3rd International Conference on Industrial Engineering and Industrial Management
XIII Congreso de Ingeniería de Organización
Barcelona-Terrassa, September 2nd-4th 2009

Modelado y resolución de un problema de programación de tareas con recursos simultáneos mediante mecanismos basados en mercado

Juan José Laviós Villahoz[1], Ricardo del Olmo Martínez[2], Alberto Araúzo Araúzo[3]

[1] Dpto. de Ingeniería Civil. Escuela Politécnica Superior. Universidad de Burgos. Av. Cantabria s/n, 09005. Burgos. jjlavios@ubu.es
[2] Dpto. de Ingeniería Civil. Escuela Politécnica Superior. Universidad de Burgos. C/ Villadiego s/n, 09001. Burgos. rdelolmo@ubu.es
[3] INSISOC- Dpto. de Organización de Empresa. ETS Ingenieros Industriales. Universidad de Valladolid. Paseo del Cauce s/n, 47011. Valladolid. arauzo@inisoc.org

Keywords: sistemas multiagente, relajación lagrangiana, scheduling

1. Introducción

El problema de scheduling o de programación de tareas consiste en asignar recursos a tareas en un determinado horizonte de tiempo, teniendo uno o varios objetivos a optimizar. El concepto de programación de tareas puede ser aplicado a distintos ámbitos: en gestión de producción se trataría de programar las operaciones de fabricación asignándolas a unas determinadas máquinas, en gestión de proyectos se programan las fases de realización del proyecto asignando cada una a los distintos equipos y recursos, en informática la ejecución de programas se necesita ser asignada a la memoria y el procesador, o en logística donde el transporte de productos debe realizarse con distintos medios de transporte y personal que deben ser programados (Pinedo 2008).

Existen multitud de problemas en los que se requiere el uso simultáneo de más de un recurso en ámbitos tan diversos como la reparación de automóviles, la programación de operaciones quirúrgicas o la planificación de tareas de equipos de trabajo (Dobson & Karmarkar 1989), aunque nosotros centraremos el problema en el ámbito industrial. Es habitual que para realizar la programación de tareas en el taller se realice la asignación de una tarea a un determinado recurso o máquina, sin embargo puede ocurrir que sea necesario programar también otros recursos complementarios como el uso de herramienta específica, determinados operarios o cualquier otro elemento.

En la actualidad la toma de decisiones de forma distribuida está siendo considerada como una alternativa a los sistemas de programación puramente centralizada puesto que permiten incorporar de modo más sencillo y natural las restricciones locales de cada recurso, sus preferencias y objetivos en el proceso de decisión (Kutanoglu & Wu 1999).

Los sistemas multiagente han demostrado ser una herramienta adecuada para el modelado de sistemas complejos y constituyen un marco apropiado para tratar de definir los procesos de toma de decisiones distribuida, puesto que esta es una de las características intrínseca a estos sistemas. Los agentes son usados para encapsular entidades físicas, lógicas o funcionales del sistema de producción. Estos sistemas se basan en la autonomía de cada agente y en la interacción y la negociación entre los agentes participantes (Shen 2002). La aplicación este paradigma en los problemas de planificación y programación de operaciones ha tenido gran importancia en los últimos años. Una revisión de los principales desarrollos de sistemas

multiagente en producción que se han llevado a cabo en los últimos años puede encontrarse en (Shen et al. 2006) y en (Lee & Kim 2008).

2. Problema

El problema planteado consiste en realizar la programación de las actividades de un conjunto de recursos (H) que deben realizar una serie de tareas (I), independientes entre sí, donde cada tarea debe de ser finalizada antes de una determinada fecha (D_i). Para poder realizar cada tarea es necesaria la participación simultánea en la tarea de varios recursos que dispongan las habilidades que la tarea requiera. El número de tareas que pueden ser programadas de forma simultánea en un mismo recurso está limitada por su capacidad (M_{hk}). En este apartado se presenta la formulación del problema utilizando la notación que se muestra en la tabla 1.

Tabla 1. Notación del modelo

i	índice de cada tarea. i = 1,...,N donde N es el número total de tareas
I	conjunto de tareas, I = { i : 1,..., N }
k	periodo de tiempo, k = 1,..., K donde K es el horizonte de programación
h	Índice de cada máquina, h = 1,..., M donde M es el número total de máquinas
H	conjunto de máquinas, H = { h : 1,..., H }
d_i	tiempo proceso de la tarea i
D_i	fecha de entrega comprometida para la tarea i
b_i	fecha inicio programada de la tarea i
c_i	fecha finalización programada de la tarea i
T_i	Retraso de la tarea i
γ_{ih}	$\begin{cases} 1 & \textit{si es necesario utilizar el recurso } h \\ 0 & \textit{si no se necesita el recurso } h \end{cases}$
γ_i	vector que define las necesidades de utilización de recursos que tiene la operación i
δ_{ihk}	$\begin{cases} 1 & \textit{si la tarea } i \textit{ se programa en el recurso } h \textit{ en el instante } k \\ 0 & \textit{en caso contrario} \end{cases}$
M_{hk}	Capacidad de la máquina h en el instante k

El problema está sometido a las restricciones de capacidad de los recursos, puesto que la cantidad de cada uno está limitada, restricciones de simultaneidad de recursos, ya que deben asignarse los recursos necesarios para realizar cada tarea en los mismos espacios de tiempo, y las restricciones de relación entre variables, que terminan de definir el problema.

El problema lo definimos como:

$$\min \sum w_i T_i^2 \qquad (1)$$

s.a.
$$\sum_{i=1}^{I} \delta_{ihk} \leq M_{hk} \qquad \begin{matrix} k = 1,2,..,K \\ h = 1,2,..,H \end{matrix} \quad (2)$$

$$\delta_{ihk} = \begin{cases} 1 & si \quad k \in [b_i, c_i \wedge \gamma_{ih} = 1] \\ 0 & si \quad k \notin [b_i, c_i] \vee \gamma_{ih} = 0 \end{cases} \qquad \begin{matrix} k = 1,2,..,K \\ i = 1,2,...,I \\ h = 1,2,..,H \end{matrix} \quad (3)$$

$$c_{ij} = b_i + d_i - 1 \quad h = h_i \quad i = 1,2,...,I \quad j = 1,2,..,N_i \qquad (4)$$

Función objetivo

Los objetivos de la programación de la producción buscan medir la calidad de una solución que responda al problema estudiado. Bajo el objetivo de los criterios se definen y utilizan para aislar las soluciones óptimas o satisfactorias. Los criterios generales más frecuentemente utilizados son la minimización de la duración total de fabricación, los tiempos de inactividad de las máquinas, el coste de la programación de la producción (p.ej. el número de productos en curso puede llevar a costes de almacenamiento suplementarios) o incluso respecto las fechas de finalización de las tareas a conseguir. En ciertos casos particulares puede ser necesario criterios más específicos como, por ejemplo, la minimización del efecto de cuello de botella producido por un recurso. La función a minimizar en este caso es la suma ponderada de los cuadrados del retraso de cada tarea, como muestra la expresión (1).

El retraso (T) se define como la diferencia entre la fecha programada de finalización de la tarea y la fecha de finalización comprometida, en caso de que la primera sea posterior a la segunda. Es decir:

$$T_i = max(0, c_i - D_i) \qquad (5)$$

Restricciones de capacidad

Las restricciones de capacidad limitan el número de operaciones (i) que pueden ser programados en una máquina (h) en un instante determinado (k). En el caso más general esta capacidad será un número real (M_{hk}), aunque en muchas ocasiones se supone que toma el valor 1, es decir, que cada máquina no puede realizar más de una operación simultáneamente. En ese caso, la expresión (2) quedaría:

$$\sum_{i=1}^{I} \delta_{ihk} \leq 1 \qquad (6)$$

Restricciones de simultaneidad de recursos

Sea una operación i que necesita un conjunto de recursos de forma simultánea para ser realizada. Se define el vector γ_i (γ_{i1}, γ_{i2},...,γ_{iH}) como el vector que define las necesidades de utilización de recursos que tiene la operación i, de modo que para cada operación (i) se tiene que cumplir que en el intervalo de tiempo en que está programada, los recursos necesarios para realizarla deben estar asignados a ella. Se definen las variables binarias γ_{ih} que toman el valor 1 en caso de que la operación i deba utilizar la máquina h , y 0 en caso contrario

Restricciones de definición de las variables

Las relaciones entre los instantes de comienzo b_i y finalización c_i de una operación i, si la operación se realiza en una máquina en la que se tarda en realizar un tiempo d_i, se muestra en la expresión (4)

3. Mecanismo de asignación basado en subastas

Los mecanismos de asignación basados en mecanismos de mercado es una de las ramas de la programación de tareas distribuida más activa en los últimos años. Se trata de establecer un programa de producción basándose en los precios que aparecen a partir de las pujas que se generan individualmente para conseguir un objetivo individual por parte de una tarea o un recurso (Kutanoglu & Wu 1999). La idea que subyace es que la asignación se realice por la interacción entre los agentes, recursos y tareas, que participen en un mercado creado ad-hoc y que establezcan sus precios a través de un proceso iterativo de búsqueda del equilibrio. Esto permite que los cálculos complejos a realizar puedan ser distribuidos entre los distintos agentes participantes y realizados en paralelo, por lo que mejora la rapidez en su resolución. La carga de información intercambiada es baja, ya que la información intercambiada son las pujas y los precios (Wellman et al. 2001).

Las subastas combinatorias son uno de los mecanismos de mercado propuesto por distintos autores como medio de negociación y coordinación en los sistemas multiagente. Se caracterizan porque los participantes realizan pujas no por un sólo producto, sino por combinaciones de productos para las que tienen distintas valoraciones. Las subastas combinatorias son adecuadas cuando el valor de cada ítem depende de cuales sean los productos adquiridos en la misma subasta. Por ejemplo, imaginemos que un viajero quiere realizar una estancia en una ciudad durante un determinado número de días. Para. Sólo puede llegar a esa ciudad en avión. Para poder realizar el viaje deberá adquirir un billete de ida, la estancia de hotel durante un número de días y un billete de vuelta. La valoración de cada noche de hotel y de los billetes de avión depende de las demás noches o billetes adquiridos. Para el viajero será importante disponer de hotel el día de llegada y tener reservado ese hotel hasta la fecha de regreso. Si, por ejemplo, no es posible reservar el hotel en una fecha determinada entre el día de llegada y regreso, el resto de días de hotel e incluso los billetes de avión dejarán de tener valor para el viajero.

Hay distintos tipos de subastas combinatorias. Es posible realizar la subasta combinatoria en una sola ronda, donde los participantes dan sus valoraciones sobre las combinaciones de productos una sola vez y el subastador realiza la asignación de productos de forma que se maximice la función objetivo que se haya marcado, o bien se puede realizar la subasta combinatoria en varias rondas. En este caso se denominan iterativas. Existen dos tipos básicos de subastas combinatorias iterativas: las de fijación de cantidades y las de fijación de precios. En las de fijación de cantidades las pujas que envían los participantes en la subasta consiste en precios sobre varios artículos. El subastador realiza una asignación provisional de los artículos que depende de los precios enviados. Los participantes van ajustando los precios en cada iteración. En el segundo tipo básico, las de fijación de precios, el subastador asigna un precio a cada uno de los artículos a subastar. La puja de cada participante consiste, en este caso, en el conjunto de elementos que pretende adquirir con los precios dados. El subastador va adaptando los precios en cada iteración en base a la demanda de cada artículo.

En (de Vries & Vohra 2003) se señalan las ventajas que las subastas combinatorias iterativas tienen sobre las de una sola ronda, como son que no es necesario que cada participante realice pujas sobre todo el conjunto de combinaciones posible, que cada participante termina revelando información privada relevante para el resto de participantes, sus preferencias, y que se adaptan bien a los entornos dinámicos, este es el caso de los entornos productivos, donde tanto los participantes como los elementos a subastar van entrando y saliendo en la subasta en momentos diferentes.

4. Modelado del problema como subasta combinatoria

Es posible modelar el problema de programación de tareas como una subasta donde cada una de las unidades de tiempo en que dividimos el horizonte de programación de cada recurso es el ítem que se adquiere en la subasta. Cada tarea participa en las subastas pujando por el conjunto de unidades de tiempo de las máquinas o recursos que le permiten realizar la operación (Dewan & Joshi 2002). El mecanismo seguirá los principios de los sistemas distribuidos de forma que la información relevante de cada uno de los agentes participantes, como el tiempo de proceso de cada operación, recursos necesarios, fecha de entrega comprometida, penalización por unidad de tiempo de retraso, no será directamente accesible al resto de agentes (Duffie 1990). Por ejemplo, ninguno de los agentes tarea conocerá qué otros agentes tarea están en el sistema, como tampoco ninguno de los recursos conocerá qué tareas necesitan de sus servicios hasta que no tenga lugar la subasta. Los precios de cada unidad de tiempo de uso de cada recurso resultantes de la subasta serán los indicadores que mostrarán la demanda relativa de cada unidad de tiempo.

El modelo que se plantea consiste en representar el problema de programación de tareas como una subasta combinatoria iterativa de fijación de precios, donde las tareas pujan por conseguir los recursos necesarios para ser llevadas a cabo. El uso de un recurso en un intervalo de tiempo determinado estará valorado por la tarea en la media que le permita que sea completada y que se realice antes, o lo más cerca posible, de la fecha comprometida de finalización de la tarea.

5. Relajación lagrangiana del problema

Existen estrechos vínculos entre las subastas combinatorias y las técnicas de relajación lagrangiana. Podemos utilizar estas técnicas para realizar la actualización de los precios en cada iteración. Este mecanismo ha sido utilizado para resolver distintos problemas de programación de la producción (Luh & Hoitomt 1993), como el problema de job shop (Kaskavelis & Caramanis 1998), (Tao Sun et al. 2006) o el de flow shop (Tang et al. 2006). En esta sección se plantea la relajación de una de las restricciones del problema original que permita representar la resolución del problema como una subasta combinatoria iterativa de fijación de precios y utilizando las técnicas asociadas a la relajación lagrangiana para la actualización de los precios.

La técnica de relajación lagrangiana propone relajar los problemas de optimización eliminando un conjunto de restricciones, haciendo el problema a optimizar más sencillo. El conjunto de restricciones se incorpora a la función objetivo como un término que penaliza el incumplimiento de la restricción.

Si se eliminan del problema las restricciones de capacidad, estas restricciones las incorporaremos a la función objetivo multiplicadas por un escalar no negativo λ_{hk} penalizando el incumplimiento de las mismas. El problema relajado del problema quedará:

$$R: \quad \min_{\delta_{ihk}} \sum_{i=1}^{I} w_i T_i^2 + \sum_{h=1}^{M} \sum_{k=0}^{K} \lambda_{hk} \left(\sum_{i=1}^{I} \delta_{ihk} - M_{hk} \right) \quad (7)$$

$$s.a \quad \lambda_{hk} \geq 0$$

restricciones (3) y (4)

Si reagrupamos términos en la expresión 7 el problema relajado queda:

R: $$\min_{\delta} -\sum_{h=1}^{M}\sum_{k=0}^{K}\lambda_{hk}M_{hk} + \sum_{i=1}^{I}\left(w_i T_i^2 + \sum_{h=1}^{M}\sum_{k=0}^{K}\lambda_{hk}\delta_{ikh}\right) \quad (8)$$

s.a $\lambda_{hk} \geq 0$

restricciones (3) y (4)

El problema dual es:

D: $$\max_{\lambda}\left\{\min_{\delta} -\sum_{h=1}^{M}\sum_{k=0}^{K}\lambda_{hk}M_{hk} + \sum_{i=1}^{I}\left(w_i T_i^2 + \sum_{h=1}^{M}\sum_{k=0}^{K}\lambda_{hk}\delta_{ikh}\right)\right\} \quad (9)$$

s.a $\lambda_{hk} \geq 0$

restricciones (3) y (4)

Para unos λ dados es posible descomponer el problema (9) en un conjunto de subproblemas correspondientes al conjunto de tareas a programar. Así, para cada tarea i dados unos determinados □ el problema a resolver será:

R_i: $$\min_{\delta} w_i T_i^2 + \sum_{h=1}^{M}\sum_{k=0}^{K}\lambda_{hk}\delta_{ikh} \quad (10)$$

s.a $\lambda_{hk} \geq 0$

restricciones (3) y (4)

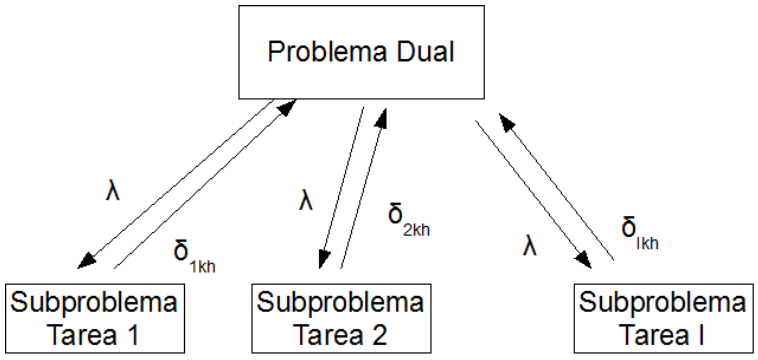

Figura 1. Estructura de resolución del algoritmo

La función lagrangiana no es diferenciable en ciertos puntos del espacio λ. Para resolver el problema dual (R) utilizaremos el método del subgradiente (Fisher 1985). En este método los multiplicadores λ son actualizados de forma iterativa según la expresión:

$$\lambda^{n+1} = max(0, \lambda^n + \alpha^n g(\lambda^n)) \quad (11)$$

donde:

n es la iteración actual

$g(\lambda^n)$ es el subgradiente de v(LR) respecto a λ con calculado como

$$g_{h,k} = \sum_{i} \delta_{ijhk} - M_{h,k} \quad (12)$$

α^n es el paso, dado por la expresión:

$$\alpha^n = \gamma \frac{L^* - L^n}{g(\lambda^n)^T g(\lambda^n)} \quad 0 < \gamma < 2 \qquad (13)$$

donde:

L^n es el valor de v(LR) en la n-enésima iteración

L^* es el valor óptimo de la función lagrangiana $L^* \geq L^n$

γ es una constante cuyo valor es $0 < \gamma < 2$

Se puede ver que el subgradiente, en este caso, evalúa en qué medida no se cumplen las restricciones de capacidad del problema. El efecto es que cuando las pujas sobre un determinado instante de un recurso superan la capacidad del mismo este hecho es penalizado aumentando el valor del multiplicador de Lagrange correspondiente. El valor del multiplicador de Lagrange se reducirá cuando la capacidad sea superior al número de pujas sobre el instante de tiempo requerido.

Dado que no se conoce el valor óptimo de la función lagrangiana, lo habitual es tomar un valor estimado. El valor de la función objetivo del problema inicial para el programa factible es una cota superior de L^*. Este valor se toma como mejor estimación de L^*.

6. Mecanismo de subasta

El modelo estará formado por agentes *Tarea* que representarán las características y objetivos de cada una de las tareas que deba ser realizada en el sistema de producción, agentes *Recurso*, con las características que los definen, y un agente *Subastador,* que realizará el papel de coordinador de los agentes. Cada agente *Recurso* subastará su uso para cada uno de los instantes de tiempo en que se divide el horizonte de programación. Los agentes *Tarea* pujarán por los instantes de tiempo que más le interesen, dados unos determinados precios. La puja que cada agente *Tarea* envía al subastador es el conjunto de instantes de tiempo de cada recurso deseados. El subastador recogerá las pujas y calculará de nuevo los precios, enviando esta información a las tareas y siguiendo un proceso iterativo que finalizará cuando los precios se estabilicen. Este proceso se detalla a continuación.

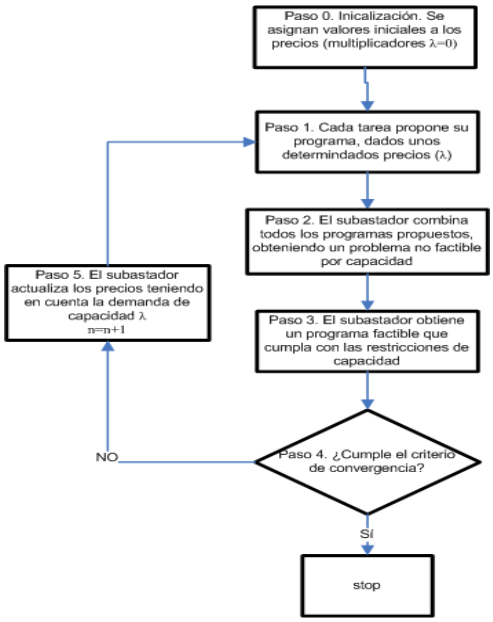

Figura 1. Diagrama de flujo de la subasta

Paso 0. Inicialización de variables. El subastador comienza fijando unos precios iniciales para los slot de tiempo de cada uno de los recursos λ^0_{hk}. Lo habitual es comenzar con un valor de cero para el conjunto de slots de tiempo $\lambda^0_{hk} = 0$.

Paso 1. Cálculo de la puja a enviar por parte de cada tarea. Cada una de las tareas resuelve su subproblema de optimización para unos precios dados (λ^i). Su objetivo es que la suma de la función de penalización por el retraso más el valor del coste de uso de los recursos, la suma de los precios de los slots de los recursos seleccionados sea mínimo (10). Para ello cada tarea evaluará todas las alternativas escogiendo la de menor valor. Cada puja (conjunto de slots seleccionados) es enviada al subastador.

Paso 2. Evaluación de las pujas enviadas. El subastador combina los programas propuestos por las tareas y evalúa las necesidades de capacidad de cada recurso que existen en cada instante de tiempo.

Paso 3. Construcción del programa factible. En general la solución propuesta a partir de las soluciones de los subproblemas de cada una de las tareas va a resultar infactible porque no se cumplirán las restricciones de capacidad de los recursos. Será necesario construir una solución factible a partir de la solución propuesta. Este punto se desarrollará en el apartado siguiente.

Paso 4. Criterio de parada. Los criterios de parada utilizados habitualmente son que el tamaño del paso α^n sea suficientemente pequeño o que se haya superado un determinado número de iteraciones. Si se cumple alguno de ellos el algoritmo se parará. En caso contrario se volverá al paso 1, comenzando una nueva iteración.

Paso 5. Actualización de los precios. La actualización de precios tendrá en cuenta la capacidad de cada máquina solicitada en cada instante de tiempo. El objetivo de actualizar los precios es reducir los conflictos entre los agentes. Los precios penalizan el incumplimiento de las restricciones relajadas, utilizando para ello la expresión (11). Así, en los periodos de tiempo en los que exista una demanda superior a la capacidad del recurso el precio de ese periodo de tiempo aumentará. Si coinciden demanda y capacidad del recurso, el precio no variará. Por último, si la demanda de un recurso en un determinado instante es inferior a su capacidad, el precio del recuso disminuirá. El precio no puede ser negativo.

7. Construcción del programa factible

Una vez resueltos los subproblemas por parte de cada tarea, estos son enviados al subastador. El subastador comprueba si existen conflictos entre las propuestas enviadas por las tareas. Existe conflicto cuando el número de tareas que solicita un instante determinado de un recurso es superior a la capacidad máxima del recurso. Si no existen conflictos, el programa propuesto por el conjunto de la tarea será factible, con lo que se habrá llegado a la solución deseada. En general el programa propuesto a partir de las soluciones individuales no será factible, por lo que será necesario construir un programa que cumpla con las restricciones del problema original a partir de las soluciones propuestas por cada tarea.

Para la construcción del problema factible se utilizará una heurística a partir de la solución del problema relajado. Las heurísticas más habitualmente utilizadas ordenan en cada recurso las tareas por la fecha de comienzo o por la fecha de finalización propuestas en el problema relajado. El método consistirá en considerar en cada instante del horizonte de programación el conjunto de tareas que puede ser programado. Si en un determinado instante el conjunto de operaciones susceptibles de ser programadas en un determinado recurso no supera la capacidad del recurso, estas son programadas en el recurso en ese instante. En caso contrario, se utilizará una heurística para determinar qué tareas serán programadas de forma que se respete las restricciones de capacidad, y cuáles permanecerán pendientes de ser programadas. Una vez programados todos los recursos para un determinado instante, se pasará al instante siguiente, incorporándose al proceso las nuevas tareas susceptibles de ser programadas. Este proceso se repetirá hasta que hayan sido programadas todas las operaciones.

En (Hoitomt et al. 1993) se propone una heurística para la generación de programas factibles utilizando un algoritmo voraz que considera el incremento en la función de coste original. En caso de conflicto se evalúa la penalización que supone para cada orden afectada el retraso de una unidad de tiempo. Se realizará aquella que su retraso suponga una penalización mayor.

8. Conclusiones

Podemos considerar el problema de programación de tareas que necesitan varios recursos de forma simultánea como un sistema donde cada tarea actúa como un agente inteligente que trata de ser realizada utilizando los servicios que le proporcionan otros agentes recurso. Cada tarea trata de minimizar el coste que le supone el uso de los recursos y de la penalización que le supone el retraso en su finalización. El coste de los recursos va a ser establecido por un mecanismo, subasta combinatoria, donde se considera la oferta (capacidad de cada recurso) y la demanda (cantidad de tareas que solicitan el uso del recurso en un determinado instante).

La relajación lagrangiana presenta la ventaja principal de descomponer problemas complejos a nivel de taller o de proyectos en problemas individuales a nivel de órdenes de trabajo, relajando las restricciones de capacidad de los recursos. De este modo se distribuye la carga principal de cálculo entre los agentes participantes.

Referencias

Dewan, P. & Joshi , S. (2002). Auction-based distributed scheduling in a dynamic job shop environment . *International Journal of Production Research,*, 40(5), 1173-1191.

Dobson, G. & Karmarkar, U.S. (1989). Simultaneous Resource Scheduling to Minimize Weighted Flow Times. *Operations Research*, 37(4), 592-600.

Duffie, N.A. (1990). Synthesis of heterarchical manufacturing systems. *Comput. Ind.*, 14(1-3), 167-174.

Fisher, M.L. (1985). An Applications Oriented Guide to Lagrangian Relaxation. *INTERFACES*, 15(2), 10-21.

Hoitomt, D., Luh, P. & Pattipati, K. (1993). A practical approach to job-shop scheduling problems. *Robotics and Automation, IEEE Transactions on*, 9(1), 1-13.

Kaskavelis, C.A. & Caramanis, M.C., (1998). Efficient Lagrangian relaxation algorithms for industry size job-shop scheduling problems. *IIE Transactions*, 30(11), 1085-1097.

Kutanoglu, E. & Wu, S.D. (1999). On combinatorial auction and Lagrangean relaxation for distributed resource scheduling. *IIE Transactions*, 31(9), 813-826.

Lee & Kim (2008). Multi-agent systems applications in manufacturing systems and supply chain management: a review paper. *International Journal of Production Research*, 46(1), 265, 233.

Luh, P. & Hoitomt, D. (1993). Scheduling of manufacturing systems using the Lagrangian relaxation technique. *Automatic Control, IEEE Transactions on*, 38(7), 1066-1079.

Pinedo, E.P.M.L. (2008). *Scheduling* 2º ed., Springer-Verlag New York, Inc.

Shen, W. (2002). Distributed manufacturing scheduling using intelligent agents. *Intelligent Systems*, 17(1), 94, 88.

Shen, W. et al. (2006). Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced Engineering Informatics*, 20(4), 415-431.

Tang, L., Xuan, H. & Liu, J. (2006). A new Lagrangian relaxation algorithm for hybrid flowshop scheduling to minimize total weighted completion time. *Computers & Operations Research*, 33(11), 3344-3359.

Tao Sun, Luh, P. & Min Liu (2006). Lagrangian relaxation for complex job shop scheduling. En *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. págs. 1432-1437.

de Vries, S. & Vohra, R.V. (2003). Combinatorial Auctions: A Survey. *INFORMS JOURNAL ON COMPUTING*, 15(3), 284-309.

Wellman, M.P. et al. (2001). Auction Protocols for Decentralized Scheduling. *Games and Economic Behavior*, 35(1-2), 271-303.