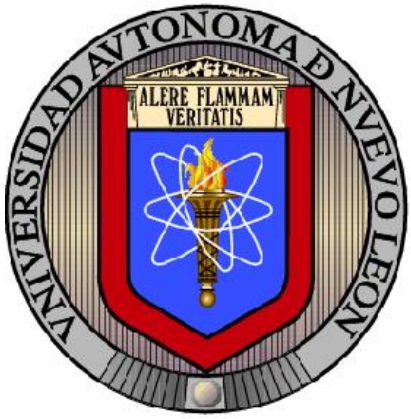
Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado



Planeación justo a tiempo: soluciones óptimas mediante reformulaciones convexas

por

Yadira Isabel Silva Soto

en opción al grado de

Maestro en Ciencias

en Ingeniería de Sistemas

San Nicolás de los Garza, Nuevo León Junio 2010

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado

Planeación justo a tiempo: soluciones óptimas mediante reformulaciones convexas

por

Yadira Isabel Silva Soto

en opción al grado de

Maestro en Ciencias

en Ingeniería de Sistemas

San Nicolás de los Garza, Nuevo León Junio 2010

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado

Los miembros del Comité de Tesis recomendamos que la Tesis «Planeación justo a tiempo: soluciones óptimas mediante reformulaciones convexas», realizada por la alumna Yadira Isabel Silva Soto, con número de matrícula 1219537, sea aceptada para su defensa como opción al grado de Maestro en Ciencias en Ingeniería de Sistemas.

El Comité de Tesis

Dra. Yasmín A. Ríos Solís
Asesor

Dr. Oscar L. Chacón Mondragón
Revisor

Dr. Roger Z. Ríos Mercado
Revisor

Vo. Bo.

Dr. Moisés Hinojosa Rivera
Subdirector
División de Estudios de Posgrado

San Nicolás de los Garza, Nuevo León, Junio 2010

A mi familia, amigos y
mi Beto por su gran apoyo

Índice general

Índice de figuras

Índice de tablas

Agradecimientos

Reciban mi agradecimiento las instituciones de las cuales recibí apoyo:

- El Consejo Nacional de Ciencia y Tecnología (CONACYT) por haberme otorgado la beca de manutención durante los dos años que estuve estudiando la maestría.
- A la Facultad de Ingeniería Mecánica y Eléctrica (FIME) por haber pagado la colegiatura correspondiente a las materias cursadas durante los cuatro semestres de la maestría.
- La Universidad Autónoma de Nuevo León (UANL) por haber pagado las cuotas de rectoría correspondientes a los cuatro semestres cursados durante la maestría.

Gracias al proyecto PROMEP 103.5/08.3125 obtenido por la Dra. Yasmín A. Ríos Solís, por haberme apoyado económicamente para exponer mi trabajo de tesis en el XIX Escuela Nacional de Optimización y Análisis Numérico, Puebla (ENOAN 2009) y en el III Taller Latino Iberoamericano de Investigación de Operaciones (TLAIO3 2009).

Mi profundo agradecimiento a Dra. Yasmín A. Ríos Solís, por haber aceptado ser mi asesora de tesis, por haber dirigido este trabajo tan bonito que realizamos juntas, por su supervisión, por todas los conocimientos que me transmitió, pe-

ro sobre todo por su gran apoyo. Fue un gusto haber trabajo con ella en este proyecto.

A los miembros del comité de tesis: Dr. Óscar Leonel Chacón Mondragón, Dr. Roger Z. Ríos Mercado, por las sugerencias dadas durante el desarrollo de esta tesis y el tiempo que invirtieron en mejorar esta investigación.

A los profesores de PISIS que, a lo largo de estos dos años de estudio, me han enseñado muchas cosas que me ayudaron a elaborar este trabajo, sobre todo de programación.

Gracias a mi familia que siempre me ha brindado su cariño y apoyo en todas las metas que me propongo cumplir y ésta fue una de ellas.

A mis compañeros de maestría por el compañerismo, apoyo y amistad que me brindaron.

A mis amigos de FCFM que ayudaron a quitarme el estrés.

A mis amigos en general, los aprecio muchísimo y gracias por todo el apoyo que me han brindado.

A mi Beto que en estos diez meses me ha apoyado bastante, me ha alentado, me ha insitido tanto para terminar a tiempo este trabajo de tesis.

Resumen

Yadira Isabel Silva Soto.

Candidata para el grado de Mestro en Ciencias
con especialidad en Ingeniería de Sistemas.

Universidad Autónoma de Nuevo León.
Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio: "Planeación justo a tiempo soluciones óptimas mediante reformulaciones convexas".

Planeación justo a tiempo: soluciones óptimas mediante reformulaciones convexas

Número de páginas: 89.

Objetivos y método de estudio: Los objetivos en este proyecto de investigación son:

- Adquirir mediante un estudio profundo, el conocimiento de la estructura del problema de planeación y secuenciación de tareas del tipo justo a tiempo y sus características particulares.
- Investigar acerca de la teoría de matrices semidefinidas positivas, de la programación cuadrática, de la programación semidefinida y cómo poder emplearlas

para resolver problemas de programación cuadrática no convexa en variables binarias.

- Desarrollar un procedimiento para la realización de una reformulación equivalente convexa mediante la teoría de matrices semidefinidas positivas, para un problema de programación cuadrática binaria no convexa. En particular, para un problema de secuenciación justo a tiempo de tareas en máquinas paralelas.

- Evaluar el desempeño del método desarrollado con base en un diseño experimental adecuado así como captar, mediante su estudio, los parámetros que ayuden a mejorar la reformulación mencionada.

La metodología que se empleó para cumplir los objetivos planteados fue:

- Estudio y análisis de la teoría matricial, programación cuadrática y programación semidefinida.

- Estudio de los trabajos que se han realizado en este campo de estudio como por ejemplo, los métodos utilizados para obtener una reformulación equivalente convexa para el problema y los resultados obtenidos mediante estos métodos.

- Diseño del procedimiento para obtener una reformulación equivalente convexa para un problema de programación cuadrática no convexa en variables $\{0, 1\}$.

- Utilizar la estructura de la matriz hessiana (matriz correspondiente de la función objetivo) ya que existen problemas en los que esta matriz tiene la propiedad de ser una matriz diagonal a bloques, para desarrollar el método de reformulación del problema, como por ejemplo, el problema de planeación justo a tiempo en máquinas paralelas.

- Evaluación extensiva del desempeño de la reformulación propuesta bajo un análisis de diseño experimental apropiado.

Contribuciones y conclusiones: La contribución que se presenta en esta tesis es un nuevo método para resolver problemas de programación cuadrática no convexa en variables binarias, el cual consiste en el estudio de la estructura que presenta la matriz hessiana asociada a la función objetivo. La estructura que presenta esta matriz es simétrica diagonal a bloques. El método consiste en descomponer la matriz original en la suma de submatrices. Cada submatriz está compuesta por cada bloque. Se aplica el método de mínimo valor propio a cada submatriz y, finalmente, se perturba la matriz hessiana con el mínimo valor propio, obteniendo una función equivalente convexa. El método propuesto fue comparado con el método que utiliza CPLEX y el método de mínimo valor propio para resolver este tipo de problemas, concluyendo que el método basado en la estructura diagonal a bloques reporta mejores resultados con respecto al método de mínimo valor propio. Comparando el método propuesto con el método que utiliza CPLEX, se concluyó que el método que utiliza CPLEX es mejor en cuanto a calidad de soluciones y en tiempo de ejecución.

Además, el método que se propone puede ser utilizado no sólo en problemas de planeación justo a tiempo, sino también en cualquier otro problema en el cual la matriz hessiana asociada a la función objetivo presente una estructura diagonal a bloques.

Firma del asesor: _____

Dra. Yasmín A. Ríos Solís

Capítulo 1

Introducción

Existen muchos problemas que se pueden modelar mediante un programa cuadrático. Dichos problemas consisten en minimizar o maximizar una función objetivo cuadrática, sujeta a restricciones lineales y variables reales, binarias o enteras. Un ejemplo es el problema combinatorio de planeación justo a tiempo en máquinas paralelas que consiste en secuenciar un conjunto de tareas o trabajos que tienen una fecha de entrega común. Cada tarea tiene una penalidad de adelanto si se termina de ejecutar antes de la fecha de entrega común. De igual manera, la tarea tiene una penalidad de retraso si se termina de ejecutar después de la fecha de entrega común. De esta manera, el objetivo es determinar el tiempo de término de cada trabajo y la máquina en donde éste será ejecutado, con el fin de minimizar la suma de penalidades de adelantos y retrasos de todos los trabajos.

La función objetivo cuadrática tiene asociada una matriz hessiana que tiene una estructura particular a bloques. Esta matriz no es semidefinida positiva por lo tanto, la función objetivo no es convexa. Existen diferentes problemas de programación cuadrática, por ejemplo, problemas en los que la función objetivo es convexa, sujeto a restricciones lineales y variables lineales, problemas en los que la función es convexa, sujeto a restricciones lineales y variables binarias, y sus variantes. Lo que se desea es resolver este tipo de problemas.

Si la función objetivo (a ser minimizada) del problema de programación

cuadrática es convexa, sujeta a restricciones lineales y con variables continuas, entonces tiene un punto mínimo global y el problema se resuelve en tiempo polinomial utilizando un método de elipsoide. Ahora bien, si la función objetivo es convexa, sujeta a restricciones lineales pero con variables binarias, el problema es NP-duro. Sin embargo, es posible obtener soluciones óptimas mediante un método de ramificación y acotamiento (B&B, por sus siglas en inglés), realizando relajaciones continuas en cada uno de los nodos. Si el problema es modelado mediante una función objetivo cuadrática no convexa, sujeta a restricciones lineales y variables binarias, como es el caso del problema definido anteriormente, entonces no existe un algoritmo que resuelva este problema en tiempo polinomial. Para resolver este tipo de problemas se pueden utilizar linealizaciones y reformulaciones convexas, entre otros métodos.

Lo que se propone en esta tesis es una nueva reformulación convexa de la función objetivo, es decir, obtener una función equivalente convexa a la función objetivo del problema no convexo, para luego emplear el método de B&B y de esta manera obtener soluciones óptimas para el problema original.

Una reformulación convexa de la función objetivo permite tener una función equivalente convexa a la función objetivo del problema no convexo. Esto quiere decir que en puntos binarios del espacio de solución, la función equivalente tiene el mismo valor objetivo que la función original, lo que permite que si estos puntos son óptimos para la función equivalente entonces también serán puntos óptimos para la función original.

El método que se propone es una nueva reformulación que aprovecha la estructura a bloques de la matriz hessiana, representando a ésta como la suma de las submatrices obtenidas de cada bloque de la matriz hessiana. Se aplica un método para perturbar cada bloque de la matriz hessiana obteniendo, de

esta manera, una función objetivo equivalente convexa y haciendo uso de un método de B&B obtener soluciones óptimas para el problema planteado.

Como se ha mencionado, también existen otros métodos basados en reformulaciones convexas para resolver problemas de programación cuadrática en el que la función objetivo es no convexa. Se estudian tres métodos: el método que utiliza CPLEX [6] para encontrar una función equivalente convexa, el método de mínimo valor propio [17] y el método basado en programación semidefinida [25].

Se realizó un estudio amplio y profundo acerca de los métodos de reformulación convexa para programarlos, probar su funcionamiento y compararlo con el método que se propone en este trabajo de tesis. Como trabajo a futuro, se propone probar el método de programación semidefinida en instancias del problema planteado. Este trabajo sienta las bases para probar el método de programación semidefinida el cual promete resolver instancias más grandes en menos tiempo, ya que la relajación continua es la mejor conocida para los problemas que se tratan en esta tesis.

Este trabajo está estructurado como sigue: en el Capítulo 2 se describen los conceptos teóricos importantes para la realización de esta tesis como conceptos y teoremas acerca de matrices semidefinidas positivas, funciones convexas y programación cuadrática. Luego, en el Capítulo 3 se presentan los cuatro métodos que se utilizaron para resolver el problema de programación no convexa que se tratan en esta tesis. Estos métodos que se mencionan, se basan en la perturbación de la matriz hessiana, mediante diferentes parámetros, para obtener una matriz semidefinida positiva y por tanto hacer que la función objetivo sea convexa. Uno de estos métodos es una de las contribuciones de esta tesis. Este método se basa en la estructura diagonal a bloques de la ma-

triz hessiana. Continuando, en el Capítulo 4, se presentan dos variantes de un problema de planeación justo a tiempo. La primera maneja una fecha de vencimiento común restrictiva y la segunda una fecha holgada. Ambos problemas presentan una función objetivo cuadrática no convexa, las restricciones son lineales y las varia-bles binarias. Se describe en qué consiste cada uno de los problemas. Luego, en el Capítulo 5, se presenta un ejemplo de una instancia del problema de planeación justo a tiempo con fecha restrictiva y los resultados experimentales obtenidos al probar cada uno de los métodos que se presentan en esta tesis. Finalmente, en el Capítulo 6 se presentan las conclusiones que se obtuvieron y el trabajo a futuro que se propone.

Capítulo 2

Teoría matricial y programación cuadrática

En este capítulo se presentan todos los conceptos que se han utilizado en el desarrollo de la tesis. Debido a que este capítulo es muy extenso, ya que se describe una gran cantidad de conceptos y teoremas, se ha divido en dos secciones, cada una con tres subsecciones. En la primera sección se definen conceptos acerca de la teoría de matrices y en la segunda sección, se presentan conceptos de la programación cuadrática.

2.1 Conceptos Básicos

Definición 2.1 *Una* matriz diagonal *es una matriz cuadrada en la que todas sus entradas no diagonales son cero.*

Ejemplo, observe las siguientes matrices de tamaño 3×3 y 4×4 [4]:

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 5 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 9 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{pmatrix}.$$

La matriz A de tamaño 3×3 es una matriz diagonal ya que los elementos de la diagonal principal son diferentes de cero y los elementos restantes son iguales

a cero. En caso contrario, observe que la matriz B de tamaño 4×4 no es una matriz diagonal ya que el elemento de la fila 2 columna 3 es distinto de cero (este elemento es igual a 9).

Este concepto es importante para el trabajo de tesis que se ha realizado ya que la matriz asociada a la función objetivo cuadrática es diagonal a bloques y el método que se propone para resolver problemas de programación cuadrática que tienen una función objetivo no convexa (ver Subsección 2.1.2 para saber más sobre funciones convexas) se basa en la estructura a bloques de la matriz asociada a la función objetivo, ver Subsección 3.2.4.

La *transpuesta de una matriz* [4] A consiste en intercambiar las filas por las columnas y se denota por A^T. Por ejemplo:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ -4 & 2 & 1 \\ 8 & 5 & -7 \end{pmatrix} \quad \Rightarrow \quad A^T = \begin{pmatrix} 1 & -4 & 8 \\ 2 & 2 & 5 \\ 3 & 1 & -7 \end{pmatrix}$$

Propiedad 2.1 [20] *Sean A y B matrices de cualquier tamaño, la transposición de una matriz cumple las siguientes propiedades.*

1.- $(A + B)^T = A^T + B^T$,

2.- $(A^T)^T = A$,

3.- $(kA)^T = kA^T$, *donde k es un escalar,*

4.- $(AB)^T = B^T A^T$, *el número de columnas de A debe ser el número de filas de B para que se pueda realizar la multiplicación de matrices.*

Dada una matriz A, una *submatriz* de A es una matriz obtenida de A al remover cualquier número de filas o columnas de A.

Ejemplo 2.1 *Observe las siguientes matrices:*

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & -1 & -2 & 5 \\ 6 & 0 & 4 & 3 \\ 7 & 9 & 8 & 6 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 2 & 3 \\ 5 & -1 & -2 \\ 6 & 0 & 4 \end{pmatrix} \qquad C = \begin{pmatrix} 6 & 4 \\ 7 & 8 \end{pmatrix}$$

Las matrices B y C son submatrices de A, la submatriz B fue obtenida al remover la fila 4 y la columna 4, la matriz C fue obtenida al remover las filas 1 y 2, y columnas 2 y 4. Si no se remueven filas y columnas se puede decir que A es su misma submatriz.

Se dice que una matriz es particionada si ésta es dividida en submatrices por medio de líneas horizontales y verticales entre filas y columnas. Entonces una matriz puede ser particionada en diferentes formas. Para la matriz A del Ejemplo 2.1, se particiona de dos formas para ejemplificar:

$$A = \left(\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 5 & -1 & -2 & 5 \\ \hline 6 & 0 & 4 & 3 \\ 7 & 9 & 8 & 6 \end{array}\right) \qquad A = \left(\begin{array}{c|ccc} 1 & 2 & 3 & 4 \\ \hline 5 & -1 & -2 & 5 \\ 6 & 0 & 4 & 3 \\ 7 & 9 & 8 & 6 \end{array}\right) \qquad A = \left(\begin{array}{cccc} 1 & 2 & 3 & 4 \\ 5 & -1 & -2 & 5 \\ 6 & 0 & 4 & 3 \\ \hline 7 & 9 & 8 & 6 \end{array}\right)$$

Partición 1 Partición 2 Partición 3

Definición 2.2 *Una matriz diagonal a bloques, es una matriz A que puede ser particionada en la siguiente forma:*

$$A = \begin{pmatrix} A_1 & & & \mathbf{0} \\ & A_2 & & \\ & & A_3 & \\ \mathbf{0} & & & \ddots \\ & & & & A_n \end{pmatrix}$$

donde la partición son las submatrices A_1, A_2, ..., A_n.

Este concepto es de suma importancia ya que la matriz con la que se trabaja en capítulos posteriores es una matriz diagonal a bloques donde cada bloque es una submatriz. Se representa a la matriz hessiana en la suma de sus submatrices para aplicar un método de reformulación convexa para la función objetivo, ver Subsección 3.2.4.

2.1.1 Matrices semidefinidas positivas

Una vez que se han definido algunos conceptos básicos acerca de matrices, ahora el siguiente paso es describir las matrices que son semidefinidas positivas, ya que la matriz asociada a la función objetivo cuadrática que se estudia en esta tesis, además de ser una matriz simétrica, cuadrada, diagonal a bloques, es una matriz que no es semidefinida positiva. En esta sección se describen conceptos importantes acerca de este tipo de matrices, así como algunas propiedades y teoremas asociados a las mismas. Se definen conceptos como: valor propio de una matriz y vectores propios. Estos conceptos serán de gran utilidad en capítulos posteriores ya que unos métodos para resolver algunos problemas de programación cuadrática se basan en estos conceptos.

Definición 2.3 [4] *Un vector $\boldsymbol{x}$ diferente de cero es un vector propio o vector característico de una matriz cuadrada A si existe un escalar λ tal que $A\boldsymbol{x} = \lambda\boldsymbol{x}$. Entonces λ es un valor propio o valor característico de la matriz A.*

Un espacio propio, autoespacio o eigenespacio es el conjunto de vectores propios con un valor propio común.

Dado $\mathbf{x}$ un vector propio de una matriz cuadrada A, entonces existe un valor propio tal que $A\mathbf{x} = \lambda\mathbf{x}$ o lo que es equivalente:

$A\mathbf{x} - \lambda\mathbf{x} = \mathbf{0}$ o $(A - \lambda I)\mathbf{x} = \mathbf{0}$, donde I es la matriz identidad de igual tamaño que A.

Sea B una nueva matriz definida como $B = A - \lambda I$. Entonces se puede escribir $B\mathbf{x} = \mathbf{0}$. Se tiene un sistema homogéneo de ecuaciones. Si B tiene matriz inversa se puede resolver el sistema, obteniendo $\mathbf{x}= B^{-1}\mathbf{0}$ o $\mathbf{x} = \mathbf{0}$. Este resultado es absurdo ya que como se ha mencionado $\mathbf{x}$ es un vector propio por esta razón $\mathbf{x} \neq \mathbf{0}$. $\mathbf{x}$ es un vector propio si y solo si B no tiene inversa. Si una matriz no tiene inversa entonces el determinante es cero, por lo que $\mathbf{x}$ será un vector propio si y solo si:

$$det(A - \lambda I) = 0$$

Esta ecuación es llamada *ecuación característica* de A, también es llamada *polinomio característico* de A[4]. Las raíces de esta ecuación son los valores propios de la matriz A.

Por ejemplo, para obtener los valores propios de $A = \begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix}$ se prosigue de la siguiente manera:

$$(A-\lambda I) = 0 \Rightarrow (A-\lambda I) = \begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix} - \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}$$
$$= \begin{pmatrix} 1-\lambda & 2 \\ 4 & 3-\lambda \end{pmatrix} = (1-\lambda)(3-\lambda) - (2)(4) = \lambda^2 - 4\lambda + 3 - 8 = \lambda^2 - 4\lambda - 5.$$

Entonces, resolviendo para λ la ecuación característica $\lambda^2 - 4\lambda - 5$=0 se encuentran los valores propios de A, $\lambda = -1$ y $\lambda = 5$ hacen que $\det(A - \lambda I) = 0$. Por lo tanto, $\lambda = -1$ y $\lambda = 5$ son los valores propios de la matriz A.

Si los elementos de una matriz son números reales y la matriz es simétrica entonces los valores propios asociados a la matriz serán también números reales.

El siguiente teorema demuestra lo anterior.

Teorema 2.1 [4]. *Los valores propios de una matriz real simétrica son reales.*

La importancia de este resultado es que dos de los métodos que se presentan en esta tesis se basan en perturbar la matriz real con su mínimo valor propio, por lo tanto la matriz perturbada será una matriz real.

Antes de introducir el concepto de matriz semidefinida positiva, primero es necesario saber que una matriz compleja es aquella en la que algunos de sus elementos son números complejos de la forma $a + bi$ donde a y b son números reales. Es necesaria la comprensión de este concepto ya que el objetivo de esta sección es definir una matriz hermitiana para luego definir una matriz semidefinida positiva.

Definición 2.4 [4] *Si $A = [a_{ij}]$ es una matriz de tamaño $n \times m$, $n, m \in \mathbb{N} = \{1, 2, \ldots\}$, entonces la transpuesta compleja conjugada de A es la matriz $m \times n$ $A^H = [\bar{a}_{ij}]$, esta matriz es obtenida, primero, encontrando el conjugado de cada elemento de la matriz A, y luego obteniendo la transpuesta de esta matriz.*

Por ejemplo, tenemos la matriz $A = \begin{pmatrix} 2+i & 3 & -i \\ 1 & 1-i & 2i \end{pmatrix}$, entonces la matriz transpuesta conjugada es: $A^H = \begin{pmatrix} 2-i & 1 \\ 3 & 1+i \\ i & -2i \end{pmatrix}$

Se puede observar que si la matriz es una matriz real entonces la transpuesta conjugada es igual a la matriz original. En este proyecto de investigación se trabaja con matrices reales. Se define la matriz conjugada debido a que la

definición de matriz hermitiana maneja este concepto, pero como ya se mencionó en este proyecto los elementos de todas las matrices con las que se trabaja son números reales.

Sea una A matriz arbitraria de tamaño $m \times n$, donde m, $n \in \mathbb{N}$,

$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$, donde $a_{ij} \in \mathbb{R}$.

Como ya se ha mencionado anteriormente la transpuesta conjugada de una matriz, es la matriz resultante de obtener el conjugado de cada elemento de la matriz. Como los elementos son reales y el conjugado de un número real es el mismo número entonces, al encontrar la matriz conjugada se obtiene la matriz $\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$ que es igual a la matriz A. Ahora, de esta matriz resultante se encuentra su transpuesta y se obtiene la matriz $A^T = \begin{pmatrix} a_{11} & a_{21} & \dots & a_{m1} \\ a_{12} & a_{22} & \dots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{nm} \end{pmatrix}$, donde A^T es una matriz de tamaño $n \times m$. Se puede observar que cuando se tiene una matriz con elementos reales la matriz transpuesta conjugada es igual a la transpuesta de la matriz.

Definición 2.5 [4, 20, 1] *Una matriz hermitiana (o hermítica) es una matriz cuadrada $A = [a_{ij}]$ de elementos complejos que tiene la característica de ser igual a su propia transpuesta conjugada. Es decir, el elemento en la i-ésima fila y j-ésima columna es igual al conjugado del elemento en la j-ésima fila e i-ésima columna, para todos los índices i y j:*

$$a_{ij} = \bar{a}_{ji} \text{ es decir si } A = \bar{A}^H$$

Por ejemplo, observe las siguientes matrices:

$$A = \begin{pmatrix} 1 & 2-i & 4i \\ 2+i & 3 & -1-i \\ -4i & -1+i & 4 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Observe que la matriz A es una matriz hermitiana, ya que al encontrar el conjugado de cada elemento de la matriz A se obtiene la matriz $\begin{pmatrix} 1 & 2+i & -4i \\ 2-i & 3 & -1+i \\ 4i & -1-i & 4 \end{pmatrix}$ y al encontrar la transpuesta de esta matriz resultante se obtiene $A^H = \begin{pmatrix} 2+i & 3 & -1-i \\ -4i & -1+i & 4 \end{pmatrix}$. Como se puede observar $A = A^H$ por lo que es una matriz hermitiana. En el caso, la matriz B no es hermitiana. Primero se obtiene el conjugado de la matriz B. Como B es una matriz real entonces la matriz conjugada de B es la misma matriz B. Ahora la transpuesta de B es $B^H = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. Como se puede observar $B \neq B^H$. Por lo tanto, no es una matriz hermitiana. Las matrices reales simétricas son un caso especial de matrices hermitianas, es decir, cuando los elementos de una matriz hermitiana son reales, entonces se tiene una matriz simétrica.

La operación $< x, y >$ define un producto interno en $\mathbb{C}$, donde $\mathbb{C}$ son los números complejos, $< x, y >= x^T \bar{y}$, donde $\bar{y}$ es el conjugado de y. Ahora si se tiene, por ejemplo, que x es un vector real y A una matriz real simétrica (hermitiana), entonces $< Ax, x >= (Ax)^T \bar{x}$. Como x es un vector real entonces $\bar{x} = x$. Ahora por propiedades de matrices transpuestas, (Propiedad 2.1), $(Ax)^T = x^T A^T$. Como A es una matriz simétrica, entonces $A^T = A$, por lo tanto, $< Ax, x >= (Ax)^T \bar{x} = x^T A^T x = x^T Ax$.

Teorema 2.2 [4, 20, 1]. *Una matriz A es hermitiana si y solo si $< Ax, x >$ es real para todo vector x*

El siguiente teorema es importante ya que, lo que se hace en este proyecto de tesis es perturbar una matriz real con el mínimo valor propio y es necesario que éste número sea real.

Teorema 2.3 [4]. *Dada A una matriz hermitiana, los valores propios de A son números reales.*

Del Teorema 2.3 se sabe que la cantidad $< Ax, x >$ es un número real. Si esta cantidad es no negativa se dice que la matriz es definida no negativa. Si la cantidad es mayor que cero, se dice que la matriz es definida positiva y si esta cantidad es mayor o igual que cero, se dice que la matriz es semidefinida positiva. A continuación se definen formalmente estos conceptos.

Definición 2.6 [4, 20, 1] *Una matriz hermitiana A de tamaño $n \times n$ es definida positiva si $< Ax, x >$ es positiva, es decir, $x^T Ax > 0$ para todo vector real o complejo x no nulo.*

De esta definición se desprende una más: matriz semidefinida positiva. En este proyecto de tesis se trabaja con matrices semidefinidas positivas.

Definición 2.7 [4, 20, 1]*: Sea A una matriz simétrica, se dice que A es semidefinida positiva(SDP) si*

$$x^T Ax \leq 0 \quad \textit{para todo} \quad x \in \Re^n.$$

A continuación se presentan tres teoremas los cuales ayudan a verificar que la matriz hessiana asociada a la función objetivo sea semidefinida positiva. En este trabajo de tesis se utiliza el Teorema 2.4 para verificar que la matriz hessiana sea semidefinida positiva. Este teorema se basa en obtener los valores

propios de la matriz hessiana. Otra manera de chequear que la matriz sea semidefinida positiva es mediante el Teorema 2.5, el cual se basa en obtener las submatrices de la matriz hessiana. El Teorema 2.6 se basa en obtener los menores principales de la matriz hessiana.

Teorema 2.4 [10] *Si Q es semidefinida positiva entonces los valores propios de Q son no negativos.*

Demostración: Sea γ un valor propio de la matriz Q, entonces $Qx = \gamma x$ para algún $x \neq 0$. Entonces, $0 \leq x^T Qx = x^T(\gamma x) = \gamma x^T x$, por lo cual $\gamma \geq 0$. ■

Teorema 2.5 [10] *Si $Q \succeq 0$ (semidefinida positiva) entonces cualquier submatriz principal de Q es semidefinida positiva.*

Teorema 2.6 [10] *Suponga que Q es una matriz simétrica. Entonces Q es semidefinida positiva si y sólo si todos los menores principales de Q son positivos.*

El siguiente teorema prueba que la suma de submatrices semidefinidas positivas es semidefinida positiva. El método que se propone en esta tesis para resolver problemas de programación cuadrática donde la función objetivo no es convexa, las restricciones son lineales y las variables son binarias, se basa en dividir la matriz asociada a la función objetivo en la suma de sus submatrices. Ver Capítulo 3.

Teorema 2.7 *Si Q es semidefinida positiva y P semidefinida positiva entonces $Q + P$ es semidefinida positiva.*

Definición 2.8 *Una matriz A simétrica es llamada diagonalmente dominante si:*

$$a_{ii} \geq \sum_{j \neq i} |a_{ij}|,\ i = 1, 2, \ldots, n.$$

El método que utiliza CPLEX para obtener una reformulación convexa de la función objetivo se basa en el Teorema 2.8.

Teorema 2.8 *Si A es diagonalmente dominante entonces A es semidefinida positiva.*

2.1.2 Funciones convexas

En esta sección se introducen conceptos importantes sobre funciones que en capítulos siguientes se utilizan, como el caso de función convexa. En este trabajo de tesis se trabaja con funciones cuadráticas, es decir, funciones de la forma $f(x) = ax^2 + bx + c$ donde $a, b, c \in \mathbb{R}$ o, en forma matricial, se tiene $f(x) = \frac{1}{2}x^T Qx + c^T x$. El objetivo es optimizar la función cuadrática.

Primeramente se definen conceptos como conjunto convexo, función y algunos conceptos relacionados con éste. Luego se define el concepto de función convexa que es el objetivo de esta sección. También se presentan algunos teoremas que se utilizaron en este proyecto. Por ejemplo, el más importante es el que dice que, si una matriz es semidefinida positiva entonces la función asociada a la función objetivo es convexa.

Definición 2.9 [26, 20] *Un subconjunto C de $\mathbb{R}$ es llamado convexo si cumple que $(1 - \lambda)x + \lambda y \in C$ para cualquier $x, y \in C$ $0 < \lambda < 1$.*

La definición se refiere a que si se tienen dos puntos en C, sean x y y, si se unen con una línea recta, esta línea está dentro del conjunto C. Como se puede observar en la Figura 2.1 a), se tienen un conjunto convexo ya que si se toman cualesquier pareja de puntos y se unen con una línea recta, esta línea

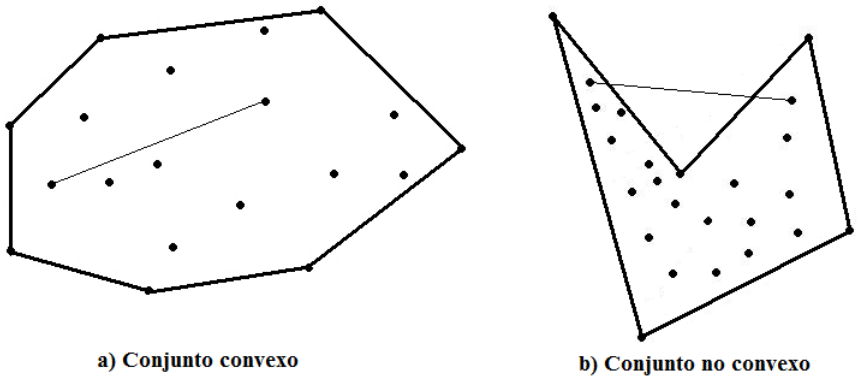


Figura 2.1: Ejemplos de convexidad de conjuntos.

está dentro del mismo conjunto. De modo contrario, en la Figura 2.1 b), la línea que une a dos puntos no está dentro del conjunto, por tanto no es un conjunto convexo.

Definición 2.10 [21] *Una función* $f(x) : \Re^n \longrightarrow \Re$ *es una función convexa si:*

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y) \tag{2.1}$$

para todo $x, y \in \Re^n$*, para todo* $\lambda \in [0,1]$*. Si la desigualdad es estricta para todo* $x \neq y$ *y* $\lambda \in (0,1)$*, se dice que la función es estrictamente convexa.*

Ahora, si la desigualdad es de mayor o igual en la Ecuación (2.1) se dice que la función es cóncava, como se define a continuación:

Definición 2.11 *Una función* $f(x) : \Re^n \longrightarrow \Re$ *es una función cóncava si:*

$$f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda)f(y)$$

Para todo $x, y \in \Re^n$*, para todo* $\lambda \in [0,1]$*. Si la desigualdad es estricta para todo* $x \neq y$ *y* $\lambda \in (0,1)$*, se dice que la función es estrictamente cóncava.*

La Figura 2.2 hace referencia a una función convexa en donde se puede observar que si se toman cualesquiera dos puntos en la función y se unen con una

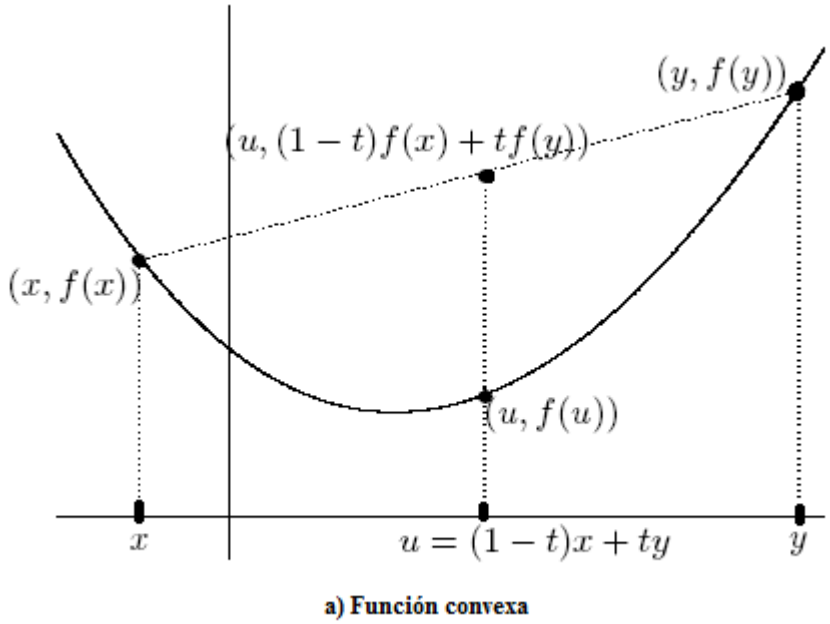


Figura 2.2: Ejemplo de función convexa.

línea recta, esta línea siempre está por arriba de la función objetivo, o lo que es lo mismo, la porción de función entre estos dos puntos siempre está por debajo de la línea recta.

Las funciones convexas aparecen muy frecuentemente en funciones objetivo de problemas de optimización. Por ejemplo, en un problema de programación lineal, donde la función es de la forma $f(x) = c^T x + d$ (c es un vector de costos y x es el vector de variables), en un problema de programación cuadrática, donde la función objetivo es de la forma $\frac{1}{2}x^T Q x + c^T x$ (c es un vector de costos, x es el vector de variables y Q es la matriz hessiana) y Q es una matriz semidefinida positiva, entre otras funciones.

Los problemas en los que la función objetivo es convexa y las restricciones forman un conjunto convexo son llamados *problemas convexos*. Estos problemas tienen numerosas propiedades teóricas importantes, por ejemplo, si el problema

de optimización tiene como objetivo minimizar, entonces garantizan la existencia de un mínimo.

El siguiente teorema garantiza que si la función es convexa entonces tiene un mínimo. Ahora si tenemos un problema de maximización solo hay que comprobar que la función sea cóncava para garantizar que existe un punto máximo.

Teorema 2.9 [10] *Suponga que Q es semidefinida positiva. La función $f(x) = \frac{1}{2}x^TQx + c^Tx$ tiene un mínimo en x^* si y solo si x^* es una solución de la siguiente ecuación:*

$$\nabla f(x) = Qx + c = 0.$$

Hasta el momento se ha definido función convexa. En este trabajo de investigación se trabaja una función cuadrática que no es convexa. Lo que se desea es que esta función cuadrática sea convexa. En el Capítulo 3 se explican algunos métodos para convexificar la función. Toda función cuadrática tiene asociada una matriz hessiana, los elementos de esta matriz son los coeficientes del término no lineal de la función cuadrática. Si esta matriz es semidefinida positiva entonces la función es convexa como se ve en el Teorema 2.10.

Teorema 2.10 [10] *La función $f(x) = \frac{1}{2}x^TQx + c^Tx$ es una función convexa si y solo si Q es semidefinida positiva.*

Demostración: Suponga que Q no es una matriz semidefinida positiva, entonces existe r tal que $r^TQr < 0$. Sea $x = \theta r$, entonces $f(x) = f(\theta r) = \frac{1}{2}\theta^2 rQr + \theta c^Tr$ es estrictamente cóncava en el subconjunto $\{x \,|x = \theta r\}$, ya que $r^TQr < 0$. En consecuencia, $f(x)$ no es una función convexa.

Ahora, suponga que Q es semidefinida positiva, entonces para $\forall\ \lambda \in [0, 1]$ y $\forall$ x, y se cumple que:

$$\begin{aligned} f(\lambda x + (1-\lambda)) &= f\left(y + (1-\lambda)\right) \\ &= \tfrac{1}{2}(y + \lambda(x+y))^T Q(y + \lambda(x-y)) + c^T(y + \lambda(x-y)) \\ &= \tfrac{1}{2}y^T Qy + \lambda(x-y)^T Qy + \tfrac{1}{2}\lambda^2(x-y)^T Q(x-y) + \lambda c^t x \\ &\quad +(1-\lambda)c^T y \\ &\leq \tfrac{1}{2}y^T Qy + \lambda(x-y)^T Qy + \tfrac{1}{2}\lambda(x-y)^T Q(x-y) + \lambda c^t x \\ &\quad +(1-\lambda)c^T y \\ &= \tfrac{1}{2}\lambda x^T Qx + \tfrac{1}{2}(1-\lambda)y^T Qy + \lambda c^T x + (1-\lambda)c^T y \\ &= \lambda f(x) + (1-\lambda)f(y), \end{aligned}$$

Entonces esto muestra que $f(x)$ es una función convexa. ■

Los métodos para convexificar funciones cuadráticas, que en este trabajo de tesis se presentan, se basan en hacer semidefinida positiva la matriz hessiana y por el Teorema 2.10 la función objetivo es convexa. Luego se aplica un B&B para obtener soluciones óptimas. Esta metodología se presentará en capítulos posteriores.

2.2 Programación cuadrática

Muchos problemas que aparecen en la vida cotidiana pueden ser representados mediante un modelo matemático analizando el comportamiento del problema o bien predecir su comportamiento futuro, de tal manera que se hacen las suposiciones y restricciones necesarias para representar las limitaciones que tiene el problema que se quiere modelar. En muchos casos se pueden utilizar modelos matemáticos que, mediante letras, números y operaciones, representan variables, magnitudes y sus relaciones.

2.2.1 Conceptos básicos

Un modelo matemático consta al menos de tres conjuntos básicos de elementos [19]:

- Variables de decisión que son incógnitas que deben ser determinadas a partir de la solución del modelo y de los parámetros que representan valores conocidos del problema o bien que se pueden controlar.
- Restricciones que son limitaciones asociadas al problema y son relaciones entre las variables de decisión y magnitudes que dan sentido a la solución del problema. Por ejemplo, si se están elaborando juguetes es evidente que no se pueden fabricar -3 juguetes. Entonces aquí se tendría una restricción de no negatividad.
- La función objetivo es una relación matemática entre las variables de decisión, parámetros y una magnitud que representa el objetivo del problema. Por ejemplo si el objetivo de un problema es minimizar los costos de operación, la función objetivo debe expresar la relación entre el costo y las variables de decisión.

Un problema de optimización consiste en encontrar la mejor solución que minimice (costos, tiempo, error, etc) o maximice (ganancias, producción, eficiencia, etc.) una función objetivo, es decir, encontrar una solución que satisfaga todas las restricciones y además que el valor de la función objetivo sea mínimo o máximo dependiendo de lo que se requiera en el modelo asociado al problema de optimización. Si la función objetivo asociada al problema de optimización es lineal, con restricciones lineales y variables continuas, entonces se tiene un problema de programación lineal.

La programación no lineal es el proceso de resolución de un sistema de igualdades y desigualdades sujetas a un conjunto de restricciones sobre un conjunto de variables reales desconocidas, con una función objetivo a maximizar, cuando

alguna de las restricciones o la función objetivo son no lineales. Si la función objetivo es modelada mediante una función cuadrática, con restricciones lineales y variables continuas entonces se tiene un problema de programación cuadrática que es un caso particular de optimización no lineal.

2.2.2 Modelación matemática

Un problema de programación cuadrática consiste en optimizar una función objetivo cuadrática compuesta de un conjunto de variables de decisión continuas sujeto a restricciones lineales. Se puede modelar matemáticamente el problema cuadrático como sigue (en su forma estándar) [8, 21, 10]:

$$\begin{aligned} \text{minimizar} \quad & f(x) = \tfrac{1}{2}x^T Q x + c^T x \\ \text{sujeto a:} \quad & Ax = b \\ & x \in \mathbb{R}^n \end{aligned} \tag{2.2}$$

En el modelo (2.2) x es el vector que está compuesto de las variables de decisión, Q es la matriz hessiana asociada a la función objetivo, donde los elementos que la componen son los coeficientes del término cuadrático de la función, A es una matriz que representa a las restricciones asociadas al problema de optimización y cuyos elementos son números reales, c es el vector de costos lineales y b es un vector cuyos elementos son números reales.

Los problemas de programación cuadrática pueden ser clasificados de la siguiente manera, basados en la estructura de su modelo matemático (2.2) [8]:

- Problemas bilineales: se presentan cuando en la matriz Q existen dos subvectores de distintas variables y y z de x tal que, el problema es lineal cuando uno de estos vectores se fija.

- Problemas cuadráticos cóncavos o convexos: se presentan cuando la matriz Q es semidefinida negativa o bien cuando tenemos una función cóncava o convexa. En este trabajo de tesis se estudian este tipo de problemas.

- Problemas cuadráticos indefinidos: aparecen cuando la matriz Q tiene valores propios tanto positivos como negativos. Desde el punto de vista de solución, estos problemas son los más intratables.

Muchos problemas de optimización son modelados mediante programación cuadrática como por ejemplo, el problema de asignación cuadrática [19], asignación de tareas [19], problemas de planeación justo a tiempo [25], problemas de diseño territorial [28], etc. Otras aplicaciones de la programación cuadrática son, por ejemplo, en el área financiera. Se pueden realizar análisis, usando modelos de programación cuadrática para determinar la selección de estrategias óptimas de inversión [19]. En el área de economía, se utilizan modelos de equilibrio para analizar expectativas de cambio en condiciones económicas, predicción de precios e incremento de la inflación [19].

2.2.3 Antecedentes

El modelo de un problema de programación cuadrática está compuesto por una función cuadrática, sujeto a restricciones lineales y variables reales, ver modelo (2.2). En esta sección se mencionan los métodos que se han desarrollado para la solución de estos modelos.

La función objetivo tiene asociada una matriz hessiana, cuyos elementos son los coeficientes del término no lineal de la función. Si esta matriz es semidefinida positiva por el Teorema 2.10 la función es convexa por lo que se convierte en un problema de programación convexa. Esto implica que cualquier óptimo local es equivalente a tener un óptimo global en problemas convexos [21] y se

puede encontrar este punto mediante numerosos algoritmos. En particular, si la función es convexa se pueden aplicar algoritmos que resuelven el problema en tiempo polinomial, como por ejemplo, el algoritmo de elipsoide de Khachiyan ó métodos de puntos interiores [25, 23, 8].

En la Sección 2.2.1 se menciona que los problemas de programación cuadrática se pueden clasificar en problemas bilineales, convexos e indefinidos. Estos problemas pueden ser resueltos de diferentes maneras. Por ejemplo, para los problemas bilineales, se han propuesto algoritmos de corte polar, reescriben el problema como un problema de max-min y se propone un algoritmo de ramificación y acotamiento para resolver este problema [8]. Para problemas convexos se aplican métodos de puntos extremos, métodos de corte de plano, reducción a un problema bilineal, etc. Para problemas cuadráticos indefinidos se pueden aplicar técnicas de descomposición [8].

Un modelo de programación cuadrática, donde la función es convexa (Q es semidefinida positiva), sujeta a restricciones lineales y variables reales, es un problema fácil de resolver, es decir, existe un algoritmo que resuelve este tipo de problemas en un tiempo polinomial (tiempo razonable), como por ejemplo, el método de elipsoide [23].

El dual asociado al problema de programación cuadrática está dado por [8]:

$$\begin{aligned} \text{maximizar} \quad & d(x,y) = b^T y - \tfrac{1}{2} x^T Q x \\ \text{sujeto a:} \quad & A^T y - Qx + s = c \\ & s \geq 0 \end{aligned} \tag{2.3}$$

donde $y \in \mathbb{R}^m$.

Otros algoritmos para resolver problemas de programación cuadrática se basan

en el dual asociado al problema de programación cuadrática, modelo (2.3). Por ejemplo, métodos de puntos interiores, métodos de gradiente conjugado, métodos de barrera logarítmica para problemas de programación cuadrática, etc. [8]. Por ejemplo, el método de barrera logarítmica [8] se basa en que la convergencia cuadrática está en una vecindad del camino central. Este método es usado para determinar cotas superiores para la diferencias entre la función de barrera y el valor objetivo entre un punto en el camino y un punto de su vecindad. Asume tres condiciones para poder aplicarlo a un problema de programación cuadrática. La primera condición es que la función sea convexa con derivadas continuas de primer y segundo orden. La segunda que la región factible sea no vacía, y tercera, que la región factible sea acotada. Además, este algoritmo realiza un número de iteraciones que es polinomial para mayores detalles veáse [8].

Se ha mencionado que un problema de programación cuadrática convexo es fácil de resolver. Ahora bien, si la matriz asociada a la función objetivo es no convexa (Q es definida negativa), el problema es NP-duro, es decir, no existe un algoritmo que lo resuelva en tiempo polinomial. Esto fue demostrado por Sahi [27] en 1974.

En este proyecto de investigación se trabaja con un modelo de programación cuadrática el cual tiene una función objetivo no convexa con variables binarias. Este problema se puede modelar en forma general de la siguiente manera:

$$\begin{aligned} \text{minimizar} \quad & f(x) = \tfrac{1}{2}x^T Q x + c^T x \\ \text{sujeto a:} \quad & Ax = b \\ & x \in \{0,1\}^n \end{aligned} \tag{2.4}$$

El problema (2.4) es NP-duro [27].

Cuando la función objetivo de un problema de programación cuadrática es

convexa, las restricciones son lineales y las variables son binarias, se pueden encontrar soluciones óptimas mediante un método de ramificación y acotamiento basado en relajaciones continuas.

Existen otros métodos para resolver problemas de programación cuadrática en donde las variables de decisión son binarias, por ejemplo: linealizaciones, relajaciones, descomposición, etc. [25]. Como ya se mencionó antes, este trabajo de tesis está basado en realizar reformulaciones convexas, para aprovechar la estructura del modelo de programación cuadrática con variables binarias.

En el Capítulo 3, se definen métodos existentes basados en convexificar la función objetivo, es decir, hacer que la matriz Q asociada a la función objetivo sea semidefinida positiva. También se presenta una nueva reformulación convexa, basada en la estructura diagonal a bloques de la matriz hessiana (en el caso en que se presente). Se presentan cuatro métodos:

- Método que utiliza CPLEX.
- Método de mínimo valor propio.
- Método basado en programación semidefinida
- Método basado en la estructura diagonal a bloques de la matriz hessiana.

Éste último es la contribución de esta tesis.

Capítulo 3

Reformulaciones convexas

En el capítulo anterior se mencionaron algunos antecedentes de problemas que se modelan mediante programación cuadrática y los métodos que existen para resolver este tipo de problemas. El problema que se estudia en esta investigación es un problema cuadrático binario (el valor de las variables de decisión es cero o uno) no convexo. El método que se emplea para resolverlos consiste en hacer que la función objetivo sea convexa. Este tipo de métodos son llamados métodos de reformulaciones convexas. El objetivo de este capítulo es describir en qué consisten algunos de ellos.

Primeramente, se define brevemente en qué consiste una reformulación convexa en la Sección 3.1. Luego, en la Sección 3.2 se presentan los cuatro métodos que se utilizan para resolver el problema de programación no convexa que se trata en esta tesis. Estos métodos se basan en la perturbación de la matriz hessiana, mediante diferentes parámetros, para obtener una matriz semidefinida positiva y por tanto hacer que la función objetivo sea convexa. Luego, mediante un método de ramificación y acotamiento se pueden encontrar soluciones óptimas al problema de programación cuadrática no convexo con variables binarias.

En este capítulo se presenta el método para resolver problemas cuadráticos no convexos que se propone en esta tesis. Este método se basá en la estructura diagonal a bloques de la matriz hessiana, si es que se presenta el caso.

3.1 Conceptos básicos

El concepto de reformulación ha tomado un rol importante dentro de la optimización, como por ejemplo en programación lineal, cuando la función objetivo es modelada mediante una función del tipo min max, funciones lineales en trozos, o en programación cuadrática, cuando la función es no lineal, no convexa. [13].

Esta sección se enfoca en el caso de programación cuadrática en donde se tiene una función cudrática no convexa en variables binarias. Una reformulación convexa de dicha función objetivo permite tener una función equivalente (tal que ésta sea convexa) a la función objetivo del problema no convexo. Esto quiere decir que, en puntos binarios del espacio de solución, la función equivalente tiene el mismo valor objetivo que la función original. Por lo tanto, si unas soluciones son óptimas para la función equivalente entonces también son óptimas para la función original. Sin embargo, para soluciones reales, los valores de las reformulaciones convexas ya no coinciden con los de la función original.

A continuación se muestra un ejemplo sencillo. Si tenemos la función cuadrática $f(x) = x^2 - 2x - 2$ y sea $x \in \{0, 1\}$. Una función equivalente a $f(x)$ es $g(x) = -x^2 - 2$ ya que cuando $x = 0$, $f(x) = g(x) = -2$. Cuando $x = 1$, $f(x) = g(x) = -3$, por lo que se concluye que en puntos binarios las funciones son iguales y por lo tanto $f(x)$ y $g(x)$ son funciones equivalentes. Ahora, si $x = 0.5$, $f(0.5) = -2.75 \neq -2.25 = g(0.5)$, se puede observar que cuando la variable x toma valores que no son binarios las funciones no son iguales. La Figura 3.1 muestra las funciones $f(x)$ y $g(x)$.

Existen diferentes reformulaciones convexas para la función objetivo, por lo que es interesante encontrar buenas reformulaciones convexas, es decir, que las relajaciones continuas sean buenas. Las buenas relajaciones son las que simpli-

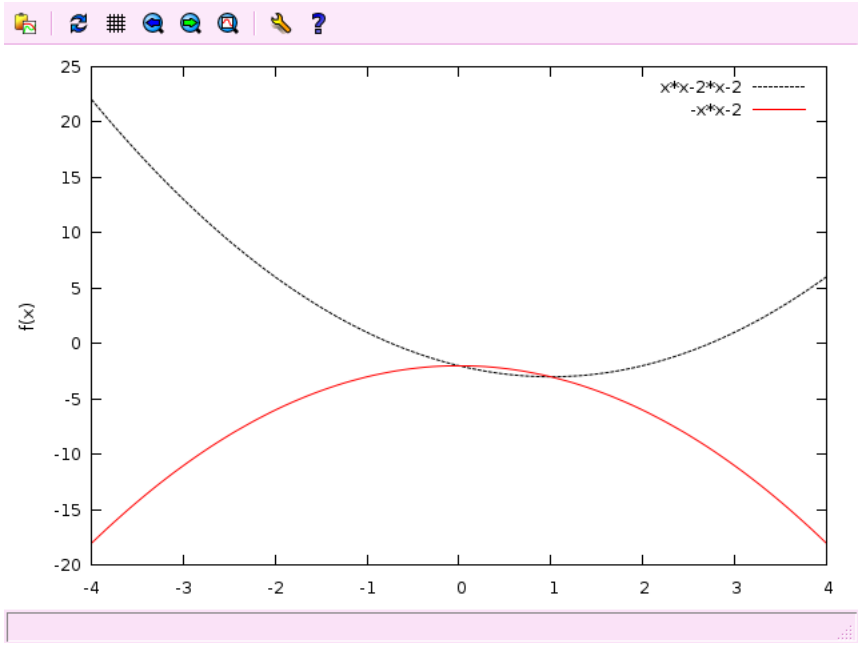


Figura 3.1: Ejemplo de función equivalentes.

fican el problema y hacen más eficiente los procedimientos de solución y cuya resolución proporciona una muy próxima a la solución del problema original.

En la siguiente sección se mencionan algunos métodos para obtener funciones equivalentes a funciones cuadráticas. Se describe, a grandes rasgos, cómo funcionan estos métodos ya existentes. Sólo se entrará en detalle en el método que se propone en esta tesis que aprovecha la estructura a bloques de la matriz hessiana.

3.2 Métodos de reformulaciones convexas

Existen varios métodos de reformulaciones convexas basados en la perturbación de la matriz hessiana. En esta sección se estudian: el método que utiliza CPLEX para convexificar la función objetivo, el método basado en la perturbación de la matriz hessiana mediante su mínimo valor propio y el método basado en programación semidefinida. El primer método se explica en la Subsección

3.2.1. Este método aprovecha el hecho de que la matriz hessiana sea simétrica diagonal dominante. El segundo método viene explicado en la Subsección 3.2.2. Este método es muy sencillo de aplicar, pero su relajación continua no es buena, lo que causa que el tiempo de ejecución en el B&B sea muy grande, mientras que el tercero se trata en la Subsección 3.2.3. Este método consume mucho tiempo para calcular los parámetros con los que se perturba la matriz hessiana, pero al aplicar el B&B, el tiempo es muy corto ya que la reformulación convexa que se obtiene al utilizar este método, es tal que su relajación continua es lo más cercana posible a la relajación continua de la función original. El método que se propone en esta tesis es una nueva reformulación que aprovecha la estructura diagonal a bloques de la matriz hessiana, el cual será definido en la Subsección 3.2.4.

Otros métodos de reformulaciones convexas han sido desarrollados por Plateau, Billionnet y Elloumi. [24] y Billionnet, Elloumi y Plateau [2] donde combinan programación semidefinida y programación cuadrática entera mixta. En estos diferentes trabajos, el problema inicial es reformulado en un problema equivalente en variables 0-1 con una función objetivo cuadrática convexa.

3.2.1 Método que utiliza CPLEX

Existe software para resolver problemas de programación cuadrática como por ejemplo: EXCEL, LINGO, LINDO, CPLEX [19]. En este proyecto de investigación también se utilizó uno de estos software para resolver el problema de programación cuadrática que presentamos en el Capítulo 4: CPLEX [6].

Cuando se utiliza CPLEX para resolver este tipo de problemas se tienen que seguir los siguientes pasos [6]:

1.- Escribir el modelo de optimización, utilizando una instancia, en el formato que requiere CPLEX, el cual es:

Minimizar

obj: Aquí escribir la función objetivo del problema. Se escribe el término cuadrático (va escrito entre [ ]/2) más el término lineal.

Subject to:

c1: Aquí va una restricción, las restricciones deben de ser de menor o igual y además del lado derecho de la desigualdad solo debe haber números.

c2: Aquí va otra restricción.

$\vdots$

cn: Aquí va la restricción número n

Bounds

Aquí van las cotas para las variables de decisión, si es que las hay

General, Integer o Binary (dependiendo del tipo de variable que sea)

Aquí van las variables.

End

2.- Cargar el archivo a CPLEX con la instrucción read.

3.- Seleccionar el tipo de modelo, en este caso es un modelo de programación cuadrática, así es que se escribe lo siguiente: *change problem qp*.

4.- Optimizar el problema con la instrucción *optimize*.

El método que utiliza CPLEX para optimizar un problema de programación cuadrática entera mixta consiste en, primero, verificar si la matriz hessiana es semidefinida positiva. Si lo es, comienza la optimización del problema de programación cuadrática 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, CPLEX convexifica, es decir, modifica la matriz hessiana para hacerla semidefinida positiva. Se basa en el hecho de que la matriz Q sea simétrica diagonal dominante (ver Definición 2.8 del Capítulo 2). Una matriz simétrica de diagonal dominante es semidefinida positiva por el Teorema 2.8. CPLEX modifica la matriz hessiana agregando el elemento de la diagonal a los otros elementos

de la fila i y se puede obtener un cambio en la diagonal que va a hacer que Q sea semidefinida positiva. Algunas heurísticas se aplican luego para reducir esta cantidad si es posible.

Como ya se ha mencionado, se utilizó el método de convexificación de CPLEX para resolver varios problemas de planeación justo a tiempo (ver Capítulo 4) para encontrar soluciones óptimas. En el Capítulo 5 se presentan a detalle los resultados que se obtuvieron.

3.2.2 Método basado en el mínimo valor propio

Este método fue propuesto por Hammer y Rubin [17] en 1970 y está basado en perturbar la diagonal de la matriz hessiana con el mínimo valor propio de ésta, es decir, a cada elemento de la diagonal se le resta el mínimo valor propio de la matriz hessiana [17]. Esto garantiza que la matriz hessiana sea una matriz semidefinida positiva y, por el Teorema 2.10 del Capítulo 2, la función objetivo será convexa.

De la formulación de un problema de programación cuadrática, ver Ecuación (2.2), si la función objetivo $f(x) = \frac{1}{2}x^TQx + c^Tx$ no es convexa, es decir, la matriz Q no es semidefinida positiva, $Q \succeq 0$, se puede, mediante el método de mínimo valor propio, hacer que la matriz Q sea semidefinida positiva. El procedimiento de este método es el siguiente:

1.- Obtener el mínimo valor propio λ de la matriz hessiana.

2.- Obtener una reformulación convexa de la función objetivo perturbando la matriz hessiana con el parámetro λ. Se resta λ a los elementos de la diagonal de la matriz, $Q - Diag(\lambda)$. Suponiendo que las variables en la función objetivo son binarias, si se resta λ a los elementos de la matriz hessiana, entonces, se tiene que sumar para mantener un equilibrio. Las

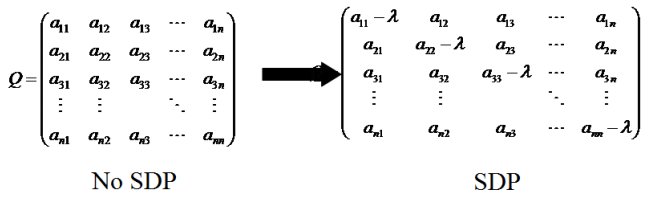


Figura 3.2: Método de mínimo valor propio.

variables son binarias, es decir, $x = 0$ ó $x = 1$, si $x = 1$, $x^2 = 1 = x$, entonces, se suma λ a cada una de las variables lineales en la función objetivo, para mantener un equilibrio. Matemáticamente se tiene que:

$$f(x) = \frac{1}{2}x^T(Q - Diag(\lambda))x + (c + \lambda)^T x \tag{3.1}$$

3.- Ahora la matriz perturbada es semidefinida positiva y por el Teorema 2.10 se tiene una función equivalente convexa a la función original. Se procede a utilizar un B&B para obtener soluciones óptimas a la función equivalente, por lo tanto estas soluciones también son óptimas para la función original.

La Figura 3.2 muestra una matriz que no es semidefinida positiva (No SDP) y perturbando la diagonal de la matriz hessiana con el mínimo valor propio de ésta se obtiene una matriz semidefinida positiva (SDP).

La reformulación convexa que resulta usando este método tarda demasiado tiempo en converger a una solución óptima cuando se aplica el B&B. Esto se debe que la calidad de la cota inferior obtenida mediante la relajación continua del problema reformulado no es buena para problemas de planeación justo a tiempo con penalidades de adelanto y retraso. Sin embargo el tiempo para obtener esta reformulación convexa es despreciable (ya que solo se requiere calcular el mínimo valor propio de la matriz). Estos comportamientos se muestran en algunos resultados obtenidos, aplicando este método a un problema de planeación justo a tiempo en el Capítulo 5.

3.2.3 Método basado en programación semidefinida

La programación semidefinida (SDP, por sus siglas en inglés) es un caso especial de la programación convexa. La idea general de la programación semidefinida [7] 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.

Los problemas de programación semidefinida son interesantes debido a que existen importantes aplicaciones en optimización combinatoria, teoría de aproximación [12], teoría de sistemas de control [12], ingeniería mecánica y eléctrica [12], etc. Algunas aplicaciones en optimización combinatoria son: función de Lovász [7], problemas de max-cut y extensiones [7] y problemas de planeación justo a tiempo [25].

La programación semidefinida minimiza o maximiza una función objetivo lineal con restricciones lineales de igualdad, con la condición de que haya una matriz semidefinida positiva [12]:

$$SDP : \begin{cases} \text{minimizar} & \sum_{i=1}^{m} c_i x_i \\ \text{sujeto a} & X = \sum_{i=1}^{m} F_i x_i - F_0 \\ & X \succeq 0 \end{cases} \tag{3.2}$$

Un aspecto muy importante de la programación semidefinida es que estos problemas pueden ser resueltos en tiempo polinomial mediante algoritmos de puntos interiores [7]. Algunos algoritmos que pueden resolver problemas de programación semidefinida son: algoritmos de funciones de barrera, métodos primal dual, métodos de escala afín (*affine-scaling methods*), métodos de reducción potencial primal-dual, métodos de inicio infactible (infeasible-start methods), métodos que solo utilizan información del gradiente, etc. [7].

Existen en la actualidad, softwares que resuelven problemas de programación semidefinida, por ejemplo: SDPA y CSDP. SDPA [11] es un algoritmo de programación semidefinida basado en el método de punto interior primal-dual, trabaja con la forma estándar de un programa semidefinido y su dual. CSDP [3, 18] maneja un algoritmo que es un predictor-corrector primal-dual de barrera, además puede trabajar con matrices generales simétricas y matrices con una estructura a bloques.

En este trabajo de tesis se utilizó el software CSDP para resolver problemas de programación semidefinida, el cual utiliza el método de punto interior primal-dual [18]. Además, se realizó el programa semidefinido asociado al modelo de un problema de planeación justo a tiempo con penalidades de adelanto y retraso para una instancia pequeña. Se deja como trabajo a futuro la implementación de este método, ver Capítulo 5.

Esta reformulación convexa [2] busca encontrar que la relajación continua sea la mejor cota inferior y así disminuir el tiempo de cómputo en el método de ramificación y acotamiento. A diferencia del método de mínimo valor propio, en este método se perturban todos los elementos de la matriz hessiana mediante parámetros encontrados, que son los valores duales de las restricciones del programa semidefinido, mediante la resolución de un programa semidefinido asociado al problema de programación cuadrática.

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 cercana a la función objetivo original. Ver Figura 3.3.

Como ya se ha mencionado a lo largo de los capítulos de esta tesis, en un

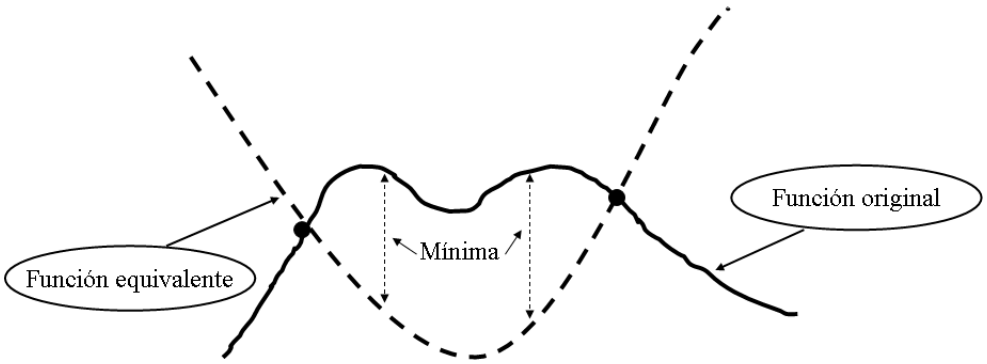


Figura 3.3: Ejemplo de función equivalentes

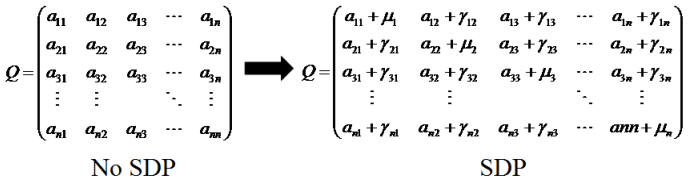


Figura 3.4: Método de programación semidefinida.

problema de programación cuadrática binaria no convexa, se tiene una matriz Q asociada a la función objetivo la cual no es semidefinida positiva. Al aplicar este método lo que se obtiene son parámetros con los cuales se perturba la matriz hessiana y de esta manera se obtiene una matriz semidefinida positiva. Por lo tanto se tiene ahora una función equivalente convexa a la función original, como se muestra en la Figura 3.4.

El método consta de los siguientes pasos:

1.- Obtener el programa semidefinido asociado al problema de programación cuadrática mediante cálculos algebráicos.

2.- Utilizar CSDP para obtener los parámetros con los que se perturban cada uno de los elementos de la matriz hessiana y de esta manera obtener una

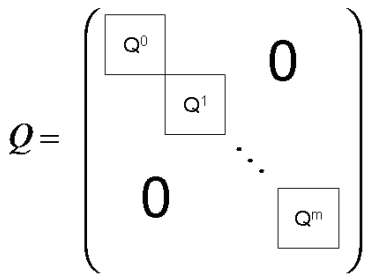


Figura 3.5: Matriz hessiana con estructura a bloques.

matriz semidefinida positiva.

3.- Aplicar un método de B&B para obtener soluciones óptimas de la función equivalente.

En este trabajo de tesis se realizó el paso 1, ver Capítulo 5. Los pasos 2 y 3 del método se dejan como trabajo a futuro.

3.2.4 Método basado en la estructura diagonal a bloques de la matriz hessiana

En esta Subsección se describe el método propuesto para convexificar funciones objetivo cuadráticas. Este método está basado principalmente en la estructura de la matriz hessiana.

Para algunos problemas de programación cuadrática, la matriz hessiana presenta una estructura diagonal a bloques, como se muestra en la Figura 3.5. Esta estructura diagonal a bloques se presenta en problemas de planeación justo a tiempo [25], en donde se desea secuenciar un conjunto de tareas en máquinas paralelas, en una formulación de un problema de diseño territorial [28], etc.

Si se cuenta con un problema en el que la función objetivo sea cuadrática y además la matriz asociada a la función objetivo presenta una estructura dia-

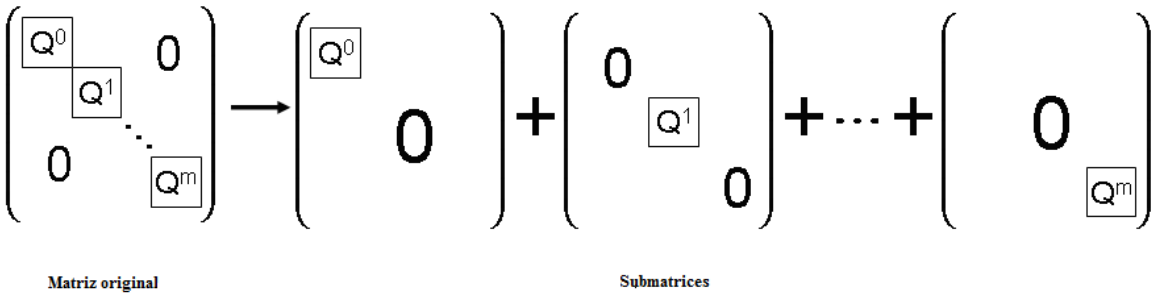


Figura 3.6: Descomposición de la matriz hessiana en submatrices.

gonal a bloques, se puede aplicar el método que se propone en esta tesis: el método basado en la estructura diagonal a bloques de la matriz hessiana.

En este método, se aprovecha esta estructura, separando la matriz original en submatrices que representan a cada bloque, tal que la suma de estas submatrices den como resultado la matriz original, como se muestra en la Figura 3.6. Esto puede realizarse ya que toda matriz puede descomponerse en submatrices (ver Capítulo 2). En este caso cada submatriz representa a un bloque de la matriz original. Luego, sumando cada submatriz se obtiene la matriz original. Si cada una de estas submatrices es semidefinida positiva, por el Teorema 2.7, entonces la suma de estas matrices da como resultado una matriz semidefinida positiva.

El objetivo de este método es obtener una matriz semidefinida positiva, para garantizar la convexidad de la función objetivo. Este método hace semidefinida positiva cada submatriz de la matriz original mediante dos métodos de convexificación mencionados en subsecciones anteriores: método del mínimo valor propio y método de programación semidefinida.

Utilizando el método de valor propio o el método de programación semidefinida se obtienen los parámetros con los que se perturban cada una de la subma-

trices. Estos parámetros hacen que las submatrices se vuelvan semidefinidas positivas y, por el Teorema 2.7, la matriz original es semidefinida positiva y, por el Teorema 2.10 la función objetivo es convexa.

Los pasos a seguir en este método son:

1.- Descomponer la matriz hessiana en submatrices.

1.- Obtener el mínimo valor propio de cada submatriz.

2.- Obtener una reformulación convexa de la función objetivo perturbando cada bloque de la matriz hessiana con el mínimo valor propio que le corresponda a cada submatriz.

3.- Utilizar un método de ramificación y acotamiento para obtener soluciones óptimas de la función equivalente y por lo tanto óptimas para la función original.

El objetivo principal de hacer semidefinida positiva cada submatriz de la matriz hessiana es disminuir el tiempo de cómputo en el método de ramificación y acotamiento. Además, el tiempo de cómputo al hacer semidefinida cada submatriz es menor que al hacer semidefinida toda la matriz.

Hasta donde se tiene conocimiento este es el primer trabajo en donde se aprovecha la estructura de la matriz hessiana para proponer nuevas reformulaciones convexas de la función objetivo. En el Capítulo 4 se presenta un problema de planeación justo a tiempo en donde la función objetivo cuadrática no es convexa. Además las variables de decisión son binarias y la matriz hessiana presenta una estructura diagonal a bloques, por lo que se utiliza la reformulación que se propone en esta tesis, es decir, se aprovecha la estructura a bloques que presenta la matriz hessiana en la función objetivo y además se utiliza tanto el método de mínimo valor propio como el método de programación semidefinida para hacer semidefinida cada una de las submatrices. En el capítulo 5

se presentan los resultados experimentales obtenidos al aplicar este método combinado con el método de mínimo valor propio.

Capítulo 4

Aplicación: planeación justo a tiempo

A lo largo de los capítulos en esta tesis, se ha hablado de cómo resolver problemas de programación cuadrática binaria no convexa. En el capítulo anterior se presentaron métodos para resolver problemas de programación cuadrática binaria no convexa. En este capítulo se presenta un pro-blema de planeación justo a tiempo, en el que la función objetivo es cuadrática no convexa, las restricciones son lineales y las variables binarias.

Primeramente, en la Sección 4.1 se define en qué consiste un problema de planeación justo a tiempo, en el que se desea secuenciar un conjunto de tareas en máquinas paralelas donde todas las tareas tiene asociada una fecha de entrega común. Luego en la Sección 4.2 se menciona cómo se han resuelto problemas de planeación justo a tiempo en la literatura. En este trabajo de tesis se manejan dos tipos de fechas de vencimiento las cuales se definen en la Sección 4.3: restrictiva y holgada.

4.1 Planeación justo a tiempo

Existen diferentes definiciones para el concepto de justo a tiempo. Una descripción precisa de este concepto es que la filosofía justo a tiempo indica cero inventarios, cero transacciones y cero disturbios. En la vida real se puede apre-

ciar lo mencionado anteriormente, por ejemplo, al fabricar un automóvil, éste se tiene que terminar de fabricar justo a tiempo a la fecha de entrega, si el automóvil se entrega antes de la fecha acordada se generan gastos de almacenaje y si se entrega después el cliente no queda satisfecho.

Mas precisamente, el enfoque de justo a tiempo tiene como objetivo reducir [22]:

- la complejidad de los detalles de planeación,
- la necesidad de controlar los centros,
- los niveles de inventario,
- las transacciones asociadas con shop-floor y la compra de sistemas.

El problema que se estudia en esta tesis, consiste en secuenciar n de tareas en m máquinas paralelas ($n < m$) que trabajan en velocidades diferentes e independientes. 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 las n tareas, $M = \{1, 2, 3, \ldots, m\}$.

A cada tarea i se tiene asociado un tiempo de procesamiento o tiempo de ejecución, dependiendo de la máquina j en la cual la tarea se este ejecutando, denotado por p_{ij}. Además, todas las tareas tienen asociada una fecha de vencimiento común, denotada como d, es decir, se desea que cada tarea quede terminada de procesar en esa fecha común.

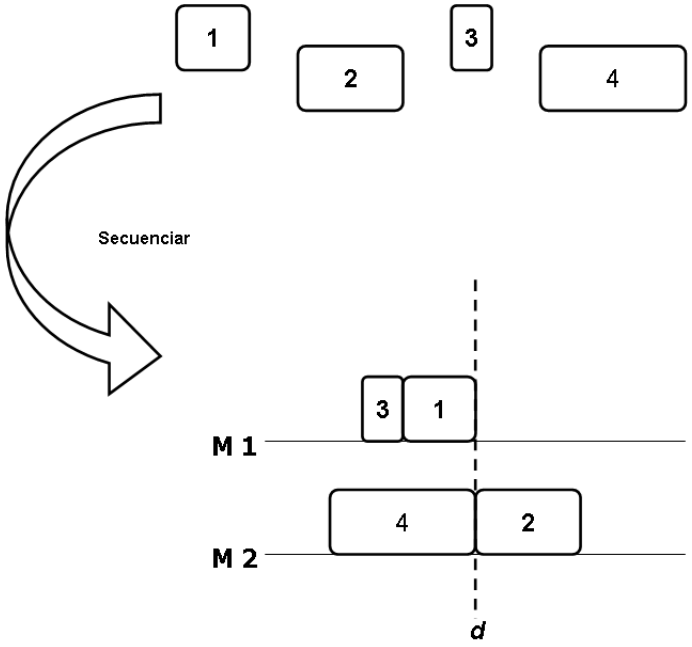


Figura 4.1: Ejemplo de secuenciación de tareas.

El tener en el problema una fecha de vencimiento común genera atrasos y adelantos en las tareas con respecto a esta fecha. Lo deseable es que las tareas terminaran de ser procesadas justo a tiempo a la fecha de entrega. Pero esto no es posible ya que todas las tareas tienen la misma fecha de vencimiento y el número de tareas es mayor al número de máquinas, por lo que algunas tareas serán terminadas de procesar antes de la fecha de entrega, llamadas tareas adelantadas, o terminan después de la fecha de vencimiento, llamadas tareas atrasadas. En la Figura 4.1 se presenta un ejemplo de secuenciación de cuatro tareas y dos máquinas, donde el tamaño indica el tiempo de procesamiento de las tareas. Las 4 tareas tienen una fecha de entrega común d. Si se observa, la tarea 1 y la tarea 4 quedaron justo a tiempo a la fecha de entrega, mientras que la tarea 2 terminó de ser ejecutada en la máquina 2 después de la fecha de entrega. Esta tarea es llamada tarea atrasada. La tarea 3 terminó de ser procesada en la máquina 1 antes de la fecha de entrega común. Esta tarea es llamada tarea adelantada.

Se asocia a cada tarea una penalidad de adelanto o atraso, lo cual permite penalizar las tareas por adelantado o por retraso para que la mayoría de las tareas se procesen justo a tiempo a la fecha de entrega común que se esté manejando. Se tiene una penalidad de adelanto α_i, por unidad de tiempo, si la tarea i termina de ser ejecutada antes de la fecha de entrega común. De lo contrario, es decir, si la tarea termina de ejecutarse después de la fecha de entrega común se tiene una penalidad β_i de atraso. El objetivo del problema de planeación es determinar el tiempo de término de cada tarea i en la máquina en donde se va a secuenciar, tal que la suma ponderada de penalidades de adelantos y retrasos sea minimizada.

En la Figura 4.2 se presenta de manera general, la estructura de un problema de planeación justo a tiempo. Se tiene que la tarea $i+1$ es ejecutada por adelantado, por lo cual se le asocia una penalidad de adelanto α_{i+1}. El costo por tener esta tarea por adelantado es $\alpha_{i+1}E_{i+1} = \max\{0, d - C_{i+1}\}$ donde E_{i+1} representa el adelanto de la tarea $i+1$, es decir, si la tarea se hace por adelantado costará $d - C_{i+1}$ lo cual representa la cantidad que faltó para terminar justo a tiempo. C_{i+1} representa el tiempo de terminacion de la tarea $i+1$. Ahora si la tarea se realiza justo a tiempo se toma el valor de cero y no tendría costo. De igual manera, si se tiene una tarea con retraso (en la Figura 4.2 la tarea i es con retraso) se tiene un costo de $\beta_i T_i = \max\{0, C_i - d\}$, donde T_i representa la cantidad de tiempo que la tarea terminó tarde.

En la Sección 4.3 se describe a detalle cómo se expresan en un modelo los adelantos y retrasos de la tarea i, E_i y T_i respectivamente.

4.2 Antecedentes

El problema de planeación justo a tiempo ha sido ampliamente estudiado desde hace 15 años [14]. Algunos trabajos que hay en la literatura se enfocan en

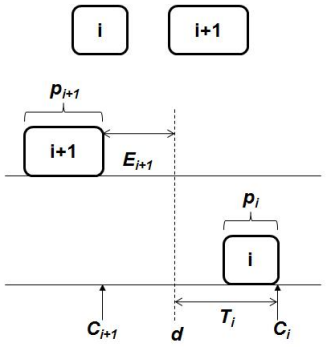


Figura 4.2: Ejemplo de secuenciación de tareas.

el estudio de diferentes tipos de problemas de planeación justo a tiempo, como por ejemplo, cuando se desean secuenciar un conjunto de tareas en una sola máquina, cuando se tienen máquinas paralelas ó máquinas paralelas idénticas. Otra variante de este tipo de problemas es cuando se tiene una fecha de vencimiento común holgada o restrictiva.

Gordon, Proth y Chu [14] presentan un estudio amplio de problemas de planeación justo a tiempo con fecha de entrega común en el caso determinista. Presentan problemas en donde la función objetivo tiene penalidades por tarea que se entrega con adelantos y atrasos. Además un punto importante de ese trabajo es que la fecha de vencimiento común también es una variable de decisión. Es decir, ademas de decidir el tipo de tarea a secuenciar (si se realiza por adelantado o atrasada), también se decide el momento en el que deben ser entregadas las tareas. Enfocan su estudio en los problemas en los que se tienen una sola máquina y en los que se presentan máquinas paralelas. Se enfocan, también en ajustes de la producción estática con un conjunto fijo de tareas disponibles para ser procesadas. En este caso consideran los modelos en los cuales se desean secuenciar n tareas (actividades) en m máquinas (recursos), donde una máquina puede procesar a lo más un trabajo y el objetivo es optimizar el tiempo de terminación de las tareas tomando en cuenta la fecha de entrega común.

Van den Akker, Hoogeveen y Van de Velde [30] presentan un trabajo muy importante en el que desarrollan un método de generación de columnas para el problema de planeación justo a tiempo con fecha de vencimiento holgada y penalidades de adelantos y atrasos. Formulan este problema como un problema de cobertura de conjuntos (*set covering problem*) con dos restricciones adicionales y un número exponencial de variables. Usan generación de columnas y relajaciones lagrangianas debido a que esta técnica es muy efectiva para resolver relajaciones lineales de problemas de cobertura de conjuntos o partición de conjuntos (*set partitioning*). La idea de utilizar generación de columnas es generar un conjunto inicial de columnas (patrones). Se resuelve la relajación lineal del problema (problema maestro) con el que se genera un patrón de soluciones. Luego, se usan los valores duales del problema maestro para resolver el subproblema generador de columnas y generar un nuevo patrón. Si existen una o más variables que quedaron fuera del programa lineal con un costo reducido negativo, dependiendo de la implementación del algoritmo, se añaden generando un nuevo patrón, ya que posiblemente pueden mejorar el valor de la solución, y se repite el procedimiento. Si no hay ninguna entonces la solución actual es óptima para la relajación lineal. La relajación lagrangiana ayuda a complementar el algoritmo de generación de columnas. Los autores muestran que la aplicación de la relajación lagrangiana puede hacer que el algoritmo de generación de columnas termine más rápido. Resuelven instancias de hasta 125 trabajos en una sola máquina aplicando esta técnica. Muestran que la cota inferior proporcionada por la relajación lagrangiana domina a la cota inferior derivada del algoritmo de generación de columnas. Además, muestran cómo la formulación de programación lineal puede ser adaptada de tal forma que la correspondiente cota inferior es igual a la cota inferior proporcionada por la relajación lagrangiana.

Otro trabajo basado en utilizar generación de columnas para problemas de

planeación justo a tiempo es el de Chen y Powell [5]. En ese trabajo presentan un algoritmo de generación de columnas basado en un algoritmo de descomposición para el problema de secuenciación de tareas en m máquinas paralelas idénticas donde la fecha de vencimiento común es holgada y el objetivo es minimizar los adelantos y retrasos de las tareas. En este trabajo el problema es formulado como un problema de programación entera para luego reformularlo, usando descomposición de Dantzing-Wolfe, como un problema de partición de conjuntos. Desarrollan un algoritmo de solución exacta basado en un método de ramificación y acotamiento. En el árbol, cada nodo es una relajación lineal del problema de conjunto particionado. Esta relajación es resuelta por medio de generación de columnas las cuales que representan secuencias en una sola máquina y son generadas resolviendo dos subproblemas de secuenciación en una sola máquina. Resuelven problemas con 60 tareas en tiempo razonable.

Cuando se tienen máquinas paralelas idénticas [22], en el caso en donde la fecha de entrega común es no restrictiva, se puede afirmar que en cada máquina la secuencia óptima es en forma de V, es decir, el tiempo total de procesamiento de las tareas que se realizan en la máquina 1 es mayor que el tiempo total de la máquina 2, el tiempo total de la máquina 2 es mayor que el de la máquina 3 y así sucesivamente. El número de tareas asignadas a cada una de las máquinas es $\lfloor n/m \rfloor$ o $\lceil n/m \rceil$, el número de tareas $\lceil n_j/2 \rceil$ terminan en el tiempo d en la máquina j, donde n_j es el número de tareas asignadas a la máquina j. Asumiendo estas ca-racterísticas, si se conoce la asignación de las tareas a las máquinas, entonces una secuencia óptima en cada máquina es encontrada usando un algoritmo, el cual encuentra una secuencia óptima en tiempo polinomial y las tareas están indexadas de acuerdo a $p_1 \geq p_2 \geq \ldots \geq p_n$.

Feldmann y Biskup [9] presentan una metaheurística para un problema de secuenciación en una sola máquina donde se tienen penalidades de adelantos y retrasos, manejan una fecha de vencimiento común restrictiva. Aplican tres me-

taheurísticas: Estrategias evolutivas, recocido simulado y umbral de aceptación (*threshold accepting*). Hacen una comparación entre estas tres metaheuristicas concluyendo que el método de umbral de aceptación es muy eficiente al obtener soluciones muy cercanas a la óptima.

Otro trabajo basado en heurísticas es el de Vredeveld y Hurkens [31] en el cual tratan un problema de secuenciación de tareas en máquinas paralelas donde cada tarea tiene asociado un peso y el objetivo es encontrar la secuencia de las tareas tal que se minimice la suma de todos los pesos asociados a las tareas. Presentan una comparación de algoritmos de aproximación basados en la relajación lineal (los cuales encuentran una solución factible en tiempo polinomial) con heurísticas de búsqueda local.

Skutella [29] considera el problema de secuenciación en máquinas paralelas en donde se desea minimizar el peso total de los tiempos de terminación de las tareas. Formula el problema como un programa entero cuadrático. La contribución es una relajación convexa para el problema. Desarrollan algoritmos de aproximación basado en relajaciones convexas y programación semidefinida. Los algoritmos de aproximación desarrollados presentan los mejores resultados conocidos para un problema de secuenciación con rechazo: una tarea puede ser rechazada o aceptada en el procesamiento. Si hay rechazo se agrega a la función objetivo una penalidad de rechazo.

Otro trabajo basado en programación cuadrática no convexa es el de Billionnet, Elloumi y Plateau [2], quienes presentan un método en el que reformulan un problema cuadrático en variables 0-1 sujeto a restricciones lineales en un programa cuadrático (0-1) donde la función objetivo es cuadrática convexa. El problema reformulado es resuelto eficientemente mediante un método de ramificación y acotamiento. Usan programación semidefinida para encontrar

una función equivalente convexa para el problema original.

Granot, Skorin-Kapov y Tamir [16] presentan un trabajo en el cual consideran un problema de secuenciación en el que un conjunto de trabajos puede ser particionado en un número pequeño de subconjuntos. Todos los trabajos de un subconjunto son idénticos. Cada trabajo debe ser procesado por una de las máquinas y todos los trabajos están disponibles para ser procesados en el tiempo 0. El objetivo es secuenciar los trabajos en las máquinas tal que el total de tiempo de terminación de los trabajos sea minimizado. En este trabajo, se presentan tres casos para el problema mencionado. En el caso 1, los pesos de los trabajos son independientes pero la máquina es dependiente, es decir, el peso de los trabajos que se procesan en la máquina i tiene el mismo valor y este peso es diferente a los trabajos que se procesen en la máquina j. En el caso 2, se tienen modelos ponderados con tiempos de procesamiento idénticos. En el caso 3, las máquinas son idénticas, es decir, procesan una misma tarea en exactamente el mismo tiempo. Para el primer y segundo casos se presenta un modelo de optimización entero mixto lineal y mediante una transformación, lo convierten en un modelo cuadrático convexo, separable cuadrática, el cual se resuelve en tiempo polinomial. En el caso 3 se desarrolla un algoritmo paramétrico que genera una representación de la solución óptima, la cual proporciona información necesaria para interpolar y calcular el valor de la función correspondiente a la solución óptima, a partir de valores críticos y una secuencia de intervalos.

Plateau y Ríos-Solis [25] presentan un problema de planeación justo a tiempo en el que se tienen penalidades de adelantos y retrasos para cada tarea. Se desea minimizar el tiempo de terminación de cada tarea. En la literatura se denota como $R\,||\sum_i w_i C_i$. El modelo matemático asociado a este problema tiene una función objetivo cuadrática no convexa. Obtienen soluciones óptimas mediante reformulaciones convexas utilizando programación semidefinida. Han

encontrado los mejores resultados en la literatura en instancias de 400 tareas y 16 máquinas para el problema.

4.3 Dos problemas de optimización de un problema de planeacón justo a tiempo

En esta sección se presentan dos variantes de un problema de planeación justo a tiempo, en los cuales se tienen penalidades de adelanto y atraso en cada una de las tareas. Además se tiene una fecha de vencimiento común. Se mencionan dos variantes, ya que se presentan dos fechas diferentes de vencimiento. El primer problema que se define es un problema en el que la fecha de vencimiento es restrictiva, es decir, la distancia que hay entre el tiempo 0 y la fecha es corta. En el segundo problema la fecha de entrega es holgada.

4.3.1 Problema de planeación justo a tiempo con fecha restrictiva

El problema de planeación justo a tiempo con fecha restrictiva, denotada como d^r con penalidades de adelanto α_i y atraso β_i para cada tarea i, implica que existe al menos una máquina j tal que la suma de los tiempos de procesamiento p_{ij} de las tareas que sean secuenciadas en dicha máquina es mayor que la fecha de entrega, $\sum_i p_{ij} > d^r$; no todos los subconjuntos de tareas pueden ser ejecutados en el intervalo $[0, d^r]$[25]. El problema es NP-duro [15].

Tener una fecha de vencimiento común para todas las tareas implica que se tiene un conjunto de soluciones dominantes, caracterizado por la siguiente proposición.

Proposición 4.1 [25] *Existe una solución óptima para el problema de planeación justo a tiempo con fecha restrictiva, tal que se cumplen las siguientes*

propiedades:

(1) No hay tiempo de ocio o tiempo muerto entre la ejecución de un par de tareas.

(2) Para cada máquina j cualquier tarea termina exactamente al tiempo d^r ó empieza la secuencia en el instante cero.

(3) Para cada máquina j, las tareas son terminadas antes o en la fecha de vencimiento común restrictiva (tareas adelantadas) y están en orden creciente conforme a la relación α_i/p_{ij}. Las tareas que empiezan a ejecutarse después de la fecha restrictiva (tareas atrasadas) están en orden decreciente conforme la relación β_i/p_{ij}.

La Proposición 4.1 implica que en cualquier máquina puede existir un trabajo que empieza antes y termina después de la fecha de vencimiento (el cual se conoce como trabajo atravesado). Además, la propiedad también nos dice que no hay tiempo de ocio entre un par de tareas. Esto quiere decir que en el instante en que termina de ejecutarse una tarea inmediatamente la siguiente tarea empieza a ejecutarse .

Como la fecha de vencimiento es restrictiva, la Proposición 4.1 dice que puede existir una máquina en donde una sola tarea empieza antes y termina después de la fecha de entrega común, pero se pagará por esta tarea una penalidad de retraso por la cantidad de tiempo que se haya pasado de la fecha restrictiva. Por ejemplo, en la Figura 4.3 se presenta una solución factible de este tipo de problema, en donde se tienen dos tareas por adelantado y dos tareas por atraso. La tarea S_s^1 es una tarea de tipo atravesada ya que empieza a ejecutarse antes y termina después de la fecha de vencimiento. Esta tarea solo tendrá un costo por atraso, por lo que se pagará la penalidad asignada a la tarea por el atraso multiplicada por el tiempo que se haya pasado de la fecha de vencimiento (G^1). Ver Figura 4.4.

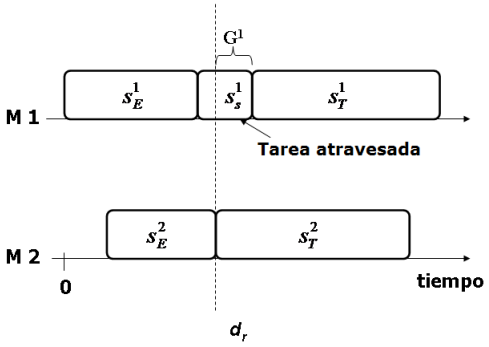


Figura 4.3: Ejemplo de una solución del problema de planeación justo a tiempo con fecha restrictiva.

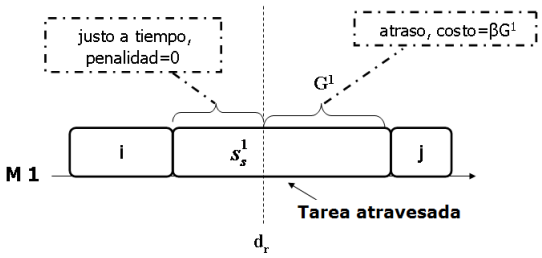


Figura 4.4: Ejemplo de una tarea tipo atravesada.

Para obtener el modelo de optimización de este problema, se definen los siguientes conjuntos, basándose en [25]:

- J, representa el conjunto de tareas que van a ser secuenciadas.
- M, es el conjunto de máquinas paralelas en las cuales se van a secuenciar las tareas. Este tipo de máquinas paralelas también pueden ser idénticas, es decir, el tiempo de procesamiento de las tareas son iguales en todas las máquinas.
- S_E^j representa el conjunto de tareas que se comienzan a ejecutar en la máquina j que son terminadas de ejecutar antes que la fecha de vencimiento (por adelantado).
- S_T^j representa el conjunto de tareas que se empiezan a ejecutar en la máquina j después de la fecha de entrega.
- S_S^j representa el conjunto de tareas en la máquina j que empiezan a ejecutarse antes y terminan después de la fecha de entrega (tareas del tipo atravesadas).

El objetivo del problema es encontrar el tiempo en el que terminan de ejecutarse cada una de las tareas, C_i. Éste es determinado por la máquina en la que las tareas son ejecutadas y la posición de las tareas en la secuencia (por adelantado, por atraso o atravesada). Considérese la variable binaria de decisión x_{ijl}, donde $l \in \{E, T, S\}$ es el tipo de tarea :

$$x_{ijl} = \begin{cases} 1 \text{ si la tarea } i \text{ de tipo } l \text{ es ejecutada en la máquina } j, \\ 0 \text{ de otro modo.} \end{cases}$$

Basándose en la Proposición 4.1 se puede encontrar un orden para las tareas conforme las relaciones α_i/p_{ij} y β_i/p_{ij}. Se definen los siguientes subconjuntos de tareas del conjunto J [25]: dado $(J, \prec_E^j)$, el orden total de tareas por adelantado en la máquina j está basado en la relación α_i/p_{ij}. Si dos tareas i y k son ejecutadas por adelantado en la máquina j, se define $i \prec_E^j k$ (una tarea i esta antes de una tarea k) si $\alpha_i/p_{ij} > \alpha_k/p_{kj}$, esto es, si la tarea i y la tarea

k serán ejecutadas por adelantado en la máquina j, entonces C_k deberá estar mas cerca de la fecha de vencimiento común que C_i. Ahora bien, si existe una máquina j en la cual un par de tareas i y k tienen igual relación de peso, es decir, $\alpha_i/p_{ij} = \alpha_k/p_{kj}$, se considerará que $i \succ_E^j k$ si $(i < k)$. De igual manera, se calcula el conjunto $(J, \prec_T^j)$, que es el orden para las tareas que serán ejecutadas por atraso, basados en la relación de peso β_i/p_{ij}: si $\beta_i/p_{ij} > \beta_k/p_{kj}$ entonces $i \prec_T^j k$, esto es C_i esta más cerca de la fecha de vencimiento común que C_k. Estos conjuntos se calculan para todas las máquinas, $j \in M$.

Cuando se tiene una tarea tipo atravesada, la tarea es divida en dos partes: una parte que termina justo a tiempo a la fecha de entrega y la otra parte que se ejecuta por retraso, ver Figura 4.4. Dado G^j la distancia entre el inicio de la siguiente tarea en la secuencia y la fecha de vencimiento común, se puede calcular esta distancia basándose en la Proposición 4.1, de la siguiente manera:

$$G^j = \text{máx}(0, \sum_{i \in J} \sum_{l \in \{E,S\}} x_{ijl} p_{ij} - d^r). \tag{4.1}$$

La ecuación (4.1), significa que, para calcular G^j en cada máquina, primero se sumarán todos los tiempos de procesamiento de las tareas por adelantado y las tareas tipo atravesadas. Si G^j es cero entonces no se tienen tareas tipo atravesadas. Ahora bien, si esta cantidad es positiva, significa que se tiene una tarea tipo atravesada y la siguiente tarea en ser procesada será por retraso y empezará la secuencia de tareas por retraso en G^j.

G^j es una cantidad entera, lo que se desea es que las variables involucradas en el modelo de optimización de este problema de planeación justo a tiempo, sean variables binarias. Por medio de una clásica transformación, expresaremos G^j como la combinación lineal:

$$G^j = \sum_{k=0}^{\lceil \log_2 u \rceil} 2^k y_k^j$$

donde u es una cota superior de G^j, $u = \text{máx}_{ij}\, p_{ij} - 1$ y las variables y_k^j son variables binarias.

Una vez definidos los conjuntos y las variables de decisión, ahora se definirán las restricciones del problema planteado.

Se tiene un conjunto de restricciones de asignación, es decir, se tiene que asegurar que una tarea sea asignada solamente a una máquina. La tarea no se puede realizar en más de una máquina. Además, si la tarea i es asignada a una máquina j, esta tarea debe ser ejecutada de un solo tipo $l \in \{E, T, S\}$. Para garantizar esto se tiene la ecuación (4.2):

$$\sum_{j \in M} \sum_{l \in \{E,T,S\}} x_{ijl} = 1, \forall\ i \in J \tag{4.2}$$

Si en el problema hay una tarea tipo atravesada se tiene que definir la distancia entre la fecha de entrega común y el inicio de la siguiente tarea en la secuencia, dadas por las restricciones (4.3) y (4.4). Además esta distancia debe ser una cantidad mayor o igual a cero, representada por las restricciones (4.5):

$$G^j \geq \sum_{i \in J} \sum_{l \in \{E,S\}} x_{ijl} p_{ij} - d^r, \forall\ j \in M \tag{4.3}$$

$$G^j \leq \sum_{i \in J} x_{ijS} p_{ij}, \forall\ j \in M \tag{4.4}$$

$$G^j \geq 0, \forall\ j \in M \tag{4.5}$$

Si en el problema se tiene una tarea tipo atravesada, por la Propiedad 4.1, solo se puede tener una tarea de este tipo:

$$\sum_{i \in J} x_{ijS} \leq 1, \forall\ j \in M \tag{4.6}$$

Una última restricción es que el tiempo de procesamiento de cada tarea del tipo por adelantado tiene que ser menor o igual a la fecha restrictiva, para garantizar que esta tarea sea por adelantado. Las restricciones (4.7) muestran este concepto.

$$\sum_{i \in J} x_{ijE} p_{ij} \leq d^r, \forall\ j \in M \tag{4.7}$$

Se ha mencionado que cada tarea tiene una penalidad de adelanto, si es ejecutada antes de la fecha de vencimiento. De igual manera, si la tarea es ejecutada después de la fecha de vencimiento, tiene una penalidad de atraso. Ahora se definen estas penalidades de adelantos y retrasos matemáticamente para incluirlas en el modelo de optimización.

Sea una tarea i que va a ser secuenciada en una máquina j, para definir su penalidad de adelanto, se tiene que observar cuales son las tareas que van a ser secuenciadas después de ésta, es decir, las tareas que pueden ser secuenciadas entre el tiempo que va terminar de ejecutarse la tarea i y la fecha de vencimiento. Esto se puede observar si se toman en cuenta las relaciones de peso α_i/p_{ij} y β_i/p_{ij}, ver Figura 4.5. También se tiene que tomar en cuenta si en la máquina j puede haber una tarea tipo atravesada. Si la hay, se tiene que tomar en cuenta, la parte de la tarea tipo atravesada que se realiza por adelantado. Se tienen que observar todos estos casos para saber si conviene o no, procesar la tarea i por adelantado, hay que recordar que la decisión está en saber qué tareas serán por adelantado, qué tareas se harán por atraso y si hay tareas tipo atraveasado.

La Ecuación (4.8) define E_i que es el adelanto de una tarea i en una máquina j: la suma de los tiempos de procesamiento de las tareas tal que, $k \succ_E^j i$ más la

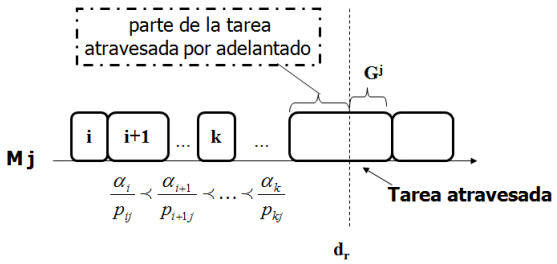


Figura 4.5: Penalidad de adelanto para las tareas.

parte que representa la parte del tiempo de duración de la tarea tipo atravesada antes de la fecha de vencimiento común, si es que existe, en la máquina j.

$$E_i = \sum_{j \in M} x_{ijE} \left(\sum_{k:k \prec_E^j i} x_{kjE} p_{kj} + \sum_{h \in J} x_{hjS} p_{hj} - G^j \right), \forall\ i \in J, \tag{4.8}$$

De igual manera se obtiene el atraso T_i de la tarea i en la máquina j. El objetivo es saber si conviene realizar la tarea por retraso y eso se ve reflejado en el tiempo de terminación de la tarea C_i, por lo que se tiene que tomar en cuenta si hay una tarea tipo atravesado antes de que la tarea i empiece a ser ejecutada. También se tienen que tomar en cuenta las tareas que puedan ser ejecutadas entre la fecha de vencimiento y el término de la tarea i. Esto se puede observar con las relaciones de pesos. Esto es, se tiene que observar que tareas pueden estar más cercanas a la fecha de vencimiento. El término más cercano se refiere a las tareas por atraso que pueden ser terminadas de ejecutarse poco tiempo después de la fecha restrictiva. La Ecuación (4.9) define la penalidad de atraso de una tarea i en la máquina j. Esta ecuación también asegura que el atraso para las tareas tipo atravesadas es igual a G^j.

$$T_i = \sum_{j \in M} x_{ijS} G^j + \sum_{j \in M} x_{ijT} \left(p_{ij} + \sum_{k:k \prec_T^j i} x_{kjT} p_{kj} + G^j \right), \forall\ i \in J. \tag{4.9}$$

Las relaciones de peso α_i/p_{ij} y β_i/p_{ij} juegan un papel muy importante a la hora de realizar el modelo de optimización. Con base a estas relaciones se pueden ordenar a las tareas en cada máquina dependiendo si se pueden hacer por adelantado o por atraso. Este orden que se menciona implica que se pueden identificar a las tareas con más peso y estas tareas son las que se deben de poner lo más cercano a la fecha restrictiva, si es que se puede, para cumplir con el objetivo de minimizar las penalidades de adelantos y atrasos para que de esta manera las tareas queden lo más cercano a la fecha de vencimiento común.

Con base en las restrucciones (4.2)-(4.9) se tiene el siguiente modelo de optimización para el problema de planeación justo a tiempo con fecha restrictiva [25]:

$$
\begin{aligned}
\min \quad & \sum_{i\in J} \alpha_i E_i + \beta_i T_i & & \\
\text{s. a} \quad & \sum_{j\in M} \sum_{l\in\{E,T,S\}} x_{ijl} = 1 & & \forall i \in J, \\
& G^j \geq \sum_{i\in J} \sum_{l\in\{E,S\}} x_{ijl} p_{ij} - d^r & & \forall j \in M, \\
& G^j \leq \sum_{i\in J} x_{ijS} p_{ij} & & \forall j \in M, \\
& \sum_{i\in J} x_{ijS} \leq 1 & & \forall j \in M, \\
& \sum_{i\in J} x_{ijE} p_{ij} \leq d^r & & \forall j \in M, \\
& G^j \geq 0 & & \forall j \in M, \\
& x \in \{0,1\}^{3nm}, y \in \{0,1\}^{\lceil \log_2 u \rceil \cdot m}, & &
\end{aligned}
\tag{4.10}
$$

donde:

$$
\begin{aligned}
G^j = \quad & \sum_{k=0}^{\lceil \log_2 u \rceil} 2^k y_k^j & & \forall j \in M, \\
E_i = \quad & \sum_{j\in M} x_{ijE} \left(\sum_{k:k\prec_E^j i} x_{kjE} p_{kj} + \sum_{h\in J} x_{hjS} p_{hj} - G^j \right) & & \forall i \in J, \\
T_i = \quad & \sum_{j\in M} x_{ijS} G^j + \sum_{j\in M} x_{ijT} \left(p_{ij} + \sum_{k:k\prec_T^j i} x_{kjT} p_{kj} + G^j \right) & & \forall i \in J.
\end{aligned}
$$

Este problema cuadrático se puede escribir en su forma estándar como:

$$\begin{aligned} \text{mín} \quad & g(z) = \frac{1}{2} z^t Q z + cz \\ \text{s. a } & Az = \mathbf{1}, \\ & A'z \leq b', \\ & z = (x, y) \in \{0,1\}^{dim \cdot m}, \end{aligned}$$

donde $dim = 3n + \lceil \log_2 u \rceil$; la matriz hessiana Q tiene un tamaño de $(dim \cdot m) \times (dim \cdot m)$; A es $n \times (dim \cdot m)$ cuyos coeficientes corresponden a las restricciones de igualdad en el modelo (4.10); A' es una matriz de tamaño $5m \times (dim \cdot m)$ que corresponde a las restricciones de desigualdad en el modelo (4.10); $\mathbf{1}$ es un vector de tamaño n cuyos elementos son iguales a 1 y b' es un vector de tamaño $5m$.

Para este problema la función objetivo cuadrática no es convexa, pero la ventaja y la razón por la cual se aplican los métodos propuestos para convexificar la función, es que la matriz hessiana presenta una estructura a bloques, donde cada bloque de la matriz representa a los coeficientes de las variables de decisión involucradas en cada máquina. A manera de ejemplo, si se tiene una instancia con cuatro tareas y dos máquinas paralelas no relacionadas, un bloque de la matriz hessiana es visualizado en la Figura 4.6.

4.3.2 Problema de planeación justo a tiempo con fecha holgada

El problema que se presenta en esta subsección es una variante del problema descrito en la Subsección 4.3.1. En este problema de planeación justo a tiempo, la fecha de entrega común es holgada.

Al igual que el problema presentado en la Sección 4.3.1, se tienen penalidades de adelanto y atraso y el objetivo principal es encontrar el término de terminación C_i de cada tarea $i \in J$ en la máquina $j \in M$. Lo que se desea es minimizar

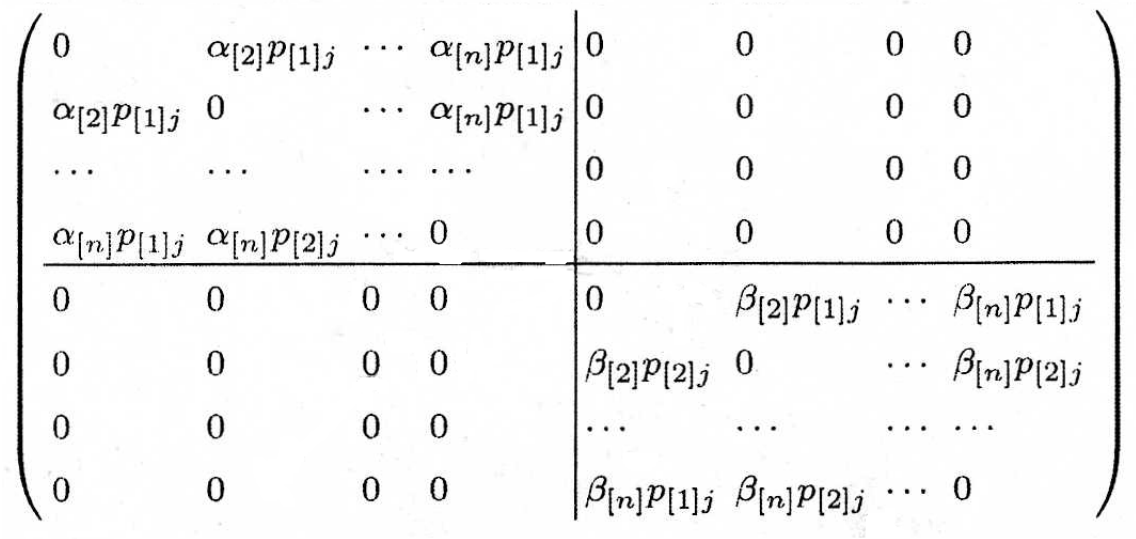


Figura 4.6: Bloque de la matriz hessiana en una instancia de 4 tareas y 2 máquinas.

la suma de penalidades de adelanto y atraso de las tareas, $\sum_{i \in J} \alpha_i E_i + \beta_i T_i$.

La diferencia entre el problema planteado en la sección 4.3.1 y el que se define en esta sección es la fecha de vencimiento común. En el problema anterior es una fecha restrictiva y la fecha del problema que que se define en esta sección es holgada. En un problema en donde la fecha es restrictiva, existe al menos una tarea i tal que $\sum_{\{i \in J\}} p_{ij} > d^r$, lo que implica que no todos los subconjuntos de tareas pueden ser ejecutados en el intervalo $[0, d^r]$.

Tener una fecha de vencimiento común holgada, denotada como d^l, implica que se pueden quitar las restricciones que forzan a la secuencia a empezar en el tiempo 0 o después para todas las máquinas [25]. Para identificar si el problema tiene una fecha de vencimiento holgada basta con verificar que la fecha de vencimiento que se esté manejando sea mayor que la suma de los tiempos de procesamiento de todas las tareas, $d > \sum_i p_{ij}\ \forall j \in M$.

Se utiliza la misma notación y los mismos conjuntos para definir el modelo de

optimización del problema de planeación justo a tiempo con fecha de vencimiento común holgada. En este tipo de problemas con fecha de entrega holgada no existen las tareas tipo atravesada.

Basándose en el problema asociado al problema de planeación justo a tiempo con fecha restrictiva que ha sido definido en la Sección 4.3.1, se define el modelo de optimización del problema planteado en esta sección.

De igual manera, la variable de decisión es x_{ijl}, $l \in \{E, T\}$, definida como:

$$x_{ijl} = \begin{cases} 1 \text{ si la tarea } i \text{ es ejecutada en la máquina } j \text{ de tipo l,} \\ 0 \text{ de otro modo.} \end{cases}$$

Como en este problema no existen tareas del tipo atravesado entonces el conjunto G^j no existe. Por lo tanto se pueden eliminar del modelo (4.10) las restricciones:
$G^j \geq \sum_{i \in J} \sum_{l \in \{E,s\}} x_{ijl} p_{ij} - d^r$, $\forall\ j \in M$,
$G^j \leq \sum_{i \in J} x_{ijs} p_{ij}$, $\forall\ j \in M$,
$G^j \geq 0$, $\forall\ j \in M$ y
$\sum_{i \in J} x_{ijs} \leq 1$, $\forall\ j \in M$.

De igual manera, los adelantos y atrasos de la tarea i en la máquina j, se definen similarmente que en la sección 4.3.1. La diferencia es que en este problema no hay tareas tipo atravesadas, se quitan de la ecuación y se tiene que los adelantos y atrasos de la tarea i en la máquina j se definen como:

$$E_i = \sum_{j \in M} x_{ijE} \left(\sum_{k : k \prec_E^j i} x_{kjE} p_{kj} \right) \qquad \forall\ i \in J, \quad (4.11)$$

$$T_i = \sum_{j \in M} x_{ijT} \left(p_{ij} + \sum_{k : k \prec_T^j i} x_{kjT} p_{kj} \right) \qquad \forall\ i \in J, \quad (4.12)$$

Basándose en el modelo (4.10), y las ecuaciones (4.11) y (4.12), el modelo de optimización asociado a un problema de planeación justo a tiempo con fecha de vencimiento común holgada, puede ser definido de la siguiente manera:

$$\begin{array}{lll} \text{minimizar} & \sum_{i \in J} \alpha_i E_i + \beta_i T_i & \\ \text{sujeto a:} & \sum_{j \in M} \sum_{l \in \{E,T\}} x_{ijl} = 1 & \forall i \in J, \\ & x \in \{0,1\}^{2nm} & \\ \text{donde:} & & \\ & E_i = \sum_{j \in M} x_{ijE} \left(\sum_{k \prec_E^j i} x_{kjE} p_{kj} \right) & \forall i \in J, \\ & T_i = \sum_{j \in M} x_{ijT} \left(p_{ij} + \sum_{k \prec_T^j i} x_{kjT} p_{kj} \right) & \forall i \in J. \end{array} \quad (4.13)$$

De igual manera (como en la Sección 4.3.1) se puede escribir el modelo de programación cuadrática en su forma estándar:

$$\begin{aligned} \text{mín} \quad & g(x) = \frac{1}{2} x^t Q x + cx \\ \text{sujeto a:} \quad & Ax = \mathbf{1}, \\ & x \in \{0,1\}^{dim \cdot m}, \end{aligned}$$

donde $dim = 2n$; la matriz hessiana tiene un tamaño de $(dim \cdot m) \times (dim \cdot m)$; A es $n \times (dim \cdot m)$ cuyos coeficientes corresponden a las restricciones de igualdad en el modelo (4.13) y $\mathbf{1}$ es un vector de tamaño n cuyos elementos son iguales a 1.

La matriz asociada a este problema no es una matriz semidefinida positiva, por lo tanto la función objetivo no es convexa.

En el siguiente capítulo se aplican los métodos presentados en el capítulo 3 para resolver este tipo de problemas.

Capítulo 5

Resultados experimentales

En el capítulo anterior se presentan dos problemas de planeación justo a tiempo como aplicación para probar experimentalmente los métodos propuestos en este trabajo de investigación. Para presentar los resultados obtenidos, este capítulo es divido en tres secciones.

En la Sección 5.1 se presenta una instancia para un problema de planeación justo a tiempo en el cual se tienen cuatro tareas que desean ser secuenciadas en dos máquinas paralelas, para esta instancia, se presenta como se define el modelo de optimización, la matriz hessiana asociada a la función objetivo y también los resultados experientales que se obtuvieron para esta instancia, aplicando los tres métodos que se presentaron en el Capítulo 3.

En la sección 5.2 se muestran los resultados que se obtuvieron al ejecutar los métodos en diferentes instancias para el problema en donde la fecha es restrictiva. De igual manera, en la Sección 5.3 se presentan resultados experimentales para el problema en donde la fecha de vencimiento común es holgada.

Tabla 5.1: Instancia 4 tareas, 2 máquinas.

Tarea i	p_{i1}	p_{i2}	α_i	β_i
1	18	19	18	17
2	15	17	5	5
3	10	12	13	1
4	2	1	17	1

5.1 Ejemplo de una instancia para el problema de planeación justo a tiempo

El objetivo de esta sección es presentar, a manera de ejemplo, como se describe el modelo de optimización asociado al problema y cual es la estructura de la matriz hessiana de la función objetivo. De esta manera, se puede observar la estructura en sí del problema que se estudia y demostrar que se utiliza este problema ya que la función objetivo es cuadrática no convexa y la matriz hessiana tiene la estructura a bloques. Se presentan dos ejemplos: el Ejemplo 1 maneja una fecha de vencimiento común restrictiva y el Ejemplo 2 maneja una fecha de vencimiento holgada.

Se tiene una instancia en la cual hay cuatro tareas con diferentes tiempos de procesamiento, las cuales se desean secuenciar en 2 máquinas que trabajan a distintas velocidades. Cada tarea tiene asociada una penalidad de adelanto y una penalidad de atraso. En la Tabla 5.1 se presenta lo antes mencionado.

La primera columna, Tarea i, son las tareas que se van a secuenciar en las 2 máquinas, en la segunda columna, p_{i1}, representa el tiempo de procesamiento de la tarea i en la máquina 1. La tercera columna esta compuesta por el tiempo de procesamiento de la tarea i en la máquina 2, la cuarta y quinta columna son las penalidades de adelanto y retraso de la tarea i, respectivamente.

Tabla 5.2: Relaciones de peso para la instancia 4 tareas y 2 máquinas.

Tarea i	α_i/p_{i1}	α_i/p_{i2}	β_i/p_{i1}	β_i/p_{i2}
1	1	0.9473	0.9444	0.8947
2	0.3333	0.2941	0.3333	0.2941
3	1.3	1.0833	0.1	0.0833
4	8.5	17	0.5	1

Teniendo los datos, se tienen que encontrar las relaciones de peso para darle un orden a las tareas. Recordemos, ver la Sección 4.3, que estas relaciones de peso se encuentran al calcular: α_i/p_{ij} y β_i/p_{ij}. Tomando los datos de la Tabla 5.1 se realizan los cálculos necesarios para encontrar las relaciones de peso. Se obtiene la Tabla 5.2 donde se presentan los resultados de estos cálculos.

Para escribir el modelo de optimización de un problema de planeación justo a tiempo con fecha restrictiva (Ejemplo 1) se necesita conocer la fecha de vencimiento, la cual se puede calcular mediante la ecuación 5.1) [25]:

$$d^r = 0.6\frac{\sum_{ij} p_{ij}}{2m} \tag{5.1}$$

Para la pequeña instancia, $d^r = 0.6\frac{\sum_{i=1}^{4}\sum_{j=1}^{2} p_{ij}}{2(2)} = 14.1$, se toma la parte entera superior de esta suma, entonces la fecha de vencimiento común restrictiva es 15.

Teniendo todos los datos necesarios, ahora se calculan los adelantos y atrasos para cada una de las tareas, E_i y T_i, respectivamente. Haciendo uso de las ecuaciones (4.8) y (4.9) se obtienen los adelantos y atrasos de cada una de las tareas:

$$E_1 = x_{11E}\left(x_{31E}p_{31} + x_{41E}p_{41} + x_{11S}p_{11} + x_{21S}p_{21} + x_{31S}p_{31} + x_{41S}p_{41} - 2^0y_{11} - 2^1y_{21}\right) + x_{12E}\left(x_{32E}p_{32} + x_{42E}p_{42} + x_{12S}p_{12} + x_{22S}p_{22} + x_{32S}p_{32} + x_{42S}p_{42} - 2^0y_{12} - 2^1y_{22}\right).$$

$$E_2 = x_{21E}\left(x_{11E}p_{11} + x_{31E}p_{31} + x_{41E}p_{41} + x_{11S}p_{11} + x_{21S}p_{21} + x_{31S}p_{31} + x_{41S}p_{41} - 2^0y_{11} - 2^1y_{21}\right) + x_{22E}\left(x_{12E}p_{12} + x_{32E}p_{32} + x_{42E}p_{42} + x_{12S}p_{12} + x_{22S}p_{22} + x_{32S}p_{32} + x_{42S}p_{42} - 2^0y_{12} - 2^1y_{22}\right).$$

$$E_3 = x_{31E}\left(x_{41E}p_{41} + x_{11S}p_{11} + x_{21S}p_{21} + x_{31S}p_{31} + x_{41S}p_{41} - 2^0y_{11} - 2^1y_{21}\right) + x_{32E}\left(x_{42E}p_{42} + x_{12S}p_{12} + x_{22S}p_{22} + x_{32S}p_{32} + x_{42S}p_{42} - 2^0y_{12} - 2^1y_{22}\right).$$

$$E_4 = x_{41E}\left(x_{11S}p_{11} + x_{21S}p_{21} + x_{31S}p_{31} + x_{41S}p_{41} - 2^0y_{11} - 2^1y_{21}\right) + x_{42E}\left(x_{12S}p_{12} + x_{22S}p_{22} + x_{32S}p_{32} + x_{42S}p_{42} - 2^0y_{12} - 2^1y_{22}\right).$$

$$T_1 = x_{11S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{12S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{11T}\left(p_{11} + 2^0y_{11} + 2^1y_{21}\right) + x_{12T}\left(p_{12} + x_{42T}p_{42} + 2^0y_{12} + 2^1y_{22}\right).$$

$$T_2 = x_{21S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{22S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{21T}\left(p_{21} + x_{11T}p_{11} + x_{41T}p_{41} + 2^0y_{11} + 2^1y_{21}\right) + x_{22T}\left(p_{22} + x_{12T}p_{12} + x_{42T}p_{42} + 2^0y_{12} + 2^1y_{22}\right).$$

$$T_3 = x_{31S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{32S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{31T}\left(p_{31} + x_{11T}p_{11} + x_{21T}p_{21} + x_{41T}p_{41} + 2^0y_{11} + 2^1y_{21}\right) + x_{32T}\left(p_{32} + x_{12T}p_{12} + x_{22T}p_{22} + x_{42T}p_{42} + 2^0y_{12} + 2^1y_{22}\right).$$

$$T_4 = x_{41S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{42S}\left(2^0y_{11} + 2^1y_{21}\right) + x_{41T}\left(p_{41} + x_{11T}p_{11} + 2^0y_{11} + 2^1y_{21}\right) + x_{42T}\left(p_{42} + 2^0y_{12} + 2^1y_{22}\right).$$

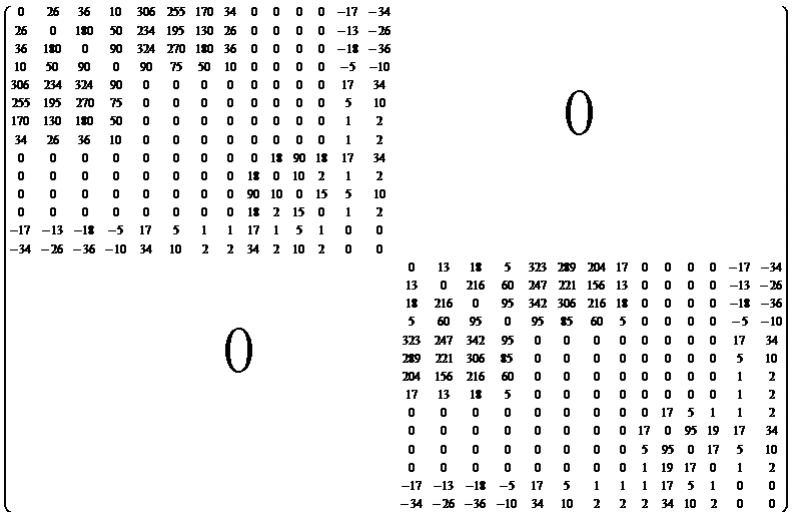


Figura 5.1: Matriz hessiana de la instancia ejemplo.

Entonces la función objetivo para el ejemplo se define como:

$$\begin{aligned}\text{minimizar } \textstyle\sum_{i\in J} \alpha_i E_i + \beta_i T_i = \quad & \alpha_1 E_1 + \beta_1 T_1 + \alpha_2 E_2 + \beta_2 T_2 + \alpha_3 E_3 + \beta_3 T_3 \\ & + \alpha_4 E_4 + \beta_4 T_4 = 18E_1 + 17T_1 + 5E_2 + 5T_2 \\ & + 13E_3 + T_3 + 17E_4 + T_4.\end{aligned}$$

donde E_1, E_2, E_3, E_4, T_1, T_2, T_3 y T_4 han sido definidas anteriormente.

La función objetivo asociada al ejemplo de instancia es una función cuadrática. Se puede observar fácilmente que el término cuadrático se obtiene de los adelantos y retrasos de las tareas.

La matriz asociada a la función objetivo de la instancia en el Ejemplo es una matriz que tiene una estructura a bloques, donde los elementos de la matriz son los coeficientes de los términos de la función objetivo. Además la matriz hessiana es simétrica, como lo podemos observar en la Figura 5.1.

Basándose en el modelo de optimización para un problema de planeación justo

a tiempo con fecha de entrega común restrictiva, las restricciones del modelo son definidas de la siguiente manera:

Restricción de asignación (4.2):

$$\begin{aligned}
&\textstyle\sum_{j=1}^{2}\sum_{l\in\{E,T,S\}} x_{1jl} = 1\\
&x_{11E} + x_{11T} + x_{11S} + x_{12E} + x_{12T} + x_{12S} = 1\\
&\textstyle\sum_{j=1}^{2}\sum_{l\in\{E,T,S\}} x_{2jl} = 1\\
&x_{21E} + x_{21T} + x_{21S} + x_{22E} + x_{22T} + x_{22S} = 1\\
&\textstyle\sum_{j=1}^{2}\sum_{l\in\{E,T,S\}} x_{3jl} = 1\\
&x_{31E} + x_{31T} + x_{31S} + x_{32E} + x_{32T} + x_{32S} = 1\\
&\textstyle\sum_{j=1}^{2}\sum_{l\in\{E,T,S\}} x_{4jl} = 1\\
&x_{41E} + x_{41T} + x_{41S} + x_{42E} + x_{42T} + x_{42S} = 1
\end{aligned}$$

Restricción (4.3):

$$\begin{aligned}
G^1 \quad & \geq \textstyle\sum_{i=1}^{4}\sum_{l\in\{E,S\}} x_{i1l}p_{i1}\\
2^0y_{11} + 2^1y_{21} \quad & \geq x_{11E}p_{11} + x_{11S}p_{11} + x_{21E}p_{21} + x_{21S}p_{21} + x_{31E}p_{31} + x_{31S}p_{31}\\
& \quad + x_{41E}p_{41} + x_{41S}p_{41} - 15\\
G^2 \quad & \geq \textstyle\sum_{i=1}^{4}\sum_{l\in\{E,S\}} x_{i2l}p_{i2}\\
2^0y_{12} + 2^1y_{22} \quad & \geq x_{12E}p_{12} + x_{12S}p_{12} + x_{22E}p_{22} + x_{22S}p_{22} + x_{32E}p_{32} + x_{32S}p_{32}\\
& \quad + x_{42E}p_{42} + x_{42S}p_{42} - 15
\end{aligned}$$

Restricción (4.4):

$$\begin{aligned}
G^1 \quad & \leq \textstyle\sum_{i=1}^{4} x_{i1S}p_{i1}\\
2^0y_{11} + 2^1y_{21} \quad & \leq x_{11S}p_{11} + x_{21S}p_{21} + x_{31S}p_{31} + x_{41S}p_{41}\\
G^2 \quad & \leq \textstyle\sum_{i=1}^{4} x_{i1S}p_{i1}\\
2^0y_{12} + 2^1y_{22} \quad & \leq x_{12S}p_{12} + x_{22S}p_{22} + x_{32S}p_{32} + x_{42S}p_{42}
\end{aligned}$$

Restricción (4.6):

$$\sum_{i=1}^{4} x_{i1S} \leq 1$$
$$x_{11S} + x_{21S} + x_{31S} + x_{41S} = 1$$
$$\sum_{i=1}^{4} x_{i2S} \leq 1$$
$$x_{12S} + x_{22S} + x_{32S} + x_{42S} = 1$$

Restricción (4.7):

$$x_{11E}p_{11} + x_{21E}p_{21} + x_{31E}p_{31} + x_{41E}p_{41} \leq 15$$
$$x_{12E}p_{12} + x_{22E}p_{22} + x_{32E}p_{32} + x_{42E}p_{42} \leq 15$$

Restricciónes (4.5), $x_{ijl} \in \{0,1\}\, 0\ \forall i \in J$, $j \in M$ y $l \in \{E,T,S\}$:

$$\begin{aligned} G^1 &\geq 0 \\ 2^0 y_{11} + 2^1 y_{21} &\geq 0 \\ G^2 &\geq 0 \\ 2^0 y_{12} + 2^1 y_{22} &\geq 0 \end{aligned}$$

$$\{x_{11E}, x_{21E}, x_{31E}, x_{41E}, x_{11T}, x_{21T}, x_{31T}, x_{41T}, x_{11S}, x_{21S}, x_{31S}, x_{41S}, x_{12E}, x_{22E},$$
$$x_{32E}, x_{42E}, x_{12T}, x_{22T}, x_{32T}, x_{42T}, x_{12S}, x_{22S}, x_{32S}, x_{42S}, y_{11}, y_{12}, y_{21}, y_{22}\} \in \{0,1\}$$

Se implementó un programa en ANSI C cuyo fin es leer los datos de una instancia del problema y escribir en un archivo de texto el modelo asociado a este problema en el formato requerido por CPLEX. Este archivo debe de ser guardado con extensión **.lp*

Se utilizó el método de ramificación y acotamiento implementado por ILOG CPLEX 11.2 [6]. Se ejecutó la instancia ejemplo en Sun Fire V440, conectado a 4 Procesadores de 1602 Hhz, Ultra SPARC III con 1 MB DE CACHE

Tabla 5.3: Comparación de métodos de convexificación en Ejemplo 1.

	CPLEX	Mínimo valor propio	Estructura a bloques
No. iteraciones	96	1165	1067
No. nodos	24	1017	1039
Tiempo CPU (s.)	0.05	0.42	0.4

y Memoria de 8 GB. El compilador que se utilizó para compilar el programa realizado en ANSI C es un compilador *gcc* de GNU versión 3.4.2.

También se probaron los métodos de mínimo valor propio y el método basado en la estructura diagonal a bloques, utilizando el método de mínimo valor propio. El mínimo valor propio de la matriz hessiana fue obtenido utilizando el software científico MATLAB versión 6.1.0. Para realizar esto se escribió un programa en ANSI C que lee la instancia y escribe en un archivo con extensión **.dat* la matriz hessiana que se obtiene con los datos de la instancia. Luego utilizando MATLAB se lee el archivo *.dat* y se obtiene el mínimo valor propio. Luego se modifica el archivo con extensión *.lp*, agregando el mínimo valor propio. De igual manera se procede para aplicar el método basado en la estructura diagonal a bloques.

Se aplicó el método que utiliza CPLEX para convexificar, el método de mínimo valor proprio y el método basado en la estructura diagonal a bloques utilizando el método de mínimo valor propio, a la instancia ejemplo y los resultados se muestran en la Tabla 5.3.

En la primera fila, se muestra el número de iteraciones que realizó el método de B&B empleando cada uno de los tres métodos; la segunda fila muestra el número de nodos que visitó el B&B y por último, en la tercera fila se muestra el tiempo de cómputo, en segundos, que se emplearon para obtener la solución

óptima de la instancia ejemplo.

Los resultados que se muestran en la Tabla 5.3 indican que, para la instancia ejemplo, el método que emplea CPLEX realizó menos iteraciones, visitó menos nodos y realizó menos tiempo en encontrar la solución óptima que los otros métodos utilizados. Por otra parte, podemos observar que el método basado en la estructura diagonal a bloques realiza menos iteraciones.

Para la misma instancia pero ahora manejando una fecha de vencimiento holgada (Ejemplo 2) se probaron nuevamente los tres métodos. El modelo (5.2) está asociado a un problema de planeación justo a tiempo con fecha de vencimiento común holgada:

$$
\begin{aligned}
\min \quad & 306x_{11T} + 323x_{12T} + 75x_{21T} + 85x_{22T} + 10x_{31T} + 12x_{32T} + 2x_{41T} \\
& + x_{42T} + \tfrac{1}{2}[100x_{21E}x_{31E} + 72x_{11E}x_{41E} + 432x_{12E}x_{32E} + 36x_{12E}x_{42E} \\
& + 180x_{21E}x_{11E} + 100x_{21E}x_{31E} + 20x_{21E}x_{41E} + 190x_{22E}x_{12E} \\
& + 120x_{22E}x_{32E} + 10x_{22E}x_{42E} + 52x_{31E}x_{41E} + 26x_{32E}x_{42E} \\
& + 34x_{12T}x_{42T} + 180x_{21T}x_{11T} + 20x_{21T}x_{41T} + 190x_{22T}x_{12T} \\
& + 10x_{22T}x_{42T} + 36x_{31T}x_{11T} + 30x_{31T}x_{21T} + 4x_{31T}x_{41T} \\
& + 38x_{32T}x_{12T} + 34x_{32T}x_{22T} + 2x_{32T}x_{42T} + 36x_{41T}x_{11T}] \qquad (5.2) \\
\text{s.a} \quad & x_{11E} + x_{11T} + x_{12E} + x_{12T} = 1 \\
& x_{21E} + x_{21T} + x_{22E} + x_{22T} = 1 \\
& x_{31E} + x_{31T} + x_{32E} + x_{32T} = 1 \\
& x_{41E} + x_{41T} + x_{42E} + x_{42T} = 1 \\
& x_{11E}, x_{11T}, x_{12E}x_{12T}, x_{21E}, x_{21T}, x_{22E}x_{22T}, x_{31E}, x_{31T}, x_{32E}x_{32T}, \\
& x_{41E}, x_{41T}, x_{42E}x_{42T} \in \{0, 1\}
\end{aligned}
$$

Los resultados se presentan en la Tabla 5.4.

Se puede observar en la Tabla 5.4 que el método basado en la estructura

Tabla 5.4: Comparación de métodos de convexificación en Ejemplo 2.

	CPLEX	Mínimo valor propio	Estructura a bloques
No. iteraciones	45	194	198
No. nodos	23	162	147
Tiempo CPU (s.)	0.01	0.07	0.06

diagonal a bloques visita menos nodos para obtener una solución óptima y reporta un tiempo menor que el método de mínimo valor propio. También se puede observar que el método que utiliza CPLEX realiza menos iteraciones, visita menos nodos y realiza menor tiempo en encontrar la solución óptima que los otros métodos.

5.2 Problema de planeación justo a tiempo con fecha restrictiva

En la sección anterior se presenta una instancia pequeña a modo de ejemplo para el problema de planeación justo a tiempo (JAT) con fecha restrictiva. En esta sección se presentan los resultados experimentales obtenidos al resolver un mayor número de instancias para este tipo de problemas, mediante los dos métodos de convexificación definidos y el método que se propone en esta tesis.

Para observar el funcionamiento de los métodos en este problema, se maneja un total de 75 instancias las cuales se desglosan en la Tabla 5.5. La Tabla 5.6 muestra el límite de tiempo que se estableció para cada tamaño de instancia. Las Tablas 5.7 a la 5.13 muestran el promedio de los datos que se recolectaron de las 5 instancias corridas para cada tamaño de instancias.

Al aplicar el método de convexificación de CPLEX a la instancia, una vez que se convexifica la matriz hessiana se aplica un B&B que también implementa

Tabla 5.5: Tamaño de instancias para un problema JAT con fecha restrictiva.

No. instancias	m	n
5	2	10
5	2	30
5	2	100
5	2	150
5	4	10
5	4	30
5	4	100
5	4	150
5	6	10
5	6	30
5	6	100
5	6	150
5	8	10
5	8	30
5	8	100

Tabla 5.6: Límite de tiempo para las instancias de un problema JAT con fecha restrictiva.

m	n	Límite de tiempo (s.)
2	30	3600
2	100	7200
2	150	10800
4	10	7200
4	30	7200
4	100	10800
4	150	10800
6	10	10800
6	30	10800
6	100	14400
6	150	14400
8	10	14400
8	30	14400
8	100	14400

Tabla 5.7: Resultados del método CPLEX para un problema JAT con fecha restrictiva.

Instancia	Iteraciones	Nodos	Tiempo	IOR (%)
(2,10)	278838	140039	117	0.01
(2,30)	3225704	1229147	3600	102
(2,100)	333856	109714	7200	101
(2,150)	93122.4	34291	10800	101
(4,10)	17719732	5946059	7200	104
(4,30)	2557571	980053	7200	102
(4,100)	139006	37648	10800	102
(4,150)	59397	18962	10801	102
(6,10)	21119768	5907935	10800	100
(6,30)	2645881	811489	10800	101
(6,100)	486461	50420	14400	101
(8,10)	21744764	7483776	14400	100
(8,30)	3671018	1015894	14400	100
(8,100)	90079	27808	14401	103

CPLEX.

En la Tabla 5.7, la primera columna pertenece al tamaño de instancia en la que se probó el método, la segunda columna representa la cantidad promedio de iteraciones que se realizaron, la tercera columna representa la cantidad promedio de nodos visitados en el método de ramificación y acotamiento, la cuarta columna representa la cantidad promedio de tiempo invertido en el B&B. Este tiempo no incluye la convexificación ya que el tiempo invertido en convexificar cada una de las instancias fue a lo mucho 4 segundos. Por último, la quinta columna representa el intervalo de optimalidad relativa (IOR) promedio reportado.

$$IOR = \frac{CotaSuperior - CotaInferior}{CotaInferior}$$

Observando los resultados de la Tabla 5.7, los tiempos de cómputo son muy grandes, inclusive para instancias pequeñas, por ejemplo para las instancias de 2 máquinas y 30 tareas, en 3600 segundos no encuentra la solución óptima y reporta un IOR relativo promedio de 102 %, lo que significa que está muy lejos de la solución óptima. En conclusión, utilizando el método de CPLEX para obtener soluciones óptimas para el problema de planeación justo a tiempo con fecha de vencimiento común restrictiva, se invierte mucho tiempo en encontrar una solución óptima. Se puede observar también que conforme crece el tamaño de las instancias se invierte más tiempo en intentar encontrar la solución óptima.

Ahora, obsérvese los resultados experimentales de la Tabla 5.8 obtenidos al utilizar el método de mínimo valor propio. Para éste método solamente se resolvieron 20 instancias, de las cuales, 5 son de 2 máquinas y 10 tareas, 5 instancias de 2 máquinas con 30 tareas, 5 instancias de 2 máquinas y 100 tareas y 5 instancias de 2 máquinas y 150 tareas, ya que los tiempos de ejecución de las instancias son demasiados altos. Para los grupos de instancias de 10, 30, 100 y 150 se establece un límite de tiempo, el cual se fija en 300 s., 3600 s., 7200 s. y 10800 s., respectivamente.

De igual manera, la Tabla 5.8 está estructurada como la Tabla 5.7. En los resultados que muestra la Tabla 5.8, el número de iteraciones realizadas y el número de nodos visitados son mucho más grandes que si se utiliza el método de CPLEX. Además se invierte mucho más tiempo en encontrar una solución. Por ejemplo al observar el tiempo invertido en las instancias de tamaño 2 máquinas y 10 tareas, el método de CPLEX invierte, en promedio, 117. 756 segundos mientras que el método de mínimo valor propio en 300.022 segundos,

Tabla 5.8: Resultados del método de mínimo valor propio para un problema JAT con fecha restrictiva.

Instancia	Iteraciones	Nodos	Tiempo	IOR (s.)
(2,10)	1135144	417023	300	1840
(2,30)	5916900	2025355	3600	2900
(2,100)	709131	171460	7200	3600
(2,150)	297067	103203	10800	3800

Tabla 5.9: Resultados experimentales del método basado en la estructura diagonal a bloques para el problema JAT con fecha restrictiva.

Instancia	Iteraciones	Nodos	Tiempo	IOR (%)
(2,10)	1095626	409966	300	1680
(2,30)	2418664	1019347	3600	2980
(2,100)	692566	177996	7200	36086
(2,150)	55573	20248	10800	3780

y no encuentra solución óptima quedándose con un IOR promedio de 1840 %.

Los resultados del método basado en la estructura diagonal a bloques (método que se propone en esta tesis), en las mismas instancias que en el método de mínimo valor propio, se muestran en la Tabla 5.9. Se observa que los tiempos de cómputo también son mucho más grandes que el método de CPLEX, pero son menores que utilizando el método de mínimo valor propio.

Comparando con el método de mínimo valor propio, el método propuesto realiza menos iteraciones, visita menos nodos y emplea un menor tiempo de cómputo. Por ejemplo, observando las instancias de 2 máquinas y 10 tareas, en 300 segundos, el método propuesto realiza, en promedio, 1095626 iteraciones mientras que el otro método realiza 1135144 iteraciones. En 300 segundos el método propuesto reporta un IOR promedio de 1680 % mientras que el otro

método reporta 1840 %. Por otro lado, el método de CPLEX encontró la solución óptima en 117 segundos, en promedio.

Como conclusión, el método de CPLEX es significativamente que los otros dos métodos. El método propuesto es mejor que el método de mínimo valor propio.

5.3 Problema de planeación justo a tiempo con fecha holgada

De manera similar que en la sección anterior, se probaron 3 métodos de los 4 que se presentan en esta tesis para el problema de planeación justo a tiempo en el que se maneja una fecha de vencimiento común holgada. Los métodos que se probaron son: Método de CPLEX, método de mínimo valor propio, método basado en la estructura diagonal a bloques utilizando el método de mínimo valor propio.

Primero, se muestran los resultados experimentales al utilizar el método de CPLEX, luego se presentan los resultados obtenidos utilizando el método de mínimo valor propio y, por último, los resultados experimentales utilizando el método basado en la estructura diagonal a bloques.

Se probó el método de CPLEX en 6 tamaños diferentes de instancias, para cada tamaño se manejan 5 instancias diferentes, es decir, se probó el método en 30 instancias. El tamaño de las instancias se muestra en la Tabla 5.10.

Los resultados al aplicar el método de CPLEX se muestran en la Tabla 5.11. La información desplegada es igual a las tablas anteriores. Como se puede apreciar para instancias de tamaño 2 máquinas y 60 tareas, en dos horas no encuentra la solución óptima y se queda con un IOR promedio de 8.10 %. Para instancias

Tabla 5.10: Tamaño de instancias para un problema JAT con fecha holgada.

No. instancias	m	n
5	2	10
5	2	30
5	2	60
5	4	10
5	4	30
5	4	100

Tabla 5.11: Resultados del método de CPLEX para un problema JAT con fecha holgada.

Instancia	Iteraciones	Nodos	Tiempo (s.)	IOR (%)
(2,10)	386	163	0.13	0
(2,30)	1427705	599364	994.99	0.01
(2,60)	2400697	997363	4390.38	8.10
(4,10)	50956	17793	12.196	0.02
(4,30)	7096786.8	1829661	7200.048	675.38
(4,100)	218487	45900.6	7200.272	701.26

de 4 tareas y 10 máquinas se encuentran soluciones óptimas pero cuando se tienen 30 tareas o 100 tareas se llega al límite de tiempo sin hallar solución. Se observa que para instancias de 4 máquinas y 100 tareas llega al límite de tiempo y finalizando con un IOR promedio muy grande.

Los resultados obtenidos al emplear el método de mínimo valor propio se despliegan en la Tabla 5.12. La Tabla 5.12 está estructurada de la misma manera que la Tabla 5.11. Con este método solo se resolvieron 15 instancias. Como se puede apreciar el método de mínimo valor propio, igual que en la sección anterior, emplea mucho más tiempo en intentar obtener una solución óptima que el método de CPLEX. De la Tabla 5.12 se puede observar que para ins-

Tabla 5.12: Resultados del método de mínimo valor propio para un problema JAT con fecha holgada.

Instancia	Iteraciones	Nodos	Tiempo (s.)	IOR (%)
(2,10)	337765	101380	44.02	0
(2,30)	7116395	3740904	7200.02	139.89
(2,60)	1710623	877702	7200.032	124

Tabla 5.13: Resultados del método basado en la estructura diagonal a bloques para un problema JAT con fecha holgada.

Instancia	Iteraciones	Nodos	Tiempo (s.)	IOR (%)
(2,10)	116793	76748	33.58	0
(2,30)	8484905	397201	7200.02	122.26

tancias de 2 máquinas y 10 tareas se emplearon, en promedio, 44. 02 segundos para obtener una solución óptima mientras que con el método de CPLEX se emplearon, en promedio, 0.132 segundos.

La tabla 5.13 despliega resultados preliminares del método basado en la estructura diagonal a bloques, los cuales muestran que para instancias de tamaño 2 máquinas y 10 tareas emplea menos tiempo en encontrar la solución óptima que el método de mínimo valor propio. Para estas instancias se tiene un tiempo de 33 segundos, el método de ramificación y acotamiento realiza, en promedio, 116793.4 iteraciones y visita 76748.2 nodos mientras que el método de mínimo valor propio realiza 337765.8 iteraciones y visita 101380.6 nodos.

En conclusión el método propuesto en esta tesis es mejor que el método de mínimo valor propio ya que reporta mejores tiempos, sin embargo el método de CPLEX es mejor que el método propuesto ya que los tiempos son mucho más cortos.

Capítulo 6

Conclusiones

El trabajo realizado consistió en proponer un nuevo método para obtener una reformulación convexa de un problema de programación cuadrática en la que la función objetivo no es convexa. Se probó el funcionamiento del método que se propone en esta tesis y se comparó con otros métodos existentes en la literatura en un caso particular de un problema de secuenciación. En este capítulo, se presentan las conclusiones que se obtuvieron a lo largo de esta investigación y se mencionan las áreas de oportunidad para trabajo a futuro.

6.1 Conclusiones

En este trabajo de tesis, el foco de estudio se centró en un problema de programación cuadrática en $\{0, 1\}$ sujeto a restricciones lineales, en el que la función objetivo no es convexa, es decir, la matriz hessiana asociada a la función objetivo no es semidefinida positiva. Además, la matriz hessiana presenta una estructura diagonal a bloques.

Para resolver este problema, se estudió la teoría matricial, la teoría de programación cuadrática y programación semidefinida para proponer un nuevo método de reformulación convexa basado en la estructura diagonal a bloques que presenta la matriz hessiana. En la literatura, existen métodos para resolver este tipo de problemas que presentan una función cuadrática no convexa en variables binarias. Por lo tanto, se hizo una revisión de la literatura en donde

destacaron tres métodos: Método que utiliza CPLEX para obtener una función equivalente convexa de la función objetivo, método del mínimo valor propio y el método basado en programación semidefinida. En esta tesis se propone el método basado en la estructura diagonal a bloques de la matriz hessiana.

Luego del estudio teórico, experimentalmente se probó el método que utiliza CPLEX, el método de mínimo valor propio y el método basado en la estructura diagonal a bloques (método propuesto) en dos problemas de planeación justo a tiempo con fecha de vencimiento común los cuales son NP-duros. Estos métodos dan una función equivalente convexa a la función objetivo y luego aplican un método de ramificación y acotamiento para obtener soluciones óptimas.

El primer problema que se estudió fue un problema de planeación justo a tiempo en máquinas paralelas en el que la fecha de vencimiento común para todas las tareas es restrictiva. Este problema tiene asociado un modelo de optimización en el que tenemos una función objetivo cuadrática no convexa, sujeto a restricciones lineales y variables binarias. El segundo problema es similar al primero, la diferencia es que se maneja una fecha de vencimiento común holgada. Ambos problemas presentan una función objetivo cuadrática no convexa y además, la matriz hessiana asociada a cada una de las funciones objetivo presenta una estructura diagonal a bloques.

Los resultados experimentales muestran que el método basado en la estructura diagonal a bloques reporta mejores resultados comparado con el método de mínimo valor propio. Los tiempos de cómputo son menores por lo que la cota inferior es mejor y la calidad de soluciones también es mejor.

Comparando el método propuesto con el método que utiliza CPLEX, se concluye que el método que utiliza CPLEX es significativamente mejor en cuanto

a calidad de soluciones y en tiempo de ejecución ya que los tiempos son mucho menores.

En resumen, se estudiaron tres problemas de secuenciación de tareas del tipo justo a tiempo, cuatro métodos de reformulaciones convexas y se evaluaron tres métodos de reformulación.

6.2 Trabajo a futuro

Al estudiar el problema planteado en este trabajo de tesis, se concluye que hay un amplio campo de estudio en esta área de investigación. Todavía hay mucho por hacer en esta área, por lo que se menciona el trabajo a futuro que puede enriquecer esta línea de investigación.

Primeramente, está pendiente por probar el método basado en programación semidefinida. Se mencionó en este trabajo que la relajación semidefinida del problema en estudio da la mejor cota inferior para el problema, por lo que, se ahorra tiempo en el método de ramificación y acotamiento.

Hasta el momento, se ha estudiado la teoría referente a la programación semidefinida, lo que falta por hacer es terminar un programa para obtener el programa semidefinido asociado al problema de planeación justo a tiempo en máquinas paralelas con fecha de vencimiento común larga y restrictiva y probar el funcionamiento del método basado en la estructura diagonal a bloques aplicando programación semidefinida, mediante un diseño de experimentos adecuado.

También falta por probar este método en un problema e planeación justo a

tiempo en el que la fecha de vencimiento común es restrictiva.

El trabajo de tesis aquí presentado, está basado en el estudio de la estructura diagonal a bloques. Se propuso descomponer la matriz original en submatrices y aplicar diferentes métodos de reformulaciones convexas a cada submatriz. Otra idea que se tiene es proponer diferentes reformulaciones convexas basadas en la estructura particular que presenta la matriz hessiana.

Observando los resultados obtenidos mediante los métodos de mínimo valor propio y mínimo valor propio a bloques, los tiempos de cómputo son muy grandes, por lo que es interesante introducir buenas cotas inferiores y/o superiores para que al aplicar el método de ramificación y acotamiento se generen cortes y ver si se disminuye el tiempo invertido en obtener una solución óptima.

En este trabajo de tesis se probó el método propuesto en un problema de planeación justo a tiempo. Es interesante probar el funcionamiento de este método en otros problemas en donde la matriz hessiana presente una estructura a bloques.

Bibliografía

[1] R. Bhatia. *Positive Definite Matrix.* Princeton University Press, New Jersey, E. U. A. (2007).

[2] A. Billionnet, S. Elloumi y M.-C. Plateau. Improving standard solvers via convex reformulation for constrained quadratic 0-1 programs: The QCR method. *Discrete Applied Mathematics* **157**(6), 1185–1197 (2009).

[3] B. Borchers. CSPD, a C library for semidefinite programming. *Optimization Methods and Software* **11**(1), 613–623 (1999).

[4] R. Bronson. *Matrix Methods: An Introduction.* Academic Press, San Diego, E. U. A., 2a. edición (1991).

[5] Z.-L. Chen y W. B. Powell. A column generation based descomposition algorithm for a parallel machine just-in-time scheduling problem. *European Journal of Operational Research* **116**(1), 220–232 (1999).

[6] ILOG CPLEX 11.2. URL:. http://www.ilog.com/products/cplex/.

[7] E. de Klerk. *Aspects of Semidefinite Programming.* Kluwer, Dordrecht, Holanda (2002).

[8] D. den Hertog. *Interior Point Approach to Linear, Quadratic and Convex Programming.* Kluwer, Dordrecht, Holanda (1994).

[9] M. Feldmann y D. Biskup. Single-machine scheduling for minimizing earliness and tardiness penalties by metaheuristics. *Computers & Operations Research* **28**(8), 787–801 (2001).

[10] R. M. Freund. Quadratic functions, optimization and quadratic forms. Manuscrito no publicado, Massachusetts Institute of Technology, Cam-

bridge, E.U.A. (2004) URL: http://ocw.mit.edu/courses/sloan-school-of-management/15-084j-nonlinear-programming-spring-2004/lecture-notes.

[11] K. Fujisawa, M. Kojima, K. Nakata y M. Yamashita. SDPA (semidefinite programming algorithm) user's manual - version 6.2.0. Research Report B-308, Departamento de Matemticas y Ciencias Computacionales, Tokyo Institute of Technology, Tokyo, Japan(2004). URL: http://www.is.titech.ac.jp/ kojima/sdp.html.

[12] K. Fujisawa, H. Sasajima y Y. Tasui. What is SDP? URL: http://sdpa.indsys.chuo-u.ac.jp/sdpa/files/whatissdp.pdf.

[13] M. Fukushima y L. Qi (Eds). *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods*. Kluwer, Dordrecht, Holanda (1999).

[14] V. Gordon, J.-M. Proth y C. Chu. A survey of the state-of-the-art of common due date assignment and scheduling research. *European Journal of Operational Research* **139**(1), 1–25 (2002).

[15] R. Graham, E. Lawler, J. Lenstra y A. R. Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics* (5), 287–326 (1979).

[16] F. Granot, J. Skorin-Kapov y A. Tamir. Using quadratic programming to solve high multiplicity scheduling problems on parallel machines. *Algorithmca* **17**(2), 100–110 (1997).

[17] P.L. Hammer y A.A. Rubin. Some remarks on quadratic programming with 0-1 variables. *RAIRO* **3**(1), 67–79 (1970).

[18] C. Helmberg, F. Rendl, R. Vanderbei y H. Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization* **6**(2), 342–361 (1996).

[19] F. S. Hillier y G. J. Lieberman. *Introduction to Operations Research*. McGraw-Hill, New York, E. U. A., 7a. edición (2001).

[20] L. Hogben. *Handbook of Linear Algebra*. Chapman and Hall, Boca Raton, E. U. A. (2007).

[21] R. Horst y P. M. Pardalos. *Handbook of Global Optimization*. Kluwer, Dordrecht, Holanda (1995).

[22] J. Jozefowska. *Just-in-time Scheduling: Models and Algorithms for Computer and Manufacturing Systems*. Springer, New York, E. U. A. (2007).

[23] M. Kozlov, S. Tarasov y L. Kachian. Polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics* **20**(5), 223–228 (1979).

[24] M.-C. Plateau, A. Billionnet y S. Elloumi. Eigenvalue methods for linearly constrained quadratic 0-1 problems with application to the densest k-subgraph problem. En *Memorias del 6to congreso de la ROADEF*, páginas 55–66, Tours, Francia (2005).

[25] M. C. Plateau y Y. A. Rios-Solis. Optimal solutions for unrelated parallel machines scheduling problems using convex quadratic reformulations. *European Journal of Operations Research* **201**(3), 729–736 (2010).

[26] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, E. U. A. (1970).

[27] S. Sahi. Computationally related problems. *SIAM Journal on Computing* **3**(4), 262–279 (1974).

[28] M. A. Salazar-Aguilar, R. Z. Ríos-Mercado y M. Cabrera-Ríos. New models for commercial territory design. Reporte técnico PISIS-2009-07, Programa de Posgrado en Ingeniería de Sistemas, Universidad Autónoma de Nuevo León, San Nicolás de los Garza, N.L. (2009).

[29] M. Skutella. Convex quadratic and semidefinite programming relaxations in scheduling. *Journal of the ACM* **48**(2), 206–242 (2001).

[30] M. van den Akker, H. Hoogeveen y S. van de Velde. Combining column generation and Lagrangean relaxation to solve a single-machine common due date problem. *INFORMS Journal on Computing* **14**(1), 37–51 (2002).

[31] T. Vredeveld y C. Hurkens. Experimental comparison of approximation algorithms for scheduling unrelated parallel machines. *INFORMS Journal on Computing* **14**(2), 175–189 (2002).

Capítulo 7

Ficha autobiográfica

Yadira Isabel Silva Soto

Candidato para el grado de Doctor en Ingeniería
con especialidad en Ingeniería de Sistemas

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Tesis:
Planeación justo a tiempo: soluciones óptimas mediante reformulaciones convexas

Lic. Yadira Isabel Silva Soto estudió la carrera de Licenciado en Matemáticas en la Facultad de Ciencias Físico-Matemáticas (FCFM) de la Universidad Autónoma de Nuevo León (UANL), en el período Agosto 2003-Diciembre 2007.

En enero 2009 fue aceptada en el programa de Posgrado en Ingeniería de Sistemas (PISIS) de la Facultad de Ingeniería Mecánica y Eléctrica (FIME) de la UANL para estudiar por dos años la maestría en esta área. Le fue asignada como asesora de tesis a la Dra. Yasmín A. Ríos Solis.

Durante estos dos años en el PISIS de la FIME, gracias a los proyectos de la Dra. Yasmín, tuvo la oportunidad de presentar este trabajo en dos congresos:

Escuela Nacional de Optimización y Análisis Numérico (ENOAN), el cual se llevó a cabo en Puebla, Puebla en Marzo de 2009 y en el III Taller Latino Iberoamericano de Investigación de Operaciones (TLAIO3) el cual se llevó a cabo en Octubre 2010 en Acapulco, Guerrero, en el cual se publicó este trabajo en las memorias del mismo. Además, participó en el programa de seminarios de la misma institución, dando dos pláticas en las cuales presentó avances de este proyecto de tesis, en Mayo 2009 y Noviembre 2009.