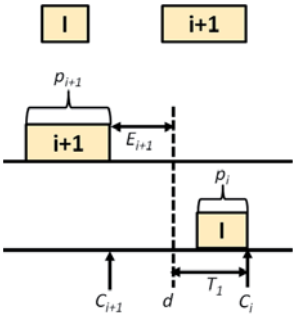
Planeación justo a tiempo:
Soluciones óptimas mediante reformulaciones convexas

Fernando Elizalde Ramírez
Instituto Tecnológico de Cerro Azul
fernandoelizalderamirez@gmail.com
Yadira I. Silva Soto, Yasmín A. Ríos-Solís
División de Posgrado en Ingeniería de Sistemas, FIME-UANL
yadis76@gmail.com , agueda.riossl@uanl.edu.mx



RESUMEN

En este trabajo se aprovecha el gran alcance de la teoría de la programación cuadrática para obtener soluciones óptimas de problemas complejos de planeación de la producción justo a tiempo. Se aplica una metodología que inicia, a diferencia de otros enfoques clásicos, con una formulación del problema de planeación justo a tiempo para máquinas paralelas mediante un programa cuadrático con variables 0-1 y restricciones lineales. Por construcción, este programa de segundo grado no es convexo por lo que se reformuló antes de someterlo a un procedimiento de ramificación y acotamiento para obtener la solución óptima, de manera que se garantiza la convexidad y se obtiene una cota inferior de alta calidad. Los resultados obtenidos muestran que esta metodología permite obtener mejores resultados en comparación con otras estrategias reportadas en la literatura científica.

PALABRAS CLAVE

Programa cuadrática, reformulación convexa, matriz semidefinida positiva, planeación justo a tiempo.

ABSTRACT

Advantage of quadratic programming theory to obtain optimal solutions of just-in-time scheduling problems is taken in this work. A methodology that begins, in contrast to more classical approaches, by formulating the just in time parallel machine scheduling problems as a 0–1 quadratic programs under linear constraints is applied. By construction this quadratic program is non-convex. Therefore, it is reformulated in such a way that we can ensure convexity and a high-quality continuous lower bound before submitting it to a branch-and-bound procedure. Experimental results show that this methodology produce better results compared to other strategies reported in the scientific literature.

KEYWORDS

Quadratic programming, convex reformulation, positive semidefinite matrix, just-in-time optimization.

INTRODUCCIÓN

En este mundo globalizado, las empresas tienen como prioridad mantener satisfecho al cliente. Para esto buscan desarrollar productos de calidad y entregarlos en la fecha acordada. Pero de igual modo tratan de disminuir los costos de producción y de almacenamiento, y es allí donde nace la Planeación Justo-a-Tiempo.

La planeación justo a tiempo es un problema complejo en que se debe secuenciar un número de tareas dadas, en un conjunto de máquinas paralelas, donde el número de tareas es mayor que el de las máquinas. Todas las tareas tienen una fecha de entrega común, así como un tiempo de procesamiento. Cada tarea tiene penalidades de adelanto (costos de almacén, vigilancia, costos por caducidad) y de atraso (descontento del cliente). El objetivo es ejecutar todas las tareas en las diferentes máquinas de manera que se minimice la sumatoria ponderada de las penalidades de adelanto y de retraso.

Este problema puede modelarse de manera natural con una función objetivo cuadrática. En la literatura, la metodología usual es hacer una linearización del modelo para que el problema sea tratable. Nuestro aporte es usar el modelo cuadrático y reformularlo de manera a hacerlo tratable. La tratabilidad en estos problemas significa que el modelo puede ser sometido a un algoritmo clásico de ramificación y acotamiento (Brach and Bound, B&B). Dicho algoritmo se encuentra en muchas librerías comerciales de la optimización discreta.

El algoritmo B&B hace relajaciones lineales (ya no considera que se requieren variables 0-1) y con esto logra explorar el espacio de soluciones de manera inteligente para finalmente encontrar la solución óptima. Sin embargo, para poder usar este método requerimos dos cualidades muy importantes en el modelo: convexidad y calidad en las relajaciones.

El modelo cuadrático del problema de producción justo a tiempo que presentaremos más adelante tiene asociada una matriz Hessiana cuya estructura es diagonal a bloques. Esta matriz no es semidefinida positiva, por lo tanto la función objetivo no es convexa.

El objetivo de este artículo es proponer métodos de reformulaciones convexas que usan la estructura diagonal a bloques de la matriz Hessiana. Con este método, obtenemos un problema equivalente convexo y podemos optimizar el problema mediante los ya mencionados métodos de ramificación y acotamiento.

Este artículo está basado en el de Plateau y Ríos-Solís[1] y en la tesis de maestría de Soto-Silva.[2] Las principales mejoras que proponemos en este estudio, son el problema nuevo de máquinas paralelas idénticas y el uso de la estructura diagonal a bloques de la matriz Hessiana para las reformulaciones convexas.

FORMULACIÓN MATEMÁTICA

Nuestro caso de aplicación es la Planeación Justo a Tiempo con fecha de entrega común, es decir, se requiere que todas las tareas sean entregadas en una misma fecha. Por ejemplo, esto sucede si hay una fase de ensamble posterior o si el cliente así lo requiere. Se busca idealmente: cero inventarios, cero transacciones y cero disturbios.

Los problemas de producción Justo a tiempo han sido ampliamente estudiados en la literatura. Sin embargo, el esfuerzo se ha centrado en los problemas con una sola máquina.[3-5] En este artículo nos enfocamos al caso de máquinas paralelas.

Formalmente, el problema que se estudia consiste en secuenciar n tareas en m máquinas paralelas que trabajan a velocidades iguales. Se tienen los siguientes conjuntos:

J es el conjunto de tareas que van a ser procesadas en las m máquinas; $J = \{1, 2, 3, \ldots, n\}$.

M es el conjunto de máquinas en las cuales se van a secuenciar (o ejecutar) las tareas; $M = \{1, 2, 3, \ldots, m\}$.

Cada tarea i tiene asociado un tiempo de procesamiento o tiempo de ejecución, dependiendo de la máquina j en la cual la tarea se realice, denotado por p_{ij}. Además, todas las tareas tienen asociada una fecha de vencimiento común, denotada como d. Revisiones de literatura acerca de las fechas de entrega comunes en problemas de justo a tiempo fueron presentadas por Baker y Scudder[6] y por Gordon *et al*.[7]

El tener en el problema una fecha de vencimiento común genera inevitablemente atrasos y adelantos en las tareas con respecto a esta fecha. Lo deseable es que las tareas terminaran de ser procesadas justo

a tiempo, es decir, a la fecha de entrega. Esto no es posible ya que todas las tareas tienen la misma fecha de vencimiento y el número de tareas es mayor que el de las máquinas, por lo que algunas tareas serán terminadas de procesar antes de la fecha de entrega, las así llamadas tareas adelantadas, las cuales pagarán una penalidad de adelanto αi por unidad de tiempo. También se tiene tareas que terminan después de la fecha de entrega, a las cuales se le conoce como tareas atrasadas y se les cobra una penalidad de atraso β_i por unidad de tiempo.

Lo que se busca es el tiempo de fin de cada una de las tareas y la máquina que las procesará con el objetivo de que la suma ponderada de penalidades de adelantos y de retrasos sea minimizada.

En la figura 1 el tamaño de cada bloque indica el tiempo de procesamiento de cada tarea. Si se observa, la tarea 1 y 4 quedaron justo a tiempo de la fecha de entrega, mientras que la tarea 2, terminó de ser ejecutada después de la fecha de entrega. La tarea 3, terminó de ser procesada antes de la fecha de entrega común.

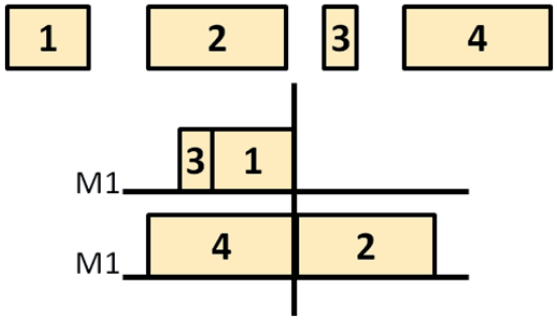


Fig. 1. Ejemplo de secuenciación de tareas.

En la figura 2, se tiene que la tarea *i+1* es ejecutada por adelantado. El costo por tener está tarea por adelanto que es terminada en el tiempo C_{i+1}, *es* $\alpha_{i+1}\ max\{0,\ d - C_i\} = E_{i+1}$. Ahora, si la tarea se realiza justo a tiempo entonces no tendrá costo. De igual manera, si se tiene una tarea con retraso, se tiene un costo de $\beta_{i+1}\ max\{0,\ C_i - d\} = T_i$.

El modelo matemático para este problema queda definido con las variables x_{ijk} que toman el valor de 1 si la tarea *i* se hace en la máquina *j* y es del tipo *k*, con *k* igual a *E* si es una tarea por adelantado y *k* igual a *T* si es una tarea que se posicionará en retraso. Dicha variable toma el valor de 0 en otro caso. El modelo matemático es el que sigue y está basado en:[2]

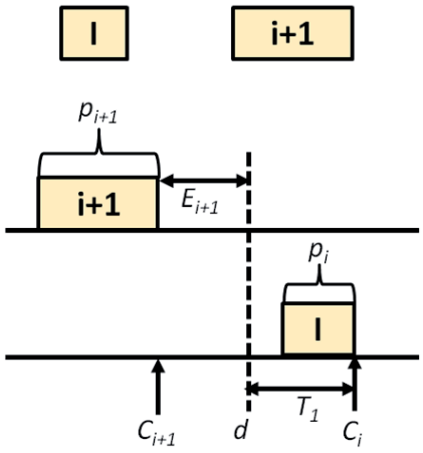


Fig. 2. Ejemplo de costos de adelanto y retraso de secuenciación de tareas.

$$min\ \Sigma_{i\in J}\alpha_i E_i + B_i T_i$$

$$s.a.\ \Sigma_{i\in J}\Sigma_{l\in\{E,\,T\}} x_{ijl} = 1\ \forall_i \in J$$

$$x \in \{0,\ 1\}^{2nm}$$

Donde

$$E_i = \Sigma_{i\in M}\ x_{ijE}\left(\Sigma_{k\prec^i_{Ei}}\ x_{kjE}\ P_{kj}\right)$$

$$T_i = \Sigma_{i\in M}\ x_{ijT}\left(P_{ij} + \Sigma_k \prec^i_{Ei}\ x_{kjT}\ P_{kj}\right)$$

Los índices de las sumatorias para E_i y T_i representan un orden parcial de las tareas que se define mediante su prioridad dada por la división entre sus penalidades por unidad de tiempo de procesamiento. Para más detalles ver.[2,8]

De igual manera se puede escribir el anterior modelo de programación cuadrática en su forma estándar (e.g.[9]):

$$ming(x) = \tfrac{1}{2}x^t Q x + cx$$

$$s.a.\ Ax = 1$$

$$x \in \{0,1\}^{2nm}$$

Donde:

x= Vector de variables de decisión.

c= Vector de costos lineales, que en este caso solo serían las penalidades de retraso.

Q= Matriz Hessiana asociada a la función objetivo.

A= Matriz de restricciones.

La matriz *Q* asociada a este problema no es una matriz semidefinida positiva, por lo tanto la función objetivo no es convexa. Una particularidad de este problema, y que se ha observado ya en otros problemas de optimización combinatoria, es su estructura diagonal por bloques como lo muestra la figura 3.

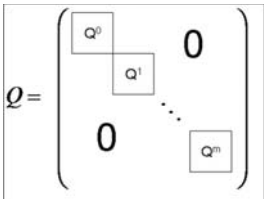


Fig. 3. Estructura diagonal por bloques de la matriz Hessiana.

Este nuevo modelo cuadrático nos permite aplicar las reformulaciones convexas que a continuación presentaremos. Mejor aún, las modificaremos para aprovechar que la matriz Hessiana tiene una estructura a bloques que podemos utilizar para crear mejores reformulaciones convexas, como se mostrará al final de este trabajo.

MÉTODOS DE REFORMULACIONES CONVEXAS

Existen varios métodos de reformulaciones convexas basados en la perturbación de la matriz Hessiana. Nuestro método, el 4, usa las reformulaciones 1.-3., pero aprovecha la estructura a bloques de la matriz Hessiana.

1. Método que utiliza CPLEX[10] para optimizar un problema de programación cuadrática: primero verifica si la matriz Hessiana es semidefinida positiva, si lo es, comienza la optimización mediante un algoritmo de barrera (puntos interiores). Ahora, si es un problema en el que se tienen variables binarias o enteras, optimiza mediante un método de B&B. Si la matriz no es semidefinida positiva la convexifica sumando un número grande en la diagonal.
2. Método basado en el mínimo valor propio[11,12] que consiste en perturbar la diagonal de la matriz Hessiana con el mínimo valor propio de ésta, es decir, a cada elemento de la diagonal principal se le resta el mínimo valor propio de la matriz Hessiana, como se observa en la figura 4. Esto garantiza que la matriz Hessiana sea una matriz semidefinida positiva y la función objetivo sea convexa.

Existen otros métodos de reformulación que en este trabajo no se consideran dado que tardan mucho en obtenerse las reformulaciones para instancias medias.

3. Método basado en programación semidefinida:[9] La programación semidefinida es un caso especial de la programación convexa. La idea general de la programación semidefinida es optimizar una función lineal sujeta a restricciones lineales con la condición de que haya una matriz semidefinida positiva, ya que el conjunto de matrices semidefinidas positivas constituye un conjunto convexo. El objetivo de este método es determinar los mejores parámetros que hacen que la función equivalente obtenida sea convexa y además, que la cota inferior obtenida mediante relajación continua sea máxima, es decir, que la función equivalente este lo más cerca de la función objetivo original.

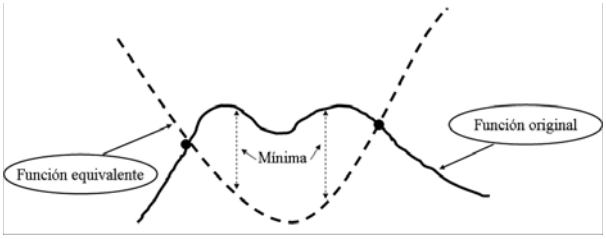


Fig. 4. Función equivalente.

Se tiene ahora una función equivalente convexa a la función original, como se muestra en la figura 4. En efecto, una reformulación no es más que un problema que en variables 0-1 da exactamente el mismo valor que el problema original. Sin embargo, cuando se relaja (para usar el método de B&B) es diferente el valor que se obtiene. Lo que se busca es una reformulación que en la relajación sea lo más cercano a la función original como lo muestra la figura 4 (los puntos de intersección son las soluciones con variables 0-1).

Es aquí donde proponemos nuestra reformulación.

4. Método basado en la estructura diagonal a bloques: Nuestro método aprovecha la estructura diagonal a bloques de la matriz Hessiana, separando la matriz original en submatrices que representan a cada bloque, tal que la suma de estas matrices den como resultado la matriz original. Como se puede observar en la figura 5.

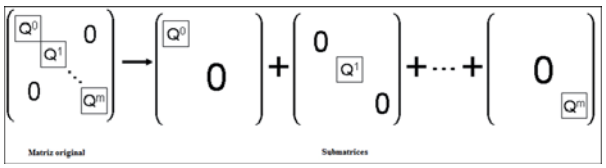


Fig. 5. Descomposición de una matriz Hessiana.

El objetivo de nuestro método es hacer semidefinida positiva cada submatriz de la matriz original mediante el método de mínimo valor propio. Mediante estos métodos se obtienen parámetros con los que se perturban cada una de las submatrices, por lo tanto las submatrices se vuelven semidefinidas positivas, y sumadas conforman la matriz original pero ahora semidefinida positiva, teniendo así que la función objetivo es convexa. Así, al convexificar matrices más pequeñas, el tiempo de cómputo total se ve drásticamente disminuido.

Los pasos a seguir en nuestro método son:

1. Descomponer la matriz Hessiana en submatrices.
2. Obtener el mínimo valor propio de cada submatriz.
3. Obtener una reformulación convexa de la función objetivo perturbando cada bloque de la matriz Hessiana con el mínimo valor propio que le corresponde a cada submatriz.
4. Utilizar un método de B&B para obtener soluciones óptimas de la función equivalente y por lo tanto estas soluciones también son óptimas para la función original.

RESULTADOS EXPERIMENTALES

Los resultados presentados en la tabla I, se lograron mediante la implementación del método propuesto de descomposición por bloques y mínimo valor propio para dos problemas de planeación de la producción justo a tiempo. El primero es el que se analizó en[2] que considera máquinas paralelas diferentes para ejecutar las tareas (lado izquierdo de la tabla I). El segundo es el que se propone en este trabajo que considera máquinas paralelas idénticas.

La tabla I tiene en la primera columna el tamaño de la instancia (m,n), es decir, el número de máquinas y el número de tareas. Es de observarse que el tamaño de instancia alcanzado es el considerado como mediano. Sin embargo, resoluciones exactas para este problema no alcanzan estas tallas. Es decir, son los mejores resultados de la literatura. Cada renglón de la tabla I es un promedio de 10 instancias. La columna Objetivo muestra el valor promedio de la función objetivo. Podemos ver que aunque parezca contra intuitivo, es frecuente que en máquinas paralelas idénticas tenemos peores valores objetivo. La columna Iteraciones corresponde al

Tabla I. Resultados obtenidos para modelos anteriores y para el modelo propuesto en este estudio.

Máquinas diferentes paralelas							Máquinas idénticas paralelas				
Instancia	Objetivo	Iteraciones	Nodos	Tiempo (segundos)	GAP %		Objetivo	Iteraciones	Nodos	Tiempo (segundos)	GAP %
(2, 30)	1917,2	9704	4264	1,59	0,00		2951	4132355	3565893	1355,7	0,00
(2, 60)	7838,2	272771	117659	101,28	0,00		12598,2	9170179	3288118	3600	0,00
(2, 100)	21023,8	2236485	1031468	1907,10	0,01		33856,4	3489428	1315821	3600	2,59
(2, 150)	45481,6	1856769	800805	3600	0,08		76622,4	1330727	531040	3600	1,96
(4, 30)	501,2	18008409	4697613	3600	0,00		1307	19276149	4722695	3600	10,06
(4, 60)	3194,2	7904528	1744828	3600	8,45		6641,5	4632296	1135753	3600	10,05
(4, 100)	5814,25	2544829	763620	3600	5,61		16524,4	1030856	313420	3600	9,65
(4, 150)	14427,2	1578912	379822	3600	4,13		39781,4	223493	67423	3600	6,98
(6, 30)	268,6	13068927	2826620	3600	95,88		744	12608695	2732882	3600	52,2
(6, 60)	1199,4	5159969	993857	3600	40,60		4075	2498753	573813	3600	25,66
(6, 100)	2929,4	1955241	357798	3600	24,73		11955	497676	115296	3600	12,45
(6, 150)	6721,4	804388	132831	3600	14,56		23828,2	93670	23123	3600	12,4
(8, 30)	130,6	9149790	2424856	3600	334,51		706,4	11242494	2233081	3600	76,59
(8, 60)	565,2	2508000	499349	3600	112,58		2033,25	2161275	360625	3600	17,6
(8, 100)	1775,8	1037400	188474	3600	48,93		7286,6	341880	65975	3600	12,82
(8, 150)	3970,4	412540	132831	3600	27,81		18532,6	78029	15250	3600	4,35

número de relajaciones lineales que se hicieron para resolver el problema. De nuevo observamos que contra intuitivamente para las máquinas idénticas tenemos un mayor número de iteraciones. Esto puede ser porque cuando las máquinas son diferentes, el algoritmo descarta malas soluciones rápidamente. La columna Nodos se refiere al número de nodos explorados por el B&B. El Tiempo refleja el tiempo total del B&B en segundos (es de notarse que se limitó el tiempo a 3600 segundos). Finalmente la columna GAP% refleja la diferencia entre la mejor solución obtenida y la mejor relajación.

Se puede observar en esta tabla, en los resultados de las instancias con dos máquinas, que los mejores resultados son obtenidos para el modelo de máquinas diferentes paralelas debido a que sus valores son menores a comparación del otro conjunto, con la misma cantidad de máquinas pero en este caso idénticas.

Para el caso donde se tienen 4 máquinas, el modelo de máquinas diferentes paralelas presenta mejores valores del gap entre los dos modelos, aunque el de las máquinas idénticas paralelas tiene valores más bajos de iteraciones y nodos explorados, tarda más en resolver que el de las máquinas diferentes.

COMPARACIÓN DE RESULTADOS

En la figura 6 se muestran comparativos entre máquinas paralelas diferentes e idénticas. En la primera línea están dos gráficas para el caso de máquinas paralelas diferentes para instancias con 30 y 150 tareas. En la segunda línea de la figura 6 se muestra el caso de máquinas paralelas idénticas también para instancias de 30 y 150 tareas. En las cuatro gráficas se muestra en el eje de las abscisas corridas de diferentes instancias (5 instancias diferentes). En el de las ordenadas se muestra el porcentaje del GAP ((Mejor solución conocida-Mejor solución relajada)/Mejor solución conocida). Se observa como el GAP disminuye cuando se incrementa el número de tareas. Contra intuitivamente, para el caso de máquinas paralelas los GAPs para instancias de 30 tareas son menores que para máquinas diferentes. Sin embargo, para instancias de 150 tareas eso es contrario.

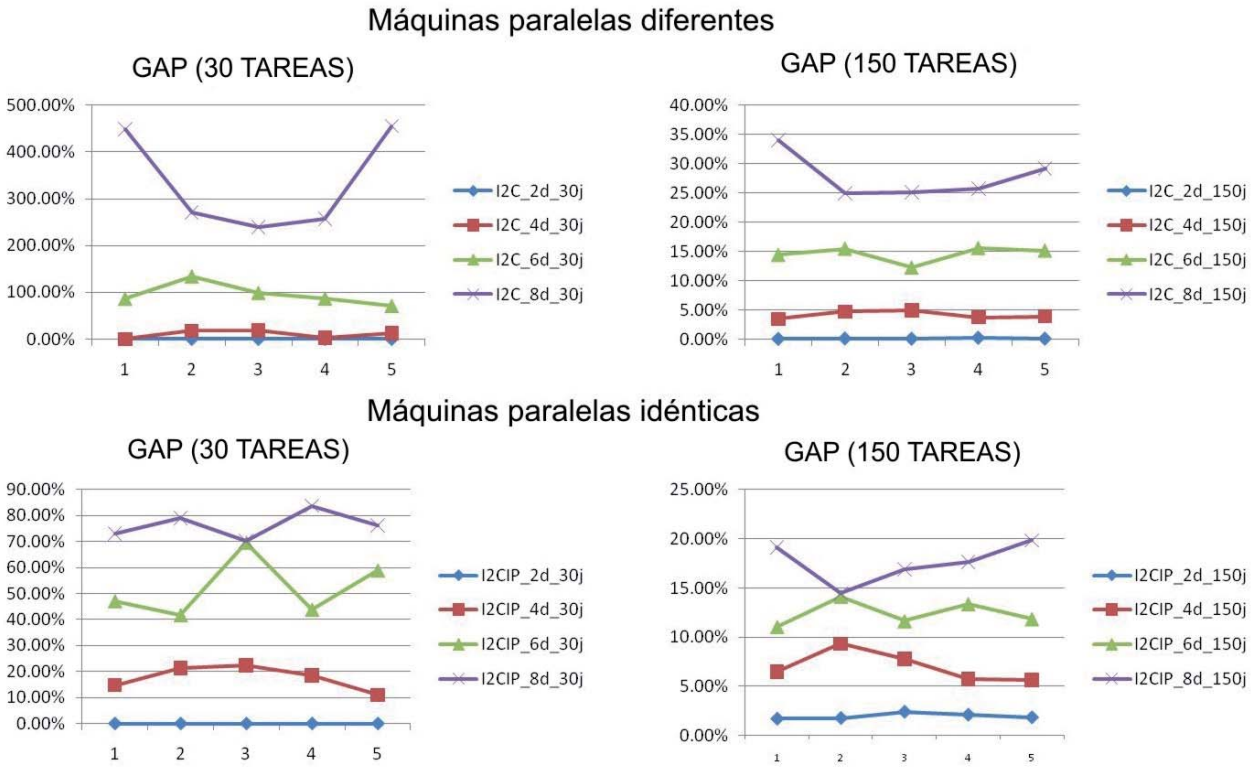


Fig. 6. GAPs y comparativo entre número de tareas y de máquinas.

En la figura 7 se muestran gráficas similares pero ahora en el eje de las ordenadas se tiene el número de iteraciones que realiza el B&B (mayor número de iteraciones refleja una instancia más difícil). En la primera columna están las dos gráficas de máquinas paralelas diferentes y en la segunda columna la de paralelas idénticas. Se puede observar que ahora el número de iteraciones para máquinas idénticas es menor para instancias de 30 tareas. Sin embargo, vemos lo contrario con las instancias de 150 tareas.

La figura 6 es relevante dado que para entender mejor el comportamiento de las instancias se requiere entender las soluciones que arrojan. De esta manera se pueden proponer mejores métodos de resolución.

Los mejores resultados del GAP los arroja el conjunto de máquinas idénticas paralelas, ya que en la mayoría de las instancias son más bajos en comparación con los resultados obtenidos para las máquinas diferentes paralelas, esto quiere decir que se encuentra más cerca del resultado óptimo.

La figura 7 es también una comparativa para entender mejor el comportamiento del método B&B cuando es utilizado en nuestras instancias y con nuestro método de reformulaciones por bloques.

Lo relevante de estas figuras es que el número de iteraciones disminuye conforme el número de tareas aumenta. Sin embargo, sigue siendo menor para el caso de máquinas diferentes. Un caso peculiar se da en las máquinas diferentes, como se puede observar en las gráficas, conforme se aumenta el número de tareas para el caso de 2 máquinas, el número de iteraciones aumenta.

CONCLUSIONES

En este trabajo se propuso un modelo matemático para el problema de la planeación de la producción justo a tiempo en máquinas paralelas y con una fecha de entrega común. Se propuso usar métodos de reformulaciones convexas que aprovecharan la estructura diagonal a bloques de la matriz Hessiana y se realizaron experimentos computacionales. Cabe mencionar que el problema que se trató es de actualidad en la literatura científica y que los resultados obtenidos son los mejores de la literatura en lo que se refiere a soluciones exactas.

El método de reformulación convexa propuesto en este trabajo, puede ser utilizado para cualquier problema de programación cuadrática, con restricciones lineales y variables binarias como son los problemas de localización, por ejemplo.

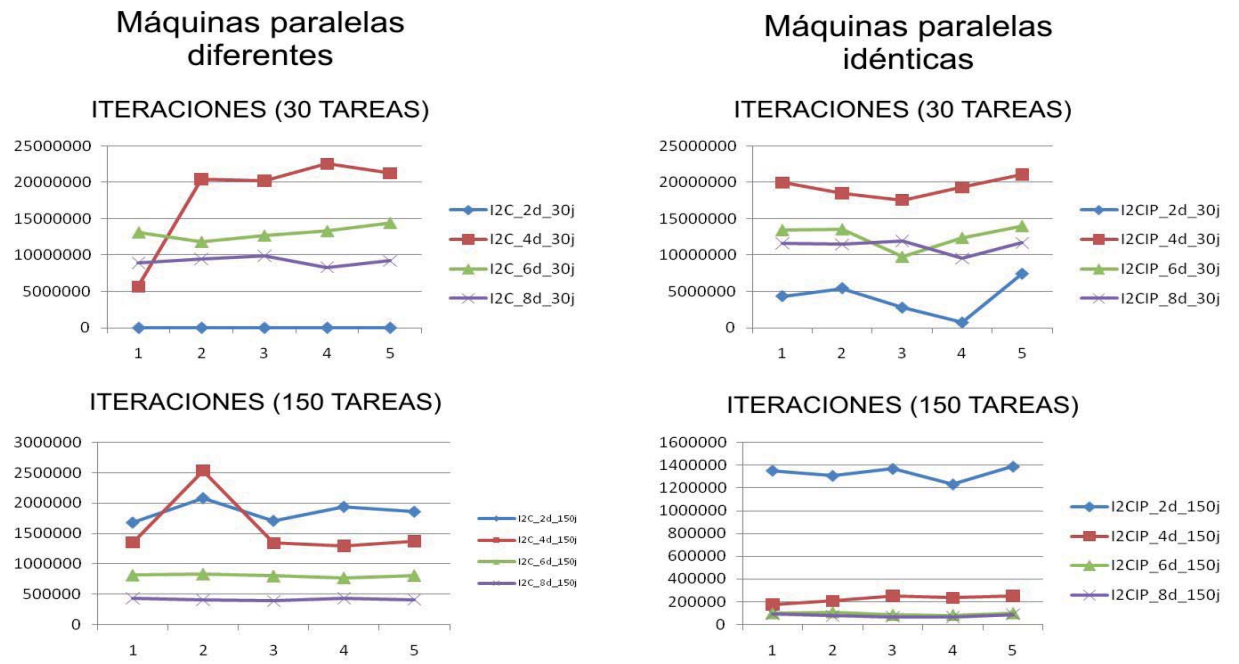


Fig. 7. Iteraciones y comparativo entre número de tareas y de máquinas.

Para el caso de este trabajo donde se estudia un sistema de planeación justo a tiempo con fecha común se compararon los resultados obtenidos para máquinas diferentes paralelas y máquinas idénticas paralelas teniendo las siguientes observaciones:

- El valor del GAP disminuye conforme se incrementa el número de tareas, pero aumenta conforme el número de máquinas se hace mayor.
- El número de iteraciones de igual modo que el GAP, disminuye conforme aumenta el número de tareas y a mayor número de máquinas más iteraciones se realizaran.

Este estudio permite que en futuras investigaciones se utilice más a fondo la estructura de los problemas para obtener mejores resultados.

AGRADECIMIENTOS

A la Academia de Mexicana de Ciencias por la beca otorgada al Ing. Fernando Elizalde Ramírez para la realización de la estancia del Verano de la Investigación Científica en la Facultad de Ingeniería Mecánica y Eléctrica de la UANL en el posgrado de Ingeniería en Sistemas.

A la FIME por la organización de los talleres, clases, seminarios y eventos realizados alrededor del Verano Científico 2010.

REFERENCIAS

1. M.-C Plateau and Y.A. Rios-Solis. Optimal solutions for unrelated parallel machines scheduling problems using convex quadratic reformulations. European Journal of Operational Research, 201(3):729–736, 2010.
2. Yadira I. Silva Soto, Planeación Justo a Tiempo Mediante Reformulaciones Convexas, tesis de maestría UANL, 2010.
3. F. Sourd, A reinforced Lagrangean relaxation for non-preemptive single machine problem, in: Tenth International Workshop on Project Management and Scheduling, 2006, pp. 330–334.
4. M. Feldmann, D. Biskup, Single-machine scheduling for minimizing earliness- tardiness penalties by meta-heuristic approaches, Computers Industrial Engineering 44 (2) (2003) 307–323.
5. M. van den Akker, J. Hoogeveen, S. van de Velde, Combining column generation and Lagrangian relaxation to solve a single-machine common due date problem, INFORMS Journal on Computing 14 (1) (2002) 37–51.
6. K. Baker, G. Scudder, Sequencing with earliness and tardiness penalties: A review, Operations Research 38 (1) (1990) 22–36.
7. V. Gordon, J. Proth, A. Chu, A survey of the state-of-the-art of common due date assignment and scheduling research, European Journal of Operational Research 139 (1) (2002) 1–25.
8. M. Skutella, Convex quadratic and semidefinite programming relaxations in scheduling, Journal of the ACM 48 (2) (2001) 206–242.
9. J. Nocedal, S. Wright, Numerical Optimization, Computer and Information Science Series, Springer-Verlag, Berlin, New York, 2006.
10. ILOG CPLEX 11.2 Web site. http:/www.ilog.com/products/cplex/
11. P. Hammer, A. Rubin, Some remarks on quadratic programming with 0–1 variables, RAIRO 3 (1970) 67–79.
12. M.-C. Plateau, A. Billionnet, S. Elloumi, Eigenvalue methods for linearly constrained quadratic 0–1 problems with application to the densest k- subgraph problem, in: P.U.F. Rabelais (Ed.), 6ème congrès ROADEF, 2005, pp. 55–66.