Programación de operaciones a corto plazo mediante la técnica de programación con restricciones: Características y resultados sobre diversos entornos productivos.

Área temática: C – Gestión de Operaciones y Logística

Novas, Juan M.

CIEM (UNC-CONICET)
Medina Allende s/n, 5000, Córdoba, Argentina
jmnovas@famaf.unc.edu.ar

RESUMEN
El presente trabajo tiene como objetivo introducir las características y ventajas principales del uso de técnicas de programación con restricciones al abordar el problema de la programación de operaciones en el corto plazo ("scheduling"). Este problema ha sido y continúa siendo de gran interés tanto para la comunidad académica como para la práctica industrial. Es por ello que numerosas metodologías de resolución se han propuesto durante décadas, siendo la programación con restricciones ("Constraint Programming, CP") una de las más ventajosas para tratarlo.
Mediante esta contribución, se presentan los principales beneficios que otorga la programación con restricciones, describiendo sus: (i) facilidades de modelado en lenguaje de alto nivel, (ii) desarrollo incremental e iterativo de modelos, (iii) posibilidades de resolución en tiempos de cómputo aceptables, entre otras características. Finalmente, se muestran los hallazgos y resultados obtenidos al utilizar CP en distintos casos de estudio provenientes de diversos entornos industriales, tales como sistemas de manufactura flexible y plantas de procesos "batch".

Palabras Claves: Programación de operaciones, programación con restricciones, gestión de la producción, optimización de la producción.

Abstract:
This work aims to introduce the main characteristics and advantages of Constrain Programming (CP) techniques when they are used to tackle the scheduling problem at manufacturing settings. During the last decades, the scheduling problem has been a challenge to face not only to the academic community but also to the industrial practitioners. Therefore, numerous resolution methodologies have been proposed, being Constrain Programming one of the most suitable techniques to apply in these types of problems.
In this work, the main benefits of Constrain Programming are described: (i) simple modelling using a high level language, (ii) incremental development of models, and (iii) instantiation of good quality solutions in short computational times, among other advantages. Finally, the obtained results using CP models for different case studies are shown. Diverse industrial environments, such as flexible manufacturing systems and batch plants, are addressed.

Keywords: Scheduling, constraint programming, production management, manufacturing production optimization.

1. INTRODUCCIÓN

En la actualidad, las compañías industriales forman parte de un contexto globalizado que se caracteriza por su naturaleza dinámica y creciente competitividad. Este entorno competitivo determina la existencia de numerosos factores que influyen sobre los sistemas productivos. Entre aquellos que tienen mayor impacto sobre los mismos, se destacan: (i) el aumento e innovación permanente en la variedad de productos, (ii) la dinámica en los requerimientos de los clientes (respecto al producto, a fechas de entrega, etc.) y el aumento de presiones provenientes del mercado, (iii) la integración de la cadena de suministro y, (iv) el creciente empleo e integración de tecnologías de información, entre otros. Sumado a estos factores exógenos al ambiente industrial, los procesos de producción se ven afectados de manera continua y permanente por su dinámica intrínseca.

Como consecuencia tanto del contexto externo como de los requerimientos internos de la organización, el área de producción debe resolver a corto plazo (diaria o semanalmente) qué órdenes de trabajo se procesarán, a qué máquinas se asignarán las mismas en las distintas etapas productivas, qué recursos serán necesarios, en qué momento se ejecutará cada tarea de manufactura, o al menos en qué secuencia se realizarán, entre otras decisiones críticas. Para poder hacer frente a estas cuestiones y resolverlas con eficiencia, las organizaciones deben contar con herramientas que den soporte a los procesos de planificación y programación de la producción.

En este trabajo se hará foco particularmente en el la función de la programación de operaciones a corto plazo o “scheduling”, considerando que la empresa ya cuenta con una planificación a mediano y largo plazo. La actividad de la programación de la producción se considera central dentro de las compañías manufactureras y debe ser percibida como un proceso continuo y permanente adaptación a cambios. La importancia de esta actividad se observa en las ventajas que la misma otorga a la industria: (i) mayor eficiencia en el manejo de sus operaciones, (ii) reducción de los costos operativos, (iii) uso eficiente de recursos (iv) disminución de tiempos muertos, (v) aumento del "throughput" y la productividad, (vi) así como de la capacidad de cumplimiento, entre otros beneficios.

A pesar de conformar una función crítica para la correcta gestión de la producción, la programación de operaciones no es una práctica común en las industrias manufactureras, especialmente en las pequeñas y medianas compañías. Existen varias razones por las cuales esto ocurre [1,2], entre las que podemos destacar: (i) los sistemas de programación de operaciones asumen cierto carácter estático de la planta, cuando la principal característica es su dinámica natural, (ii) no es posible adaptar las agendas de producción generadas cuando ocurren cambios, quedando desactualizadas y sin posibilidad de ser implementadas, (iii) la comunicación de los sistemas de “scheduling” con los de negocio (como los ERP o MRP) y con los sistemas de recolección de datos, no es trivial, (iv) el desarrollo de herramientas de soporte para la programación de la producción es complejo y de difícil mantenimiento, (v) los sistemas “enlatados” son demasiado estándares y tienen dificultades para adaptarse a las particularidades de cada planta.

Una metodología que permite dar soporte a la resolución del problema, salvando algunos de los inconvenientes por los cuales estos desarrollos no se emplean en las industrias, y que durante los últimos años ha demostrado ser eficiente para tal fin, es la programación con restricciones o “Constraint Programming” (CP) [3,4]. Esta técnica ha sido empleada para agendar actividades sobre distintos dominios, tales como: sistemas ferroviarios [5], líneas de ensamble [6,7], gestión de personal [8], plantas de procesos [9,10], plantas de manufactura discreta y flexible [11,12], entre otros.

El presente trabajo tiene como objetivo presentar las ventajas y características distintivas de la programación con restricciones, como metodología de resolución al problema de la programación de operaciones a corto plazo, en diferentes tipos de procesos industriales. En la sección que se presenta a continuación se define el problema de la programación de operaciones en plantas industriales, para luego, en la Sección 3, describir la técnica de programación con restricciones. En la Sección 4 se muestran los resultados logrados al emplear modelos CP en entornos productivos de distinto tipo. Finalmente, se presentan conclusiones y trabajo futuro.

2. EL PROBLEMA DE LA PROGRAMACIÓN DE OPERACIONES (“SCHEDULING”)

La programación de operaciones a corto plazo o “scheduling” es una actividad cuyo principal objetivo puede definirse, de manera general, como la búsqueda de: (i) la asignación de recursos de capacidad limitada a un conjunto de trabajos o actividades que deben ser ejecutados dentro de un dado período temporal (u horizonte de programación) y/o ajustarse a determinados vencimientos, y (ii) la definición de una agenda de ejecución de dichas tareas, o al menos una secuencia de ejecución de las mismas. Con frecuencia, la programación de tareas busca lograr su propósito de manera eficiente respecto a una o más medidas de desempeño o performance.

Según el dominio o entorno donde se busca resolver un problema de “scheduling”, los recursos donde deben asignarse las operaciones o actividades pueden ser, por ejemplo, unidades de

procesamiento en un entorno computacional, pistas en un aeropuerto, dársenas en un puerto, personal en una construcción, equipos en una planta industrial, etc. Las tareas, por su parte, pueden ser ejecuciones de programas de computación, aterrizajes y arribos en un aeropuerto, etapas en una obra en construcción, operaciones en un proceso productivo, etc. [13]. Los recursos se describen en términos de su tipo y su capacidad, mientras que cada tarea lo hace en relación al tiempo en que debería iniciarse y finalizarse, su duración y sus requerimientos de recursos para poder ejecutarse. Luego también surgen condicionamientos o restricciones que el problema debe considerar, vinculados a la relación entre conjuntos de recursos (por ej. su conectividad, secuencia en una línea, etc.), conjuntos de tareas (por ej. relaciones de precedencia, secuencias prohibidas, etc.) y el vínculo entra tareas y recursos (por ej. asignaciones prohibidas, capacidad utilizada, etc.).

El problema de "scheduling" es complejo tanto desde un punto de vista académico como práctico. Como consecuencia, ha despertado un notable interés en la comunidad científica desde hace más de medio siglo, la que ha desarrollado innumerables propuestas para el tratamiento del mismo, logrando importantes avances provenientes de distintas áreas de estudio (investigación operativa, informática, sistemas inteligentes). Por otra parte, existe un claro interés práctico en dar adecuadas respuestas al problema en los distintos dominios donde éste existe. Los ambientes industriales conforman una de estas áreas donde es de vital importancia desarrollar un correcto tratamiento y generar herramientas computacionales de soporte.

En este trabajo, se presentan el problema de la programación de operaciones en plantas de procesos "batch" [14], y sistemas de manufactura flexible [12]. El problema de "scheduling" vinculado otros dominios industriales, tales como estaciones "wet-etch" y entornos de tipo taller o "job-shop", puede estudiarse en [15] y [16], respectivamente. Luego de introducir las características principales de los entornos, se presenta un apartado describiendo algunas medidas de desempeño.

2.1. Plantas de Procesos "Batch"

Es común encontrar en la industria química (farmacéuticas, pinturas, etc.), muchas plantas multiproducto con múltiples etapas, cada una de las cuales posee equipos diferentes en paralelo (ver Figura 1). Este tipo de ambientes generalmente opera programando un cierto número de órdenes de producción (o "batches"), para cumplir con una determinada demanda, las cuales deben estar procesadas con anticipación a una dada fecha de entrega. El desafío consiste en resolver el problema de programar las órdenes considerando las múltiples restricciones que posee el entorno: asignaciones de producto a equipo prohibidas, "changeovers" dependientes de la secuencia, restricciones topológicas, diferentes tipos de recursos renovables (personal, vapor, etc.), distintos tipos de políticas de almacenamiento y espera, etc.

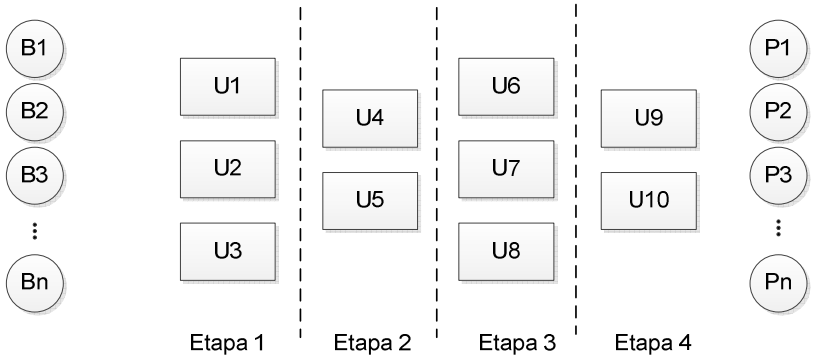

Figura 1. *Esquema de una planta de procesos tipo "batch". B.batches; U.unidades; P.productos elaborados.*

Se considera que al inicio del horizonte de "scheduling" existe un conjunto de órdenes de producción a procesar, el cual fue definido de antemano en base a pedidos de clientes. Una dada orden puede estar compuesta por más de un "batch", teniendo cada uno de éstos una fecha de entrega y un tiempo de disponibilidad para ser procesado.

En cada etapa del proceso productivo, existe un conjunto de equipos o unidades que operan en paralelo, las cuales pueden ser diferentes. Cada equipo pertenece a una única etapa y puede estar conectada con todos o sólo algunos equipos de la etapa siguiente. Las unidades también tienen un tiempo de disponibilidad propio. El tiempo de procesamiento de un "batch" en un equipo es determinístico, conocido de antemano y depende del "batch" y del equipo. Además, pueden existir tiempos de limpieza sobre las unidades, el cual ocurre al cambiar de "batches" en un equipo. Este tiempo se conoce como tiempo de "chageover" y puede depender de la secuencia de los productos y del equipo.

Entre otras restricciones que se deben considerar, se encuentran: secuencias de productos prohibidas, asignaciones de producto a equipo prohibidas, limitaciones en la disponibilidad de otros recursos distintos de las unidades. Sumado a ello, pueden existir distintas políticas de almacenamiento y espera intermedia, tales como almacenamiento intermedio ilimitado (unlimited intermediate storage, UIS), sin almacenamiento intermedio (non-intermediate storage, NIS) con espera ilimitada (unlimited wait, NIS-UW), sin espera (zero wait, NIS-ZW), y espera limitada (finite wait, NIS-FW).

2.2. Sistemas de Manufactura Flexible

Los sistemas de manufactura flexible (Flexible Manufacturing Systems, FMS) son entornos de producción de partes discretas, altamente automatizados, compuestos por múltiples centros de maquinado multipropósito, pulmones de almacenamiento y al menos un vehículo guiado automáticamente (automated guided vehicles, AVG), todo integrado mediante un sistema de control informático (ver Figura 2). Este tipo de ambientes productivos se caracterizan por ser altamente flexibles y poseer alta productividad. El desafío concerniente a la actividad de programación de la producción en estos entornos, consiste esencialmente en lograr programar eficientemente no sólo las actividades de producción, sino también las de almacenamiento y transporte. Entre los aspectos que condicionan la correcta programación de tareas, se encuentran: las características propias del sistema FMS, el nivel de automatización, y qué tipos de recursos componen el ambiente. Otra cuestión relevante a tener en cuenta en los FMS es la correcta gestión de las herramientas.

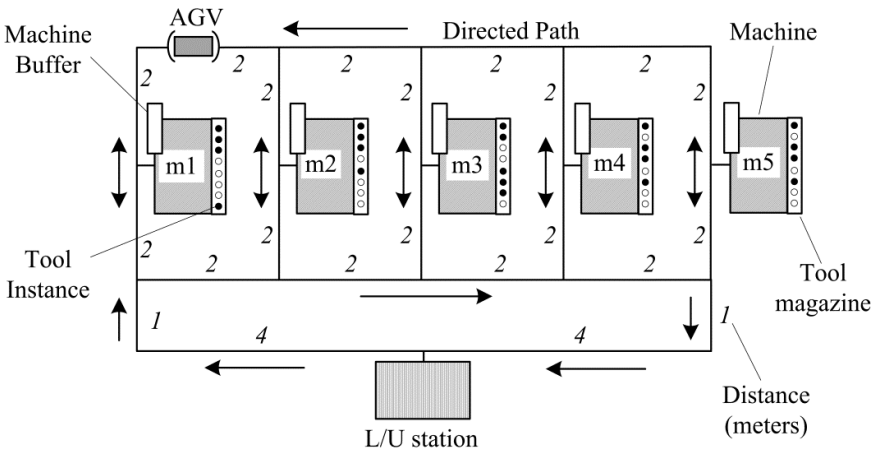

Figura 2. *Esquema de un sistema de manufactura flexible (Fuente [12]). AGV.vehículo; L/U. estación de carga y descarga de partes.*

La configuración de FMS que se trata aquí y de la cual se muestran resultados en la sección 4, consiste en: (i) una estación de carga y descarga de partes (L/U), (ii) máquinas multipropósito, cada una con un contenedor de herramientas de capacidad limitada, (iii) herramientas, (iv) un almacén local de partes asociado a cada máquina, para contener partes hasta que la máquina esté disponible, (v) un único vehículo automático para el transporte de partes, (vi) una red con sentido de circulación preestablecido.

El problema a resolver consiste en agendar un conjunto de partes que requieren ser procesadas en el FMS, disponibles al inicio del horizonte de programación. Todas las partes entran/salen a través de la estación L/U y cada parte requiere de una determinada secuencia de operaciones. Cada operación es ejecutada en una máquina perteneciente al conjunto de máquinas alternativas habilitadas para dicha operación. Cada máquina puede realizar una operación a la vez. Las herramientas son clasificadas por tipo, existiendo un número limitado de copias por tipo. Éstas son asignadas a los dispositivos herramentales de capacidad limitada. El tiempo de procesamiento de una parte dependerá de la máquina y la herramienta empleada. Como las herramientas permanecen fijas en los herramentales durante el período de fabricación, las partes son trasladadas entre las máquinas mediante un AGV.

Debe entonces considerarse que existe un tiempo de transporte de las partes y que, debido a esto, no todas las partes comenzarán a ser ejecutadas en su primer operación al momento cero. La Figura 3 ilustra todos los movimientos del AGV que deben ser considerados: viaje con carga, viaje sin carga y AGV en espera.

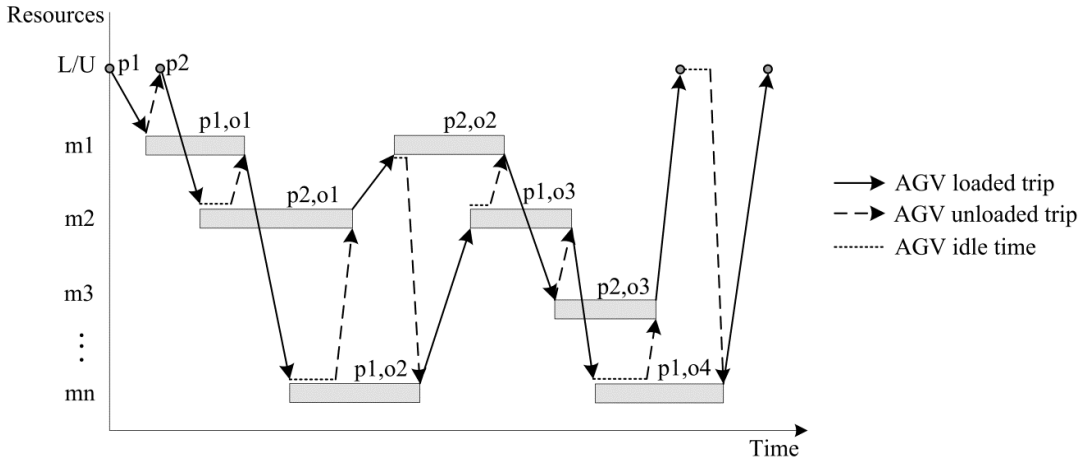

Figura 3. *Ilustración de viajes requeridos por el AGV para transportar las partes p1 y p2 durante su procesamiento (Fuente [12]).*

2.3. Medidas de desempeño

Si bien en teoría, para un dado problema, se podría buscar soluciones factibles que no persigan ningún objetivo en particular, esto no ocurre cuando se abordan dominios industriales. Por lo general, en las contribuciones realizadas en el área de "scheduling" de plantas industriales se busca optimizar alguna medida de desempeño o performance. Existen medidas que tratan de lograr soluciones eficientes respecto al tiempo, otras que buscan optimizar costos, etc. Entre las que pertenecen al primer grupo, encontramos: makespan (tiempo de finalización de todas las operaciones agendadas), tardanza total, anticipación total, número de órdenes tardías, etc. Por otra parte, existen las medidas que persiguen la reducción de costos operativos por cada equipo, por uso de herramientas, por uso de energía u otro recurso, etc.

A continuación, definiciones de tres medidas que se encuentran entre las más empleadas:

(i) Makespan (Mk): El makespan es la medida más empleada para evaluar la performance de una propuesta. Es igual al tiempo de finalización de la última operación realizada, el momento en el cual todas las órdenes de producción has sido procesadas. También puede definirse como el máximo tiempo de fin entre las últimas operaciones de todas las partes o lotes.

(ii) Tardanza Total: Esta medida se define como la suma de las tardanzas individuales de cada parte o lote. A su vez, la tardanza de cada parte o lote se mide como la magnitud de la demora en la finalización de dicha parte o lote respecto a su fecha de entrega definida al inicio del horizonte. Cuando el tiempo de finalización es menor a la fecha de entrega, se considera que su valor es igual a cero.

(iii) Adelanto Total: Se define como la suma de los adelantos de todas las órdenes de trabajo. El adelanto de una parte o lote se establece como la diferencia entre la fecha de entrega pactada y el tiempo de fin efectivo. Si la finalización de la parte o lote es mayor a su fecha de entrega, entonces no hay adelanto y su valor es cero (habrá tardanza).

3. CARACTERÍSTICAS DE LA PROGRAMACIÓN CON RESTRICCIONES

Como se mencionara previamente, existen numerosas técnicas para resolver el problema de "scheduling" en plantas industriales. En este trabajo, describiremos los principales aspectos que caracterizan a la Programación con Restricciones o CP.

La técnica de programación con restricciones se refiere a la implementación computacional de algoritmos avanzados creados para abordar eficientemente los problemas de satisfacción de restricciones ("Constraint Satisfaction Problems",CSP) [3]. Esta técnica ha sido reconocida por muchos expertos como una herramienta adecuada para el modelado y resolución de problemas combinatorios de optimización, siendo el problema de "scheduling" uno de ellos [4]. CP proviene principalmente del área de Inteligencia Artificial, desde donde se creó con el objeto de reducir los tiempos de modelado y resolución de problemas del tipo CSP.

Un problema CSP se define como una terna (X, D, C) donde:

- X representa a un conjunto finito de variables $\{x_1,\ldots,x_n\}$, donde n > 0,
- D es un conjunto de finito de dominios $D_1,\ldots,D_n$, donde la i-ésima componente D_i es el dominio que contiene todos los posibles valores que la variable x_i puede adoptar.
- C representa a un conjunto de restricciones, $C_1,\ldots,C_k$, donde k puede adoptar cualquier valor entero, y cada restricción (C_j) se define sobre un el conjunto (o un subconjunto) de variables $\{x_1,\ldots,x_n\}$. De esta manera, queda restringido el conjunto de valores que las variables involucradas pueden tomar de manera simultánea.

Una solución a un determinado problema CSP consiste en la asignación de valores a todas las variables, de forma que se satisfagan todas las restricciones. Cuando se aborda un problema CSP, se enfrenta alguna de las siguientes tres situaciones. Se pretende encontrar (i) una única solución factible, sin ninguna preferencia en particular, (ii) todas las soluciones posibles del problema, (iii) una solución óptima, o al menos una solución de buena calidad, considerando alguna medida de desempeño o performance expresada en función de algunas o todas las variables del problema. En la actualidad existen diferentes paquetes de "software" que permiten formular y resolver problemas de tipo CSP, tales como: IBM ILOG CPLEX CP Optimizer [17], ECLipSe [18], OZ [19].

Dado un problema a solucionar mediante CP, en el cual se persigue la obtención de una solución óptima o de buena calidad, se debe atravesar dos fases de resolución: (i) *Modelado del problema.* Para la especificación de un modelo CP, se requiere definir las variables y sus respectivos dominios, el conjunto de restricciones que las vincula, una función objetivo y su relación con las variables de decisión. Las variables pueden ser de distinto tipo (enteras, reales, booleanas, simbólicas), al igual que las restricciones (algebraicas, lógicas, híbridas). La flexibilidad de esta forma de representación lleva a que un problema específico pueda ser modelado de distintas maneras, por lo que definir el modelo más adecuado no es una tarea trivial. (ii) *Resolución del problema.* En esta etapa se procesan las restricciones mediante algoritmos de propagación de restricciones basados en técnicas de consistencia y/o algoritmos de búsqueda. En general, estas herramientas se combinan ya que las técnicas de consistencia reducen el espacio de soluciones, mientras que los algoritmos de búsqueda exploran dicho espacio. Las herramientas comerciales disponibles en la actualidad incorporan este tipo de "solvers" y algunas permiten que quien formula el modelo incorpore estrategias de solución que toman ventaja de la estructura propia del problema.

Los problemas de "scheduling" pueden representarse como problemas CSP donde, generalmente, los aspectos temporales referidos a las tareas son las variables del problema (tiempos de inicio, duración y fin), las cuales pueden adoptar valores, es decir instanciarse, a partir de un conjunto de valores posibles, definido por el horizonte de "scheduling" y acotado por restricciones propias del problema. Las restricciones que vinculan las variables del problema pueden ser muy variadas, dependiendo del tipo de problema de "scheduling" abordado. Entre las más usuales se encuentran las restricciones de precedencia de actividades, de asignación de tareas a recursos, de cotas en los tiempos de inicio y fin de las actividades, etc.

Entre las numerosas ventajas que provee CP, es posible destacar:

- Las implementaciones de CP se realizan en lenguajes declarativos de alto nivel, lo cual permite el desarrollo de modelos de manera más sencilla, permitiendo el uso de expresiones lógicas;
- Permite el uso de constructores o funciones predefinidas, lo que agiliza el proceso de modelado;
- Brinda flexibilidad y facilidad para la incorporación, eliminación o modificación incremental de restricciones, y otros componentes de un dado modelo;
- Facilita, en general, la detección de especificaciones no factibles del problema de manera inmediata o en tiempos de cómputo muy reducidos;
- Brinda la capacidad para encontrar soluciones factibles iniciales en forma rápida e instanciar soluciones óptimas o subóptimas en bajos tiempos de CPU.

Estas características, que fueron tratadas extensamente por Zeballos y colaboradores [20], justifican ampliamente la adopción de la programación con restricciones como herramienta para abordar problemas de "scheduling" en ambientes industriales.

La representación de los modelos CP por medio de los cuales se obtienen las soluciones que se muestran en la siguiente sección, se realizó mediante el lenguaje de alto nivel OPL, subyacente en el ambiente IBM ILOG CPLEX Optimization Studio. El empleo de este ambiente de modelado para la representación y resolución de los modelos CP, implica aceptar la existencia de dos suposiciones subyacentes importantes, a saber: el tiempo es una variable discreta que gestiona en forma implícita el "solver" del ambiente y la capacidad de los recursos debe expresarse mediante valores enteros.

Es necesario hacer notar que OPL junto al paquete IBM ILOG CP Optimizer provee algunas variables, funciones y restricciones específicas del problema de "scheduling", cuyo objetivo es facilitar el modelado de los problemas de "scheduling". Algunas de estas posibilidades de modelado son:

- **Variables de intervalo**: En OPL, las actividades son representadas por variables denominadas "interval variables". Una variable de intervalo representa un tiempo durante el cual una actividad es realizada, pero cuya ubicación temporal no es conocida previo a la resolución. Estas variables se caracterizan por los atributos de inicio, duración y fin. También pueden ser declaradas como opcionales. Existe una serie de funciones disponibles para operar sobre intervalos: *startOf(), endOf(), sizeOf(), presenceOf()*. Si la variable es declarada como opcional,

la función *presenceOf* devuelve el valor 1 cuando la variable está presente en la solución, y 0 en caso contrario. La Figura 4 ilustra estos conceptos.

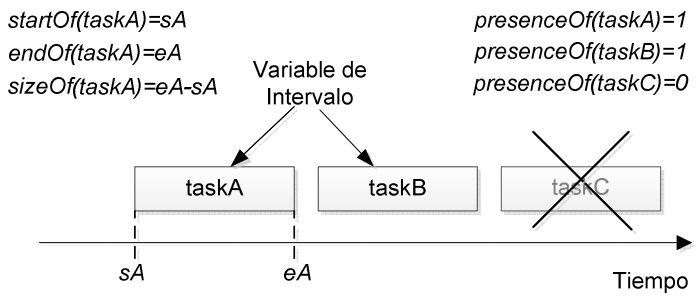

Figura 4. Ilustración de una solución donde sólo taskA y taskB están presentes.

- **Variables de secuecia**: representan un orden sobre un conjunto de variables de intervalo, incluidas en la solución de un dado problema. Estas variables son de utilidad para expresar la secuencia de tareas o actividades asignadas a recursos. Existen distintos tipos de funciones y restricciones que pueden ser aplicadas sobre las variables de secuencia, tal como *noOverlap()*, que impide a actividades ubicadas en un mismo recurso superponer sus intervalos de ejecución. Esta función también facilitan la representación de los tiempos de changeover.
- **Restricción de acumulación**: es usual encontrar problemas donde existen limitaciones en la capacidad de los recursos renovables, por ejemplo, un tanque, un almacén, etc. Para abordar esta característica, se emplean funciones acumulativas sobre recursos renovables, las cuales modelan la variación de una cantidad en el tiempo, usualmente demandada por una tarea o actividad. Una tarea normalmente incrementa el uso acumulado de un recurso al iniciar su ejecución, liberando dicha capacidad cuando termina la ejecución de la misma. Existen varios tipos de funciones acumulativas, dos de las cuales son las más distintivas: *step()* y *pulse()*. Su uso depende del tipo de consumo de la capacidad de un recurso que realicen las actividades. La Figura 5 muestra una ilustración para las funciones *f = pulse(taskA,1)* y *f = pulse(taskB,1)*.

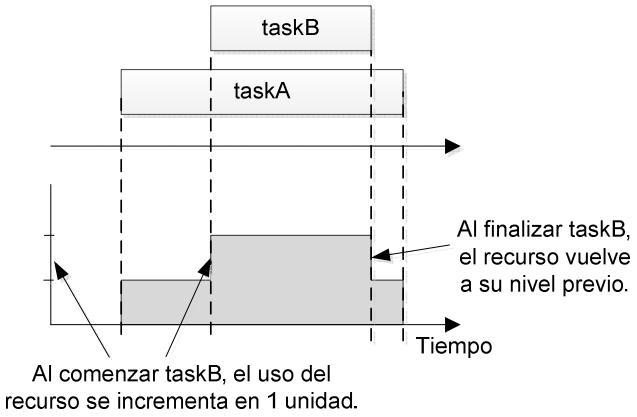

Figura 5. Ilustración de cómo trabaja la función acumulativa *pulse()* .

- **Restricciones de precedencia**: este tipo de restricciones, comunes a los problemas de "scheduling", son empleadas para restringir la posición relativa que pueden tener las variables de intervalo en una dada solución. Permiten especificar cuando una tarea debe iniciar o finalizar con respecto al inicio o fin de otra tarea. Entre estas restricciones se encuentran: *endBeforeStart(), startBeforeEnd(), endAtStart(), etc*. Por ejemplo, la función *endBeforeStart(taskA,taskB)* asegura que *taskA* finalice antes del inicio de *taskB* (tal como se observa en la Figura 4).
- **Restricción "Alternative"**: este constructor permite controlar la ejecución y sincronización de diferentes tareas. La declaración *alternative(taskA,{taskB1, taskB2, taskB3})* establece que la variable de intervalo *taskA* (no opcional) es ejecutada si y sólo si uno de los intervalos del conjunto *{taskB1, taskB2, taskB3}* (intervalos declarados como opcionales) es ejecutado. Existe una relación de tipo *XOR* entre este conjunto de tareas alternativas opcionales (Figura 6). La actividad *taskA* se sincronizará con la tarea seleccionada, es decir, compartirán el mismo tiempo de inicio y fin.

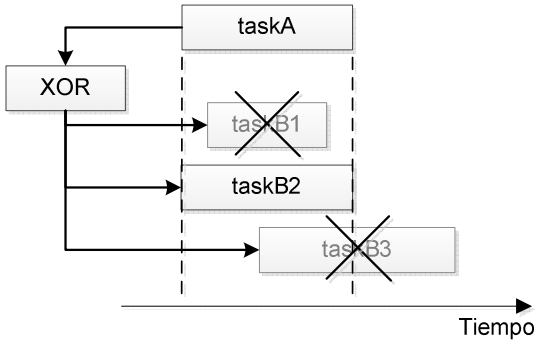

Figura 6. Ilustración de *alternative(taskA,{taskB1, taskB2, taskB3})*

- **Restricción "Span"**: Esta restricción se emplea para sincronizar un intervalo con un conjunto de otros intervalos. La expresión *span(taskA,{taskB1, taskB2, taskB3})* indica que taskA debe abarcar todo el conjunto de tareas taskB. Esto es, taskA inicia junto al primer intervalo presente de tipo B y finaliza junto a la última tarea presente de tipo B. Esta situación se ilustra mediante la Figura 7.

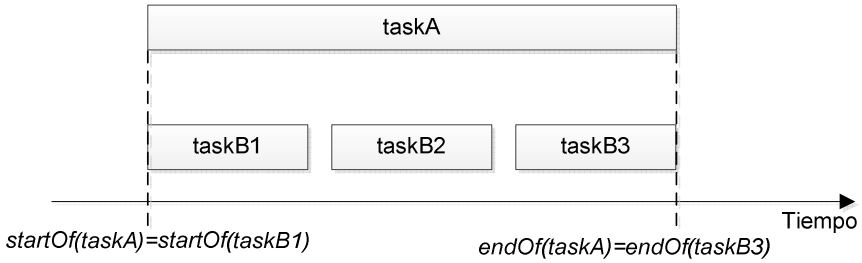

Figura 7. Span de *taskA* sobre *{taskB1, taskB2, taskB3}*.

4. RESULTADOS DE MODELOS CP DESARROLLADOS

En esta sección se presentan algunos de los resultados alcanzados al emplear CP para abordar la resolución de problemas de "scheduling". En una primera sección, se muestran ejemplos de plantas de procesos "batch", mientras que en una segunda, resultados sobre sistemas de manufactura flexible. No es el objetivo de esta sección presentar en detalle los modelos CP, así como tampoco analizar exhaustivamente todas las soluciones, sino ilustrar sobre el potencial de la técnica al presentar algunos casos resueltos. Es por ello que se deja a consideración del lector la extensión de la lectura de los modelos y los resultados hallados, lo cual puede realizar a través de los documentos referenciados [12] y [14].

4.1. Plantas de Procesos "Batch"

A continuación se presentan algunos casos de estudio abordados, los cuales varían en el número de órdenes de producción consideradas, el tipo de planta, diferentes aspectos limitantes y medidas de desempeño. Los resultados son comparados con los presentes en la literatura. También se muestran soluciones obtenidas al tratar problemas de gran tamaño (sin referencias en la literatura). Todos los casos de estudio han sido resueltos mediante CPLEX Optimization Studio [21] en una computadora con procesador Intel(R) Core(TM) i5-2450M CPU @ 2.50 GHz, con 4 Mb de memoria RAM.

Los casos de estudio P1 a P4, asumen que cada orden de producción se compone de un único "batch", mientras que P5 aborda un problema con órdenes multi-batch. Una breve descripción de cada ejemplo tratado se presenta a continuación:

- *Ejemplo P1*: Consiste en una planta farmacéutica con 6 etapas de producción y 17 unidades de procesamiento diferente. Se deben agendar 50 batches de diferentes productos, minimizando makespan y considerando diversas restricciones [22].
- *Ejemplo P2*: Un conjunto de 60 órdenes deben agendarse en una planta similar a la descripta para el ejemplo 1, también minimizando makespan. La variante P2.1 considera una política de almacenamiento ilimitado (UIS), mientras que la variante P2.2 trabaja con una política sin almacenamiento y cero espera (NIS-ZW) [23].
- *Ejemplo P3*: Este caso de estudio considera una planta multiproducto de 5 etapas, con 25 unidades diferentes y restricciones topológicas. También considera tiempos de changeover dependientes de la secuencia. Se minimiza makespan. Los problemas considerando 12 órdenes y políticas UIS, NIS-ZW y NIS-UW, se denominan P3.1, P3.2 y

P3.3, respectivamente. Los casos para 22 órdenes, bajo las mismas políticas, son P3.4, P3.5 y P3.6 [20].

- *Ejemplo 4*. Este caso corresponde a una planta con 5 etapas y 12 unidades de procesamiento. Se deben agendar 12 órdenes considerando changeover dependiente de la secuencia y disponibilidad limitada de ciertos recursos renovables, como vapor y electricidad, los cuales son requeridos por ciertos batches en algunas etapas. El objetivo perseguido es la minimización de la tardanza, bajo política UIS [24].
- *Ejemplo 5.* Este ejemplo toma como base el caso 1 y lo modifica para considerar producción mediante campañas, de un gran número de batches. Un conjunto de 9 órdenes, donde cada una tiene entre 4 y 10 batches, necesita programarse en planta (un total de 70 batches). Entre campañas de productos disímiles, se debe respetar un tiempo de changeover. Se presentan cuatro instancias del caso: P5.1, minimiza makespan; P5.2, minimiza tardanza total; P5.3, optimiza el número total de órdenes tardías [14].

Los resultados obtenidos mediante modelos CP para los ejemplos introducidos, se presentan en la Tabla 1. Allí se observan los resultados existentes para estos caos en la literatura y los obtenidos mediante los desarrollos CP. Estos últimos se dividen en dos secciones: primer solución y mejor solución hallada en 900 segundos de cómputo. Los resultados han sido obtenidos empleando la configuración por defecto del solver.

Tabla 1. *Resultados computacionales para los casos considerados. Comparación con otras contribuciones.*

Problema	**Resultados reportados en la literatura**		**Modelo CP**			
	Primera solución	**Mejor solución**	**Primera solución**		**Mejor solución (900 s)**	
	Valor de función objetivo	**Valor de función objetivo**	**Valor de función objetivo**	**Tiempo CPU(s)**	**Valor de función objetivo**	**Tiempo CPU(s)[a]**
P1.1	--	30.05	29.40	1.26	28.53	169.12
P2.1	49.16	48.54	59.73	1.24	47.18	799.66
P2.2	58.10	56.06	79.36	1.04	54.07	899.34
P3.1	354.00	293.10	215.00	0.53	199.00[b]	85.48
P3.2	381.00	311.20	213.00	1.27	199.00[b]	42.46
P3.3	370.10	301.20	229.00	0.67	199.00[b]	5.55
P3.4	534.40	509.40	304.10	0.98	260.00	353.27
P3.5	--	--	379.00	1.80	263.00	708.61
P3.6	592.40	550.40	390.50	1.35	264.00	863.65
P4.1	--	31.60	181.10	0.49	31.60	20.15
P5.1	-	-	68.34	232.62	51.91	1737.77
P5.2	-	-	72.72	153.59	9.08	1797.11
P5.3	-	-	7	155.98	6	426.52

a. Tiempo requerido para obtener una solución óptima o instanciar una solución subóptima en 900 segundos de cómputo.
b. Solución óptima

Se observa que con el enfoque CP se mejoran las soluciones reportadas por otros autores (quienes utilizaron modelos matemáticos). Para los casos de P5, se encuentran buenas soluciones en los tiempos establecidos como límite.

4.2. Sistemas de Manufactura Flexible

Los resultados que se presentan en esta sección, han sido obtenidos sobre un conjunto de casos de estudio. Los datos de cada uno de estos ejemplos, han sido generados combinando datos de ejemplos provenientes de diferentes contribuciones, ya que no existía un trabajo de referencia que considerara todos los aspectos que aquí se abordaron. En [12] pueden encontrarse las características de los diferentes problemas resueltos, para el caso de un FMS con 5 máquinas,

donde hay 22 tipos de herramientas, y cada máquina cuenta con un dispositivo herramental con 60 slots.

Los casos han sido resueltos mediante la versión ILOG OPL Studio 3.7, empleando una computadora con procesador Pentium Dual Core 3.0 GHz, con 3 Gb de memoria RAM.

Con se mencionara en la sección 2.2, cuando se manejan aspectos vinculados a tiempos de transporte, debe tenerse en cuenta la magnitud relativa entre los tiempos de transferencia respecto de los tiempos de procesamiento. Esta relación es representada por el ratio tt / pt, donde el numerador es el tiempo promedio de transporte y el denominador el tiempo promedio de procesamiento. Con el fin de evaluar esta relación, diferentes tiempos de transporte fueron propuestas para cada caso, sin variar los tiempos de procesamiento, obteniéndose ratios en el rango [0.04 – 0.13].

Se muestran aquí los resultados para el escenario donde se consideran viajes con carga y sin carga del AGV. Se asume que el tiempo de viaje descargado entre dos equipos corresponde a la mitad del tiempo de viaje con carga entre esos mismos recursos. En la Tabla 2 se presentan los resultados, observándose que la performance del modelo CP se deteriora a medida que el ratio tp se incrementa. La mayor parte de las soluciones encontradas son de buena calidad, aunque no pudo comprobarse el óptimo en el tiempo de resolución preestablecido (1000 s de CPU).

Tabla 2. Resultados de instancias de problemas P1 – P5 considerando viajes con y sin carga de los AGV. Minimización de makespan.

				Solución óptima/mejor en 1000 s			
Problema	**Variables**	**Restric.**	***tp* ratio**	**Makespan**	**Número total de herramien-tas**	**Número total de viajes con carga**	**Tiempo CPU[a]**
P1	803	3869	0.04	370	26	7	213.6
			0.05	374	25	5	221.3
			0.07	381	26	5	37.5
			0.09	401[b]	25	5	29.3
			0.13	433[b]	25	5	126.5
P2	1151	4898	0.04	377	31	8	138.5
			0.05	385	32	8	848.2
			0.07	398[b]	31	9	586.7
			0.09	438[b]	32	8	877.8
			0.13	478[b]	30	9	609.5
P3	2696	10983	0.04	706[b]	53	14	8.8
			0.05	707[b]	54	17	94.7
			0.07	750[b]	54	10	54.2
			0.09	817[b]	58	16	14.2
			0.13	973[b]	56	12	3.5
P4	1586	6704	0.04	590[b]	23	11	552.9
			0.05	626[b]	23	13	2.7
			0.07	611[b]	22	11	456.5
			0.09	756[b]	21	12	21.6
			0.13	768[b]	22	12	40.4
P5	1569	5961	0.04	455[b]	36	14	2.1
			0.05	487[b]	40	13	2.2
			0.07	536[b]	38	11	343.2
			0.09	686[b]	42	12	36.4
			0.13	702[b]	42	8	2.5

a. Tiempo requerido para obtener una solución óptima o instanciar una solución subóptima.
b. Mejor solución subóptima instanciada en 1000 s.

La propuesta CP tiene la capacidad de encontrar una primer solución factible en un tiempo razonablemente corto, lo cual es una característica a destacar. La Tabla 2 también muestra el número total de copias de herramientas que se requiere en cada solución.
En la Figura 8 se presenta el diagrama de Gantt correspondiente a la solución óptima del problema P2 con tp=0.05, mostrando la carga de máquina, la agenda de operaciones, las herramientas empleadas para cada operación, y la agenda de los almacenes locales de las máquinas. Mediante las líneas, también se muestran los viajes del AGV con carga, sin carga y cuando se encuentra en espera.

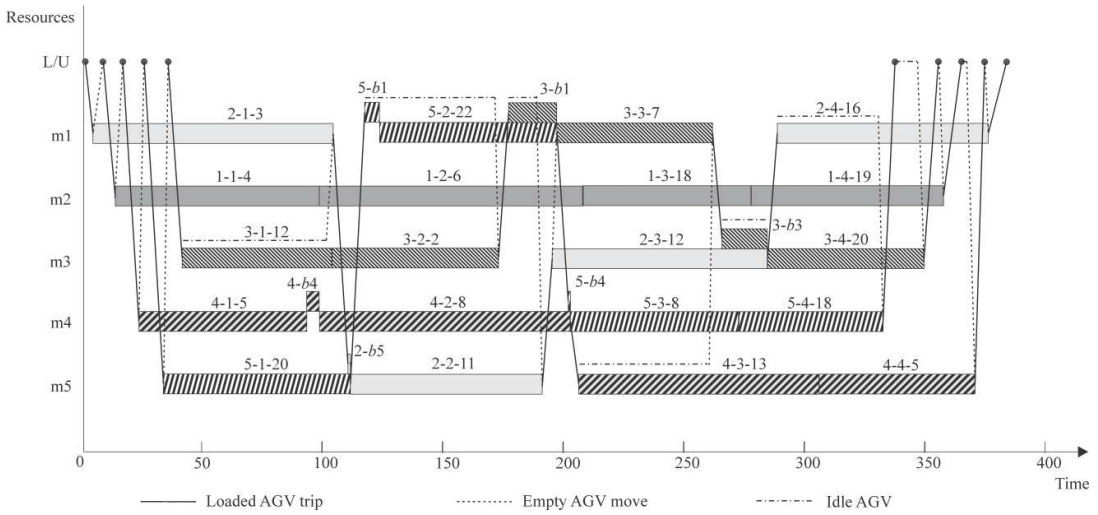

Figura 8. *Diagrama de Gantt. Solución óptima P2, tp=0.05.*

Un análisis más detallado de las soluciones, así como otros escenarios propuestos y la comparación entre los mismos, puede encontrarse en [12].

5. CONCLUSIONES

El principal objeto de este trabajo es dar a conocer los beneficios de abordar el problema de la programación de la producción, en entornos fabriles de procesos y de manufactura discreta, mediante la programación con restricciones. En este sentido, en una primera sección se describió el problema y los aspectos característicos de dos tipos de ambientes productivos: procesos “batch” y sistemas de manufactura flexible. Luego, se introdujeron los principales aspectos y ventajas de la programación con restricciones. Finalmente se mostraron algunos resultados que se han hallado mediante modelos basados en dicha técnica, para los ambientes previamente introducidos.
Los resultados expuestos y un análisis exhaustivo de los mismos, así como los modelos que se emplearon para obtener dichas soluciones, pueden encontrarse en las contribuciones referenciadas.
El estado del arte en el área, así como los trabajos realizados por el autor, permiten llegar a la conclusión que la técnica de programación con restricciones es una de las herramientas más eficaces y eficientes para abordar el problema de “scheduling” en distintos entornos productivos.
Actualmente, se está estudiando el uso de CP en plantas de tipo taller (“job shop” flexibles), las cuales se encuentran en gran número en la región, representadas principalmente por industrias metalmecánicas.

6. REFERENCIAS

[1] Henning, G. H. (2009). Production scheduling in the Process Industries: Current trends, emerging challenges and opportunities. In R. de Brito Alves, C. Oller do Nascimento, & E. Biscaia (Eds.), Computer-Aided Chemical Engineering-27. Elsevier Science Ltd.

[2] Harjunkoski, I., Maravelias C., Bongers, P., Castro, P., Engell, S., Grossmann, I., Hooker, J., Méndez, C., Sand, G., & Wassick, J. (2014). Scope for Industrial Applications of Production Scheduling Models and Solution Methods. Computers and Chemical Engineering, 62, 161-193.

[3] Brailsford, S. C., Potts, C. N., & Smith, B. M. (1999). Constraint satisfaction problems algorithm and applications. European Journal of Operational Research, 119, 557–581.

[4] Baptiste, P., Le Pape, C., & Nuijten, W. (2001). Constraint-based scheduling: applying constraint programming to scheduling problems. New York: Springer+Business Media.

[5] Rodriguez, J. (2007). A constraint programming model for real-time train scheduling at junctions. Transportation Research Part B: Methodological, 41, 231–245.
[6] Topaloglu, S., Salum, L., & Supciller, A. A. (2012). Rule-based modeling and constraint programming based solution of the assembly line balancing problem. Expert Systems with Applications, 39, 3484–3493.
[7] Öztürk, C., Tunali, S., Hnich, B., & Örnek, A. M. (2012). A Constraint Programming Model for Balancing and Scheduling of Flexible Mixed Model Assembly Lines with Parallel Stations. In T. Borangiu, A. Dolgui, I. Dumitrache, F. G. Filip (Eds.), Proc. 14th IFAC symposium of information control problems in manufacturing (pp. 420–425).
[8] Bourdais, S., Galinier, P., & Pesant, G. (2003). HIBISCUS: A constraint programming application to staff scheduling in health care. Lectures Notes in Computer Science, 2833, 153–167.
[9] Castro, P. M., Grossmann, I. E., & Novais, A. Q. (2006). Two new continuous-time models for the scheduling of multistage batch plants with sequence dependent changeovers. Industrial & Engineering Chemistry Research, 45, 6210–6226.
[10] Novas, J. M., & Henning, G. P. (2010). Reactive scheduling framework based on domain knowledge and constraint programming. Computers & Chemical Engineering, 34, 2129–2148.
[11] El Khayat, G. E., Langevin, A., & Riopel, D. (2006). Integrated production and material handling scheduling using mathematical programming and constraint programming. European Journal of Operational Research, 175, 1818–1832.
[12] Novas, J. M., & Henning, G. P. (2014). Integrated scheduling of resource-constrained flexible manufacturing systems using constraint programming. Expert Systems with Applications, 41, 2286 - 2299.
[13] Pinedo, M. (2008). “Scheduling. Theory, Algorithms and Systems”, 3º Ed. Springer Science + Business Media.
[14] Novara, F., Novas, J.M., Henning, P. (2013). A comprehensive CP approach for the scheduling of resource-constrained multiproduct multistage batch plants. Computer-Aided Chemical Engineering, 32- Proceedings of ESCAPE 23, Andrzej Kraslawski, Ilkka Turunen (Eds.), Elsevier, ISBN: 978-0-444-63234-0, 589-594.
[15] Novas, J.M., Henning, P. (2012). A Comprehensive Constraint Programming Approach for the Rolling Horizon-based Scheduling of Automated Wet-etch Stations. Computers and Chemical Engineering Journal, 42, 189-205. Elsevier (Ed), ISSN 0098-1354.
[16] Sirolla, M.D., Novas, J.M., Henning, P. (2014). Programación de la producción a corto plazo y de tareas de mantenimiento preventivo en ambientes Job Shop flexibles. 43° JAIIO-SII 2014, Buenos Aires, Argentina.
[17] IBM ILOG CP Optimizer, http://www-01.ibm.com/software/commerce/optimization/cplex-cp-optimizer/ . Accedido 10/09/2015.
[18] Wallace, M., Novello, S., Schimpf, J. (1997). ECLiPSe: A platform for constraint logic programming. ICL Systems Journal, 12, 159–200.
[19] mOZart. The Mozart Programming System. http://mozart.github.io/. Accedido 10/09/2015.
[20] Zeballos, L., Novas, J.M., Henning, P. (2011). A CP Formulation for Scheduling Multiproduct Multistage Batch Plants. Computers and Chemical Engineering Journal, 35, 2973-2989.
[21] IBM ILOG, CPLEX Optimization Studio. http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud. Accedido 10/09/2015.
[22] Castro, P.M., Harjunkoski, I., Grossmann, I.E., (2009). Optimal short-term scheduling of large-scale multistage batch plants. Industrial and Engineering Chemistry Research 48, 11002-11016.
[23] Kopanos, G.M., Méndez, C.A., Puigjaner, L. (2010). MIP-based decomposition strategies for large-scale scheduling problems in multiproduct multistage batch plants: A benchmark scheduling problem of the pharmaceutical industry. European Journal of Operational Research, 207, 644-655.
[24] Marchetti, P.A., Cerdá, J. (2009). An approximate mathematical framework for resource-constrained multistage batch Scheduling. Chemical Engineering Science, 64, 2733-2748.