prada Moraga, César De
Mazaeda Echevarria, Rogelio

Tutores:

Departamento de Ingeniería de Sistemas y Automática

Valladolid, 7 de Julio de 2017

Resumen

La optimización para problemas de scheduling ha sido ampliamente estudiada en la literatura de los últimos años, como consecuencia del exponencial desarrollo de los solvers (o softwares de optimización) y del creciente interés en la investigación operativa como herramienta para mejorar procesos.

En este Trabajo Fin de Master (TFM) se pretende realizar el scheduling para las tachas de primera cristalización en el cuarto de azúcar en una planta azucarera. Por medio de la formulación en GAMS de un modelo matemático en tiempo continuo que represente la planta real con el mayor detalle posible, se obtendrá una solución cuya factibilidad será comprobada introduciendo los resultados obtenidos en una simulación del sistema real implementada en EcosimPro.

De esta forma, se pretende demostrar la exactitud del modelo, de manera que si en algún momento se decidiese implementar en un entorno productivo, los resultados se adecuarían al comportamiento del mismo.

Palabras clave

Scheduling, planta azucarera, GAMS, tiempo continuo, optimización, EcosimPro.

Abstract

Optimization for scheduling problems has been widely surveyed over the last years, as result of both the high improvement of solvers (optimization software) and the increasing interest in operational research as a tool used in processes' improvement.

Within this document, we want to do the scheduling for the first crystallizer step (compose by three crystallization tanks) in a sugar production process. The development of a time continuous-optimization model by using GAMS will let us represent accurately the process, with all its boundaries. Once we get it, we will execute the optimization software and obtain the scheduling solution, which feasibility will be ensured by running a simulation in EcosimPro.

Therefore, we will give evidences about model's exactitude, which will ensure that the results would suit the real process activity in a hypothetical production application.

Key words

Scheduling, sugar plant, GAMS, continuous time, optimization, EcosimPro.

ÍNDICE

ÍNDICE DE FIGURAS

CAPÍTULO 1.

INTRODUCCIÓN Y OBJETIVOS.

1.1. Introducción.

Cualquier proceso industrial es el resultado de una serie de operaciones coordinadas que requieren de precisión y eficacia en su ejecución, para garantizar la calidad deseada en el producto de salida. Dichas operaciones suelen ser complejas, requiriendo conocimientos intrínsecos para su ejecución y dificultando en gran medida la flexibilidad del citado proceso.

Con el fin de optimizar todos estos aspectos, en el último medio siglo se ha comenzado a delegar trabajos anteriormente manuales en softwares específicos, aliviando la carga soportada por los encargados de la gestión de una empresa, evitando así posibles errores debidos al "factor humano" y permitiendo resolver tareas complejas, imposibles de abarcar para una persona de cualquier otro modo.

Tras su uso, dichos softwares se han demostrado extraordinariamente efectivos ya que, con un pequeño conocimiento de lenguajes de programación y con el requerimiento de breves periodos de tiempo para su ejecución, son capaces de generar importantes ahorros en los procesos, suponiendo un factor determinante de la calidad competitiva de una empresa.

En el caso concreto del scheduling (secuenciamiento de las actividades de un proceso en el tiempo), una leve mejora en el tiempo de ciclo (periodo de tiempo desde que se comienza la primera actividad del proceso hasta que finaliza la última actividad del mismo) puede suponer un importante ahorro para la empresa, ya que suelen ser realizados repetidamente.

Intrínsecamente, si se reduce el tiempo de ciclo podremos incrementar la producción, o bien reducir el tiempo que las máquinas están activas (con el ahorro económico que esto conlleva).

1.2. Objetivos.

En base a la justificación anterior, los tres Objetivos Principales (OP) del presente Trabajo Fin de Master (TFM) son:

- OP 1: realizar un modelo matemático en GAMS que represente el cuarto de azúcar de una planta azucarera. Más concretamente, el cuarto de cristalización en donde tiene lugar la primera critalización del azúcar (tachas de primera).

  Esto se llevará a cabo por medio de una formulación basada en tiempo continuo (evitando dividir el espacio temporal en un conjunto finito de elementos).

- OP 2: empleando datos empíricos tomados de la planta real, realizar una optimización en la que se tengan en cuenta las restricciones reales del sistema, y que de como resultado un scheduling para las tres tachas.

- OP 3: a partir del scheduling que obtengamos en la optimización, comprobar que efectivamente se trata de una solución factible, introduciendo esos resultados en una simulación del sistema realizada en EcosimPro y evaluando su output.

1.3. Estructura.

Este TFM se ha estructurado como sigue:

- Capítulo 2: INVESTIGACIÓN OPERATIVA.

  Este capítulo pretende dar una visión general de los fundamentos de modelado matemático y optimización necesarios, de manera que permita una mejor comprensión de la formulación realizada.

- Capítulo 3: OPERACIONES DE SCHEDULING.

Introducción a las operaciones de scheduling y a los procesos batch, así como unos breves ejemplos de formulaciones matemáticas para modelar y optimizar estos procesos.

- Capítulo 4: DESCRIPCIÓN DEL PROBLEMA.

  En este capítulo se describe de manera detallada el proceso de obtención del azúcar, haciendo especial incapié en la etapa objeto de interés: el cuarto del azúcar. También se aporta una breve descripción del enfoque que daremos para realizar el modelado y la posterior optimización.

- Capítulo 5: MODELO MATEMÁTICO Y RESULTADOS.

  Aquí exponemos por fin la formulación que se ha llevado a cabo para modelar el sistema real. Aportamos argumentos y justificaciones acerca de los valores considerados para algunos de parámetros y llevamos a cabo la optimización. Por último, evaluamos la factibilidad de los resultados obtenidos trasladándolos a una simulación en EcosimPro.

- Capítulo 6: CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO.

  Para terminar, se lleva a cabo una revisión de los resultados obtenidos, y se realiza una exposición de las posibles líneas de trabajo futuras con el objetivo de mejorar el presente trabajo.

CAPÍTULO 2

INVESTIGACIÓN OPERATIVA.

2.1. Introducción.

La investigación operativa ha impulsado en las últimas décadas las actividades de optimización, haciendo posible la resolución de problemas que hasta hace poco se antojaban irresolubles.

A continuación, describiremos este concepto, hablaremos de los principales algoritmos de optimización existentes y de las herramientas en las que hoy en día se apoya para resolver problemas (solvers).

2.2. La investigación operativa.

También conocida como Investigación de operaciones u Operations Research (nombre en inglés), se trata de una metodología surgida en torno a los años 50, y que se aplica a multitud de sectores de producción y manufactura, distribución, transporte, telecomunicaciones y salud, entre otros muchos otros (Sáez Aguado, 2014).

Esencialmente consta de dos partes: la modelización, o construcción de modelos matemáticos y la optimización, o resolución de estos modelos. Dichas partes se encuentran fuertemente relacionadas y son definidas brevemente a continuación.

2.2.1. Modelización matemática.

La modelización matemática trata de la formulación de modelos de optimización. Típicamente, un modelo de optimización tiene la siguiente forma:

$$
\begin{aligned}
\text{Maximizar o minimizar} \quad & f(x_1,x_2,\ldots,x_n) \\
\text{Sujeto a} \quad & g_1(x_1,x_2,\ldots x_n) \leq 0 \\
& g_2(x_1,x_2,\ldots x_n) \leq 0 \\
& \ldots \\
& g_m(x_1,x_2,\ldots x_n) \leq 0 \\
& h_1(x_1,x_2,\ldots x_n) = 0 \\
& h_2(x_1,x_2,\ldots x_n) = 0 \\
& \ldots \\
& h_p(x_1,x_2,\ldots x_n) = 0
\end{aligned}
$$

Como vemos, consta de varios elementos, los cuales podemos clasificar de la siguiente manera:

1. Variables de decisión ($x_1,x_2,\ldots x_n$): son las incógnitas cuyo valor se quiere determinar. Dependen del tipo de problema, con lo cual es muy común encontrarse variables de decisión de diferentes tipos (reales si pueden tomar cualquier valor, enteras si sólo pueden tomar valores enteros, binarias si pueden valer 1 ó 0…).

2. Función objetivo ($f(x_1,x_2,\ldots,x_n)$): es una función matemática que representa aquello que deseamos optimizar, ya sea maximizándola o minimizándola. Para que nos hagamos una idea, un ejemplo tipo de función objetivo que se desee minimizar sería un coste o una distancia recorrida, en tanto que una que se desee maximizar sería un beneficio.

3. Restricciones ($g_1(x_1,x_2,\ldots x_n) \leq b_1$): son las condiciones que debe cumplir la solución del problema. El $\leq$ es el tipo de restricción, y en este caso se trata de un signo de desigualdad, pudiendo también ser de igualdad. Cuanto más fielmente representemos las restricciones del sistema real, mejor se adecuará nuestra solución a la realidad.

En función de las características que posean estos tres elementos, existen diferentes tipos de problemas de programación matemática, pudiendo clasificarse los más importantes en:

a) Problemas de programación lineal: tanto la función objetivo como las restricciones deben de ser lineales, es decir, de la forma:

Función objetivo: $f(x) = c_1x_1 + c_2x_2 + ... + c_nx_n$

Restricciones: $a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \quad \leq, =, \geq \quad b_1$

$a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \quad \leq, =, \geq \quad b_2$

...

$a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n \quad \leq, =, \geq \quad b_m$

Siendo $c_1,...c_n$ y $a_{11}...a_{mn}$ los coeficientes de la función objetivo y de las restricciones, respectivamente, no pudiendo existir ninguna variable no lineal (sinusoidal, exponencial...) ni ningún producto o cociente entre variables.

El problema de programación lineal consiste en determinar los valores de las variables $x_1,x_2,...x_n$ continuas que proporcionen el mejor valor para la función objetivo f(x), cumpliendo todas las restricciones propuestas.

b) Problemas de flujo en redes: caso particular de la programación lineal, en donde las variables corresponden a los flujos definidos sobre un grafo o red. El concepto de grafo fue introducido por Euler para resolver el problema de los puentes de Königsberg (Euler, 1741) y es la base para resolver los ampliamente estudiados problemas de rutas de vehículos (Alonso, 2016).

c) Problemas de programación entera: otro caso particular de la programación lineal, en donde se restringe el formato de las variables continuas que teníamos en los problemas de programación lineal al considerarlas números enteros. Distinguimos dos tipos de variables, enteras generales (≥ 0) y enteras binarias ($\in [0,1]$), de forma que si ambos tipos de variables están presentes en el problema, este pasa a de nominarse "Problema de programación mixta entera".

d) Programación no lineal: en contraposición con los problemas de programación lineal, en este caso la función objetivo y/o las restricciones pueden ser funciones no lineales (puede existir producto/cociente entre variables, funciones seno, coseno, exponencial...).

En función de si las variables existentes son únicamente enteras o hay mezcla entre enteras y binarias, distinguimos los problemas NLP (Nonlinear Programming) o MINLP (Mixed Integer Nonlinear Programming), respectivamente y por sus siglas en inglés.

Debe prestarse mucha atención a la definición del problema, ya que, por ejemplo, la selección de unas variables erróneas o una elevada cantidad de las mismas nos puede condicionar de manera negativa el modelo. Por el contrario, la correcta definición de la función objetivo y las restricciones nos permiten representar mejor el sistema real, haciendo que los resultados obtenidos sean más fiables.

2.2.2. Optimización. Resolución de los modelos.

Una vez planteado el modelo de forma adecuada, se puede aplicar alguno de los métodos disponibles para resolverlo, obteniéndose como resultado unos valores de las variables y de la función objetivo planteada.

Si la modelización ha sido correcta, dichos resultados constituirán una solución real del problema.

2.2.3. Métodos de resolución para problemas lineales mixto enteros

2.2.3.1. Introducción.

Para resolver problemas de optimización lineal mixto-entera disponemos de diversos métodos que se suelen agrupar en tres grandes categorías: métodos exactos, heurísticas y meta heurísticas (Rocha, González , & Orijuela, 2011). Los primeros buscan soluciones óptimas por medio de modelos matemáticos resueltos mediante el empleo de solvers específicos, lo cual suele acarrear tiempos de computación elevados. Las heurísticas y meta heurísticas, por su parte, buscan dar soluciones aproximadas suficientemente buenas con reducidos tiempos de computación.

Otra diferencia entre estos tipos de métodos de resolución es que los métodos exactos precisan de un software específico para obtener algún resultado, mientras que las heurísticas y meta heurísticas no, pudiendo implementarse en cualquier lenguaje de programación y ejecutarse sin necesidad de tener algún algoritmo implantado.

A continuación, aparece un resumen de las principales subdivisiones que se suelen hacer para cada uno de los métodos citados.

2.2.3.2. Métodos exactos

Estos métodos pueden presentar limitaciones cuando se trata de resolver problemas de gran tamaño, dado que el tiempo de ejecución crece exponencialmente.

Típicamente se suele clasificar estos métodos en tres grupos (Rocha, González , & Orijuela, 2011):

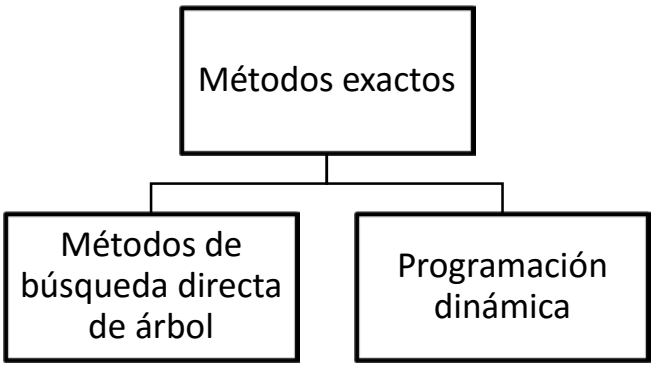

Figura 2.1: taxonomía para los métodos de resolución exactos.

Como el número de algoritmos existentes dentro de cada uno de estos métodos es muy elevado, hablaremos sólo de los algoritmos de Branch & Bound y de Branch & Cut para búsqueda directa de árbol y del método simplex para la programación lineal. La programación dinámica, por su parte, es menos empleada, por lo que sólo se mencionará.

- Métodos de búsqueda directa de árbol: la búsqueda se realiza sobre los nodos de un árbol de acuerdo con criterios específicos de cada método.

    - Algoritmo de Branch & Bound: basados en los conceptos de relajación, ramificación y poda, este algoritmo de enumeración parcial está basado en realizar una "poda" en los distintos nodos del árbol, en base a planos de corte y cotas dadas por determinadas restricciones. Estas podas permiten, resolviendo la relajación lineal del problema entero, determinar si una solución no es factible o es peor que la mejor solución factible encontrada hasta el momento, y detener la ejecución

ahí, pasando a evaluar otras ramas del árbol que sí puedan mejorar la solución actual.

- Algoritmo de Branch & Cut: similar al anterior, pero en vez de resolver una relajación lineal sencilla, emplea planos de corte para "podar" muchos más nodos al tratarse de una relajación muy fuerte.

  La Figura 2.2 muestra un ejemplo de este algoritmo, empleado por muchos solvers para resolver problemas de optimización. Ahí se pueden apreciar con más claridad conceptos como árbol (definido por las ramas que van surgiendo) y poda (si obtenemos una solución no factible, de ese punto no surgirán ramas).

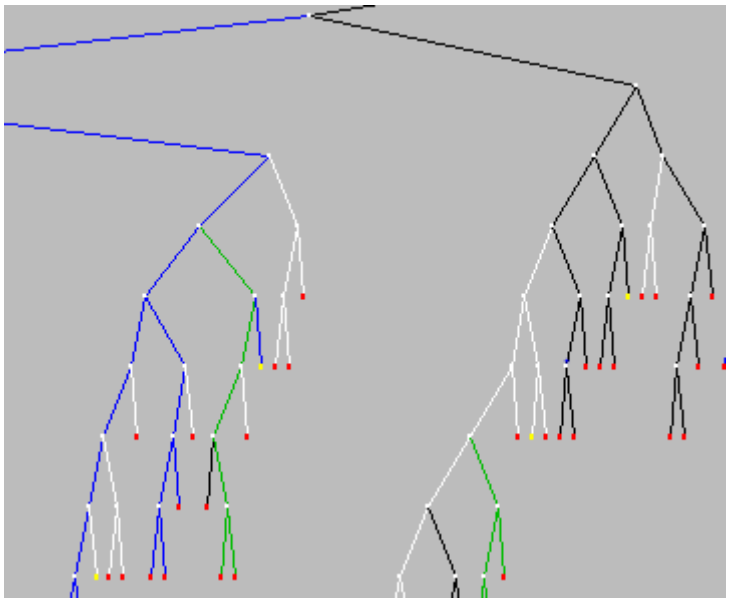

Figura 2.2: ejemplo práctico del algoritmo de Branch & Cut.

- Programación dinámica: hay que reprogramarlos para cada problema tratado, por lo que no resultan muy utilizados y no se hablará más de ellos.

2.2.3.3. Métodos heurísticos y meta heurístios.

Como ya se mencionó anteriormente, la finalidad de estos métodos es encontrar soluciones lo mejores posibles (que no tienen por qué ser óptimas, como ocurría en los métodos exactos) con tiempos de ejecución muy reducidos, aportando una mayor flexibilidad y haciendo que su ejecución resulte más cómoda.

La diferencia principal entre los métodos heurísticos y los meta heurísticos reside en que los primeros suelen quedarse atascados en óptimos locales, de manera que podemos creer que una solución es muy buena, pero resulta que sólo lo es en el entorno local, mientras que los segundos tratan de superar estos óptimos locales, evaluando todo el rango de soluciones posibles (entorno global).

En lo relativo a su clasificación, tanto de los métodos heurísticos como de los meta heurísticos existen multitud de implementaciones posibles, más numerosas a medida que pasan los años. Como no son objeto de estudio por parte de este TFM, simplemente nombraremos algunas de las más importantes, dejando para futuros trabajos su análisis.

- Métodos heurísticos.
    - De construcción: buscan construir una solución factible. Aproximadamente todas están basadas en el método Greedy.
        - Algoritmo de ahorros de Clark & Wright.
        - Inserción secuencial.
    - De mejora: a partir de una solución factible dada por los métodos de construcción, tratan de mejorarla.
        - 2 opt.
        - 3 opt.
        - Algoritmo de intercambios.

- Métodos meta heurísticos.
    - Recocido simulado (Simulated Annealing).
    - Búsqueda tabú.
    - Algoritmos genéticos.
    - Colonia de hormigas.

2.2.4. Métodos de resolución para problemas no lineales.

Para el caso de problemas no lineares se puede realizar la misma clasificación que la ya hecha para los lineares: métodos exactos, heurísticas y meta heurísticas, siendo objeto de nuestro interés los métodos exactos.

2.2.4.1. Métodos exactos

Los distintos algoritmos exactos que podemos encontrar no difieren mucho de los presentados anteriormente. La gran mayoría de ellos están basados en

Branch & Bound, difiriendo unos de otros en si se emplean problemas lineares (LP), no lineares (NLP) o no lineares mixtos (MINLP) para realizar la poda. Sin embargo, existen otros que también son utilizados, y que definimos muy brevemente a continuación (Lee & Leyffer, 2012):

- Outer Approximation: método propuesto por Duran y Grossmann para resolver problemas MINLP, basado en la suposición de que un problema MINLP puede asemejarse a un problema MILP (Mixed Integer Linear Programming) por medio de una sucesión de aproximaciones lineales exteriores, que acotan la región factible.

- Benders Descomposition: método propuesto por Benders para resolver problemas MINLP que son lineares en las variables "fáciles" y no lineares en las variables "difíciles".

- Métodos basados en planos de corte, para resolver tanto problemas NLP como MINLP.

2.2.5. Herramientas de implementación.

Desde su creación, el método simplex, los algoritmos de Branch & Cut, los algoritmos de Branch & Bound… han sido incluidos en una gran variedad de programas (cada uno de ellos incorporando alguna modificación en el algoritmo de resolución), a los cuáles se les denomina comúnmente solvers, dado que se encargan de resolver problemas de optimización.

2.2.5.1. Solvers existentes para resolver problemas lineales.

Podríamos dividir los principales solvers existentes en dos grupos (Meindl & Templ, 2012):

- Solvers comerciales (código cerrado):

    - CPLEX: comercializado por IBM, se caracteriza por poder soportar problemas con un gran número de variables. Incorpora además varias interfaces para conectar el solver con diferentes programas y lenguajes.

    - Xpress-IVE: desarrollado íntegramente pare resolver problemas lineales, incorpora también interfaces y librerías para ampliar

sus prestaciones. Las herramientas de tuning y mapping son muy útiles y frecuentemente empleadas para evaluar los valores de los parámetros del sistema y representar las soluciones, respectivamente.

- Gurobi: solver empleado tanto para resolver problemas lineales como no lineales. Está escrito en C y es accesible para diversos lenguajes de programación.

- Solvers gratuitos (código abierto):
  - LP_SOLVE.
  - GLPK.
  - CLP.
  - SCIP.
  - SoPlex.

2.2.5.2. Solvers existentes para resolver problemas no lineales.

Como ocurría en el caso anterior, debido al creciente interés en la optimización de procesos han ido surgiendo en los últimos 40 años multitud de solvers para resolver tanto problemas NLP como MINLP.

- Algunos solvers para resolver problemas NLP (Lee & Leyffer, 2012):
  - SNOPT.
  - FilterSQP.
  - CONOPT.
  - IPOPT.
  - LOQO.

- Algunos solvers para resolver problemas MINLP (R. Bussieck & Vigerske, 2014):

    - AlphaBB: desarrollado por la Universidad de Princeton, emplea algoritmos de Branch & Bound para realizar la optimización, utilizando problemas NLP para realizar las podas.

    - ANTIGONE: desarrollado por la Universidad de Princeton, emplea algoritmos de Branch & Bound para realizar la optimización, utilizando problemas MINLP para realizar las podas.

    - BARON: desarrollado por la Universidad Carnegie Mellon, emplea algoritmos de Branch & Bound para realizar la optimización, utilizando problemas LP para realizar las podas.

    - KNITRO: desarrollado por Zienea Optimization Inc., resuelve los problemas de optimización por medio de Branch & Bound, pudiendo utilizar tanto problemas lineares como no lineares para realizar la poda.

2.2.6. GAMS.

Como hemos podido observar, existen diferentes variantes de problemas de optimización, y para cada uno de ellos existen multitud de solvers capaces de resolver dichos problemas. Disponer de todas las licencias es prácticamente imposible debido al elevado precio de las mismas, por lo que puede ser habitual trabajar únicamente con un solver, perdiendo flexibilidad en el caso de que otro pudiera darme una solución mejor.

En este contexto nosotros vamos a utilizar para la realización de este TFM la herramienta GAMS (GAMS, 2017). GAMS (General Algebraic Modeling System) es un sistema de modelado para programación matemática en alto nivel, con la principal ventaja de que nos permite seleccionar el solver que queremos utilizar entre un amplio abanico de posibilidades disponibles. De esta forma podemos evitar centrarnos en un único programa o en un único tipo de problema de optimización, obteniendo una mayor flexibilidad en función de nuestras necesidades.

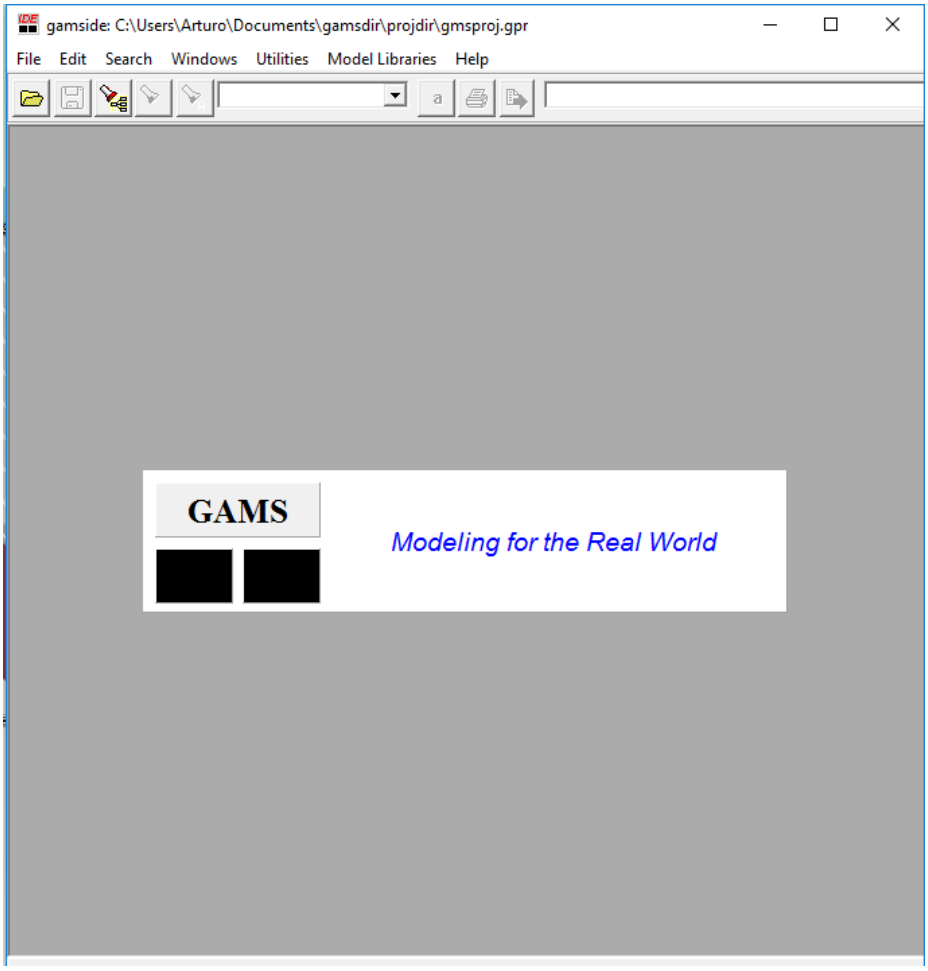

Figura 2.3: vista general de GAMS.

Dentro de la lista de solvers disponibles, el propio GAMS nos indica los problemas de optimización para los que ese solver está diseñado. De esta manera, sabemos en todo momento que si queremos optimizar, por ejemplo, un problema mixto entero no lineal (MINLP) qué solvers están preparados para resolver dicho problema.

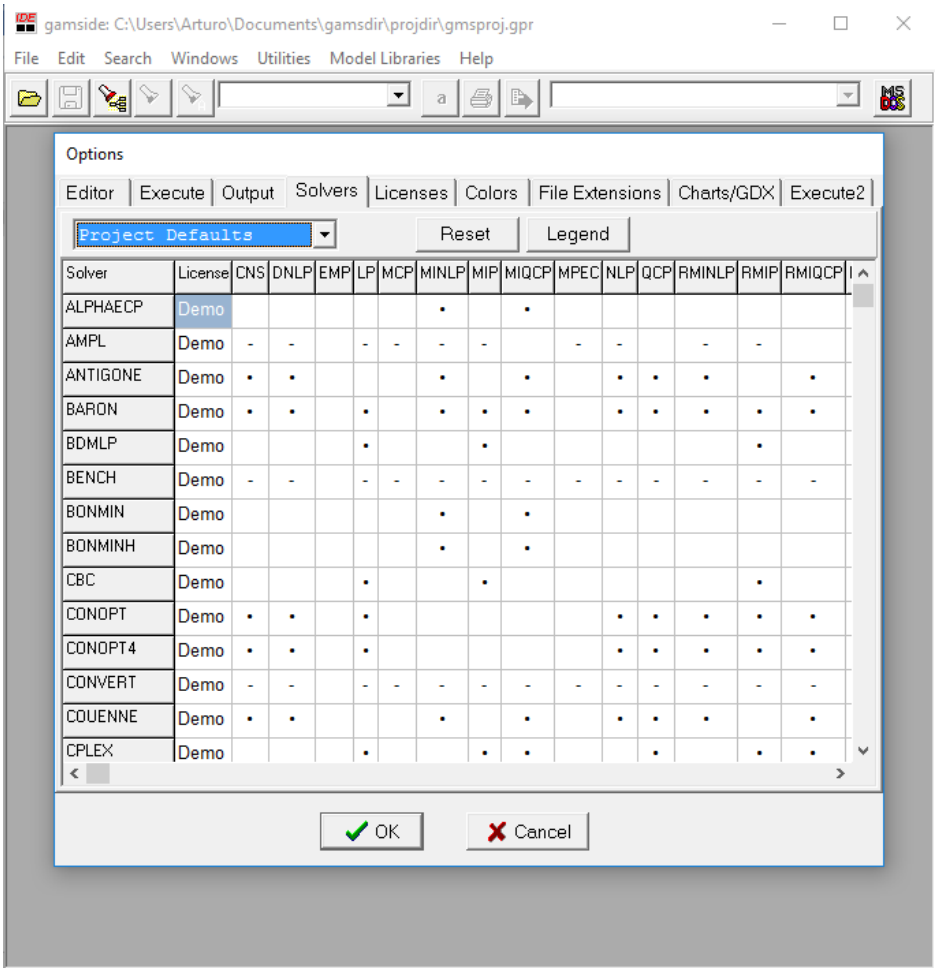

Figura 2.4: algunos de los solvers disponibles en GAMS.

La programación, como es habitual en todos estos programas orientados al usuario, es muy intuitiva. Se estructura en un sistema por bloques, de manera que existe un bloque set para definir nuestros conjuntos, un bloque parameters para introducir nuestros datos, un bloque variables donde definiremos las variables a optimizar de nuestro problema y un bloque equations donde definiremos las restricciones que vamos a imponer.

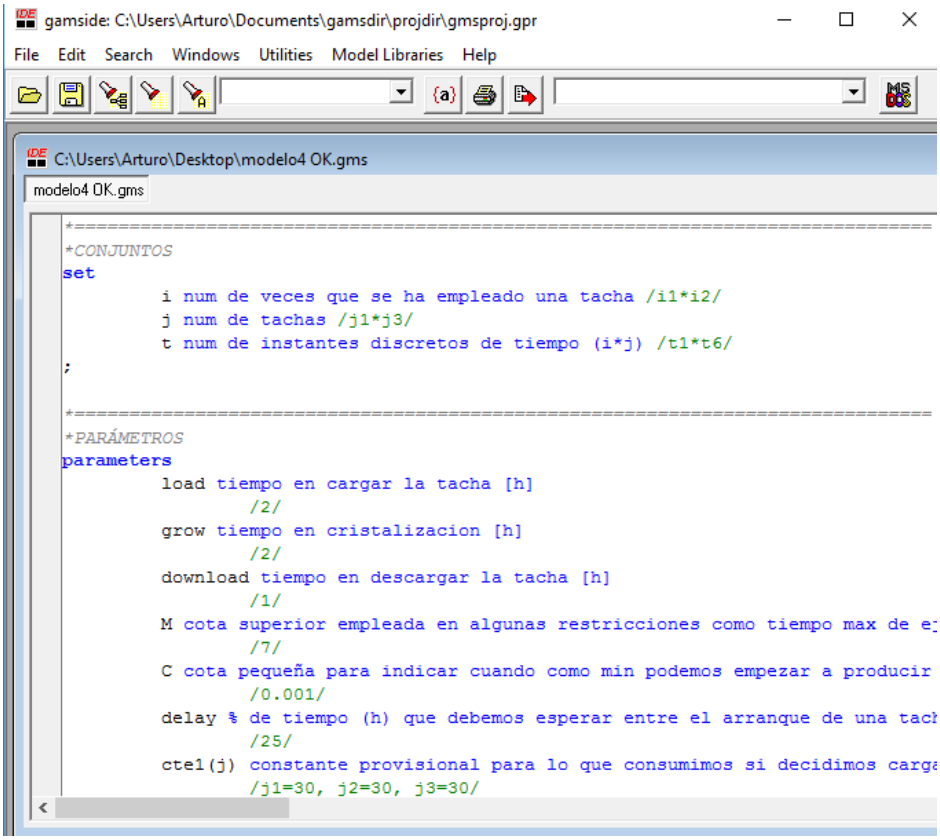

Figura 2.5: ejemplo de programación en GAMS.

CAPÍTULO 3

OPERACIONES DE SCHEDULING

3.1. Introducción

El scheduling (L.Pinedo, 2016) es un proceso de toma de decisiones muy empleado para la programación de operaciones en industrias manufactureras y/o de servicios. En una visión muy general, consiste en decidir qué recursos se utilizan y en qué instantes de tiempo con el fin de planificar y secuenciar la producción, buscando la optimización de ciertos objetivos.

Todos estos elementos pueden ser muy variados: los recursos pueden ser, por ejemplo, máquinas en un taller, pistas de despegue/aterrizaje en un aeropuerto o unidades de procesamiento en un entorno computacional; en estos casos, las peticiones podrían ser la obtención de un determinado producto en la industria manufacturera, programar los despegues/aterrizajes en el aeropuerto o la ejecución de un programa informático; por último, los objetivos pueden abarcar desde abastecer una cierta demanda en el menor tiempo posible, respetar los horarios estipulados de vuelo o la minimización de los tiempos de ejecución.

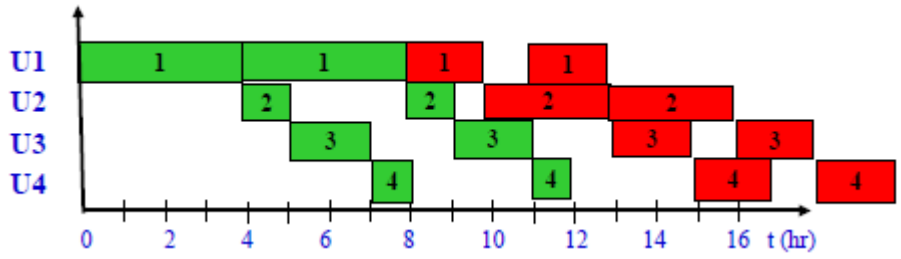

Figura 3.1: ejemplo de scheduling en un diagrama de Gantt.

En adelante, cuando hablemos de operaciones de scheduling estaremos haciendo referencia exclusivamente a la industria, dejando de lado a las operaciones de scheduling aplicadas a servicios, ya que no son objeto de estudio por parte de este TFM.

3.2. Ejemplo: fábrica de bolsas de papel.

A modo de ejemplo, tomaremos uno que aparece en (L.Pinedo, 2016).

Consideremos una fábrica que produce bolsas de papel para cemento, carbón o comida para perros, entre otros mercados. El proceso productivo de una bolsa de papel consta de tres etapas: la impresión del logo en el papel, el encolado de los laterales de la bolsa y el cosido de uno o ambos extremos de la bolsa. Cada una de estas etapas puede ser realizada por un grupo de máquinas, que no tienen por qué ser necesariamente idénticas. Por ejemplo, pueden existir diferencias en el tiempo de operación en función del tipo de bolsa fabricada, el número de colores que pueden imprimir la máquina o el tamaño de la bolsa de papel que pueden soportar.

Las órdenes de producción indican la cantidad de cada tipo de bolsa de papel que se debe fabricar para una determinada fecha límite, de manera que entregar la mercancía fuera de plazo conlleva importantes pérdidas para la compañía, tanto financieras como a nivel cliente.

Por último, se debe saber que cuando una máquina pasa de fabricar un tipo de bolsa de papel a otro, se requieren operaciones de cambio de herramienta que hacen que no sea operativa durante un determinado periodo de tiempo, mayor cuanto mayor sea la diferencia entre los dos tipos de bolsa.

Sabiendo esto, se desea decidir qué máquinas se ponen en marcha y en qué orden, de manera que se minimicen tanto las penalizaciones por entregar mercancía fuera de plazo como el tiempo total ocioso por la realización de cambios de herramienta.

3.3. Procesos por lotes.

En el entorno productivo podemos identificar básicamente dos tipos de procesos, atendiendo a la forma en que estructuran la producción: procesos continuos y procesos por lotes (batch o semibach).

Un proceso continuo se caracteriza porque las materias primas están constantemente entrando por un extremo del sistema, al mismo tiempo que

en el otro extremo se obtiene de forma continua el producto elaborado. Es por ello que los volúmenes de producción son muy elevados, debido a la necesidad de amortizar unas elevadas inversiones de capital para la adquisición de equipos especializados en producir un determinado producto.

Figura 3.2: una planta petroquímica es un proceso continuo.

Los procesos por lotes (batch o semibatch), según la definición del estándar ISA-S88 (1995), son aquellos procesos que conducen a la producción de cantidades finitas de un determinado producto a partir de una cantidad de materiales de entrada. En este caso, los volúmenes de producción son más reducidos, y el rango de productos que podemos fabricar es mayor, debido al uso de equipos flexibles. La diferencia existente entre los procesos batch o semibatch es que en los semibatch se mantiene la alimentación de entrada durante el lote, mientras que en los batch no (García, 2001).

Figura 3.3: las tachas en una industria azucarera son procesos por lotes.

A partir de ahora, nos centraremos en las operaciones de scheduling para procesos por lotes batch, ya que son el objeto de estudio de este TFM.

3.4. Optimización de las operaciones de scheduling.

Como el lector puede ir deduciendo, programar el scheduling de una planta no es una tarea sencilla, ya que dependerá, entre otros muchos factores, del tipo de proceso, del número de equipos disponibles, del número de productos a fabricar o de las limitaciones impuestas por dicha planta. De esta forma, programar el scheduling manualmente es una opción cada vez menos recomendable, por la elevada carga de trabajo y la escasa flexibilidad que esto supone.

A lo largo de las últimas décadas, se ha optado por delegar esta programación en softwares especializados (ver Capítulo 2) de forma que, si nosotros conseguimos modelar matemáticamente nuestra planta y establecer unos objetivos claros, podremos obtener un óptimo global o local del scheduling requerido. Esto permite reducir enormemente el tiempo invertido en realizar el scheduling, así como obtener unos mejores resultados que la opción manual y otorgar mayor flexibilidad (si, por ejemplo, compramos una máquina extra, simplemente habría que cambiar un parámetro del modelo matemático).

Para que nos hagamos una idea del amplio abanico de opciones que existen a la hora de realizar un scheduling en relación a las características de la planta, podemos observar la siguiente imagen, en donde aparece una clasificación para los procesos batch (A.Méndez, Cerdá, E.Grossmann, Harjunkoski, & Fahl, 2006):

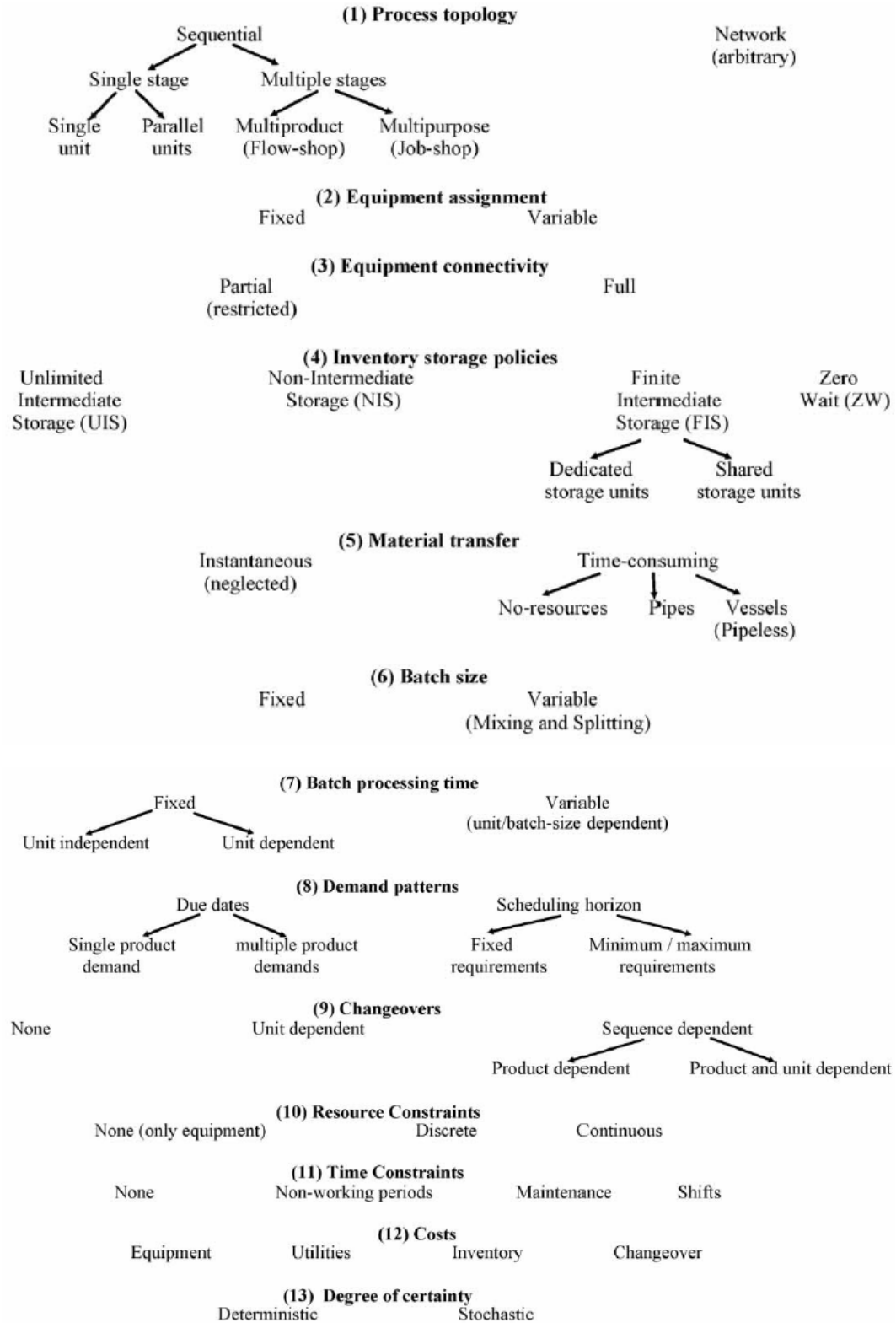

Figura 3.4: tipos de scheduling para procesos batch.

3.4.1. Clasificación de los modelos de optimización para procesos batch.

Asimismo, se pueden implementar diferentes modelos de optimización para realizar el scheduling, cada uno de ellos con unas características propias, si bien podemos identificar cuatro aspectos principales (A.Méndez, Cerdá, E.Grossmann, Harjunkoski, & Fahl, 2006):

- Representación temporal: podemos decidir realizar una programación basada en tiempo discreto o en tiempo continuo.

  La primera consiste en dividir el horizonte de planificación en intervalos finitos de tiempo con una duración determinada, de manera que los eventos deberán comenzar/terminar en los límites de dichos periodos. Esto presenta inconvenientes, dado que la carga computacional necesaria, así como la precisión de nuestra programación será mayor cuanto mayor sea el número de intervalos seleccionado.

  En la representación continua, las decisiones temporales están representadas por variables continuas, de manera que se pueden producir eventos en cualquier instante de tiempo. De esta forma evitamos ese crecimiento exponencial de la carga computacional conforme queremos representar periodos de tiempo más reducidos.

- Balances de materia: la secuenciación de los lotes y/o el dimensionamiento de los mismos dan lugar a dos categorías de problemas. La primera de ellas trata de optimizar el número y tamaño de los lotes, en tanto que la segunda asume que el número de lotes de un determinado tamaño es conocido.

- Representación de eventos: básicamente, consiste en secuenciar la distribución de los eventos a lo largo del horizonte temporal, de manera que se garantice que la capacidad máxima de los recursos nunca sea excedida.

- Función objetivo: como ya se vio en el Capítulo 2, es el elemento de nuestro modelo que nos va a indicar qué es lo que deseamos optimizar. En las operaciones de scheduling, podemos querer maximizar el beneficio obtenido durante la producción, o minimizar el makespan (tiempo total de producción), el número de retrasos, la acumulación de stocks... entre otras muchas posibilidades.

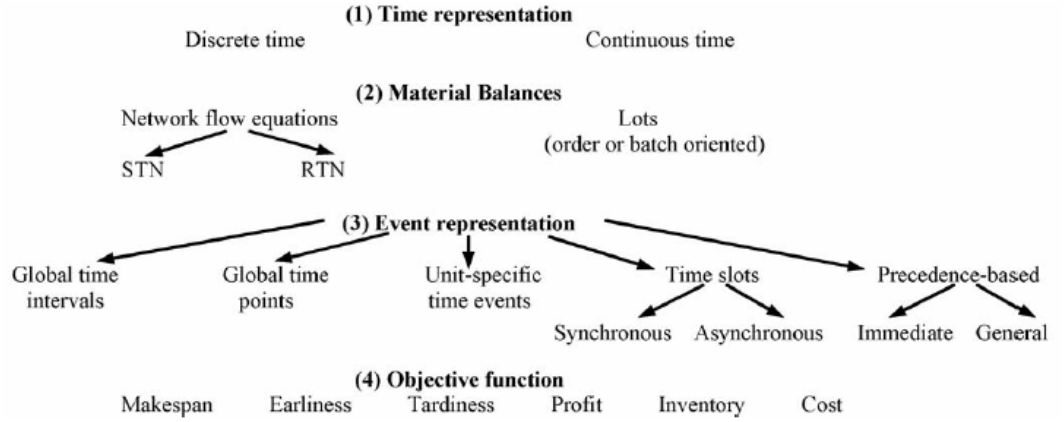

Figura 3.5: modelos de optimización para procesos batch.

Como podemos observar en la figura anterior, la diversidad de problemas es amplia en función de las características que vayamos a manejar. Si a esto le añadimos la clasificación vista en la Figura 3.4, nos hacemos una idea de la carga de trabajo que llevaría resolver estos problemas de scheduling de una manera manual, y la escasa flexibilidad que esto conllevaría en el caso de que deseásemos modificar algún parámetro del problema.

3.4.2. Ejemplo sencillo de optimización de scheduling.

Con el objetivo de dar una visión más concreta de la programación matemática necesaria para optimizar operaciones de scheduling, a continuación aparece un ejemplo, que si bien es muy básico, puede servirnos para asentar conceptos (De Prada, 2016).

Supongamos que tenemos un conjunto de I tareas que deseamos programar en M equipos. Para cada tarea $i \in I$, tenemos un tiempo mínimo en que la tarea debe comenzar (r_i), así como un tiempo máximo de finalización (d_i). Cada tarea tendrá una duración determinada (p_{im}), que dependerá del equipo $m \in M$ en que se procese. Por último, al procesar una tarea i en un equipo m, se incurre en un coste (C_{im}). El problema consiste en decidir en qué equipo se procesa cada tarea ($y_{im} \rightarrow$ variable binaria que vale 1 si la tarea i se procesa en el equipo m, y vale 0 en caso contrario), así como el tiempo de inicio de dicha tarea (t_i), de manear que se minimice el coste total de procesamiento (problema de secuenciamiento y temporización).

Tendremos también una variable binaria z_{ij} que valdrá 1 si la tarea i precede a la tarea j y 0 en caso contrario. Esto es necesario para conocer la secuencia de operación dentro de un equipo.

Tarea	r_i (min)	d_i (min)
A	0	5
B	0	8
C	1	4

		Máquina	
	p_{im} (min)	M1	M2
Tarea	A	5	3
	B	2	4
	C	3	3

		Máquina	
	C_{im} (€)	M1	M2
Tarea	A	2,5	1
	B	0,5	3
	C	2	2

Traduciendo el problema a un modelo matemático que podamos optimizar, obtenemos lo siguiente:

$$Minimizar \quad \sum_{i \in I} \sum_{m \in M} C_{im} * y_{im}$$

Sujeto a ciertas restricciones:

$$t_i \geq r_i$$

(1) El tiempo de comienzo de la tarea debe ser superior al mínimo exigido.

$$t_i + \sum_{m \in M} p_{im} * y_{im} \leq d_i$$

(2) El tiempo en que finaliza una tarea no puede ser superior al máximo permitido.

$$\sum_{m \in M} y_{im} = 1$$

(3) Cada tarea debe ser asignada a un único equipo.

$$\sum_{i \in I} p_{im} * y_{im} \leq M$$

(4) El tiempo total de procesamiento en un equipo está sometido a cotas.

$$1 \geq z_{ij} + z_{ji} \geq y_{im} + y_{jm} - 1 \qquad i > j$$

(5) Si las tareas i y j se procesan en el equipo m, se tiene que cumplir que o j precede a i o i precede a j.

$$t_j \geq t_i + \sum_{m \in M} p_{im} * y_{im} - M * (1 - z_{ij})$$

(6) Si la tarea i precede a la tarea j, entonces el tiempo de procesamiento de la tarea j será como mínimo $t_i + p_{im}$.

Si ahora trasladamos dicho modelo a GAMS (ver Capítulo 2), podemos lograr obtener la solución óptima del problema, que minimiza el coste total de operación.

```
Proven optimal solution.

MIP Solution:            3.500000
Final Solve:             3.500000

Best possible:           3.500000
Absolute gap:            0.000000
Relative gap:            0.000000
```

Figura 3.6: función objetivo óptima calculada por GAMS.

```
----      74 VARIABLE t.L

i2 4.00,     i3 1.00


----      74 VARIABLE y.L

                m1          m2

i1                          1.00
i2           1.00
i3           1.00
```

Figura 3.7: valores de las variables calculados por GAMS.

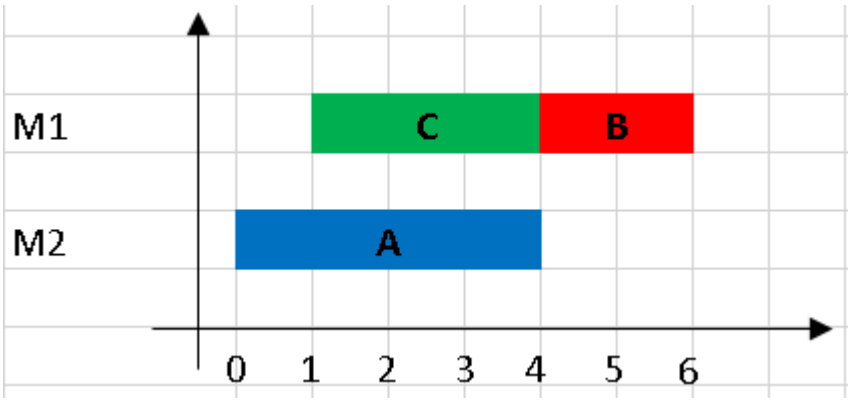

Figura 3.8: ejemplo visual de la solución de GAMS.

CAPÍTULO 4

DESCRIPCIÓN DEL PROBLEMA

4.1. Introducción

Como ya se enunció previamente, uno de los objetivos de este TFM es el desarrollo de un modelo matemático en GAMS que permita simular y optimizar las operaciones de scheduling para un sistema real. En este caso, se ha escogido una planta azucarera, más concretamente el cuarto de azúcar donde se lleva a cabo la cristalización.

Dado que este sistema no es trivial, a lo largo del presente capítulo se llevará a cabo una descripción del mismo, indicando las etapas en las cuáles centraremos nuestra atención.

4.2. Descripción del proceso de extracción del azúcar

El objetivo de la industria azucarera consiste en la extracción de la sacarosa de alguna materia prima (típicamente remolacha o caña) y la cristalización de la misma para formar el azúcar tal y como lo conocemos, con el fin último de su comercialización o distribución.

De ahora en adelante, nos centraremos en las fábricas de azúcar que emplean remolacha como materia prima, ya que son hacia las que está orientado este proyecto.

Típicamente, una fábrica de azúcar consta de dos etapas bien definidas: el cuarto de la remolacha y el cuarto de azúcar, que describimos a continuación

(ACOR, 2017) (AZUCARERA, 2017) (Jarquin, 2015) (Mazaeda, De Prada, & P. Cristea).

4.2.1. Cuarto de la remolacha

Esta etapa tiene como principal objetivo la extracción y pre procesamiento del jugo de sacarosa contenido dentro de la remolacha, y consiste básicamente en un conjunto de operaciones continuas:

- Recepción: la remolacha llega en vehículos a la fábrica, donde es pesada. Se toma una muestra para determinar el porcentaje de impurezas (tierras, piedras, hierbas…) de la recepción, así como su contenido en azúcar.

Figura 4.1: remolacha azucarera.

- Almacenamiento: los vehículos vuelcan las remolachas en las tolvas de descarga. Por medio de cintas transportadoras y tras una operación de desterrado en seco, se mandan a un carro distribuidor que las deposita en silos de almacenamiento, a los cuáles se les insufla aire ambiente para una mejor conservación del producto.

- Lavado: se suelen emplear tamices vibrantes a los que se les proyecta agua a presión o un tambor giratorio en el que entra agua a contracorriente, con el fin último de eliminar las impurezas presentes en la remolacha.

- Cortado: la remolacha lavada se trocea en tiras (denominadas cosetas) de sección triangular o cuadrangular y espesor de entre 2 y 3 mm, característica necesaria para la siguiente etapa.

Figura 4.2: coseta o remolacha cortada en tiras.

- Difusión: las cosetas avanzan por medio de dos tornillos sin fin situados en el interior de difusores continuos por los que circula agua caliente a contracorriente. Esto provoca transformaciones en las paredes celulares que permiten la difusión de la sacarosa al medio externo. Así se logra la extracción de la sacarosa contenida en la remolacha, obteniéndose el jugo bruto o jugo de difusión.

  Por el extremo opuesto del difusor se descarga la coseta ya agotada, que se denomina pulpa. Esta pulpa es sometida a tres procesos: prensado, para extraer casi por completo la abundante agua que contienen y recircularla a la etapa de difusión; secado, donde se termina de eliminar el agua contenida en la pulpa; granulado, donde se toma la pulpa seca y se trata en unas máquinas especiales para obtener gránulos de pulpa llamados pellets, que se almacenan para su venta posterior.

- Depuración del jugo: el jugo bruto procedente de la difusión contiene partículas en suspensión, así como una gran cantidad de no azúcares disueltos y sustancias coloidales procedentes de la remolacha. Además, es ácido, lo que puede provocar la descomposición de la sacarosa en su posterior procesamiento.

  Por todo ello, es necesario depurar este jugo, separando las partículas en suspensión, los no azúcares y eliminando los coloides, así como proporcionar un pH adecuado para eliminar la acidez. Esto se logra fundamentalmente mediante la adición de lechada de cal y anhídrido carbónico para formar aglomerados de carbonato cálcico e impurezas que son eliminados posteriormente mediante filtración.

Figura 4.3: depuración del jugo bruto.

- El jugo procedente de la depuración es una disolución azucarada con una gran cantidad de agua y poco contenido en materia seca (grados Brix: cantidad de materia seca disuelta en un líquido). El objetivo de la evaporación es incrementar este contenido de materia seca evaporando gran parte del agua que contiene el jugo por medio de una serie de evaporadores de efecto múltiple.

  El jugo concentrado que sale de la instalación de evaporación se denomina jarabe. En este jarabe se disuelven azúcares de los bajos productos para la obtención del jarabe estándar, que constituye la alimentación de los cristalizadores encargados de la producción del azúcar comercial.

4.2.2. Cuarto de azúcar

El objetivo de este proceso es la cristalización de la sacarosa disuelta en el jarabe estándar, obteniendo de este modo azúcar sólido cristalizado, apto para la comercialización. Este es un proceso complejo, con sucesivas cristalizaciones y refundición con el objetivo de obtener un producto de alta calidad para la venta, a la vez que se reutilizan al máximo las soluciones de sacarosa (también denominadas mieles) hasta que ya no es posible la cristalización de las mismas, dando origen a la melaza.

- Primera cristalización: se realiza en unos evaporadores al vacío o cristalizadores, denominados tachas, mediante un proceso de cocción. Es un proceso por lotes, que transcurre por varias etapas, siguiendo una receta configurable por el operario. Primero se introduce una cantidad determinada de jarabe estándar que se conoce como “pie de

tacha" y que se concentra mediante evaporación hasta conseguir una solución sobresaturada controlada. En este momento, se introduce una cantidad de cristales de azúcar de un tamaño aproximado de 5-10 micras denominada siembra. La tacha se sigue alimentando con jarabe estándar a la vez que se va evaporando el agua controlando en todo momento la sobresaturación de la solución, evaporando al vacío la masa cocida, lo que provoca el aumento progresivo del tamaño de los cristales de la siembra al ir cristalizando la sacarosa sobre la superficie de dichos cristales.

Cuando acaba el proceso, la tacha tiene en su interior una masa formada por cristales, de un tamaño aproximado de 500 μm, y una solución de azúcar que no ha cristalizado y de no azúcares. A la masa de cristales se le denomina masa cocida de 1ª y a la solución miel madre.

Las tachas trabajan en vacío con el objetivo de rebajar el punto de ebullición de los productos en ellas procesados, y de este modo evitar la descomposición térmica del azúcar o caramelización. Como ya se ha indicado al comienzo de este apartado, la operación de las tachas es discontinua, por lo que terminada ésta, la masa cocida se descarga a unos depósitos denominados malaxadores.

Figura 4.4: tachas donde tiene lugar la cristalización.

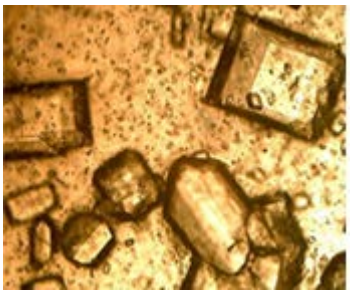

Figura 4.5: visión microscópica de cristales de azúcar.

- Centrifugación de primer producto: por medio de esta operación de desea lograr separar los cristales de azúcar de la masa cocida de la miel madre. Se realiza en unas centrífugas, donde el contenido de los malaxadores se somete a la fuerza centrífuga de un tambor rotatorio que la envía hacia las paredes del cesto, provistas de una tela metálica por donde pasa la miel madre, pero no los cristales de azúcar. Las centrifugas de esta masa trabajan por lotes. La primera miel madre separada se denomina miel pobre y continua el proceso hacia la 2ª cristalización. En la parte final de la centrifugación se introduce vapor para lavar los cristales, lo que genera miel (rica) con una mayor pureza debido a la disolución de parte de los cristales que es reciclada para ser usada como jarabe estándar en las tachas de primera.

- Secado y acondicionamiento del azúcar: el azúcar separado en la anterior centrifugación se denomina azúcar blanquilla, que previo proceso de secado y acondicionamiento se almacena en silos para su posterior envasado y comercialización[1].

Figura 4.6: azúcar blanquilla.

- Segunda cristalización: la miel pobre obtenida en la centrifugación de primer producto se somete a un proceso de cristalización análogo al de la primera en las llamadas tachas de segunda.

  El producto obtenido en estas tachas se conoce como masa cocida de 2ª (formada por el denominado azúcar de 2ª y una solución conocida como miel de 2ª).

- Centrifugación de segundo producto: el objetivo es el de separar el azúcar de 2ª y la miel de 2ª obtenidas en la segunda cristalización, y se

[1] A modo de curiosidad, señalar que el azúcar blanquilla se comercializa en unos tamaños de cristales comprendidos entre 0,4 y 0,6 mm.

lleva a cabo mediante un proceso análogo al de la centrifugación de primer producto.

El azúcar de 2ª se disuelve de nuevo, recirculándose hacia el ya citado jarabe estándar, que alimenta las tachas de 1ª.

- Tercera cristalización: proceso análogo a la segunda cristalización, obteniéndose como producto azúcar de 3ª y miel de 3ª.

- Centrifugación de tercer producto: la centrifugación de la masa de 3ª, de modo similar a las anteriores, da lugar a un azúcar de 3ª y a una solución de azúcar no cristalizable conocida como melaza. El azúcar de tercera se pone en contacto con miel de 2ª en una afinadora, con lo que se consigue un tratamiento de lavado de cristales.

  La melaza, por su parte, es utilizada para la producción de alcohol etílico, levadura, ácido cítrico u otros usos industriales

Pueden co-existir otros elementos, como tachas continuas o torres de enfriamiento para recuperar parte del azúcar que se iría con las melazas, pero el esquema básico de fabricación en la mayor parte de las fábricas es el mencionado.

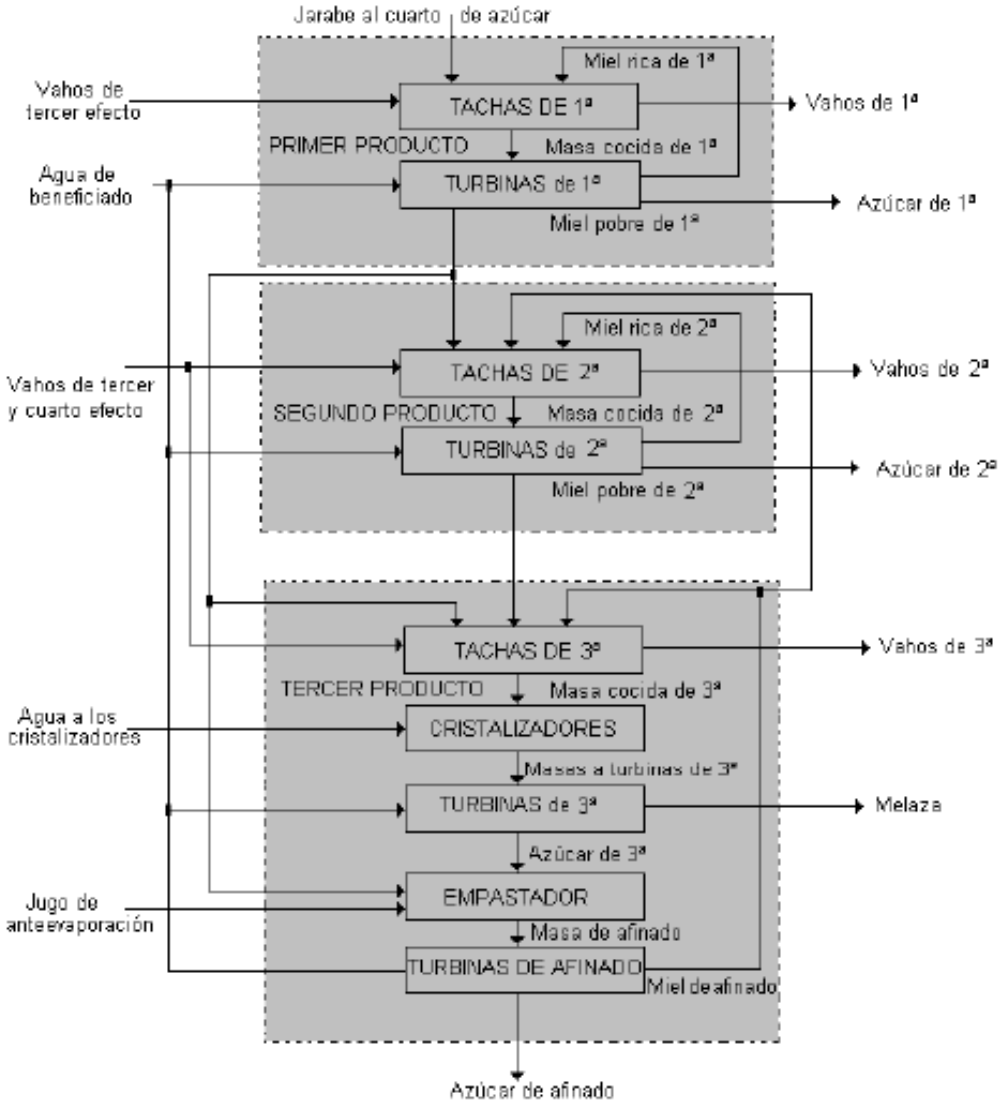

Figura 4.7: esquema del proceso completo en el cuarto del azúcar (García, 2001).

4.3. Problemática objeto de interés

Aunque todo el proceso de fabricación anteriormente descrito es interesante desde el punto de vista del control y la automatización, seguramente una de las partes más críticas sea el cuarto del azúcar ya que de estas operaciones dependerá la calidad del producto final, pudiendo suponer cualquier fallo una gran pérdida tanto a nivel económico como a nivel cliente.

En especial centraremos nuestro interés en la etapa de cristalización dentro de las tachas, ya que las operaciones de arranque de las mismas deben ser planificadas con mucho cuidado, por diversos motivos:

- Por una parte, durante la cristalización se va a consumir una cierta cantidad de jarabe estándar que no es constante a lo largo del tiempo de operación de la tacha, requiriéndose un mayor volumen en los instantes iniciales para realizar la siembra, luego deteniendo por un tiempo el consumo para dejar que comience la cocción, y por último volviendo a consumir para garantizar unas condiciones estables dentro de la tacha.

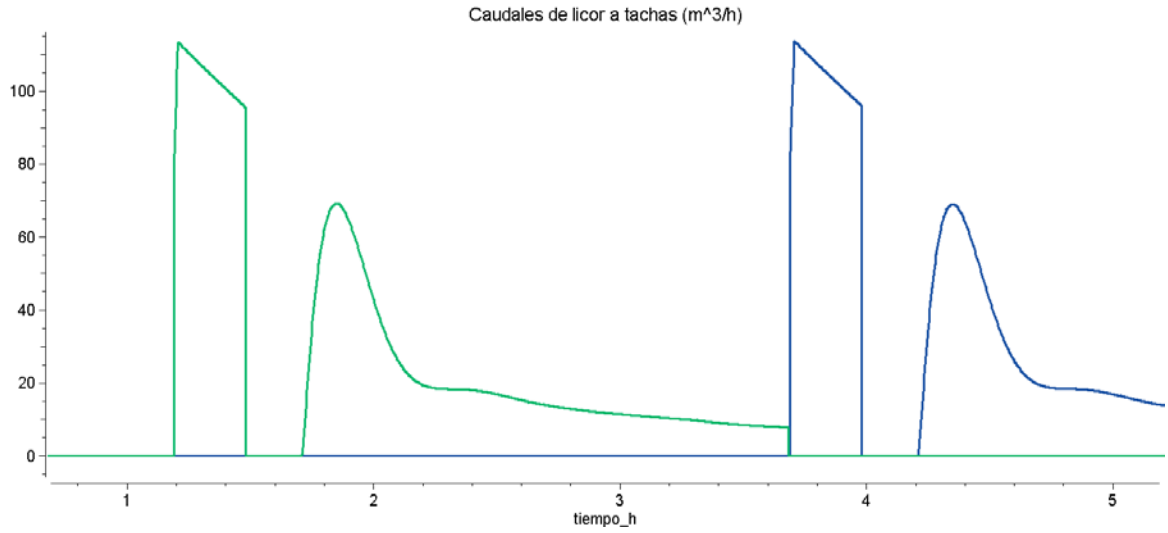

Figura 4.8: consumo de jarabe estándar a lo largo del tiempo para dos tachas (verde y azul).

Como este jarabe estándar proviene de un depósito donde se encuentra almacenado, y al cual sigue llegando de manera continua más jarabe, hay que tener cuidado en que no se exceda la capacidad de dicho depósito, así como de evitar su vaciamiento por debajo de un determinado límite crítico. Esto debe respetarse tanto para el scheduling de un ciclo productivo (ejecución de las tachas) como para la sucesión de ciclos productivos, de manera que se puedan ejecutar estos procesos batch de una manera continuada sin que ello aporte inestabilidad al sistema.

- Por otro lado, para conseguir las condiciones de sobresaturación necesarias durante el proceso de cocción, se consume vapor que proviene normalmente de la sección de evaporación por motivos de integración energética. Al igual que ocurría con el consumo de jarabe estándar, el consumo de vapor no es constante a lo largo de la cocción, presentando una forma similar al caso del jarabe que puede verse en la Figura 4.8, con una zona inicial en la que no se consume vapor, simplemente se llena la tacha con jarabe.

Como el consumo de este vapor es limitado, podría suceder que si no realizamos un scheduling de las tachas adecuado, la demanda de vapor supere a la disponibilidad del mismo, ocasionando un fallo en el proceso consistente en una menor presión de vapor que trae consigo una mayor duración del ciclo de las tacha. Obviamente, debe garantizarse la factibilidad del uso tanto de vapor como de jarabe para una adecuada operación de las tachas.

La siguiente imagen muestra un esquema simplificado del proceso de cristalización para las tachas de primera que considera tres tachas y no incluye los malaxadores ni las centrífugas (Mazaeda, De Prada, & P. Cristea):

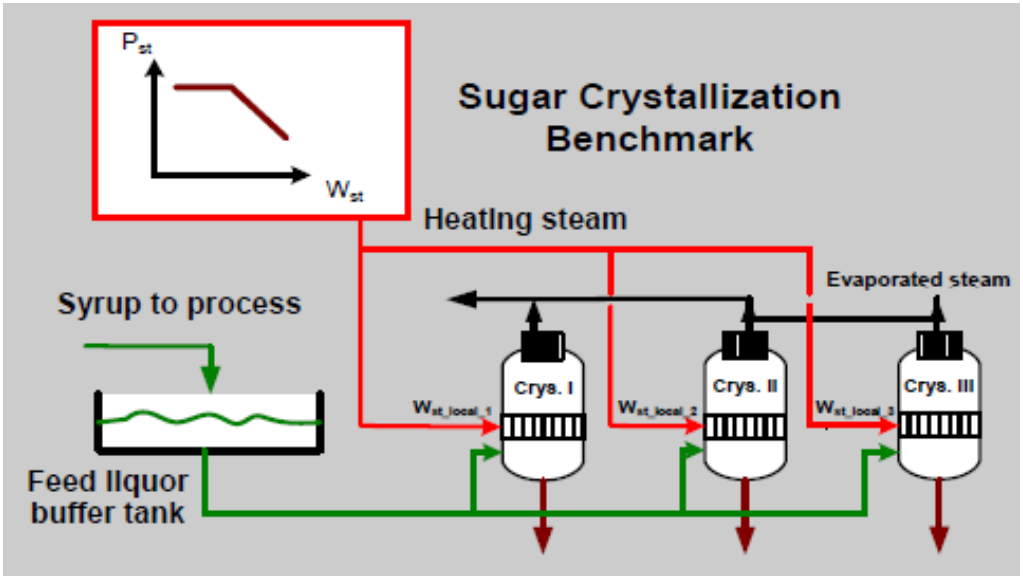

Figura 4.9: esquema simplificado de las tachas de primera.

Dicho esquema será el que tengamos en cuenta a la hora de formular los problemas de scheduling y comprobar la validez de las soluciones obtenidas, pudiéndose posteriormente ampliar a otras configuraciones. Ha sido propuesto como un benchmark en el Proyecto europeo HYCON2 y será utilizado para el dimensionamiento de los equipos y condiciones de operación típicas reflejadas en el Capítulo 5.

4.4. Enfoque del problema

Este problema ya ha sido estudiado previamente en la literatura (Jarquin, 2015), abarcando un entorno más amplio como es el cuarto de azúcar en su

totalidad. En ese caso, se optó por realizar un modelo matemático basado en instantes de tiempo discretos, que puede presentar alguno de los inconvenientes citados en el apartado 3.4.1.

Nuestra meta ha sido intentar evitar esta dependencia de la división del espacio temporal en un conjunto finito de elementos por medio de la realización de un modelado basado en tiempo continuo. Si bien es cierto que va a ser necesario tomar medidas en instantes discretos de tiempo, coincidentes con los eventos que se producen en el sistema, con el fin de garantizar el cumplimiento de ciertas restricciones (ej. altura del depósito de jarabe estándar), se prevé que esta formulación puede acarrear unos mejores resultados desde el punto de vista de tiempos de ejecución y un menor tamaño del problema.

En resumen, mediante el desarrollo de un modelo matemático en tiempo continuo y su posterior simulación en GAMS, se pretende optimizar el scheduling de la etapa de cristalización en el cuarto de azúcar, garantizando unas condiciones estables de operación tanto para el depósito de jarabe estándar como para el flujo de vapor que llega al proceso. Se ha optado por comenzar por las tachas de primera, pudiendo extenderse posteriormente a otras zonas del proceso.

CAPÍTULO 5

MODELO MATEMÁTICO Y RESULTADOS

5.1. Introducción

En este capítulo vamos a exponer la formulación matemática que se ha desarrollado para optimizar el scheduling de las tres tachas, así como los resultados que han sido obtenidos en las simulaciones realizadas, con el fin último de mostrar la validez de las mismas cuando se aplican al sistema real.

Como ya se ha indicado en capítulos anteriores, el modelo matemático ha sido escrito en GAMS. Las simulaciones dentro de GAMS han sido realizadas con el solver BARON, adecuado para resolver problemas no lineales. Para comprobar la factibilidad de los resultados en el sistema real, se ha empleado un simulador de las tachas desarrollado en EcosimPro.

Todos los parámetros de volumen y caudales considerados se han hecho en base a datos tomados del benchmark del proyecto HYCON2 que reflejan de forma realista las condiciones de una planta azucarera real.

5.2. Consumo de las tachas

Como ya se vio en el capítulo anterior, el consumo de jarabe estándar para una tacha sigue aproximadamente la siguiente estructura:

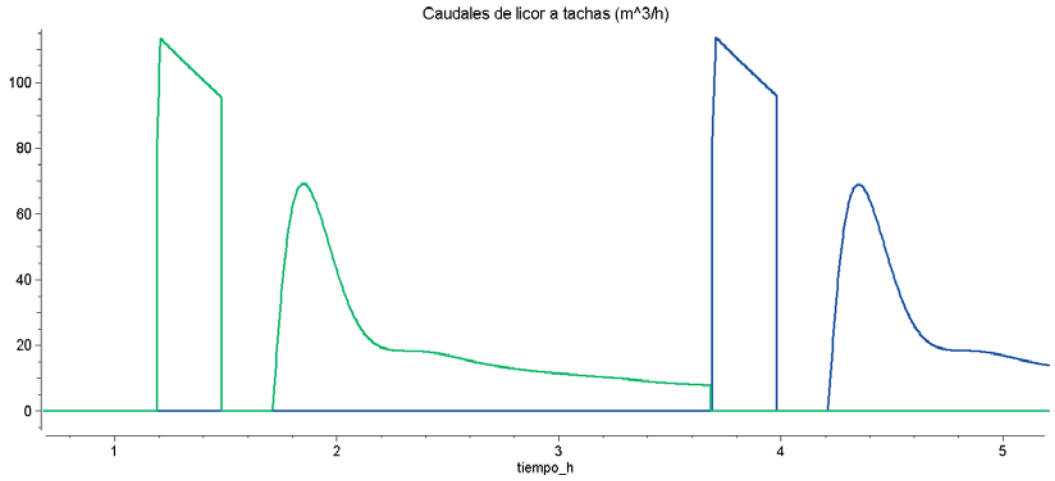

Figura 5.1: consumo de jarabe estándar a lo largo del tiempo para una tacha (verde).

Es muy importante tener en todo momento presente este perfil de consumo durante la simulación, ya que la altura del depósito de jarabe en cualquier instante dependerá de lo que haya ido consumiendo cada tacha a lo largo del tiempo, así como del jarabe que haya ido alimentando dicho depósito. Debido a esto, una aproximación del perfil de consumo poco precisa puede llevarnos a que los resultados no se ajusten todo lo que deberían a la realidad.

Para el desarrollo del modelo matemático, se ha realizando un perfil de consumo de las tachas bastante preciso, con la siguiente forma:

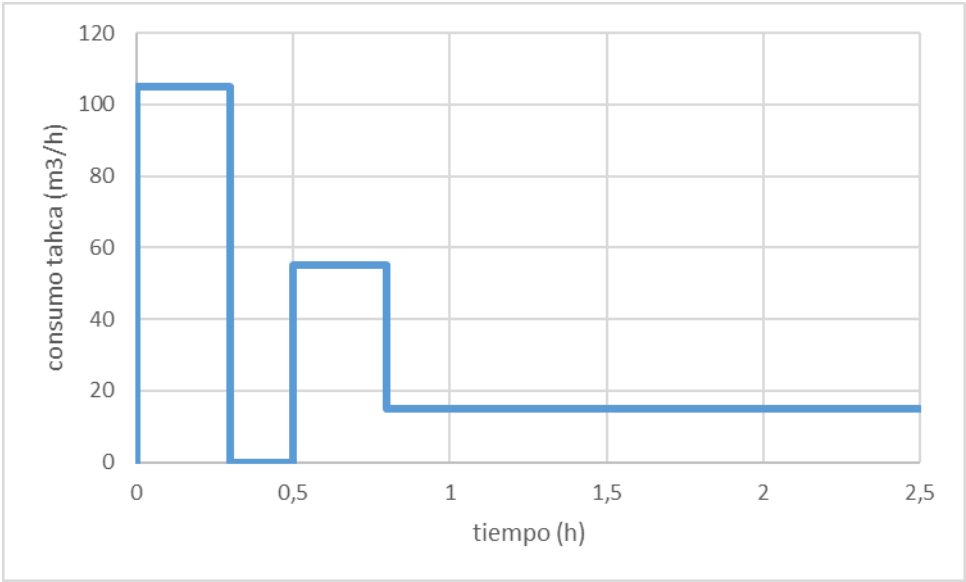

Figura 5.2: perfil final de consumo modelado.

Vemos que ahora se asemeja mucho más al perfil de consumo real. Si bien es cierto que no deja de ser una aproximación, hay que destacar la dificultad de simular exactamente el perfil de consumo de la tacha, siendo dicha aproximación necesaria si se quiere obtener algún resultado.

5.3. Caudal de alimentación al depósito de licor

En esta primera formulación, se ha establecido un caudal de alimentación al depósito de licor constante a lo largo del tiempo, con un valor de 39,61 m^3/h que puede ser una aproximación aceptable de la situación real, que podemos ver en la siguiente imagen:

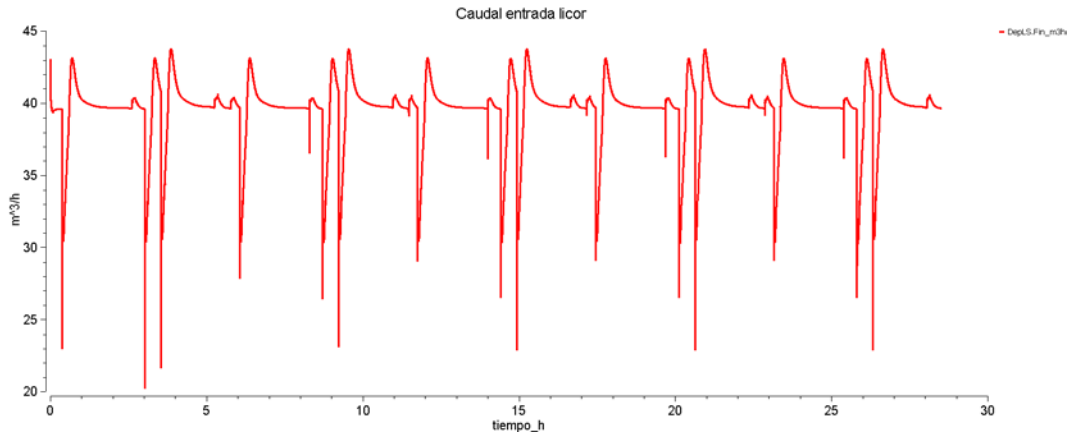

Figura 5.3: caudal de alimentación real al depósito de licor.

Sin embargo, como el lector puede hacerse idea, modelar de manera exacta este comportamiento real resulta imposible, por lo que la aproximación una vez más resulta necesaria.

5.4. Depósito de licor estándar.

De igual manera que en los casos anteriores, el depósito de licor ha sido modelado atendiendo a datos del ejercicio, es decir, un área del depósito de 24,3 m^2, con una altura de 3,2 m.

Estos datos nos permiten conocer las limitaciones que puede tener el depósito en cuanto a la altura máxima y mínima permitidas, factores críticos para evitar el desbordamiento y el vaciamiento del mismo.

5.5. Modelo matemático

Una vez explicados esos conceptos iniciales, pasamos a describir la formulación matemática que se ha desarrollado. Cabe destacar que, debido a la imperiosa necesidad de evaluar la altura del depósito en instantes concretos de tiempo, la formulación ha dado como resultado un problema no lineal mixto entero (MINLP), que si bien es complejo, es también necesario para realizar esta optimización.

Sin más demora, pasamos a describir la formulación matemática empleada:

- CONJUNTOS

  i =1..X → nº de tachas

  s =1..Y →nº de slots

  j, j2, j3 =1..Z→ instantes de tiempo

Para la simulación que hemos llevado a cabo, se han establecido 3 tachas, 1 slot y 15 posibles instantes de tiempo (5 por tacha y slot, como veremos posteriormente).

- PARÁMETROS

  maxs: num max de slots

  /1/

  maxj: num max de instantes de tiempo

  /15/

  deltat: tiempo total de operación de una tacha [h]

  /2.5/

  M: cota superior empleada como tiempo max de ejecución [h]

  /10/

  bigM: valor muy grande (cota superior en algunas restricciones)

  /1000/

  C: cota pequeña para indicar cuándo podemos empezar a producir

/0.001/

F1(i): consumo tacha intervalo 1 [m3 por h]

/i1=105, i2=105, i3=105/

F2(i): consumo tacha intervalo 2 [m3 por h]

/i1=0,001, i2=0,001, i3=0,001/

F3(i): consumo tacha intervalo 3 [m3 por h]

/i1=55, i2=55, i3=55/

F4(i): consumo tacha intervalo 4 [m3 por h]

/i1=15, i2=15, i3=15/

A1: area del depósito 1 [m2]

/24.3/

V01: volumen inicial del depósito 1 [m3]

/30/

h1max: máxima altura permitida para el depósito 1 [m]

/3.1/

h1min: mínima altura permitida para el depósito 1 [m]

/0.5/

q1: caudal cte de entrada al depósito 1 [m3 por h]

/39.61/

profit: ingresos por tacha arrancada

/50/

- VARIABLES BINARIAS

Ys(i,s,j): 1 si se decide activar el slot s de la tacha i en el instante j.

Yi1(i,s,j): 1 si en el instante j se produce el primer cambio en el consumo.

Yi2(i,s,j): 1 si en el instante j se produce el segundo cambio en el consumo.

Yi3(i,s,j): 1 si en el instante j se produce el tercer cambio en el consumo.

Yf(i,s,j): 1 si se decide terminar el slot s de la tacha i en el instante j.

Y(i,s,j): 1 si en el instante j está activa la tacha i realizando el slot s.

La siguiente imagen muestra los instantes de tiempo a los que se refieren cada una de estas las variables (salvo Y(i,s,j)), con el fin de que el concepto quede claro:

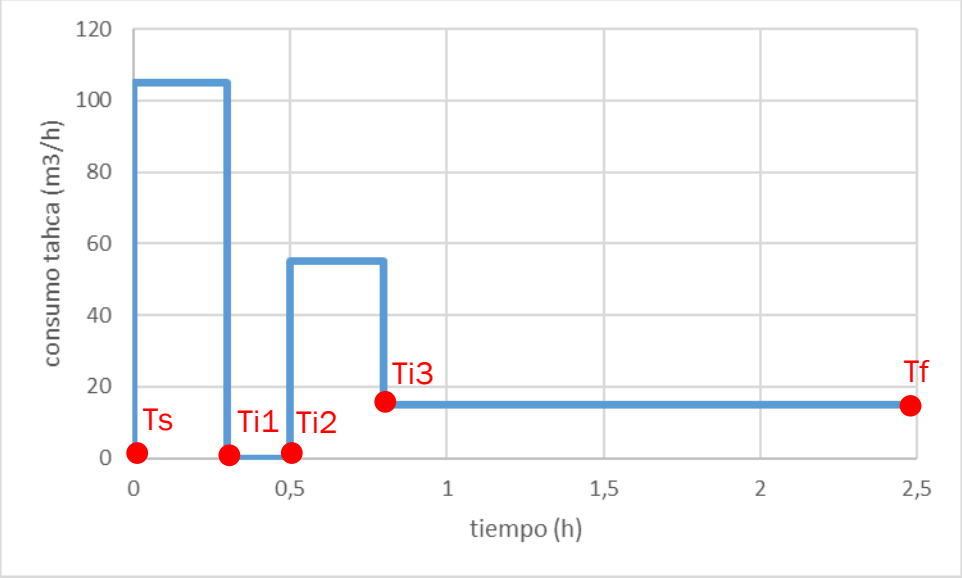

Figura 5.4: ejemplo gráfico de algunas variables binarias.

- VARIABLES POSITIVAS

t(j): tiempo de comienzo o fin de un slot en una tacha o de un punto intermedio de cambio de caudal.

Ts(i,s): tiempo de comienzo del slot s en la tacha i.

F(i,s,j): consumo de flujo entrada del slot s de la tacha i en el instante j (m3/h).

h1(j): altura del depósito 1 en el instante j.

- FUNCIÓN OBJETIVO

Como la propia formulación se va a encargar de garantizar que los niveles del depósito sean adecuados, así como de que se pueda repetir el ciclo productivo seleccionado de forma secuencial debido a las restricciones que hemos impuesto, nuestro único objetivo va a ser el de arrancar todas las tachas.

$$profit * \sum_i \sum_s \sum_j Y_{isj}^s$$

- RESTRICCIONES

res1*: un slot de una tacha puede arrancar a lo sumo en un instante de tiempo*

$$\sum_j Y_{isj}^s \leq 1 \qquad \forall\, i, s$$

res2*: si se empieza un slot debe terminarse*

$$\sum_j Y_{isj}^s = \sum_j Y_{isj}^f \qquad \forall\, i, s$$

res3*: si se arranca una tacha se tiene que pasar por el instante intermedio 1*

$$\sum_j Y_{isj}^s = \sum_j Y_{isj}^{i1} \qquad \forall\, i, s$$

res3_1*: si se arranca una tacha se tiene que pasar por el instante intermedio 2*

$$\sum_j Y_{isj}^s = \sum_j Y_{isj}^{i2} \qquad \forall\, i, s$$

res3_2*: si se arranca una tacha se tiene que pasar por el instante intermedio 3*

$$\sum_j Y_{isj}^s = \sum_j Y_{isj}^{i3} \qquad \forall\, i, s$$

res4: *en el instante de comienzo la tacha esta activa*

$$Y_{isj} \geq Y_{isj}^{s} \qquad \forall\, i, s$$

res5: *si la tacha esta activa necesariamente Ys tiene que haber valido 1*

$$\sum_{j} Y_{isj}^{s} \geq Y_{isj2} \qquad \forall\, i, s$$

res6: *en el instante final la tacha deja de estar activa*

$$Y_{isj} \leq 1 - Y_{isj}^{f} \qquad \forall\, i, s, j$$

res7: *la tacha está activa en el instante intermedio 1*

$$Y_{isj} \geq Y_{isj}^{i1} \qquad \forall\, i, s, j$$

res7_1: *la tacha está activa en el instante intermedio 2*

$$Y_{isj} \geq Y_{isj}^{i2} \qquad \forall\, i, s, j$$

res7_2: *la tacha está activa en el instante intermedio 3*

$$Y_{isj} \geq Y_{isj}^{i3} \qquad \forall\, i, s, j$$

res8: *secuenciamiento de los instantes de tiempo*

$$t_{j+1} \geq t_{j} \qquad \forall\, j < jmax$$

res9: *si se ha activado una tacha en el instante j entonces t(j) tiene que ser >=0 pero tomar un valor*

$$t_{j} \geq C * Y_{isj}^{s} \qquad \forall\, i, s, j$$

res10_1, res10_2: *si la tacha i inicia el slot s en el instante j y lo termina en j2 entonces tj2=tj+delta t con delta t tiempo de operación*

$$t_j - t_{j2} + deltat \leq \left(2 - Y^s_{isj} - Y^f_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

$$t_j - t_{j2} + deltat \geq -\left(2 - Y^s_{isj} - Y^f_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

res11_1, res11_2: *el instante intermedio 1 difiere en 0,3 segundos del instante de arranque*

$$t_j - t_{j2} + 0.3 \leq \left(2 - Y^s_{isj} - Y^{i1}_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

$$t_j - t_{j2} + 0.3 \geq -\left(2 - Y^s_{isj} - Y^{i1}_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

res11_1_1, res11_2_1: *el instante intermedio 2 difiere en 0,2 segundos del instante intermedio 1*

$$t_j - t_{j2} + 0.2 \leq \left(2 - Y^{i1}_{isj} - Y^{i2}_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

$$t_j - t_{j2} + 0.2 \geq -\left(2 - Y^{i1}_{isj} - Y^{i2}_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

res11_1_2, res11_2_2: *el instante intermedio 3 difiere en 0,3 segundos del instante intermedio 2*

$$t_j - t_{j2} + 0.3 \leq \left(2 - Y^{i2}_{isj} - Y^{i3}_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

$$t_j - t_{j2} + 0.3 \geq -\left(2 - Y^{i2}_{isj} - Y^{i3}_{isj2}\right) * M \qquad \forall\, i, s, j, j2 \;/\; j2 > j$$

res12_1: *si se arranca en un instante debemos tener Yisj activo a lo largo del proceso*

$$Y_{isj3} - 1 \geq -\left(2 - Y^s_{isj} - Y^f_{isj2}\right) * M \qquad \forall\, j3 > j \;and\; j3 < j2$$

res12_2: *fuera del intervalo de proceso de la actividad Yisj debe valer 0*

$$Y_{isj3} \leq \left(2 - Y^s_{isj} - Y^f_{isj2}\right) * M \qquad \forall\, i, s, j, j2, j3 \mid j3 < j \;or\; j3 > j2$$

res13: *no puede hacerse el ciclo s si no se ha hecho el s-1*

$$\sum_j Y^s_{i\,s+1\,j} \le \sum_j Y^s_{i\,s\,j} \qquad \forall\, i,s \mid s < smax$$

res14_1, res14_2: *el consumo desde el instante de arranque hasta el instante intermedio 1 es F1*

$$F_{isj3} - F1_i \ge -\left(2 - Y^s_{isj} - Y^{i1}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid j3 > j \;and\; j3 \le j2$$

$$F_{isj3} - F1_i \le \left(2 - Y^s_{isj} - Y^{i1}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid j3 > j \;and\; j3 \le j2$$

res14_1_1, res14_2_1: *el consumo desde el instante intermedio 1 hasta el instante intermedio 2 es F2*

$$F_{isj3} - F2_i \ge -\left(2 - Y^{i1}_{isj} - Y^{i2}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid j3 > j \;and\; j3 \le j2$$

$$F_{isj3} - F2_i \le \left(2 - Y^{i1}_{isj} - Y^{i2}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid j3 > j \;and\; j3 \le j2$$

res14_1_2, res14_2_2: *el consumo desde el instante intermedio 2 hasta el instante intermedio 3 es F3*

$$F_{isj3} - F3 \ge -\left(2 - Y^{i2}_{isj} - Y^{i3}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid\; j3 > j \;and\; j3 \le j2$$

$$F_{isj3} - F3_i \le \left(2 - Y^{i2}_{isj} - Y^{i3}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid j3 > j \;and\; j3 \le j2$$

res15_1, res15_2: *el consumo desde el instante intermedio hasta el instante final es F2*

$$F_{isj3} - F2_i \ge -\left(2 - Y^{i}_{isj} - Y^{f}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid\; j3 > j \;and\; j3 \le j2$$

$$F_{isj3} - F2_i \le \left(2 - Y^{i}_{isj} - Y^{f}_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2,j3 \mid\; j3 > j \;and\; j3 \le j2$$

res16: *el consumo de un slot fuera de sus puntos de comienzo y fin es 0 (en el punto de comienzo también es 0)*

$$\sum_{\substack{j3 \\ j3 \le j \;or\; j3 > j2}} F_{isj3} \le \left(2 - Y^s_{isj} - Y^f_{isj2}\right) * bigM \qquad \forall\, i,s,j,j2$$

res17: *si no se ha arrancado una tacha para hacer un slot su consumo debe valer 0*

$$F_{isj} \leq bigM * \sum_{j2} Y^s_{isj2} \qquad \forall\, i,s,j$$

res18_1, res18_2: *para calcular el tiempo de comienzo del slot s de la tacha i*

$$T^s_{is} - t_j \leq \left(1 - Y^s_{isj}\right) * M \qquad \forall\, i,s,j$$

$$T^s_{is} - t_j \geq -\left(1 - Y^s_{isj}\right) * M \qquad \forall\, i,s,j$$

res19_1: *altura para el primer instante de tiempo (t_1)*

$$A1 * h1_1 = V01 + q1 * t_1$$

res19_2: *altura para el resto de instantes de tiempo*

$$A1 * h1_j = A1 * h1_{j-1} + (q1 - \sum_i \sum_s F_{isj}) * \left(t_j - t_{j-1}\right) \qquad \forall\, j > 1$$

res20_1: *cota superior para la altura*

$$h1_j \leq h1max \qquad \forall\, j$$

res20_2: *cota inferior para la altura*

$$h1_j \geq h1min \qquad \forall\, j$$

res21_1: *la altura final del depósito debe de estar en unos límites respecto a la altura inicial, para de esta forma garantizar la reproducibilidad del ciclo*[2]

$$h1_j \geq \left(\frac{V01}{A1}\right) * 0.9 \qquad \forall\, j = jmax$$

[2] Si al haber realizado un ciclo a partir de un volumen de depósito inicial se obtiene una solución factible en el sistema real, al partir el segundo ciclo de una altura similar a la del primero, la factibilidad del ciclo debería estar asegurada.

***res21_2**: la altura final del depósito debe de estar en unos límites respecto a la altura inicial, para de esta forma garantizar la reproducibilidad del ciclo*

$$h1_j \leq \left(\frac{V01}{A1}\right) * 1.4 \qquad \forall\, j = jmax$$

5.6. Resultados obtenidos

Tras realizar el modelado, hemos ejecutado GAMS empleando el solver BARON, adecuado para resolver problemas MINLP (lo necesitamos debido a la restricción res19_2). Tras 459,15 segundos (7,5 minutos), hemos obtenido una solución que implica la ejecución de las tres tachas, y que respeta todas las restricciones propuestas. A continuación aparece un resumen de estos resultados:

---- 205 VARIABLE Ys.L 1 si se decide arranque el slot s de la tacha i en elinstante j

	j1	j6	j8
i1.s1		1	
i2.s1			1
i3.s1	1		

---- 205 VARIABLE Yf.L 1 si se decide terminar el slot s de la tacha i en el instante j

	j5	j12	j13
i1.s1		1.00	
i2.s1			1.00
i3.s1	1.00		

---- 205 VARIABLE Y.L 1 si en el instante j está activa la tacha i realizando el slot s

	j1	j2	j3	j4	j6	j7
i1.s1					1	1
i3.s1	1	1	1	1		

	+	j8	j9	j10	j11	j12
i1.s1	1	1	1	1		
i2.s1	1	1	1	1	1	

---- 205 VARIABLE F.L consumo de caudal de entrada de la tacha i para el slot s en el instante j (m3)

	j2	j3	j4	j5	j7	j8
i1.s1					105	0.001
i3.s1	105	0.001	55	15		

	+	j9	j10	j11	j12	j13
i1.s1	55	15	15	15		
i2.s1	105	0.001	55	15	15	

---- 205 VARIABLE t.L tiempo de comienzo o fin de un slot en una tacha

j1 0.04, j2 0.34, j3 0.54, j4 0.84, j5 2.54, j6 2.72

j7 3.02, j8 3.22, j9 3.52, j10 3.72, j11 4.02, j12 5.22

j13 5.72, j14 5.72, j15 5.87

---- 205 VARIABLE h1.L altura del deposito 1 en el instante j

j1 1.31, j2 0.50, j3 0.83, j4 0.64, j5 2.36, j6 2.64

j7 1.83, j8 2.16, j9 0.67, j10 0.88, j11 0.50, j12 0.97

j13 1.48, j14 1.48, j15 1.73

Trasladando estos resultados a EcosimPro, donde está simulado el proceso con modelos realistas de primeros principios y realizamos una simulación en la que se reproduzca ese ciclo 5 veces (25 horas de producción). A continuación, se muestran los resultados obtenidos:

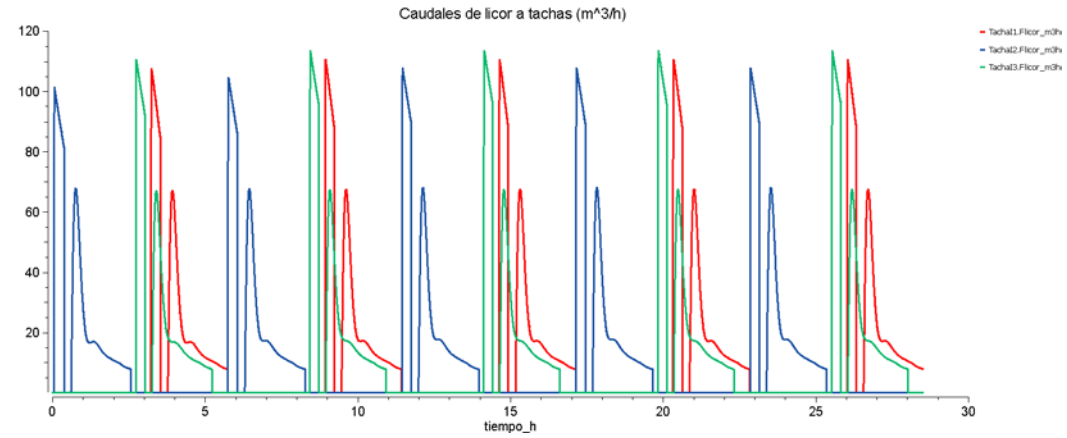

Figura 5.5: ejecución de las tachas a lo largo del tiempo.

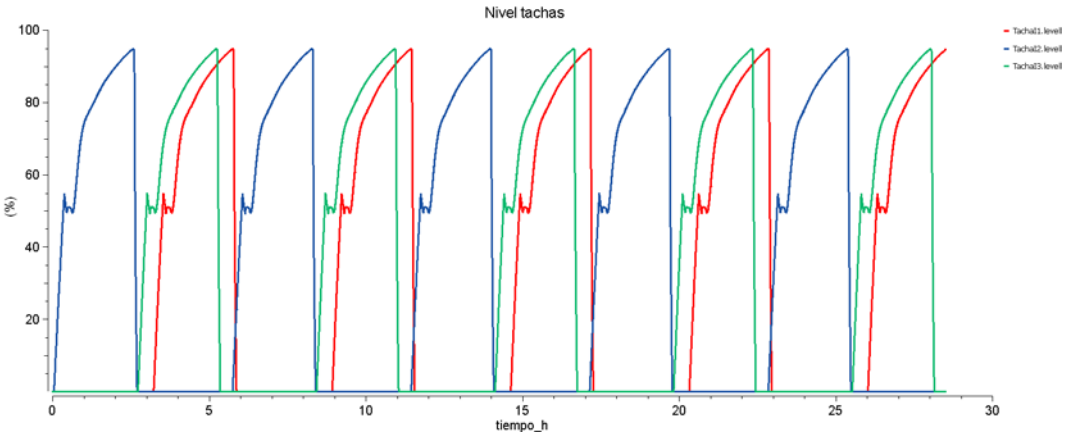

Figura 5.6: % de llenado de las tachas a lo largo del tiempo.

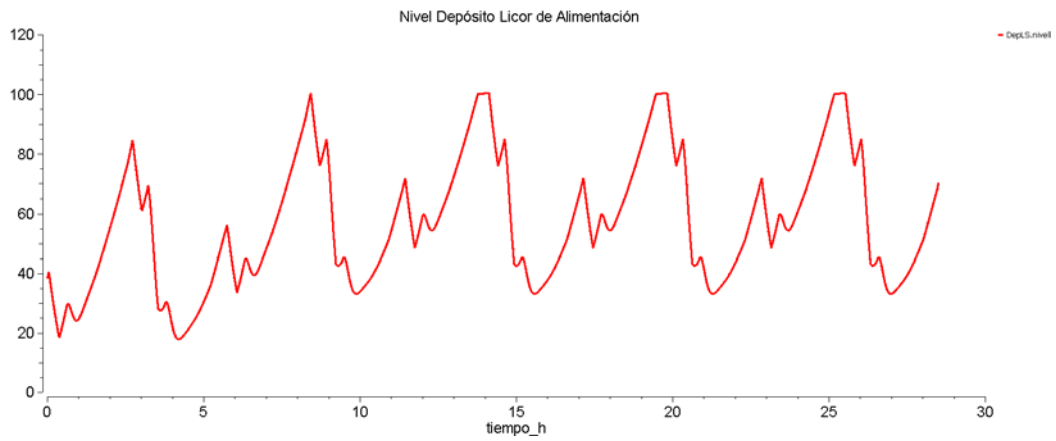

Figura 5.7: altura del depósito a lo largo del tiempo.

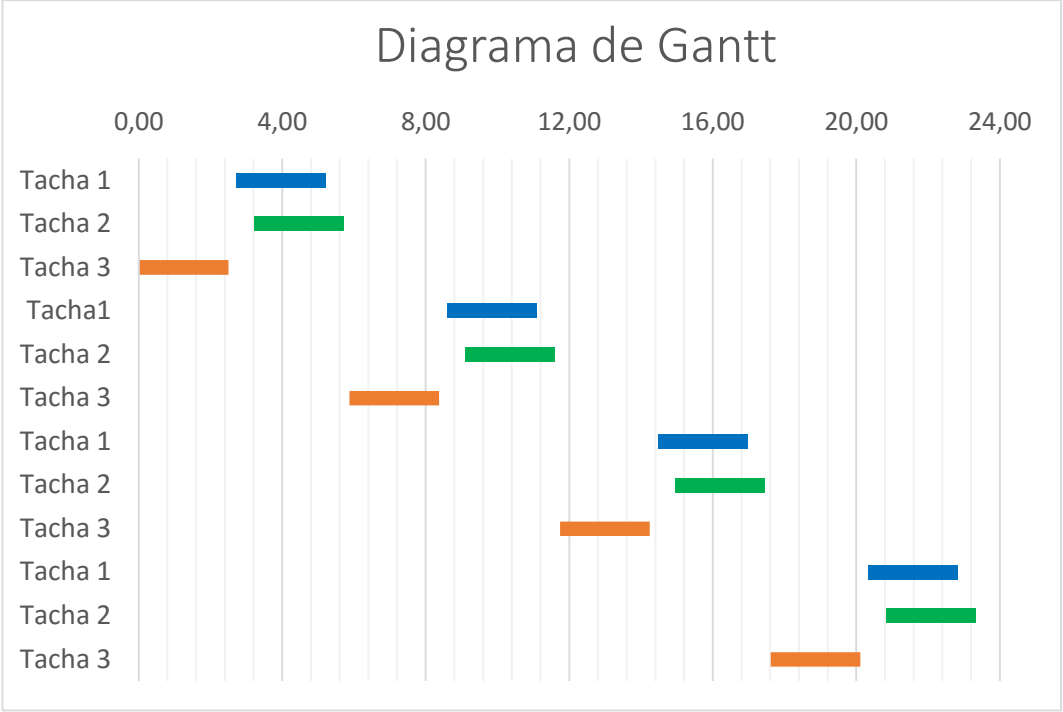

Figura 5.8: diagrama de Gantt del scheduling obtenido.

Como podemos observar, pese a que la optimización de GAMS daba unos resultados que cumplían las restricciones impuestas, de manera que el depósito mantuviera su altura entre unos límites aceptables, al pasarlo al sistema real no se obtienen exactamente los mismos resultados. Esto es debido a que tanto los perfiles de consumo de cada tacha como el caudal de

entrada al depósito se han modelado de una forma aproximada, de manera que no se corresponden exactamente con el sistema real. Como hemos dicho anteriormente, la dificultad de esos perfiles de consumo/entrada es elevada, resultando imposible su modelado exacto y siendo por tanto necesarias dichas aproximaciones.

Pese a todo ello, los resultados son muy buenos, ya que hemos conseguido simular correctamente un ciclo productivo de 25 horas respetando las alturas del depósito en todo momento salvo en unos instantes muy pequeños de tiempo. Este error puede ser solventado sin problemas cambiando algún parámetro del modelo como por ejemplo la altura máxima permitida, pero consideramos que los resultados son correctos y merecen ser mostrados.

CAPÍTULO 6

CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

6.1. Conclusiones finales

Creemos que los resultados han sido satisfactorios, dado que hemos logrado cumplir todos los objetivos planteados en el Capítulo 1. Por una parte, el modelo en tiempo continuo ha sido planteado teniendo en cuenta las restricciones del sistema real, habiendo reproducido los perfiles de caudales de una forma bastante precisa, algo nada fácil teniendo en cuenta la dificultad que entrañan los perfiles reales. En segundo lugar, se han logrado obtener una salida de la simulación en un tiempo bastante bueno teniendo en cuenta la dificultad del problema tratado. Por último, tras haber comprobado la factibilidad de los resultados en EcosimPro, podemos decir que estos se ajustan bastante bien al sistema real (las diferencias son debidas sobre todo a que el perfil de caudal de entrada al depósito es imposible de modelar fielmente, por lo que se recurre a aproximaciones que conllevan un ligero error), por lo que podrían perfectamente ser utilizados en una hipotética fase productiva.

6.2. Futuras líneas de trabajo

- Extender el modelo a otras etapas del cuarto de azúcar, no sólo a las tachas de primera. Esto puede conllevar introducir alguna recirculación de flujo hacia los depósitos.

- Definir otras funciones objetivo: en este caso simplemente hemos optado por crear un scheduling que fuese reproducible en el tiempo, pero podría ser interesante proponer otras funciones objetivo como pueden ser minimizar el tiempo de ciclo, o introducir un coste por uso de equipo que deba ser minimizado.

- Proponer cambios en el modelo que lo simplifiquen: que se trate de un modelo no lineal es un inconveniente. Sin embargo, se ha creído necesario dado que hay que evaluar la altura del depósito en instantes concretos de tiempo, pero podría ser interesante intentar proponer otra formulación que evitase este inconveniente.

Bibliografíá

A.Méndez, C., Cerdá, J., E.Grossmann, I., Harjunkoski, I., & Fahl, M. (2006). "State of the art review of optimization methods for short term scheduling of batch processes". *ELSEVIER*.

ACOR. (Mayo de 2017). *Cooperativa ACOR*. Obtenido de http://www.cooperativaacor.com/es/extraccion/art/189/

Alonso, A. A. (2016). *"Comparativa de modelos y solvers para la optimización de variantes de problemas de rutas de vehículos".* Valladolid.

Anónimo. (2016). *"Clasificación de procesos industriales".*

AZUCARERA. (Mayo de 2017). *AZUCARERA*. Obtenido de http://www.azucarera.es/descubre_fases_proceso.php#

De Prada, C. (2016). *"Scheduling. Apuntes UVa".* Valladolid.

E. Rosenthal, R. (2017). *"GAMS —A User's Guide".*

Euler, L. (1741). "Comm. academiae scientiarum Petropolitanae". 128-40.

GAMS. (2017). Obtenido de https://www.gams.com/

García, A. G. (2001). *"Modelado y simulación de procesos batch: sección de cristalización de una azucarera".* Valladolid.

Jarquin, J. A. (2015). *"Secuenciamiento óptimo del proceso de cristalización de una azucarera".* Valladolid.

L.Pinedo, M. (2016). *"Scheduling: Theory, Algorithms and Systems".* Springer.

Lee, J., & Leyffer, S. (2012). *Mixed Integer Nonlinear Programming.* Springer.

Mazaeda, R., De Prada, C., & P. Cristea, S. (s.f.). "Plant-wide hierarchical optimal control of a crystallization process". 1-6.

Meindl, B., & Templ, M. (2012). *"Analysis of commercial and free and open source solvers for linear optimization problems".*

R. Bussieck, M., & Vigerske, S. (2014). MINLP Solver Software.

Rocha, L., González , C., & Orijuela, J. (2011). "Una revisión al estado del arte del problema del ruteo de vehículos: Evolución histórica y métodos de resolución". *Ingeniería*, 35-55.

Sáez Aguado, J. (2014). *"Métodos matemáticos en ingeniería de organización I".* Apuntes de clase.