ISBN: 978-950-42-0200-4
DOI: https://doi.org/10.33414/ajea.5.752.2020

Desarrollo sistemático de modelos basados en programación con restricciones para problemas de *scheduling* industrial

Systematic development of Constraint Programming formulations for industrial scheduling problems

Presentación: 06-07/10/2020

Doctorando:
Mauricio Daniel Sirolla
Grupo de Investigación y Desarrollo en Informática Avanzada, Instituto de Desarrollo Tecnológico para la Industria Química (INTEC), Consejo Nacional de Investigaciones Científicas y Técnicas - Universidad Nacional del Litoral (CONICET-UNL), Santa Fe, Argentina
msirolla@intec.unl.edu.ar

Director/a:
Gabriela Patricia Henning

Resumen

Se aborda el problema de *scheduling* industrial desde una perspectiva holística. Para ello se propone la construcción de un sistema de *scheduling* como un producto de *software*. Se sintetizan los aportes en relación a la adopción de una metodología de desarrollo de este sistema. El trabajo se ha circunscripto a los módulos para dar soporte a los problemas de *scheduling* predictivo, en entornos de producción de manufactura discreta. Para la resolución de los problemas, se seleccionó el enfoque de programación con restricciones. Se propone la generación automática de los modelos en forma sistemática a partir del análisis de la estructura de los problemas, la identificación de las decisiones y restricciones, y la representación de las mismas de acuerdo a una propuesta de estructuras de modelado comunes. Se emplea un conjunto de casos de la bibliografía para evaluar si el enfoque desarrollado y la forma de construcción de los modelos dan lugar a resultados satisfactorios.

Palabras clave: *scheduling* industrial, programación con restricciones, sistema de *scheduling*.

Abstract

The problem of industrial scheduling is approached from a holistic perspective. This implies the construction of a scheduling system envisioned as a *software* product. In order to do that, a software development methodology is proposed and the most relevant contributions reached are synthesized. This work has been limited to modules and capabilities for supporting predictive scheduling problems in discrete manufacturing production environments. For the resolution of problems, constraint programming has been selected. An approach for automatic model generation in a systematic way is proposed, based on the analysis of the structure of the problems, the identification of decisions and restrictions, and their representation according to a proposed set of common modelling structures. A set of examples from the bibliography is used to assess whether the developed approach and the way models are generated lead to satisfactory results.

Keywords: *Industrial scheduling*, *constraint programming, scheduling system*.

Introducción

Los problemas de planificación de producción a corto plazo (*scheduling*) son inherentemente complejos, tanto por su naturaleza combinatoria (problemas de tipo NP-*Hard*) (Brucker y Knust, 2012), como por la gran variedad de características de los distintos

ambientes industriales en los que se presenta. La actividad de *scheduling* forma parte del proceso de planificación, programación y control de la producción que se lleva a cabo en toda industria, el cual brinda información crítica para la realización de las actividades en la empresa misma y de aquellas que conforman sus cadenas de suministros.

Un problema de *scheduling* industrial consiste como mínimo en las decisiones de asignación, secuenciamiento y *timing* asociadas a las tareas de manufactura requeridas para llevar a cabo la fabricación de las partes o lotes de productos y a otras que sean necesarias (movimiento de materiales, etc.) durante un horizonte de planificación. Las actividades deben ser agendadas a un conjunto de recursos limitados (máquinas, servicios auxiliares, etc.). Como resultado del *scheduling* se obtiene un plan o agenda que detalla las tareas que lleva a cabo cada recurso productivo, su orden de ejecución, así como sus momentos de inicio y fin previstos. Dicha agenda debe satisfacer las restricciones que imponga naturalmente el ambiente de manufactura, así como aquellas asociadas a los trabajos que se realizan. La calidad de la agenda resultante es otro aspecto importante. Se espera que el plan generado sea "eficiente" respecto de una o más medidas de desempeño preseleccionadas, que contemplen la minimización de tiempos o costos.

La resolución de los problemas de *scheduling* en el ámbito industrial ya ha sido contemplado de alguna forma en las funcionalidades de paquetes de *software* comerciales disponibles, como los sistemas APS (*Advanced Planning Systems*). Sin embargo, su orientación a entornos muy específicos, en general su escasa interoperabilidad con otros sistemas informáticos de las organizaciones (MES/ERP), y su elevado costo de implementación, ha llevado a la proliferación del uso de soluciones que podrían catalogarse como primitivas (planillas de cálculo, soluciones manuales). Éstas y otras dificultades de implantación de estas soluciones ya han sido discutidas por algunos autores como Harjunkoski (2016).

La comunidad académica se ha interesado en la resolución de este tipo de problemas recurriendo a enfoques de muy variada naturaleza. Desde métodos algorítmicos aproximados, basados en la exploración estocástica de soluciones, hasta los exactos (como los métodos de enumeración). Sin embargo, las propuestas son de difícil implementación en la práctica. En la mayoría de los casos se ignora completamente la existencia de sistemas informáticos que las deban realizar, así como su interoperabilidad con otros sistemas existentes en la organización que le provea o a los cuales alimente de información. Suelen enfocarse en cuestiones de performance computacional, y usualmente simplifican mucho el problema dando lugar a soluciones “inviables”. Otra dificultad importante es que generar o modificar un método de resolución requiere un acabado conocimiento de la técnica que se esté empleando y de la forma de representación que la misma maneje, destrezas y tiempo que suelen ser difíciles de encontrar en los planificadores para que, en el día a día, puedan hacer uso de las mismas.

Debido a las limitaciones identificadas, es necesario abordar estos problemas con una visión holística. La atención ya no debe ser puesta meramente en las cuestiones algorítmicas, sino en cómo llevar a cabo la implementación de los métodos de la academia en una aplicación de *software* que le confieran usabilidad en entornos industriales. El enfoque que se ha adoptado es el de encarar las etapas metodológicas de construcción de un sistema de *scheduling* desde la perspectiva de la ingeniería de *software*. Para la resolución de problemas se decidió emplear el enfoque de programación con restricciones, entre otras razones por su elevada capacidad de representación de los problemas. Para ello se propusieron un conjunto de estructuras de modelado comunes y se explicitaron los razonamientos, para reducir la labor “artesanal” con la que se generan los modelos, lo que facilitaría su incorporación en un prototipo de herramienta de *scheduling*. En este resumen, se abordan los avances para el desarrollo de algunos de los módulos asociados a dicho prototipo de sistema en un ambiente de manufactura discreta. Se acota el análisis a los problemas de tipo predictivo.

Desarrollo

El desarrollo del prototipo de sistema tiene como punto de partida un conjunto relevante de las funcionalidades, el cual ha sido generado a partir del relevamiento de la bibliografía del área (Framinan y colab., 2014) y del estudio de algunas herramientas de índole comercial, que han permitido identificar requerimientos funcionales y de calidad que debería satisfacer un sistema de *scheduling*. La Figura 1 presenta un extracto de las funcionalidades comunes que han sido identificados/propuestos para estos sistemas. Además de las funcionalidades, se han considerado un conjunto de requerimientos de calidad, relacionados a Adecuación Funcional, Eficiencia en el desempeño, Compatibilidad, Usabilidad, Confiabilidad, Seguridad, Mantenibilidad, y Portabilidad.

1. Gestionar ambiente productivo.
2. Especificar problema predictivo.
3. Gestionar análisis de capacidad de recursos productivos.
4. Gestionar indicador de calidad de agenda.
5. Resolver problema predictivo.
6. Visualizar plan de operaciones.
7. Comparar agendas

(a)

2. Especificar problema predictivo.

2.1. Especificar horizonte de planificación.
2.2. Seleccionar recurso productivo a utilizar.
2.3. Recuperar/Importar perfil de disponibilidad de recurso.
2.4. Modificar perfil de disponibilidad de recurso.
2.5. Recuperar perfil de costos de uso de recurso productivo.
2.6. Modificar perfil de costos de uso de recurso productivo.
2.7. Recuperar/Importar orden de producción a cumplir.
2.8. Seleccionar orden a satisfacer.
2.9. Especificar restricción temporal entre operaciones (inicio conjunto, fin concurrente) no contenidas en las recetas.
2.10. Especificar tipo de *preemption.*
…

(b)

Figura 1. (a) Grupos de requerimientos funcionalidades identificados. (b) Extracto del desglose de uno de los grupos.

Modelo de dominio

Otra cuestión vinculada al desarrollo de la herramienta es contar con la representación explícita de conocimiento del dominio. La necesidad de disponer de modelos que capturen, de manera explícita y sin ambigüedad, los elementos y relaciones presentes en un dominio ha sido identificada por la comunidad académica, aunque fueran omitidos la mayor parte de las contribuciones. La representación del dominio se realizó en base a un enfoque de orientación a objetos. Se desarrollaron Diagramas de Clases, de acuerdo a la sintaxis estándar que propone el lenguaje UML y consistente con los lineamientos de los estándares ampliamente aceptados en el área (ANSI/ISA-88, 2001; ANSI/ISA-95, 2010).

El conocimiento de dominio se ha modelado a través de diferentes perspectivas, las cuales organizan los conceptos de la representación en tres grandes grupos: (i) vista del Ambiente de manufactura, que describe: los tipos de partes que pueden fabricarse y los diferentes tipos de actividades que demanda su manufactura; los recursos que forman parte del ambiente y que intervienen en las actividades productivas; (ii) vista de la disponibilidad de recursos, que especifica los intervalos de tiempo y la capacidad de cada recurso productivo concreto durante el horizonte de planificación. (iii) vista de la actividad de *scheduling*, que consiste en los conceptos asociados a las tareas específicas con sus atributos temporales, y a la información relacionada con los programas de operaciones de cada recurso. La Figura 2 presenta el modelo integrado del dominio para un entorno de producción simple por cuestiones de claridad (ambiente *Job Shop* Flexible). Se supone que, como actividades se llevan a cabo únicamente operaciones de manufactura, y entre los recursos del ambiente se dispone de máquinas y herramientas.

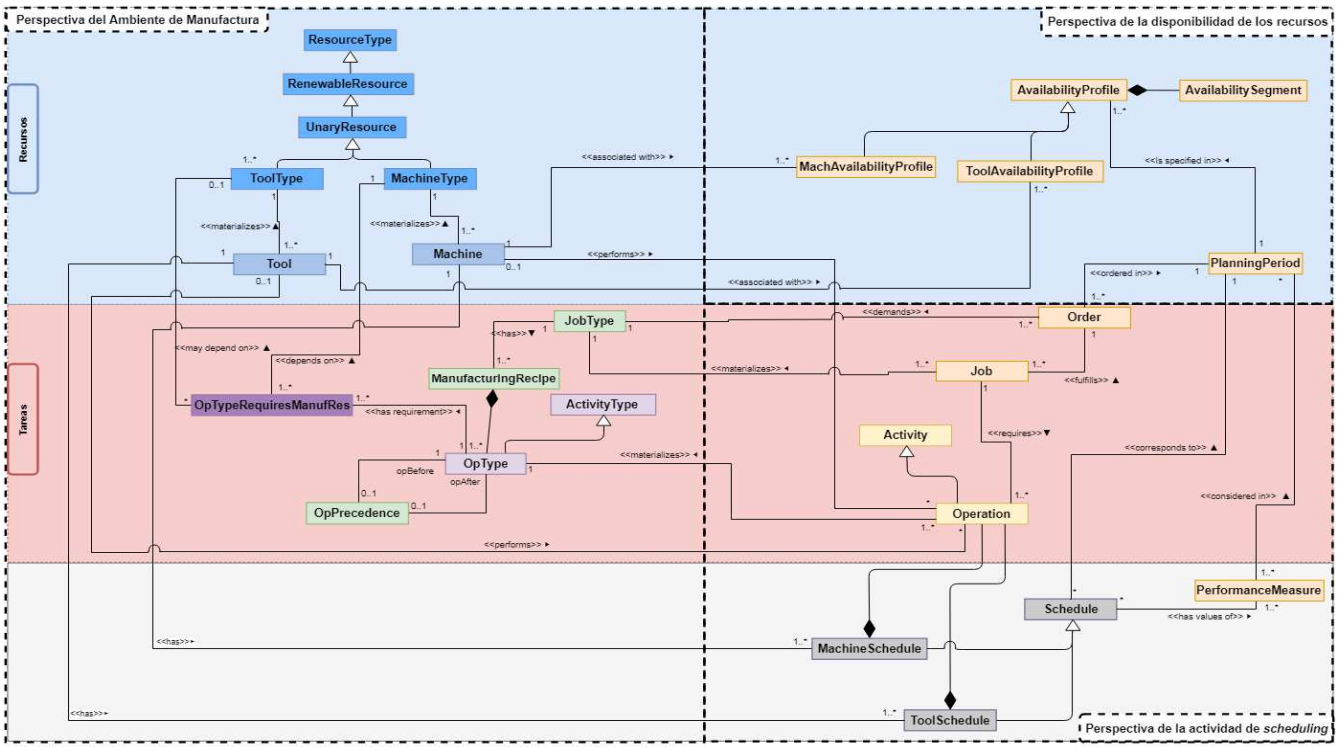


Figura 2. Modelo del dominio en ambientes de manufactura discreta que integra tres perspectivas propuestas.

Estructura de problemas y su modelado

Especificar un problema de *scheduling* puede ser interpretado como la combinación de una estructura de problema (que representa las decisiones que se toman, las restricciones que deben cumplirse, etc.) que debe ser definida y el conjunto de datos específicos, los cuales deben ser consistentes con la misma. Los datos de los problemas se organizan bajo el esquema estático que propone la conceptualización del dominio presentada en forma previa. La definición de la estructura, por su parte, requiere la intervención del planificador, que debe establecer cuáles son las tareas que se contemplan, los recursos con los que se cuenta para llevarlas a cabo, las medidas de desempeño que evalúan la calidad de las agendas generadas, etc.

La estructura de un problema puede representarse mediante el enfoque de programación con restricciones (Rossi y colab. 2006). Cabe destacar que podrían generarse "n" modelos de la misma, y que en la bibliografía se vislumbra que el proceso de su formulación generalmente no parte de una lógica explícita, pues está basado principalmente en la experiencia y en el conocimiento implícito de quien lleva a cabo esa tarea. Para orientar esas actividades, se propone un conjunto de subestructuras generales que sirven de lineamientos para la propuesta de variables y expresiones, basadas en la similitud de los conceptos que representan y los que relacionan entre sí. Así, las decisiones que tienen una naturaleza similar se representan mediante ciertos tipos de variables, y las restricciones que capturan limitaciones parecidas, mediante expresiones que tienen una misma "forma". Específicamente, se ha recurrido al lenguaje OPL (IBM, 2013) para representar las estructuras de problemas, por las facilidades de modelado de cuestiones inherentes al *scheduling*.

La Figura 3 presenta una porción de las subestructuras generales y de aquellas generadas para el siguiente caso particular. El problema consiste en un conjunto de máquinas multipropósito y de herramientas suficientes que le posibiliten realizar distintos tipos de operaciones a todas ellas, pero solo efectúan una a la vez. En dichos equipos se agendan las tareas de manufactura requeridas por un conjunto de piezas. A su vez, se conoce que se deberán realizar actividades de mantenimiento sobre las máquinas aunque no está preespecificado el momento exacto de su realización, y por ello deben coordinarse. Las actividades cuya realización es "obligatoria" durante el horizonte de planificación (operaciones de manufactura y de mantenimiento en este caso), se representan mediante variables especiales de OPL denominadas variables de intervalo (que la representan como un bloque desde que inicia hasta que finaliza). La ejecución de una tarea en un recurso (de una operación en una máquina, en una herramienta) se representa mediante una variable de intervalo de tipo opcional, pues es una decisión acerca de qué máquina y herramienta emplear. El uso de cualquier recurso productivo se captura mediante una función acumulativa en OPL, que permite la construcción del perfil de uso de ese recurso, con forma escalonada, durante el horizonte de planificación, y posibilita la coordinación de todas las tareas que lo requieren. En este caso en particular, las tareas de mantenimiento y de manufactura que se llevan a cabo en la misma máquina deben coordinarse, puesto que cuando una se lleva a cabo, no puede realizarse la otra. El extracto de la figura requiere completarse con otras expresiones que permitan capturar el problema completo.

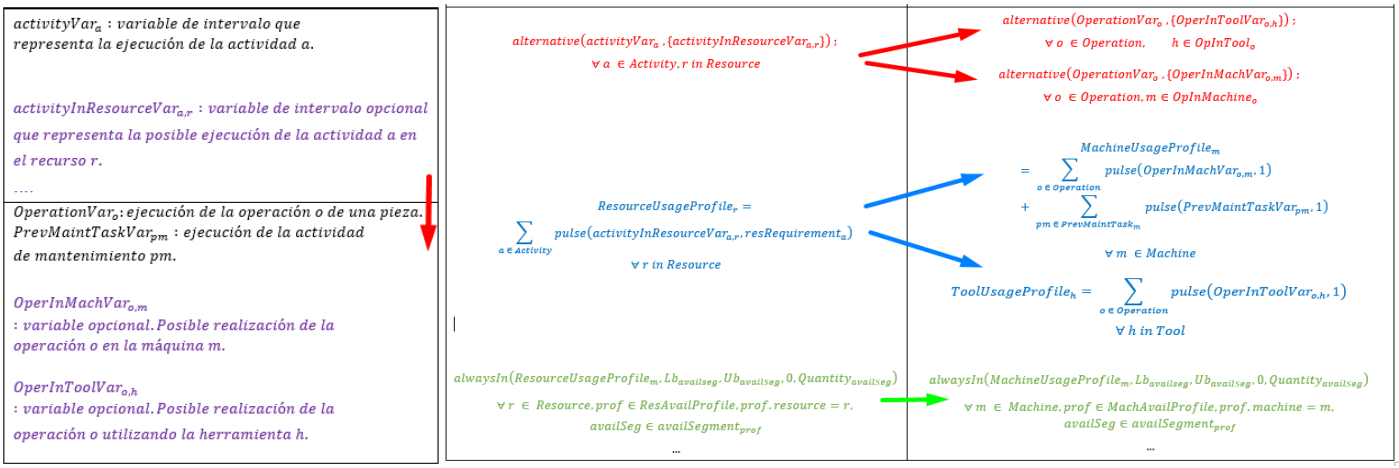


Figura 3. Extracto de declaraciones de variables y expresiones de modelado generales y las particulares.

Resultados

La validación de la propuesta se realizó a partir de variadas estructuras de problema, su representación en modelos de programación con restricciones (*Constraint Programming*, CP), concluyéndose que los mismos sirven para resolver determinados conjuntos de ejemplos. Por cuestiones de espacio no se describen en detalle los modelos utilizados, y se presentan de manera resumida los resultados alcanzados. Se han generado las agendas de un grupo de problemas simples tomados de la bibliografía, que

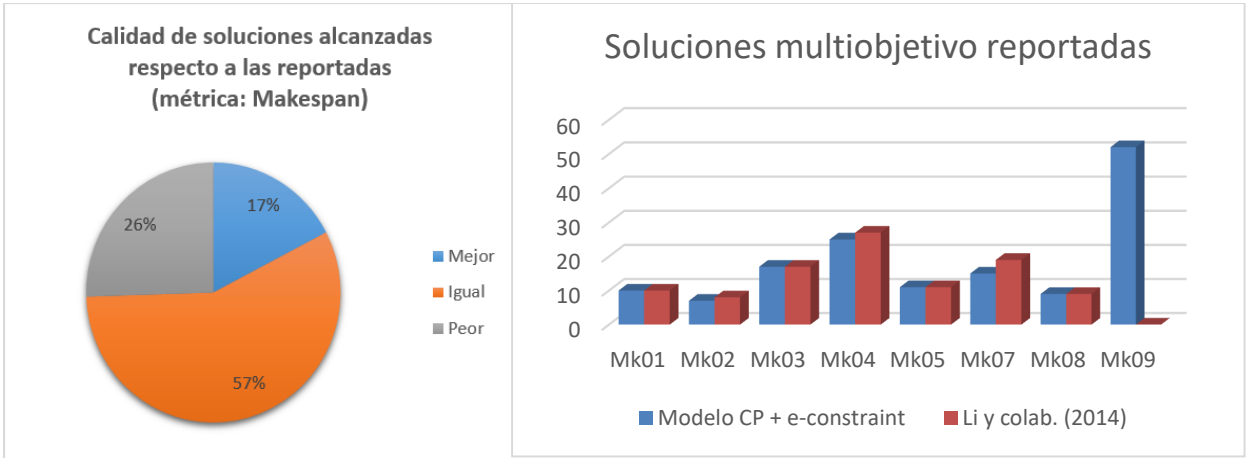
corresponden a ambientes de manufactura discreta *Job Shop* Flexible, cuyas máquinas son de tipo multipropósito. Se agendan en dichos recursos las actividades demandadas por un conjunto acotado de piezas, y la duración de dichas tareas depende del equipo que la realiza. La calidad de las agendas se evaluó considerando una sola métrica (subconjunto de casos I) y más de una en simultáneo (subconjunto de casos II).

Se presenta también un ejemplo de un problema con menores simplificaciones (III). Se trata de un ambiente de manufactura discreta con los siguientes recursos productivos: máquinas multipropósito, herramientas con cierta vida útil asignadas a cada equipo al comienzo del horizonte, y dos dispositivos de movimiento de partes, que llevan a cabo el traslado de piezas (no se considera el movimiento del dispositivo si no tiene partes). Las máquinas tienen espacios de almacenamiento en su entrada y salida, con espacio suficiente. La calidad de las soluciones se evalúa empleando una única medida de desempeño.

Las agendas fueron generadas empleando el ambiente *IBM CPLEX Optimization Studio v12.5.1* (ILOG-IBM, 2013), con una PC Dell Precision T1650, Intel® Xeon® CPU E3-1220, 3.10 GHz y 8 GB of RAM para el primer grupo de ejemplos y una PC Intel i5-8250 1.6 GHz y 8 Gb de RAM para el segundo caso.

I. Problemas simplificados de tipo Mono-objetivo: Se seleccionó como medida de desempeño el *makespan*. Se seleccionaron 98 ejemplos de ambientes FJS simples, comúnmente presentes en la bibliografía, y se compararon los resultados con aquellos alcanzados por Karimi y colab., (2012), Singh and Mahapatra, (2016), entre otros trabajos. La Figura 4.a resume la calidad de las soluciones obtenidas, respecto a las soluciones alcanzadas en la bibliografía. Los 98 ejemplos se resolvieron, alcanzando soluciones óptimas en 41 de las instancias. Primeras soluciones factibles fueron obtenidas en pocos segundos de ejecución. En todos los casos se ejecutó el *solver* por 4500 s de CPU.

II. Problemas simplificados de tipo Multi-objetivo: Se seleccionaron las medidas de desempeño *makespan*, carga máxima de trabajo y carga total de trabajo de los equipos durante el horizonte. Se eligieron los casos de estudio que se presentaron en el trabajo de Brandimarte (1993). Los resultados se compararon con los reportados por Li y colab. (2014). En este caso, el modelo CP requirió como complemento la definición de un *script* en lenguaje OPL el cual representó la aplicación del método de *e-constraint* para la exploración de soluciones en la frontera de Pareto (Mavrotas, 2009). Por cuestiones de espacio, la Figura 4.b presenta el comportamiento del modelo, mostrando las bondades del enfoque para encarar problemas multiobjetivo de tamaño pequeño y mediano al hallar las mismas soluciones que Li y colab. (2014) en los casos Mk01, Mk03, Mk05 y Mk08, soluciones no dominadas que dominan las de Li y colab. (2014) en los ejemplos Mk02, Mk04, Mk07, y finalmente la posibilidad de explorar la frontera de Pareto para el caso Mk09.



(a) (b)

Figura 4. (a). Calidad de las soluciones alcanzadas en los ejemplos I (b) Resultados de la exploración de la frontera de Pareto en ejemplos de II.

III. Modificación del caso de estudio Mk09 de Brandimarte (1993). Se definió que la duración de las actividades de transporte sea un valor entre 1 y 10 unidades de tiempo. La Figura 5 presenta el diagrama de Gantt con la solución subóptima alcanzada en 4500 s de CPU, cuya calidad en términos de *makespan* es 459 unidades de tiempo. Se presenta la utilización de los dispositivos de transporte sólo para las tareas asociadas al movimiento del *job* 1 por cuestiones de claridad.

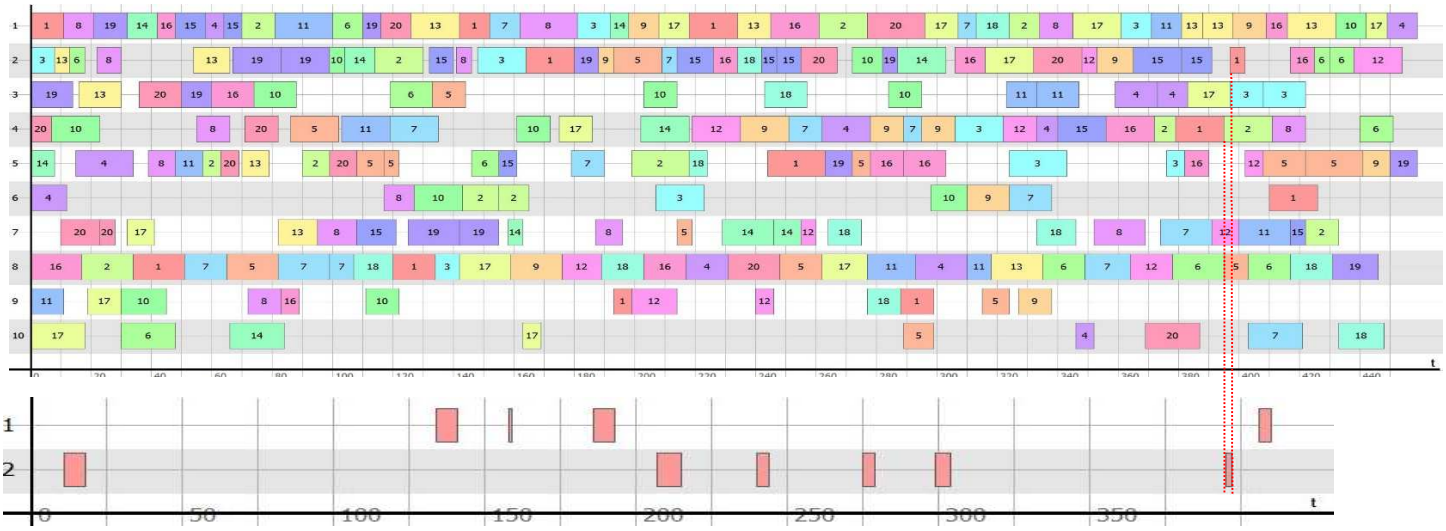


Figura 5. Diagrama de Gantt que representa la solución del problema con menos simplificaciones. Actividades de traslado de la pieza 1 destacadas.

Conclusiones

Se han abordado las actividades conducentes al desarrollo de algunos módulos de un sistema de *scheduling*, que incluya la participación del planificador, y la posibilidad de emplear uno de los enfoques provenientes de la academia para la solución de problemas de *scheduling*. Entre los aportes del trabajo se tienen la Especificación de Requerimientos del sistema y la conceptualización del dominio de trabajo. La propuesta de modelos CP, que modelen la estructura de un problema orientado por las similitudes de las decisiones y expresiones presentes se muestra como un enfoque que facilita su representación y modificación de formulaciones. Como trabajos futuros se plantea evaluar la escalabilidad en la capacidad de resolución de problemas, la definición de otras estructuras e incorporar metodológicamente los módulos tendientes a dar soporte al problema de *scheduling* online.

Referencias

ANSI/ISA-88 (2010). ANSI/ISA-88.01-2010, Batch Control Part 1: Models and terminology.

ANSI/ISA-95 (2010). ANSI/ISA-95.00.01-2010 (IEC 62264-1 Mod), Enterprise-Control System Integration, Part 1: Models and Terminology.

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. Annals of Operations Research, 41, 157-183. https://doi.org/10.1007/BF02023073

Brucker, P., & Knust, S. (2012). Complex Scheduling. Springer-Verlag Berlin Heidelberg.

Framinan, J. M., Leisten, R., & Ruiz García, R. (2014). Manufacturing Scheduling Systems. Manufacturing Scheduling Systems: An Integrated View on Models, Methods and Tools. Londres: Springer. https://doi.org/10.1007/978-1-4471-6272-8

Harjunkoski, I. (2016). Deploying scheduling solutions in an industrial environment. Computers and Chemical Engineering, 91, 127-135. https://doi.org/10.1016/j.compchemeng.2016.03.029

ILOG-IBM (2013). https://www.ibm.com/products/ilog-cplex-optimization-studio. Version 12.5.1. Último acceso: 20/08/2020.

Li, J.-Q., Pan, Q.-K., & Fatih Tasgetiren M. (2014). A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities. Applied Mathematical Modelling, 38, 1111–1132. https://doi.org/10.1016/j.apm.2013.07.038

Mavrotas, G. (2009). Effective implementation of the e-constraint method in Multi-Objective Mathematical Programming problems. Applied Mathematics and Computation, 213, 455-465. https://doi.org/10.1016/j.amc.2009.03.037

Rossi, F., van Beek, P., & Walsh, T. (2006). Handbook of constraint programming. New York: Elsevier Science.

Karimi, H., Rahmati, S.H.A., & Zandieh, M. (2012). An efficient knowledge-based algorithm for the flexible job shop scheduling problem. Knowledge-Based Systems, 36, 236–244. http://dx.doi.org/10.1016/j.knosys.2012.04.001

Singh, M. R., & Mahapatra S. S. (2016). A Quantum Behaved Particle Swarm Optimization for Flexible Job Shop Scheduling. Computers and Industrial Engineering, 93, 36-44. https://doi.org/10.1016/j.cie.2015.12.004