Universidad Nacional del Litoral

Facultad de Ingeniería Química

Doctorado en Ingeniería Química

TESIS DOCTORAL

TÉCNICAS DE OPTIMIZACIÓN BASADAS EN LÓGICA PARA PROBLEMAS DISCRETOS/CONTINUOS EN INGENIERÍA DE PROCESOS

por

ALDO VECCHIETTI

-Junio de 2000-

Universidad Nacional del Litoral

Facultad de Ingeniería Química

Se presenta esta tesis en cumplimiento de los requisitos exigidos
por la Universidad Nacional del Litoral para el grado académico de

Doctor en Ingeniería Química

TÉCNICAS DE OPTIMIZACIÓN BASADAS EN LÓGICA PARA PROBLEMAS DISCRETOS/CONTINUOS EN INGENIERÍA DE PROCESOS

presentada por
Aldo Vecchietti

Director
Dr. Ignacio E. Grossmann

Co-Director
Dr. Jaime Cerdá

Jurados: Propuesta presentada para su consideración por la Comisión de Posgrado de la Facultad de Ingeniería Química de la Universidad Nacional del Litoral

Junio, 2000

A mi padre y a mi madre

A Laura, Juan Andrés, María Julia, Santiago y Clara

A ellos les dedico el esfuerzo de todos estos años

AGRADECIMIENTOS

Quisiera agradecer en primer lugar a mi director el Dr. Ignacio Grossmann por sus invalorables guías y consejos a lo largo del desarrollo de este trabajo. Mi agradecimiento al Dr. Jaime Cerdá por el apoyo brindado en todo momento.

Quisiera agradecer también el soporte financiero y el material facilitado a las siguientes Instituciones: CONICET, Universidad Tecnológica Nacional, Proyecto FOMEC 824, National Science Foundation y GAMS Development Co.

A mis compañeros de trabajo de INGAR que compartieron conmigo este trabajo, por tantos momentos memorables, por su ayuda desinteresada, muestras de aprecio y por el ambiente creado que me hicieron crecer intelectualmente y como persona.

A mis compañeros de la Universidad Tecnológica Nacional quienes me ofrecieron el apoyo constante para el logro de este objetivo.

A los integrantes del Departamento de Ingeniería Química de la Universidad de Carnegie Mellon, porque con ellos compartí y discutí muchos aspectos del desarrollo de esta tesis.

Por último y muy especialmente, quiero expresar mi agradecimiento a Laura por su paciencia, su comprensión y su soporte en todo momento, sin su aliento y consejos no hubiese sido posible la culminación de este trabajo. A mis hijos Juan, Julia, Santiago y Clarita que sin darse cuenta apoyaron esta aventura.

Resumen

Este trabajo trata de los modelos de optimización basados en lógica para problemas discretos/continuos en Ingeniería de Procesos. Debido a que en Ingeniería de procesos es muy frecuente encontrar restricciones y ecuaciones no lineales en sus modelos, se concentrará el estudio en los problemas no lineales. Los modelos empleados tradicionalmente para este tipo de problemas son programas matemáticos mixto-enteros no lineales (MINLP). Los modelos basados en lógica surgieron más recientemente, aquí las decisiones discretas se modelan con disyunciones y proposiciones lógicas y presentan una alternativa a los MINLP. En este trabajo se propone una formulación híbrida de los problemas: con restricciones lógicas (disyunciones y proposiciones lógicas) y mixto-enteras. Esta es la base de este trabajo. Se muestra la generalidad de la formulación híbrida por medio de transformaciones de restricciones lógicas de otras áreas de la optimización basada en lógica, y de disyunciones embebidas, en la forma de la representación híbrida.

En este trabajo se analiza cuándo es conveniente formular una decisión discreta como restriccion mixto-entera ó cómo disyunción. El estudio se realiza en base a las propiedades que presentan los términos de las disyunciones y a las posibles relajaciones de un conjunto disyuntivo: “Big-M”, subrogada de Beaumont ó la cáscara convexa. Las dos primeras relajaciones están relacionadas con una representación mixta-entera, la cáscara convexa con una formulación disyuntiva. Existen dos casos extremos cuando se analizan los términos de una disyunción: que estén incluidos unos en otros ó que sean completamente disjuntos. El caso intermedio es que los términos tengan alguna intersección. Si los términos están incluidos unos en otros, todas las relajaciones se comportan de la misma forma. En este caso la disyunción se puede reemplazar por el término que tiene la región factible mayor. Si los términos de una disyunción no se intersectan conviene formular la decisión discreta como disyunción y aplicar la cáscara convexa como relajación. Si se está en el caso intermedio no es posible establecer claramente que formulación es conveniente. Se propone un algoritmo de preprocesamiento para determinarlo.

Existen diversos algoritmos para la solución de problemas MINLP y disyuntivos. Aquí se proponen extensiones de los algoritmos para la formulación híbrida. Se realiza una implementación de estos algoritmos en la primera versión del software LOGMIP, y se presentan los resultados obtenidos en una serie de ejemplos de ingeniería de procesos resueltos con este programa.

A partir de la implementación de la primera versión de LOGMIP surgió la necesidad de extender los lenguajes de programación matemática para incluir la expresión de disyunciones, restricciones y proposiciones lógicas. Por lo tanto, se propone esta extensión por medio de: sentencias IF..THEN..ELSE..ENDIF para la expresión de los distintos tipos de disyunciones, la implementación de los operadores lógicos: $\wedge$(“and”), $\vee$(“or”), $\underline{\vee}$(“or exclusivo”), ~(“not”,!,$\neg$), ->(implicación), <-> (equivalencia) y de algunas sentencias especiales que simplifiquen la expresión de proposiciones lógicas: *atmost, atleast, exactly, adjacent y alldifferent*.

Por último se propone una implementación del lenguaje y los algoritmos de resolución en una nueva versión de LOGMIP. La propuesta se basa en la implementación de un analizador sintáctico y semántico del lenguaje en una etapa de precompilación de un lenguaje matemático. Esto permite trabajar de manera independiente la parte lógica de la matemática También se presentaron distintos detalles acerca de cómo llevar a cabo las distintas implementaciones de los métodos de resolución de los problemas disyuntivos e híbridos. Actualmente se posee un prototipo avanzado de LOGMIP en su segunda versión con una implementación parcial de las ideas presentadas en este trabajo.

Indice

Lista de Figuras

Lista de Tablas

1

Introducción

1.1.Desarrollo de modelos matemáticos en Ingeniería de Procesos

La abstracción del comportamiento de un proceso en un modelo matemático requiere un profundo entendimiento de los fenómenos fisicoquímicos que ocurren en el mismo como de los objetivos perseguidos con la aplicación del modelo (Marquardt y otros, 1999). La resolución de modelos matemáticos de optimización, ya sea en Ingeniería de Procesos como en otras áreas comprende, en una primera etapa, la abstracción de los hechos de la vida real en ecuaciones matemáticas que formulen el objetivo perseguido y las restricciones a las que está sujeto el problema. También implica la selección de la representación más adecuada de las mismas que, directa ó indirectamente, involucra la selección del algoritmo para la resolución del problema. En esta primera etapa, por lo tanto, el problema debe ser pensado y analizado de modo que se puedan especificar las ecuaciones del modelo y la mejor representación para estas ecuaciones. Esta tarea no es trivial, el modelador debe elegir entre las diversas opciones que representan un mismo problema, sabiendo que esto afecta en mayor o menor medida la solución del mismo (Nemhauser y Wolsey, 1988). El análisis, desarrollo e implementación de modelos de optimización matemáticos son tareas difíciles de ejecutar y que consumen mucho tiempo. Por lo tanto, las tareas de esta primera etapa tienen un impacto importante a la hora de obtener los beneficios de esta técnica, en muchas casos involucran decisiones que afectan el tamaño del problema, principalmente en cuanto al número de variables y ecuaciones que tendrá el mismo.

Acabada la primera etapa, sigue el proceso por el cual las ecuaciones deben ser traducidas y escritas con las reglas y lenguajes de representación que poseen las herramientas computacionales que se disponen para resolver el problema. Este proceso no es tan claro, directo y sistemático como parece en un principio, ya que aquí también existen diversas opciones que el usuario elige de acuerdo con sus preferencias. El modo que se escribe el problema afecta el proceso de obtención de la solución. La solución del problema es un proceso iterativo que implica la revisión, modificación, eliminación y adición de variables y restricciones hasta que la solución obtenida satisfaga los requerimientos mínimos. Por lo tanto, existen diversas soluciones de compromiso entre distintos extremos, por ejemplo: entre una escritura compacta, pero generalmente más compleja y difícil de entender, o una escritura más larga, pero más simple, con una mejor

comprensión del modelo. La traducción de las ecuaciones por medio del lenguaje computacional lleva implícita también la selección de identificadores, operadores, operandos, etc., para la expresión del modelo. Un lenguaje con muchos elementos brinda mayores posibilidades de una traducción adecuada del modelo, pero por otra parte se deberá tener más cuidado en la selección de los elementos más apropiados para hacerlo. Un lenguaje con pocos elementos de expresión facilita la selección de sus componentes para la expresión del modelo, pero hará más dificultosa la lectura y comprensión del mismo.

La evolución de los lenguajes para la especificación de un programa matemático ha acompañado la evolución de las herramientas computacionales: desde rígidos archivos con formato (archivos MPS), pasando por lenguajes más naturales de expresión de ecuaciones matemáticas (GAMS, AMPL), hasta contar con interfaces gráficas más amigables llamadas IDE ("Integrated Development Environment") desde las cuales el usuario puede realizar todas las tareas: especificación y edición de un problema, ejecución, análisis de resultados, etc.. Aquí también se pueden mencionar las nuevas versiones de GAMS (Brooke y otros, 1996) y AMPL (Fourer y otros, 1993), otros ejemplos son OPL(van Hentenryck, 1999) y AlphaECP (Westerlund y Lundqvist, 2000). Estos progresos que en principio no parecen tan fundamentales comparados con el desarrollo de algoritmos más eficiente, o de un nuevo modelo de representación, ayudan y facilitan, sin embargo, el proceso iterativo de resolución de un problema. Por lo tanto, el contar con herramientas adecuadas y flexibles para la especificación, ejecución y análisis de los resultados obtenidos es vital para el proceso de resolución de un programa matemático.

Por lo expuesto en los párrafos anteriores, el análisis, desarrollo e implementación de modelos matemáticos de optimización en Ingeniería de Procesos, son aspectos claves que afectan directamente la eficiencia en la solución de los problemas. Un mismo problema puede ser desarrollado de manera diferente según las preferencias y habilidades del constructor del modelo. No existen patrones al respecto, si prácticas generales, que según sea el caso de estudio y el modelador involucrado tendrán un gran impacto a la hora de resolverlo, este es uno de los aspectos que se abordan en esta tesis: el análisis de las distintas opciones de modelado que presentan los problemas de optimización de

Ingeniería de Procesos. Otro aspecto importante que se aborda en esta tesis está relacionado con el desarrollo de herramientas flexibles que permitan la especificación y ejecución más adecuadas de los problemas de acuerdo las características de los mismos.

1.2.Modelos matemáticos en Ingeniería de Procesos

1.2.1. Programación mixta-entera.

Un gran porcentaje de los problemas de optimización que se deben resolver en Ingeniería de Procesos involucran ecuaciones lineales, no lineales y decisiones discretas. Ejemplos de decisiones discretas en problemas de Ingeniería de Procesos son: el número de platos de una columna de destilación, la cantidad de equipos en paralelo y el número y ubicación de tanques de almacenamiento intermedio en una planta batch, en síntesis de procesos se debe determinar que unidades de proceso conformarán el esquema de producción y cuáles son las conexiones entre las diferentes unidades. Los modelos más empleados para representar problemas de este tipo son programas matemáticos mixto-enteros ("Mixed Integer Programming" - MIP) (véase Nemhauser y Wolsey, 1988). La investigación sobre este tipo de problemas ha sido significativa en los últimos años. Los esfuerzos fueron dirigidos fundamentalmente a resolver problemas mixto enteros lineales ("Mixed Integer Linear Program" - MILP) aunque se han hecho también significativos progresos en los programas mixto enteros no-lineales ("Mixed Integer Nonlinear Program" - MINLP). En lo que sigue de la presentación nos referiremos a los programas mixto enteros en general como MIP, a los mixto enteros lineales como MILP y a los mixto enteros no-lineales como MINLP.

La optimización mixta entera (MIP) representa una infraestructura poderosa para el modelado matemático de múltiples problemas de optimización que involucran variables continuas y discretas. En los últimos cinco años ha habido un crecimiento pronunciado en el desarrollo de estos modelos en ingeniería de sistemas de proceso (Grossmann y otros, 1996; Grossmann, 1996a; Grossmann y Daichendt, 1996; Pinto y Grossmann, 1998; Shah, 1998; Grossmann y otros, 1999; Grossmann, 1999).

Los métodos y códigos de programación lineal mixta entera (MILP), han estado disponibles y se han aplicado a muchos problemas prácticos por más de veinte años.

La forma más básica de un problema MIP cuando se representa en forma algebraica es la siguiente:

$$\begin{aligned} & \min Z = f(x, y) \\ s.a. \quad & g_j(x, y) \le 0 \quad j \in J \\ & x \in X, y \in Y \end{aligned} \qquad \textit{(MIP)}$$

donde *f(x,y), g (x,y)* son funciones diferenciables convexas, *J* es el conjunto de índices de las desigualdades, *x* e *y* son variables continuas y discretas, respectivamente. *X* comúnmente se considera como un conjunto compacto convexo:

$$X = \left\{ x \mid x \in R^n, Dx \le d, x^{lo} \le x \le x^{up} \right\}$$

donde x^{lo} y x^{up} son la cota inferior y superior de la variable *x* respectivamente, mientras que *Y* corresponde a un conjunto discreto poliédrico de puntos enteros:

$$Y = \left\{ y \mid y \in Z^m, Ay \le a \right\}$$

que en la mayoría de las aplicaciones está restringido a valores de 0-1, $y \in \{0,1\}^m$.

Problemas Lineales

En la mayoría de las aplicaciones de interés, la función objetivo y las funciones de restricción *f(x,y), g(x,y)* son lineales en *y*. Si consideramos que las funciones son lineales en *x* e *y* el problema MIP se reduce a un problema MILP. La forma del problema resultante sería:

$$\begin{aligned} & \min Z = a^T x + b^T y \\ & s.a.: \\ & \quad Cx + Dy \le d \\ & \quad x \in X, \; y \in Y \end{aligned} \qquad \textit{(MILP)}$$

El método de solución que se emplea ampliamente para la resolución del problema MILP es el de Ramificación y Acotamiento ("Branch and Bound" - B&B) propuesto por Lan y Doig (1960) y formalizado por Dakin (1965). El objetivo de este algoritmo es realizar una enumeración de las combinaciones de las variables binarias pero sin la necesidad de examinar todas las combinaciones posibles. La idea básica es representar las posibles combinaciones de variables 0-1 por medio de un árbol binario, en cada nodo del árbol binario resolver un problema lineal relajado en el cual un subconjunto de variables están fijas en valores 0 ó 1, mientras que el resto puede tomar cualquier valor entre esos extremos (incluyendo los valores extremos). El programa lineal relajado que se resuelve en el nodo raíz es aquel en el cual todas las variables binarias se relajan pudiendo tomar cualquier valor entre 0 y 1. La resolución del problema relajado en el nodo raíz Z_I es una cota inferior ("lower bound") para el problema que se está resolviendo. La solución del nodo raíz es la mejor solución que se puede obtener para el problema planteado. Si al resolver este problema todas las variables binarias asumen valores enteros, se ha encontrado la solución del problema, si no es así, se deberá continuar buscando la solución entera. Para ello se abre un nueva rama del árbol en la cual algunas variables binarias se fijan en 0, otras en 1 y el resto se relaja. Existen diversos métodos para elegir las variables que se fijan y las que se dejan libres, desde simples a más complicados. Una vez que se seleccionó el subconjunto de variables binarias que tendrán valores fijos y aquellas que serán variables, se resuelve el problema lineal resultante. Cuando se obtiene una solución entera se dice que se ha obtenido un cota superior del problema Z_S ("upper bound") del problema. Se detiene una ramificación del árbol cuando se encuentra un problema infactible (no es posible encontrar una solución al problema lineal del nodo), cuando la cota superior obtenida es mayor que la cota superior actual ó cuando todas las variables binarias tienen valor 0 ó 1. Si al resolver un problema lineal en un nodo determinado se obtiene una solución entera menor que la cota superior vigente, esta solución obtenida es la nueva cota superior del problema. El algoritmo termina cuando no se pueden abrir más ramas del árbol. En este caso la solución óptima del problema Z^* es igual a la solución obtenida en el nodo con menor cota superior (cota superior vigente) Z_S. La diferencia entre la solución del nodo raíz o cota inferior *(Z_I)* y la solución encontrada *(Z^*)* es conocida como la diferencia de

integralidad ("integrality gap"). Existen numerosos programas de solución comerciales y académicos que resuelven problemas MILP por el método B&B o alguna variante del mismo. Los programas más conocidos son: LINDO, ZOOM, XPRESS, OSL, CPLEX, XA.

Problemas no lineales

Si en cambio las funciones $g(x,y)$ y $f(x,y)$ son no lineales en las variables continuas, el problema es del tipo MINLP y los métodos desarrollados y más empleados para la solución del mismo son:

- ✓ Ramificación y Acotamiento ("Branch and Bound"-B&B) (Gupta y Ravindran, 1985; Nabar y Schrage, 1991; Borchers y Mitchell, 1994; Stubbs y Mehrotra, 1996; Leyffer, 1998) de características similares al expuesto para el caso lineal,
- ✓ Descomposición Generalizada de Benders (DGB) ("Generalized Benders Descomposition" -GBD) (Geoffrion, 1972),
- ✓ Aproximación Exterior (AE) ("Outer Approximation"-OA) (Durán y Grossmann, 1986; Fletcher y Leyffer, 1994),
- ✓ Planos de Corte Extendido (PCE) ("Extended Cutting Plane" -ECP) (Westerlund y Pettersson, 1995).

Los métodos mencionados están relacionados de algún modo con la posibilidad de resolver subproblemas y obtener cotas del problema original con la solución de estos subproblemas. A continuación se describen cada uno de estos algoritmos.

a) Ramificación y Acotamiento ("Branch and Bound")

El método de "Branch and Bound" para el caso no lineal es similar al caso lineal. El mismo consiste en resolver un problema no lineal completamente relajado en las variables binarias en el nodo raíz del árbol obteniéndose así la cota inferior Z_I del problema. Luego se procede al proceso de ramificación y obtención de la cota superior resolviendo en cada uno de los nodos del árbol un problema no lineal ("Nonlinear Program" – NLP) en donde un subconjunto de las variables binarias se han fijado en valores 0 ó 1 y el resto se ha dejado que tomen cualquier valor entre

esos dos extremos. Referencias sobre el algoritmo de "Branch and Bound" no lineal se pueden encontrar en: Gupta y Ravindran, 1985; Nabar y Schrage, 1991; Borchers y Mitchell, 1994.

Este método resulta atractivo si los subproblemas NLP se resuelven con poco esfuerzo, o cuando sólo es necesario resolver unos pocos de estos subproblemas. Esto último sucede cuando se tienen pocas variables discretas, o cuando la diferencia de integralidad ("integrality gap") entre la solución del nodo raíz (problema completamente relajado) y la solución entera es pequeña.

b) Aproximación Exterior

Este método es por lo general iterativo, por cada iteración se resuelven dos subproblemas del problema original, un programa NLP y uno MILP que se denomina problema maestro. El programa no lineal (NLP) es uno en el cual las variables binarias han sido fijadas en valores 0 ó 1. Para la iteración k-ésima la forma de este subproblema es la siguiente:

$$\begin{aligned} & \min Z^k_S = f(x, y^k) \\ s.a. \quad & g_j(x,y^k) \le 0 \qquad j \in J \qquad\qquad (S\text{-}NLP) \\ & x \in X \end{aligned}$$

La solución de este problema da una cota superior Z^k_S para el problema MINLP siempre y cuando (S-NLP) tenga una solución factible. Cuando no es éste el caso, se considera el subproblema siguiente:

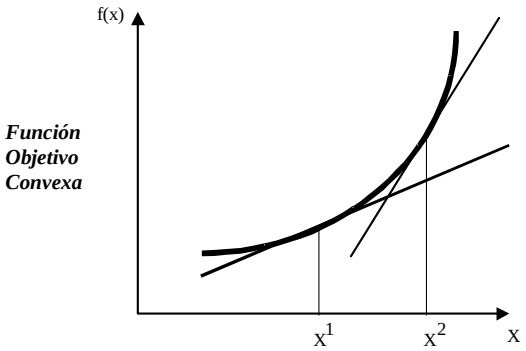
$$\begin{aligned} & \min u \\ s.a. \quad & g_j(x,y^k) \le u \qquad j \in J \qquad\qquad (S\text{-}NLPF) \\ & x \in X \end{aligned}$$

que puede interpretarse como la minimización de la medida de norma infinita de no factibilidad del correspondiente subproblema NLP. Nótese que para un subproblema no factible, la solución de (S-NLPF) da como resultado un valor estrictamente positivo de la variable escalar *u*.

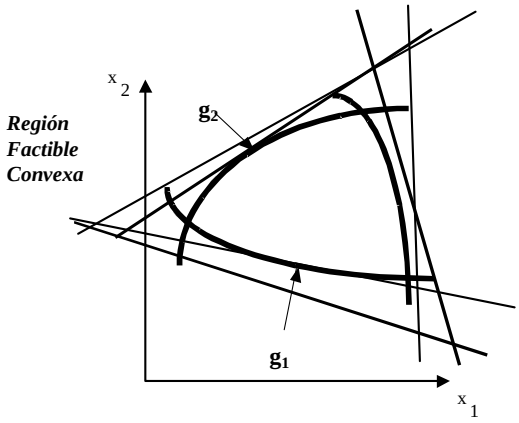
El subproblema maestro MILP es el encargado de proveer una cota inferior para el problema MINLP. La forma del problema maestro para la iteración k es la siguiente:

$$\min \; Z_I^k = \alpha$$

$$\left.\begin{aligned} s.a. \;\; \alpha &\geq f(x^k, y^k) + \nabla f(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \\ g_j(x^k, y^k) &+ \nabla g(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \leq 0 \qquad j \in J^k \end{aligned}\right\} \; k \in K \qquad (M - MILP)$$



Función Objetivo Subestimada



Región Factible Sobreestimada

Figura 1.1: Interpretación geométrica de las linealizaciones en el problema (M-MILP)

En general los valores de y^k *(ó* (x^k,y^k)*)* se obtienen de un problema maestro M-MILP cuyas restricciones están basadas en los planos cortantes que se obtienen (linealizaciones de las ecuaciones no lineales) con la solución de los subproblemas *S-NLP* en las *K* (*k=1..K)* iteraciones previas. Se debe notar aquí que a medida que las iteraciones se van sucediendo el tamaño del problema *(M-MILP)* va creciendo porque se van acumulando estos planos de corte en las iteraciones sucesivas. En la Figura 1.1 se puede ver la interpretación geométrica de las planos de corte que surgen de linealizar las restricciones no lineales del problema *(S-NLP)*, donde se puede ver que con estos hiperplanos, la función objetivo convexa es subestimada y la región factible convexa sobreestimada.

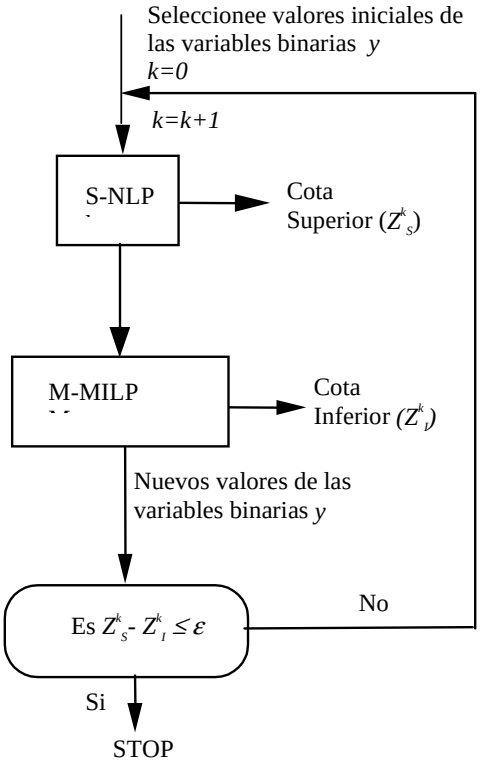


Figura 1.2: Diagrama de flujos del método de Aproximaciones Externas

En la Figura 1.2 se presenta un diagrama de flujos del algoritmo de Aproximaciones Exterior. El algoritmo AE, de acuerdo a la propuesta de Duran y Grossmann (1986), consiste en la ejecución de un ciclo de iteraciones mayores, $k=1,..K$, en las cuales el subproblema *S-NLP* se resuelve para los valores de y^k fijos correspondientes, y el problema maestro M-MILP relajado se actualiza y resuelve con las linealizaciones de las funciones correspondientes en el punto (x^k,y^k) obtenidos en la solución del *S-NLP*.

Con la premisa de convexidad de las funciones *f(x,y)* y *g(x,y)* se puede establecer la siguiente propiedad:

Propiedad 1. *La solución del problema (M-MILP), Z_I^K, corresponde a una cota inferior de la solución del problema (MINLP).*

Esta propiedad se puede verificar en la Figura 1.1. También, dado que las linealizaciones de las funciones se acumulan a medida que se suceden las iteraciones, los problemas maestros (*M-MILP*) generan una secuencia no decreciente de cotas inferiores, $Z_I^1 ... \leq Z_I^k \leq .. \leq Z_I^K$.

El método AE en general requiere de relativamente pocos ciclos de iteraciones mayores. Una razón para este comportamiento proviene de la siguiente propiedad:

Propiedad 2. *El algoritmo AE converge trivialmente en una iteración si f(x,y) y g(x,y) son lineales.*

La prueba se deriva simplemente de que si *f(x,y)* y *g(x,y)* son lineales en *x* y en *y*, el problema maestro (M-MILP) es idéntico al problema original (MINLP).

c) *Descomposición Generalizada de Benders* (DGB)

El método DGB fue propuesto por Geoffrion en 1972. El método DGB (Flippo y Kan 1993) es de naturaleza similar al de Aproximación Exterior. La diferencia se presenta en la definición del problema maestro (M-MILP). En el método DGB sólo se consideran las

desigualdades activas del conjunto $J^k = \{j \mid g_j(x^k, y^k) = 0\}$. El conjunto de las variables continuas $x \in X$ se desecha. En particular, si se asume una aproximación exterior en un punto dado (x^k, y^k),

$$\alpha \geq f(x^k, y^k) + \nabla f(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \qquad (AE^k)$$

$$g_j(x^k, y^k) + \nabla g(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \leq 0 \qquad j \in J^k$$

Haciendo uso de las condiciones de Karush-Kuhn-Tucker y eliminando las variables continuas x, las desigualdades en (AE^k) se pueden reducir como a continuación se indica (Quesada y Grossmann (1992)):

$$\alpha \geq f(x^k, y^k) + \nabla_y f(x^k, y^k)^T (y - y^k) + (\mu^k)^T \left[g_j(x^k, y^k) + \nabla_y g(x^k, y^k)^T (y - y^k) \right] \qquad (CL^k)$$

que corresponde al corte Lagrangiano proyectado en el espacio-y, en la expresión (CL^k), μ^k corresponde a los multiplicadores de Lagrange de la restricción g_j. Esto se puede interpretar como una restricción subrogada de las ecuaciones (AE^k), ya que se obtiene como una combinación lineal de éstas.

De esta manera, suponiendo que todos los subproblemas S-NLP son factibles el problema (M-MILP) se reduce a un problema proyectado en el espacio-y :

$$\min \; Z_L^k = \alpha$$

s.a.:

$$\alpha \geq f(x^k, y^k) + \nabla_y f(x^k, y^k)^T (y - y^k) + \qquad (M\text{-}MILPB)$$
$$+ (\mu^k)^T \left[g_j(x^k, y^k) + \nabla_y g(x^k, y^k)^T (y - y^k) \right] \quad k \in K$$

El algoritmo de resolución del método DGB es el mismo que se mostró en la Figura 1.2 del método AE, reemplazando el problema maestro *M-MILP* por *M-MILPB.*

Como el problema maestro (M-MILPB) se puede derivar del problema maestro (M-MILP), en el contexto del problema (MINLP), la Descomposición Generalizada de Benders se puede considerar como un caso particular del algoritmo de Aproximación Exterior. De hecho, la siguiente propiedad se aplica para los dos métodos (Duran y Grossmann, 1986):

Propiedad 3. *Dado el mismo conjunto de K subproblemas, la cota inferior predicha por el problema maestro relajado del método AE (M-MILP) es mayor o igual a la que se predice por el problema maestro relajado del algoritmo DGB (M-MILPB).*

La prueba de esta propiedad surge del hecho de que los cortes Lagrangiano (CL^k) son subrogados de las aproximaciones exteriores (AE^k). Dado que las cotas inferiores de DGB son generalmente más débiles, este método requiere comúnmente de un mayor número de iteraciones.

La siguiente propiedad de convergencia se aplica al método DGB (Sahinidis y Grossmann, 1991):

Propiedad 4. *Si el problema (MINLP) tiene una diferencia de integralidad ("integrality gap") cero, el algoritmo DGB converge en una sola iteración una vez que se encuentra el óptimo* (x^*, y^*).

La propiedad anterior implica que el único caso en el que se puede esperar que el método DGB termine en una iteración, es cuando el vector discreto inicial es el óptimo, y cuando el valor de la función objetivo de la relajación inicial del problema (MINLP) es el mismo que el de la función objetivo de la solución óptima mixta entera. Dada la relación del método DGB con el algoritmo AE, la Propiedad 4 también la hereda el método AE.

d) Método de Planos de Corte Extendido (PCE) (Westerlund y Pettersson, 1995).

El método PCE, que es una extensión del algoritmo de planos cortantes de Kelley (Kelley, 1960) para problemas NLP convexos, no depende del uso de subproblemas y algoritmos de NLP, porque al algoritmo se basa en la resolución de una secuencia de problemas MILP que tienen la siguiente forma:

$$
\begin{aligned}
& \min\ Z_I^k = \alpha \\
& s.a.\ \ \alpha \geq f(x^k, y^k) + \nabla f(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \\
& \qquad g_j(x^k, y^k) + \nabla g(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \leq 0 \qquad j \in J^k
\end{aligned}
\Bigg\} \quad k \in K \quad \textit{(M - MILPE)}
$$

Cabe notar que es la misma forma que el problema maestro del método de aproximaciones exteriores. El algoritmo se basa en primer lugar el proveer un valor inicial (x^0, y^0) para las variables x e y, evaluar las ecuaciones no lineales en ese punto e incorporar al problema *M-MILPE* una linealización de la restricción más severamente violada en el punto. A medida que las iteraciones avanzan se van incorporando nuevas linealizaciones de la restricción más violadas en el punto (x^k, y^k). La convergencia se alcanza cuando la máxima violación de las restricciones se encuentra dentro de la tolerancia especificada. El valor óptimo de la función objetivo genera una secuencia no decreciente de cotas inferiores. Variantes del algoritmo original contemplan el agregado al problema (M-MILPE) las linealizaciones de todas las restricciones violadas en el conjunto J^k, o linealizaciones de todas las restricciones no lineales $j \in J$.

Nótese que debido a que las variables discretas y continuas convergen simultáneamente, el método PCE puede requerir un gran número de iteraciones. La función objetivo tal como fue definido el método tiene que definirse como lineal, si no es así se puede introducir una nueva variable en la función objetivo y transferir la función no lineal como una restricción.

Los autores del método concluyen que el mismo es apropiado para problemas MINLP grandes pero que tenga pocas restricciones que son suavemente no lineales.

1.2.2.Programación basada en lógica.

Una alternativa para la representación modelos de optimización discretos-continuos es el empleo de lógica (Hooker, 2000). El empleo de lógica en los problemas de programación matemática y de optimización se ha incrementado en los últimos años. Las principales razones se deben a que en muchos casos la lógica presenta una manera

mas natural, directa y sistemática de modelar un problema. La Programación Disyuntiva ("Disjunctive Programming") y la Programación con Restricciones Lógicas ("Constraint Logic Programming" - CLP) (Hajian et al. 1995; Darby-Dowman et al, 1997) son dos áreas del conocimiento que hacen uso de este recurso en el modelado de sus problemas. Ambas han evolucionado de manera independiente. La Programación con Restricciones Lógicas combina un lenguaje poderoso para expresar problemas combinatorios, posee una gran capacidad para propagar las restricciones y una serie de técnicas que aceleran la solución de los problemas.

La programación disyuntiva puede ser vista como un programa mixto-entero lineal o no lineal que involucra disyunciones para el modelado de las decisiones discretas(Balas, 1985; Beaumont, 1990; Raman y Grossmann, 1993, 1994; Turkay y Grossmann, 1996a; Lee y Grossmann, 1999a). En particular, el programa mixto entero (MINLP) también se puede formular como un programa disyuntivo generalizado, de acuerdo a lo propuesto por Raman y Grossmann (1994):

$$
\begin{aligned}
&\min \quad Z = \Sigma_k c_k + f(x) \\
&\text{sujeto a:} \\
&\qquad g(x) \le 0 \\
&\qquad \bigvee_{i \in D_k} \begin{bmatrix} Y_{ik} \\ h_{ik}(x) \le 0 \\ c_k = \gamma_{ik} \end{bmatrix} \quad k \in SD \qquad \textbf{(PDG)} \\
&\qquad \Omega(Y) = True \\
&x \in R^n \ , \ Y_{ik} \in \{True, False\}^m, \ c_k \ge 0
\end{aligned}
$$

en donde Y_{ik} son las variables booleanas que establecen si un término dado en una disyunción es verdadero [$h_{ik}(x) \le 0$], mientras que $\Omega(Y)$ son las relaciones lógicas expresadas en forma de lógica proposicional, que incluyen sólo variables booleanas. Y_{ik} son variables auxiliares que controlan la parte del espacio factible en la que se encuentran las variables continuas, x, y las variables c_k representan costos fijos que se activan a un

valor γ_{ik} si el término correspondiente de la disyunción es verdadero. Finalmente, las condiciones lógicas, $\Omega(Y)$, expresan relaciones entre los conjuntos disyuntivos y por lo general pueden expresarse en Forma Conjuntiva Normal (FCN):

$$\bigwedge_{l=1,2,...L} \left[\bigvee_{(j,k)\in P_l} (Y_{ik}) \bigvee_{(j,k)\in Q_l} (\neg Y_{ik}) \right]$$

donde P_l es el subconjunto de variables Booleanas Y_{ik} que son verdaderas, y Q_l es el subconjunto de variables Booleanas que son falsas en las cláusulas l, l=1,2,…,L. En el problema *(PDG)* $f(x)$, $g(x)$ y $h_{ik}(x)$ se asumen convexas y diferenciables, también se asume que el problema *(PDG)* tiene una región factible compacta y que cada término de la disyunción tiene una región factible no-vacía.

El problema *(PDG)* representa un extensión de la programación disyuntiva de Balas(1985), que en el pasado se usó fundamentalmente como una teoría para derivar planos de corte en problemas MILP. La introducción de los modelos basados en lógica son una opción a los modelos mixto-enteros y en muchos casos forman un complemento y una expansión de este tipo de modelos, ya que éstos pueden ser representados de un modo más natural sin por ello perder ventajas a la hora de resolverlos.

1.3.Ejemplos

A continuación se presenta un ejemplo de síntesis de procesos químicos que es modelado como un programa matemático mixto-entero no lineal (MINLP) y cómo un programa disyuntivo. El problema consiste de 8 procesos interconectados entre si y se debe seleccionar de la superestructura propuesta, el diagrama de flujos que tiene menor costo de inversión y operación. Cada proceso tiene un costo de operación, y las restricciones están planteadas por los balances de masa de cada proceso y una función que representa la eficiencia del mismo. Existen además especificaciones de diseño y de producción del proceso global. En la Figura 1.3 se presenta la superestructura de los 8 procesos

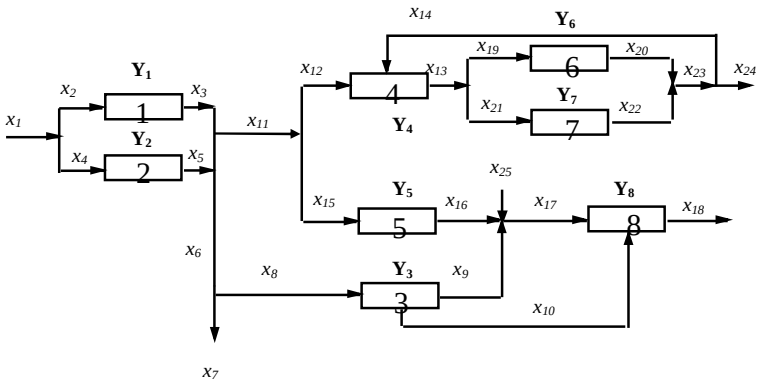


Figura 1.3 : Superestructura de 8 procesos

a) Modelo MINLP

Coeficientes de costo:

$a^T = (a_1 = 0,\ a_{2=}10,\ a_3=1,\ a_4=1,\ a5 = -15,\ a_6=0,\ a_7=0,\ a_8=0,\ a_9=-40,\ a_{10}=15,\ a_{11}=0,\ a_{12}=0,\ a_{13}=0,\ a_{14}=15,\ a_{15}=0,\ a_{16}=0,\ a_{17}=80,\ a_{18}=-65,\ a_{19}=25,\ a_{20}=-60,\ a_{21}=35,\ a_{22}=-80,\ a_{23}=0,\ a_{24}=0,\ a_{25}=-35)$

$x_i^{lo}=0 \quad \forall i$

$$min\ Z = \sum_{k=1}^{8} c_k y_k + a^T x + 122$$

sujeto a :

Balances deMasa :

$$x_1 = x_2 + x_4$$
$$x_6 = x_7 + x_8$$
$$x_3 + x_5 = x_6 + x_{11}$$
$$x_{11} = x_{12} + x_{15}$$
$$x_{13} = x_{19} + x_{21}$$
$$x_9 + x_{16} + x_{25} = x_{17}$$
$$x_{20} + x_{22} = x_{23}$$
$$x_{23} = x_{14} + x_{24}$$

Especificaciones :

$$x_{10} - 0.8\, x_{17} \le 0$$
$$x_{10} - 0.4\, x_{17} \ge 0$$
$$x_{12} - 5\, x_{14} \le 0$$
$$x_{12} - 2\, x_{14} \ge 0$$

Performance de los procesos :

$$exp(x_3) - 1 - x_2 = 0$$
$$exp(x_5 / 1.2) - 1 - x_4 = 0$$
$$1.5 x_9 + x_{10} - x_8 = 0$$
$$1.25(x_{12} + x_{14}) - x_{13} = 0$$
$$x_{15} - 2\, x_{16} = 0$$
$$exp(x_{20} / 1.5) - 1 - x_{19} = 0$$
$$exp(x_{22}) - 1 - x_{21} = 0$$
$$exp(x_{18} / 1.5) - 1 - x_{10} - x_{17} = 0$$

Decisiones discretas

$$x_2 \le 10\, y_1$$
$$x_4 \le 10\, y_2$$
$$x_9 \le 10\, y_3$$
$$x_{12} + x_{14} \le 10\, y_4$$
$$x_{15} \le 10\, y_5$$
$$x_{19} \le 10\, y_6$$
$$x_{21} \le 10\, y_7$$
$$x_{10} + x_{17} \le 10\, y_8$$
$$y_1 + y_2 = 1$$
$$y_4 + y_5 \le 1$$
$$y_6 + y_7 - y_4 = 0$$
$$y_3 - y_8 \le 0$$

donde x es el vector de variables continuas e y el vector de variables 0-1.

b) Modelo Disyuntivo

El modelo disyuntivo para el mismo problema es el siguiente:

$$min\ \ Z = \sum_{k=1}^{8} c_k + a^T x + 122$$

sujeto a :

Balances de masa

$$x_1 = x_2 + x_4$$
$$x_6 = x_7 + x_8$$
$$x_3 + x_5 = x_6 + x_{11}$$
$$x_{11} = x_{12} + x_{15}$$
$$x_{13} = x_{19} + x_{21}$$
$$x_9 + x_{16} + x_{25} = x_{17}$$
$$x_{20} + x_{22} = x_{23}$$
$$x_{23} = x_{14} + x_{24}$$

Especificaciones

$$x_{10} - 0.8\, x_{17} \le 0$$
$$x_{10} - 0.4\, x_{17} \ge 0$$
$$x_{12} - 5\, x_{14} \le 0$$
$$x_{12} - 2\, x_{14} \ge 0$$

Disyunciones

$$\begin{bmatrix} Y_1 \\ exp(x_3) - 1 - x_2 = 0 \\ c_1 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_3 = x_2 = 0 \\ c_1 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_2 \\ exp(x_5 / 1.2) - 1 - x_4 = 0 \\ c_2 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_4 = x_5 = 0 \\ c_2 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_3 \\ 1.5x_9 + x_{10} - x_8 = 0 \\ c_3 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_9 = 0 \\ c_3 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_4 \\ 1.25(x_{12} + x_{14}) - x_{13} = 0 \\ c_4 = 10 \end{bmatrix} \vee \begin{bmatrix} \neg Y_4 \\ x_{12} = x_{13} = x_{14} = 0 \\ c_4 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_5 \\ x_{15} - 2\,x_{16} = 0 \\ c_5 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_5 \\ x_{15} = x_{16} = 0 \\ c_5 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_6 \\ exp(x_{20}/1.5) - 1 - x_{19} = 0 \\ c_6 = 7 \end{bmatrix} \vee \begin{bmatrix} \neg Y_6 \\ x_{19} = x_{20} = 0 \\ c_6 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_7 \\ exp(x_{22}) - 1 - x_{21} = 0 \\ c_7 = 4 \end{bmatrix} \vee \begin{bmatrix} \neg Y_7 \\ x_{21} = x_{22} = 0 \\ c_7 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_8 \\ exp(x_{18}/1.5) - 1 - x_{10} - x_{17} = 0 \\ c_8 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_8 \\ x_{10} = x_{17} = x_{18} = 0 \\ c_8 = 0 \end{bmatrix}$$

Logica Proposicional:

$Y_1 \Rightarrow Y_3 \vee Y_4 \vee Y_5$

$Y_2 \Rightarrow Y_3 \vee Y_4 \vee Y_5$

$Y_3 \Rightarrow Y_1 \vee Y_2$

$Y_3 \Rightarrow Y_8$

$Y_4 \Rightarrow Y_1 \vee Y_2$

$Y_4 \Rightarrow Y_6 \vee Y_7$

$Y_5 \Rightarrow Y_1 \vee Y_2$

$Y_5 \Rightarrow Y_8$

$Y_6 \Rightarrow Y_4$

$Y_7 \Rightarrow Y_4$

$Y_8 \Rightarrow Y_3 \vee Y_5 \vee (\neg Y_3 \wedge \neg Y_5)$

$Y_1 \veebar Y_2$

$Y_4 \veebar Y_5$

$Y_6 \veebar Y_7$

donde Y es el vector de variables de Boole, $\Rightarrow$ es el operador implicación, $\vee$ es el operador "o" ("or") y $\veebar$ es el operador "o exclusivo" ("exclusive or").

2

Modelos discretos/continuos basados en disyunciones y variables binarias

2.1. Introducción

En el capítulo anterior se planteo el hecho que los problemas que se deben resolver en Ingeniería de Procesos frecuentemente involucran ecuaciones no lineales y decisiones discretas. Se introdujeron dos maneras diferentes de modelar un problema discreto/continuo no lineal: de manera algebraica (MINLP) ó a través de la introducción de lógica en los modelos, basados en disyunciones y proposiciones lógicas. Se presentaron dos teorías que introducen lógica en sus modelos: Programación con Restricciones Lógicas y la Programación Disyuntiva. La Programación con Restricciones Lógicas se aplica fundamentalmente a problemas de secuenciamiento ("scheduling"), asignación de recursos y problemas de planificación de la producción que son altamente combinatorios (Van Hentenryck, 1989, Van Hentenryck y Saraswat, 1996).

Para la construcción de sus modelos, la Programación con Restricciones Lógicas tiene una sintaxis muy expresiva basada en operadores lógicos ("and", "or" implicación, equivalencia, etc.) y sentencias especiales como por ejemplo "todos-diferentes" ("all-different"). La resolución de estos modelos están basadas en la enumeración de nodos en un árbol (como en el método de Ramificación y Acotamiento), en cada rama del árbol se aplican técnicas de propagación de las restricciones y de reducción del dominio de las variables. Esto hace que la Programación con Restricciones Lógicas esté enfocada fundamentalmente a problemas especiales del tipo combinatorio. Existen diversos programas comerciales y académicos, que están basados en la Programación con Restricciones Lógicas, ejemplos de tales sistemas son ECL^{i}PS^e (Wallace y otros, 1997) e ILOG (ILOG, 1998). En estos sistemas se pueden especificar restricciones condicionales, lógicas, restricciones que involucran restricciones. Estos también cuentan con un lenguaje expresivo de alto nivel para la construcción de sus modelos.

La Programación Disyuntiva (Balas, 1995; Raman y Grossmann, 1994) en cambio, es más general, y se puede aplicar a una gama más amplia de problemas. Es por ello que nuestro estudio se abocará fundamentalmente a profundizar sobre los modelos disyuntivos. Siempre es posible transformar un problema modelado como MINLP en uno PDG y viceversa. Parece más natural, por ejemplo, comenzar un problema formulado como PDG y proposiciones lógicas y luego reformularlo por medio de la transformación

de las disyunciones en desigualdades del tipo “Big-M” o por otras técnicas que se describirán en el capítulo 3 de esta tesis. También es posible transformar las proposiciones lógicas en desigualdades matemáticas. Si bien existen ciertas reglas del arte acerca de como modelar problemas de este tipo, no existe un consenso ni teoría general de cómo desarrollar modelos de manera sistemática. Por lo general los desarrolladores de tales modelos emplean reglas del arte propias adecuadas a las características y necesidades del problema que están resolviendo, que se corrigen a medida que se va experimentando con la solución del mismo

En este capítulo y basado en los trabajos de Raman y Grossmann (1994) y Turkay y Grossmann (1996a) que propusieron esquemas de modelado basados en disyunciones y variables Booleanas, se propone una formulación híbrida que contempla la posibilidad de modelar un problema con disyunciones como en el problema PDG presentado en el primer capítulo, por medio de términos que incluyen variables binarias en ecuaciones algebraicas como en los modelos MINLP ó con una combinación de ambas técnicas. La propuesta tiene como objetivo contar con una representación que sea general (aplicable a cualquier problema) que deje la posibilidad a quien desarrolla el modelo de representar un problema con la técnica más conveniente en cada caso: algebraica, disyuntiva ó híbrida. También se analiza aquí la generalidad de la representación híbrida propuesta. Es por ello que se la compara con la empleada por la Programación con Restricciones Lógicas, y se proponen una serie de transformaciones para convertir restricciones lógicas en la forma disyuntiva propuesta.

2.2. La representación híbrida

La formulación que se presenta a continuación corresponde a la representación híbrida para modelar problemas continuos/discretos lineales o no lineales, en los cuales las decisiones discretas se representan por medio de disyunciones, variables binarias y variables de Booleanas (Vecchietti y Grossmann, 1999):

$$
\begin{aligned}
\min \quad & Z = \Sigma_k c_k + f(x) + d^T y \\
st \quad & \\
& g(x) \le 0 \\
& r(x) + Dy \le 0 \\
& Ay \ge a \qquad \textbf{(PH)} \\
& \bigvee_{i \in D_k} \begin{bmatrix} Y_{ik} \\ h_{ik}(x) \le 0 \\ c_k = \gamma_{ik} \end{bmatrix} \quad k \in SD \\
& \Omega(Y) = True \\
& x \in R^n \ , y \in \{0,1\}^q \ , Y_{ik} \in \{True, False\}^m, c_k \ge 0
\end{aligned}
$$

donde x y c_k son variables continuas, y son variables binarias 0-1, Y_{ik} son variables Booleanas que se emplean para indicar si un determinado término de la disyunción $[h_{ik}(x) \le 0]$ se satisface (verdadero) o no (falso), $\Omega(Y)$ representan a las relaciones (proposiciones) lógicas que existen entre las variables Booleanas, $g(x) \le 0$ son desigualdades lineales/no-lineales que son independientes de las decisiones discretas, $f(x)$ función objetivo lineales/no-lineales, $r(x) + Dy \le 0$ corresponde a ecuaciones algebraicas mixto-enteras, $Ay \ge a$ es un conjunto de desigualdades enteras y $d^T y$ representan a términos de costo lineales. En la formulación presentada previamente se asume que las funciones $f(x)$, $r(x)$, $g(x)$ y $h_{ik}(x)$ son convexas y diferenciables. También se asume que la representación PH tiene una región factible compacta no vacía, y que cada disyunción tiene una región factible no vacía.

En la formulación previa, cuando el problema (PH) no posee restricciones con variables binarias, implicando que no se tendrán decisiones discretas formuladas de esta forma, PH se reduce entonces al Problema Disyuntivo General (PDG) propuesto por Raman y Grossmann (1994):

$$
\begin{aligned}
&\min \quad Z = \Sigma_k c_k + f(x) \\
&\text{sujeto a:} \\
&\qquad g(x) \le 0 \\
&\qquad \bigvee_{i \in D_k} \begin{bmatrix} Y_{ik} \\ h_{ik}(x) \le 0 \\ c_k = \gamma_{ik} \end{bmatrix} \quad k \in SD \qquad \textbf{(PDG)} \\
&\qquad \Omega(Y) = True \\
&x \in R^n \ , \ Y_{ik} \in \{True, False\}^m, \ c_k \ge 0
\end{aligned}
$$

A partir del problema (PH) es posible obtener la representación algebraica correspondiente a los problemas mixto-enteros no lineales (MINLP). Esto ocurre cuando no se emplean disyunciones ni variable Booleanas en la formulación de las decisiones discretas del problema, resultando un problema con la siguiente representación:

$$
\begin{aligned}
&\min \quad Z = f(x) + d^T y \\
&\text{sujeto a:} \\
&\qquad g(x) \le 0 \qquad \textbf{(MINLP-1)} \\
&\qquad r(x) + Dy \le 0 \\
&\qquad Ay \ge a \\
&x \in R^n \ , y \in \{0,1\}^q
\end{aligned}
$$

De esta forma el problema (PH) brinda la flexibilidad de modelar un problema donde las decisiones discretas se basan solamente en disyunciones (PDG), o en ecuaciones algebraicas solamente (MINLP-1), o como una combinación de ambos (PH).

Sin perder generalidad, y a los efectos de enfocar la presentación de este trabajo, en lo que sigue de la presentación hablaremos siempre, a no ser que se indique lo contrario, de problemas no lineales.

2.3. Lógica Proposicional

La integración de la lógica proposicional en problemas mixto-enteros lineales (MILP) de síntesis de procesos fue presentada por Raman y Grossmann (1993). Allí las relaciones lógicas entre los distintos componentes de la superestructura están dadas por una conjunción (términos separados por condiciones "and") de *q* cláusulas diferentes:

$$\Lambda = \{ L_1 \wedge L_2 \wedge ... \wedge L_q \}$$

donde L_i es una proposición lógica expresada en términos de los operadores $\wedge$ ("and"), $\vee$ ("or"), $\neg$(negación, "not"), $\Rightarrow$ (implicación) y $\Leftrightarrow$ (equivalencia), alternativamente también se puede incluir un operador $\veebar$ "o exclusivo".

Este conjunto de cláusulas se puede transformar en la Forma Conjuntiva Normal (Conjunctive Normal Form, CNF) (Clocksin and Mellish, 1981) que puede expresarse como sigue:

$$\Omega = \left[\bigvee_{i \in P_1} Y_i \quad \bigvee_{i \in \overline{P}_1} \neg Y_i \right] \wedge \left[\bigvee_{i \in P_2} Y_i \quad \bigvee_{i \in \overline{P}_2} \neg Y_i \right] \wedge ..... \wedge \left[\bigvee_{i \in P_s} Y_i \bigvee_{i \in \overline{P}_s} \neg Y_i \right]$$

donde P_i y $\overline{P}_i$ son subconjuntos de las variables Booleanas, *s* es el número de cláusulas disyuntivas. La Forma Conjuntiva Normal implica que toda cláusula del conjunto Ω se debe satisfacer. Si bien en los trabajos previos la lógica proposicional se implementó fundamentalmente para reflejar las relaciones estructurales entre las distintas unidades de procesos (Raman y Grossmann, 1993, Turkay y Grossmann, 1996a), la lógica proposicional puede emplearse para reflejar cualquier relación que exista entre las variables Booleanas y por ende entre los distintos términos de las disyunciones, mas allá de los problemas de síntesis de procesos. El conjunto Ω que se encuentra en la forma FCN se puede transformar automáticamente en un set equivalente de desigualdades enteras, reemplazando las variables Booleanas por variables 0-1. La forma de estas transformaciones son las siguientes:

Relación Lógica	***Expresión Lógica***	***Desigualdad equivalente***
O Lógico	$P_1 \vee P_2 \vee ... \vee P_n$	$y_1 + y_2 + ... + y_n \geq 1$
Y Lógico	$P_1 \wedge P_2 \wedge ... \wedge P_n$	$y_1 \geq 1; y_2 \geq 1; ... y_n \geq 1$
Implicación	$P_1 \Rightarrow P_2$	$1 - y_1 + y_2 \geq 1$
Equivalencia	$P_1 \Leftrightarrow P_2$	$y_1 - y_2 \leq 0$
O Exclusivo	$P_1 \veebar P_2 \veebar ... \veebar P_n$	$y_1 + y_2 + ... + y_n = 1$

Michel Tourn (1995) y Juan Jose Gil (2000) han realizado programas de transformación de expresiones lógicas en desigualdades matemáticas en los lenguaje PROLOG y en C^{++} respectivamente.

Es importante destacar que para el caso especial en el cual se tiene un restricción que debe seleccionar a lo sumo un ítem o ninguno, que se puede expresar como desigualdad de la siguiente manera:

$$\sum_i Y_i \leq 1 \qquad (2\text{-}1)$$

es complicado emplear la lógica proposicional para representarlo. Por ejemplo en el caso que se deba seleccionar una o ninguna opción entre tres variables diferentes, empleando la lógica proposicional se tendría:

$$\left[\neg Y_1 \wedge (\neg Y_2 \vee \neg Y_3)\right] \wedge \left[\neg Y_2 \wedge (\neg Y_1 \vee \neg Y_3)\right] \wedge \left[\neg Y_3 \wedge (\neg Y_1 \vee \neg Y_2)\right] \quad (2\text{-}2)$$

Mucho más complicado sería si el conjunto *i* es más grande. No obstante, si se emplea una variable Booleana "slack" *z* junto con un "o exclusivo" para el resto de las variables Booleana Y_i, se llega a la siguiente ecuación:

$$Y_1 \veebar Y_2 \veebar ... \veebar Y_s \veebar z \Rightarrow \sum_i Y_i + z = 1 \qquad (2\text{-}3)$$

De esta forma si *z* es verdadero *(z=1)* el resto de las variables Y_i son falsas $(Y_i=0)$, y si *z* es falsa *(z=0)* solo una Y_i puede ser verdadera $(Y_i=1, Y_j=0, i\neq j)$.

Ejemplo

Sea la lógica proposicional del ejemplo de los 8 procesos presentado al final del capítulo anterior:

Logica Proposicional:

$Y_1 \Rightarrow Y_3 \vee Y_4 \vee Y_5$
$Y_2 \Rightarrow Y_3 \vee Y_4 \vee Y_5$
$Y_3 \Rightarrow Y_1 \vee Y_2$
$Y_3 \Rightarrow Y_8$
$Y_4 \Rightarrow Y_1 \vee Y_2$
$Y_4 \Rightarrow Y_6 \vee Y_7$
$Y_5 \Rightarrow Y_1 \vee Y_2$
$Y_5 \Rightarrow Y_8$
$Y_6 \Rightarrow Y_4$
$Y_7 \Rightarrow Y_4$
$Y_8 \Rightarrow Y_3 \vee Y_5 \vee (\neg Y_3 \wedge \neg Y_5)$
$Y_1 \veebar Y_2$
$Y_4 \veebar Y_5$
$Y_6 \veebar Y_7$

la transformación del conjunto anterior en desigualdades pasa en primer lugar por generar la Forma Conjuntiva Normal:

$\neg Y_1 \vee Y_3 \vee Y_4 \vee Y_5$
$\neg Y_2 \vee Y_3 \vee Y_4 \vee Y_5$
$\neg Y_3 \vee Y_1 \vee Y_2$
$\neg Y_3 \vee Y_8$
$\neg Y_4 \vee Y_1 \vee Y_2$
$\neg Y_4 \vee Y_6 \vee Y_7$
$\neg Y_5 \vee Y_1 \vee Y_2$
$\neg Y_5 \vee Y_8$
$\neg Y_6 \vee Y_4$
$\neg Y_7 \vee Y_4$
$\neg Y_8 \vee Y_3 \vee Y_5 \vee \neg Y_3$
$\neg Y_8 \vee Y_3 \vee Y_5 \vee \neg Y_5$
$Y_1 \veebar Y_2$
$Y_4 \veebar Y_5$
$Y_6 \veebar Y_7$

que puede ser transformada en el siguiente conjunto de desigualdades (manualmente o por medio de los programas de computadora previamente descriptos):

$1-y_1+y_3+y_4+y_5 \geq 1$
$1-y_2+y_3+y_4+y_5 \geq 1$
$1-y_3+y_1+y_2 \geq 1$
$1-y_3+y_8 \geq 1$
$1-y_4+y_1+y_2 \geq 1$
$1-y_4+y_6+y_7 \geq 1$
$1-y_5+y_1+y_2 \geq 1$
$1-y_5+y_8 \geq 1$
$1-y_6+y_4 \geq 1$
$1-y_7+y_4 \geq 1$
$1-y_8+y_5 \geq 1$
$1-y_8+y_3 \geq 1$
$y_1+y_2=1$
$y_4+y_5=1$
$y_6+y_7=1$

2.4. Transformaciones de restricciones lógicas en disyunciones

En esta sección se pretende demostrar que es posible transformar restricciones lógicas, como aquellas que se encuentran frecuentemente en otras áreas como la Programación con Restricciones Lógicas u otro tipo de restricciones lógicas, en la forma propuesta por el problema (PH). De este modo se pretende mostrar que la representación es lo suficientemente general y que sirve para desarrollar complejos problemas que involucren decisiones discretas en su formulación.

caso a)

El primer caso a contemplar es aquel en el cual el cumplimiento de una restricción $(g(x) \leq 0)$ implica la satisfacción de otra $(f(x) \leq 0)$, término de la derecha de la implicación:

$$g(x) \leq 0 \Rightarrow f(x) \leq 0 \qquad (2\text{-}4)$$

donde ambas $g(x)$ y $f(x)$ son funciones escalares. La implicación puede ser transformada en la siguiente disyunción (sus tablas de verdad son iguales):

$$\neg\, g(x) \leq 0 \ \vee \ f(x) \leq 0 \qquad (2\text{-}5)$$

donde $\vee$ es el operador "o" y $\neg$ es la negación de la restricción o el no cumplimiento de la misma por lo que (2-5) se puede escribir como:

$$g(x) \geq \varepsilon \ \vee \ f(x) \leq 0 \qquad (2\text{-}6)$$

donde ε es un valor suficientemente pequeño para que la restricción no se satisfaga. Los valores típicos que se pueden emplear en la implementación oscilan entre 0.001 y 0.0001.

Si asignamos una variable Booleana a la disyunción anterior (6), se puede obtener una disyunción de dos términos que contempla el modelo (PH):

$$\begin{bmatrix} Y_1 \\ g(x) \geq \varepsilon \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ f(x) \leq 0 \end{bmatrix} \qquad (2\text{-}7)$$

Cuando *g(x)* y *f(x)* son vectores de funciones la transformación se puede hacer de la siguiente forma:

$$\bigwedge_{i \in I} g_i(x) \leq 0 \Rightarrow \bigwedge_{j \in J} f_j(x) \leq 0 \qquad (2\text{-}8)$$

Si se elimina la implicación en (2-8) nos queda:

$$\neg \left(\bigwedge_{i \in I} g_i(x) \leq 0 \right) \vee \left(\bigwedge_{j \in J} f_j(x) \leq 0 \right) \qquad (2\text{-}9)$$

si se traslada la negación adentro de los paréntesis, tenemos:

$$\bigvee_{i \in I} \left(\neg g_i(x) \leq 0 \right) \vee \left(\bigwedge_{j \in J} f_j(x) \leq 0 \right) \qquad (2\text{-}10)$$

lo cual puede ser transformado de la siguiente forma:

$$\bigwedge_{i \in I} \left(\bigvee_{j \in J} \neg g_i(x) \leq 0 \ \vee f_j(x) \leq 0 \right) \qquad (2\text{-}11)$$

finalmente llegamos a la forma disyuntiva comprendida en el problema (PH):

$$\bigvee_{j \in J} \begin{bmatrix} Y_i \\ g_i(x) \geq \varepsilon \end{bmatrix} \vee \begin{bmatrix} Y_j \\ f_j(x) \leq 0 \end{bmatrix} i \in I \qquad (2\text{-}12)$$

Si en cambio se tuviese que la satisfacción de una disyunción de un vector de funciones implica la satisfacción de un vector de funciones la transformación resulta de la siguiente forma:

$$\underset{i\in I}{\vee}\ g_i(x)\le 0 \Rightarrow \underset{j\in J}{\wedge}\ f_j(x)\le 0 \qquad (2\text{-}13)$$

si se elimina la implicación en (2-13) nos queda:

$$\neg(\underset{i\in I}{\vee}\ g_i(x)\le 0\ ) \vee (\underset{j\in J}{\wedge}\ f_j(x)\le 0\ ) \qquad (2\text{-}14)$$

si se traslada la negación adentro de los paréntesis, tenemos:

$$\underset{i\in I}{\wedge}(\ \neg g_i(x)\le 0\ )\vee(\underset{j\in J}{\wedge}\ f_j(x)\le 0\ ) \qquad (2\text{-}15)$$

que puede reescribirse de la siguiente forma:

$$(\underset{i\in I}{\wedge} g_i(x)\ge \varepsilon\ )\vee(\underset{j\in J}{\wedge}\ f_j(x)\le 0\ ) \qquad (2\text{-}16)$$

a partir de (2-16) se puede derivar la forma disyuntiva comprendida en el problema (PH):

$$\begin{bmatrix} Y_i \\ g_i(x)\ge\varepsilon\ \ i\in I \end{bmatrix} \vee \begin{bmatrix} Y_j \\ f_j(x)\le 0\ j\in J \end{bmatrix} \qquad (2\text{-}17)$$

caso b)

$$Y_i \Rightarrow g_i(x) \le 0 \qquad (2\text{-}18)$$

en (2-18) *Yi* es una variable Booleana y $g_i(x)$ un vector de funciones. Para tener en cuenta el caso en el cual la variable Booleana tome el valor falso, se debe expandir la implicación:

$$Y_i \Rightarrow g_i(x) \leq 0 \ \vee \ \neg Y_i \Rightarrow x \in D \qquad (2\text{-}19)$$

El segundo término de la disyunción expresa que si la variable Booleana Y_i es falsa, x puede tomar cualquier valor en el dominio relajado D. La expresión en (2-19) puede transformarse directamente en una disyunción de dos términos contemplada en (PH),

$$\begin{bmatrix} Y_i \\ g_i(x) \leq 0 \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ x \in D \end{bmatrix} \qquad (2\text{-}20)$$

Del mismo modo cuando se tiene la implicación inversa a (18) :

$$g_i(x) \leq 0 \Rightarrow Y_i \qquad (2\text{-}21)$$

La primera transformación resulta en la siguiente expresión:

$$\neg\, g_i(x) \leq 0 \vee Y_i, \qquad (2\text{-}22)$$

A partir de (2-22) se debe contemplar que sucede las variables continuas x cuando es verdadero Y_i. En este caso se dice que x pertenece a un cierto dominio D. Para llegar a la forma final del problema (PH) decimos que el primer término de (2-22) se cumple cuando Y_i es falso:

$$\begin{bmatrix} \neg Y_i \\ g_i(x) \geq \varepsilon \end{bmatrix} \vee \begin{bmatrix} Y_i \\ x \in D \end{bmatrix} \qquad (2\text{-}23)$$

caso c)

$$\Omega_i(Y) \Rightarrow h_i(x) \leq 0 \ \vee \ \neg\Omega_i(Y) \Rightarrow g_i(x) \leq 0 \qquad (2\text{-}24)$$

donde en (2-24) $\Omega_i(Y)$ es un conjunto de proposiciones lógicas que son verdaderas si todas ellas se cumplen. Si se introduce una variable de Boole Z_i que es verdadera cuando el conjunto $\Omega_i(Y)$ es verdadero y falsa en caso contrario, (2-24) se puede expresar como:

$$Z_i \Rightarrow h_i(x) \leq 0 \ \vee \ \neg\, Z_i \Rightarrow g_i(x) \leq 0 \qquad (2\text{-}25)$$

A partir de (2-25) es trivial arribar a una disyunción en la forma (PH):

$$\begin{bmatrix} Z_i \\ h_i(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} \neg Z_i \\ g_i(x) \le 0 \end{bmatrix} \tag{2-26}$$

$$\Omega_i(Y) \Leftrightarrow Z_i$$

Caso d) Disyunciones Embebidas

Un caso adicional que es de interés son las disyunciones embebidas (disyunciones que contienen disyunciones) y que no están directamente contempladas en la formulación propuesta. Un caso de este tipo se puede presentar en problemas de diseño multiperíodo (Van der Heever and Grossmann, 1999). Por ejemplo supongamos que se tiene la siguiente disyunción:

$$\begin{bmatrix} Y_1 \\ \begin{bmatrix} Z_1 \\ h_1(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} Z_2 \\ h_2(x) \le 0 \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ \\ g(x) \le 0 \end{bmatrix} \tag{2-27}$$

que implica que si Y_1=*Verdadero*, una disyunción adicional existe y esta se maneja con las variables Booleanas Z_1 *y* Z_2. Las disyunciones como (2-27) se pueden transformar en disyunciones no-embebidas de la forma (PH). En primer lugar se quita la disyunción embebida en el primer término y ésta se reemplaza por una proposición lógica:

$$\begin{bmatrix} Y_1 \\ Z_1 \vee Z_2 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ g(x) \le 0 \end{bmatrix} \tag{2-28}$$

$$\begin{bmatrix} Z_1 \\ h_1(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} Z_2 \\ h_2(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} \neg Z_1 \wedge \neg Z_2 \\ x \in D \end{bmatrix} \tag{2-29}$$

(28) puede transformarse en el siguiente conjunto de proposiciones lógicas:

$$\begin{aligned} & Y_1 \vee Y_2 \\ & Z_1 \vee Z_2 \Leftrightarrow Y_1 \\ & Y_2 \Rightarrow g(x) \le 0 \end{aligned} \tag{2-30}$$

Si a la implicación (2-30) se aplica la transformación presentada en el caso b) se tiene:

$$\begin{bmatrix} Y_2 \\ g(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x \in D \end{bmatrix} \tag{2-31}$$

Teniendo en cuenta (2-29), (2-30) y (2-31), la forma final de la disyunción (2-27) que contempla (PH) es la siguiente:

$$\begin{bmatrix} Z_1 \\ h_1(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} Z_2 \\ h_2(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} \neg Z_1 \wedge \neg Z_2 \\ x \in D \end{bmatrix} \tag{2-32}$$

$$\begin{bmatrix} Y_2 \\ g(x) \le 0 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x \in D \end{bmatrix} \tag{2-33}$$

$$\begin{gathered} Y_1 \vee Y_2 \\ Z_1 \vee Z_2 \Leftrightarrow Y_1 \end{gathered} \tag{2-34}$$

Lo que las transformaciones desarrolladas en las secciones previas han mostrado es que un rango amplio de restricciones lógicas pueden convertirse a la forma propuesta por el problema PH. Se han transformado restricciones lógicas pertenecientes a Programación con Restricciones Lógicas (CLP) y disyunciones embebidas con lo que un amplio conjunto de problemas pueden formularse como un problema (PH).

Ejemplo de un modelo de Programación con Restricciones Lógicas (CLP) convertido a la forma GDP.

En lo que sigue vamos a mostrar un ejemplo de un problema con restricciones lógicas que se convirtió a la forma GDP por medio de las transformaciones presentadas anteriormente. El problema se formuló originalmente para ser resuelto con el programa comercial ILOG.

Variables continuas: $x \ge -5, \quad y \ge 0$
Variables enteras: $k=0,1,2,3,4$

Restricciones:

$$x^3 + 10\,x = y^x + 2^k$$

$$kx + 7.7\,y = 2.4$$

$$(k-1)^{y+1} \le 10 \qquad (2\text{-}35)$$

$$\{\,[\log(y + 2x+12) \le k+5] \;\vee\; [\,y \ge k^2\,]\,\} \Rightarrow \{\,x \le 0 \wedge y \le 1\,\}$$

$$x \le 0 \Rightarrow\; k > 3$$

Con las transformaciones descritas anteriormente en (6) y (16) el problema se puede modelar en la forma GDP como sigue:

Variables continuas: $x \ge -5,\quad y \ge 0$
Variables enteras: $k=0,1,2,3,4$
Restricciones:

$$x^3 + 10\,x = y^x + 2^k$$

$$kx + 7.7\,y = 2.4$$

$$(k-1)^{y+1} \le 10$$

$$\begin{bmatrix} Y_1 \\ log(y+2x+12)-(k+5) \ge \varepsilon \\ y - k^2 \le \varepsilon \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ x \le 0 \\ y \le 1 \end{bmatrix} \qquad (2\text{-}36)$$

$$\begin{bmatrix} W_1 \\ x \ge \varepsilon \end{bmatrix} \vee \begin{bmatrix} W_2 \\ k > 3 \end{bmatrix}$$

3

Caracterización de las disyunciones y sus relajaciones

3.1. Introducción

En el capítulo anterior se propuso una representación híbrida (problema PH) para los problemas discretos/continuos no lineales en donde las decisiones discretas se formulan como disyunciones o como términos algebraicos que involucran variables binarias. En este punto un aspecto que no esta claro es cómo se deben expresar las decisiones discretas: como una disyunción ó de forma mixta entera. En este contexto existen dos aspectos fundamentales para analizar: una de ellas es la diferencia de integralidad ("integrality gap") que existe entre la solución óptima entera y la relajación continua del problema. Muchos de los algoritmos para la solución de estos problemas involucran en cierto modo la solución de un problema relajado. En este capítulo se van a explorar las diferentes relajaciones posibles que existen de un conjunto disyuntivo: la formulación "Big-M", la subrogada de Beaumont (Beaumont, 1990) y la cáscara convexa ("convex hull") (Balas, 1998, Lee y Grossmann, 1999a), de modo tal que podamos caracterizarlas de acuerdo a cuán ajustada es la aproximación. La formulación "Big-M" y la subrogada de Beaumont implican que las ecuaciones/desigualdades han sido modeladas en forma mixto-entera. La cáscara convexa está directamente relacionada con la formulación disyuntiva. El otro aspecto a analizar es que la formulación, y por lo tanto la relajación elegida, incrementa de diferentes formas el número de variables discretas y continuas, como también las ecuaciones que posee el problema original. Este número incrementado de variables y ecuaciones pueden hacer que un problema sea imposible de resolver, fundamentalmente en problemas grandes que involucran un gran número de variables y ecuaciones. En ciertos casos, la mejor formulación y relajación posible de un problema, implican una cantidad muy grande de variables y ecuaciones. Por lo tanto, dependiendo del caso, existe una situación de compromiso entre la mejor relajación posible y la medida del problema.

El objetivo de este capítulo es el de caracterizar y analizar tanto matemáticamente como a través de ejemplos las relajaciones de las formulaciones "Big-M" y la cáscara convexa. Primero se introducen las definiciones y propiedades de un conjunto disyuntivo, luego se presentan las diferentes relajaciones y sus propiedades. La sección siguiente discute acerca de las relajaciones de un conjunto disyuntivo y se establecen procedimientos y criterios para el modelado de las decisiones discretas. Finalmente se

discutirá el modelado de decisiones discretas cuando se tienen ecuaciones no-lineales. Esto se debe al interés especial que las ecuaciones no-lineales presentan en Ingeniería de Procesos para la satisfacción de los balances de masa y energía de las plantas.

3.2. Definiciones y propiedades de un conjunto disyuntivo.

Un conjunto disyuntivo F puede ser expresado como un conjunto de restricciones separados por el operador "o":

$$F = \bigvee_{i \in D} \left[h_i(x) \leq 0 \right] \quad x \in R^n \tag{3-1}$$

En la expresión (3-1) se asume que $h_i(x) \leq 0$ es continua y convexa. F puede ser considerado como una expresión lógica representada por desigualdades y donde una de ellas debe ser satisfecha. La región factible de cada término de la disyunción se puede expresar como todos los puntos de la variable *x* que satisfacen la desigualdad (región factible):

$$R_i = \{ x \mid h_i(x) \leq 0 \} \tag{3-2}$$

Un conjunto disyuntivo se puede expresar de muchas maneras diferentes que son lógicamente equivalentes y que se pueden obtener unas de otras. F se puede expresar también como la unión de los puntos factibles de los términos de la disyunción que se llama Forma Normal Disyuntiva (FND):

$$F = \bigcup_{i \in D} \left[h_i(x) \leq 0 \right] \quad x \in R^n \tag{3-3}$$

$$F = \bigcup_{i \in D} R_i \tag{3-4}$$

Si la unión de las regiones factibles de los términos de la disyunción es igual a uno de sus términos, por ejemplo R_j, luego este término es aquél que tiene la región factible más grande y el conjunto disyuntivo en este caso es llamado *impropio*, por el

contrario el conjunto disyuntivo es llamado *propio* (Balas, 1985). El conjunto disyuntivo *impropio* se puede escribir de la siguiente manera:

$$F = \bigcup_{i \in D} R_i = R_j \tag{3-5}$$

El conjunto disyuntivo impropio tiene la siguiente propiedad:

$$R_i \subseteq R_j \quad \forall i \neq j \tag{3-6}$$

El significado de (3-6) es que las regiones factibles de los términos i-ésimos, para $i \neq j$ en el conjunto disyuntivo F están incluidos en la región factible del término j-ésimo, luego, dado que F está expresado como la unión de los diferentes términos, el conjunto disyuntivo se puede reducir a:

$$F = \{\, x \mid h_j(x) \leq 0 \,\} \tag{3-7}$$

Por otro lado, el conjunto disyuntivo *propio* es aquel donde la intersección de las regiones factibles de sus términos, o está vacía o está no vacía pero no se aplica la condición (3-5). Luego, para un conjunto disyuntivo propio, se sostiene una de las siguientes condiciones:

$$\bigcap_{i \in D} R_i = \varnothing \tag{3-8}$$

$$\bigcap_{i \in D} R_i \neq \varnothing \quad , \bigcup_{i \in D} R_i \neq R_j \tag{3-9}$$

El significado de la propiedad (3-8) es que todas las regiones factibles de los términos del conjunto disyuntivo *propio* son disyuntos, mientras que (3-9) expresa que las regiones factibles del conjunto disyuntivo propio tienen alguna región de intersección.

3.3 Relajaciones de un conjunto disyuntivo

Dado un conjunto disyuntivo como en (3-1) existen diversas relajaciones que se pueden formular, las más comunes corresponden a: la "Big-M", la subrogada de Beaumont y la cáscara convexa. Dado que las ecuaciones y las restricciones que representan a la cáscara convexa son diferentes dependiendo si $h_i(x) \leq 0$ es lineal o no lineal, en lo que sigue se van a presentar ambos casos separadamente.

Caso lineal

Para el caso lineal el conjunto F en (3-1) se puede escribir como:

$$F = \bigvee_{i \in D} \left[a_i^T x \leq b_i \right] \quad x \in R^n \tag{3-10}$$

a)Relajación "Big-M"

La relajación "Big-M" para el conjunto F en (3-10) es la siguiente:

$$a_i^T x \leq b_i + M_i (1 - y_i) \tag{3-11}$$

$$\sum_i y_i = 1 \tag{3-12}$$

donde $y_i \in \{0,1\}$ y M_i es un escalar cuyo valor es suficientemente grande tal que si $y_i=0$, (3-11) es redundante, si $y_i=1$ la restricción (3-11) se debe satisfacer. El valor más ajustado que se puede determinar para M_i esta dada por:

$$M_i = max\{ a_i^T x - b_i \mid x^{lo} \leq x \leq x^{up} \} \tag{3-13}$$

Notar que para poder calcular el valor de M_i se deben proveer las cotas superior e inferior de las variables. Cuanto más ajustados son los valores de las cotas mejor puede ser la relajación. La determinación de M_i no es siempre trivial, especialmente cuando las cotas de las variables no están bien definidas, en este caso se deben proveer estimaciones de los valores de las cotas.

b) Relajación de Beaumont

Beaumont(1990) propuso una desigualdad equivalente para el conjunto disyuntivo (3-10). Esta desigualdad representa una relajación continua comparada con la formulación 0-1 tradicional de (3-10). En primer lugar, se debe calcular el valor de relajación de cada disyunción:

$$M_i = max\{ a_i^T x - b_i \mid x^{lo} \le x \le x^{up} \} \tag{3-14}$$

cada término de la disyunción se debe relajar como:

$$a^T_i x \le b_i + M_i (1-y_i) \tag{3-15}$$

luego dividiendo cada término de la disyunción por el valor correspondiente por M_i:

$$\frac{a_i^T x}{M_i} \le \frac{b_i}{M_i} + 1 - y_i \tag{3-16}$$

y luego sumando todos los términos de la disyunción se obtiene la relajación propuesta por Beaumont:

$$\sum_i \frac{a_i^T x}{M_i} \le \sum_i \frac{b_i}{M_i} + N - 1 \tag{3-17}$$

donde N es el número de términos de la disyunción (3-10). La expresión (3-17) se llama la subrogada de Beaumont.

c) Relajación por cáscara convexa

La relajación por medio de la cáscara convexa para el conjunto disyuntivo (3-10) fue propuesto en primer lugar por Balas(1979) y se expresa por medio de las siguientes restricciones:

$$
\begin{aligned}
& x - \sum_{i \in D} v_i = 0 \qquad x, v_i \in R^n \\
& a_i^T \; v_i - b_i \; y_i \leq 0 \\
& \sum_{i \in D} y_i = 1 \quad , 0 \leq y_i \leq 1, \; i \in D \\
& 0 \leq v_i \leq y_i \; v_i^{up}
\end{aligned}
\tag{3-18}
$$

Es importante notar que mientras la relajación “Big-M” agrega nuevas variables binarias y una ecuación al conjunto disyuntivo F, la cáscara convexa además de agregar variables binarias, las variables continuas son desagregadas en un nuevo conjunto v_i y nuevas ecuaciones se agregan al conjunto disyuntivo original, mientras que con la relajación de Beaumont la disyunción es reemplazada por una restricción sin agregados de variables ni ecuaciones extras. Este aspecto es importante en el momento de resolver problemas de gran tamaño. Es también importante recalcar en este punto que la subrogada de Beaumont tiene una región factible igual a la de la relajación “Big-M” como fue mostrado por el mismo autor (Beaumont, 1990).

Ejemplos:

Sea la disyunción siguiente:

$$[x_2 \geq x_1 + 1] \; \vee \; [x_1 \geq x_2 + 1] \tag{3-19}$$

donde las cotas de las variables son: $0 \leq x_1 \leq 8$, y $0 \leq x_2 \leq 8$. La representación geométrica de la disyunción se muestra en la Figura 3.1. La formulación “Big-M” para el conjunto disyuntivo (3-19) se muestra en la Figura 3.2. El valor de M_i que se obtiene de aplicar (3-13) es 9.

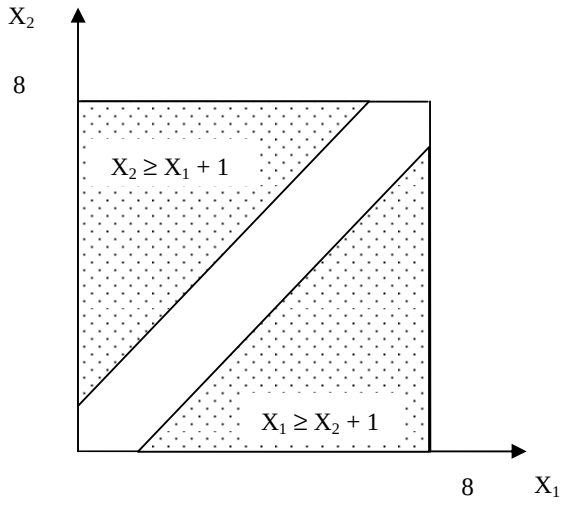


Figura 3.1: representación geométrica de la disyunción (3-19)

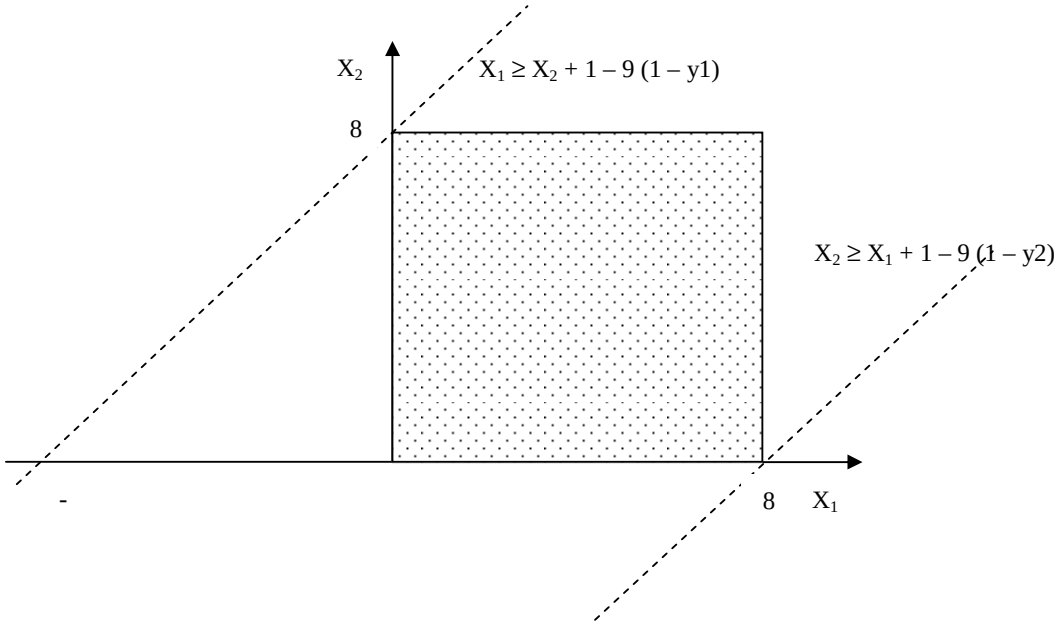


Figura 3.2: Representación geométrica de la relajación "Big-M"de (3-19)

La relajación por medio de la cáscara convexa proyectada en el espacio (x_1, x_2) se muestra en la Figura 3.3. Las ecuaciones de la cáscara convexa para este ejemplo son las siguientes:

$$
\begin{aligned}
& x_1 = v_{11} + v_{12} \\
& x_2 = v_{21} + v_{22} \\
& v_{21} - v_{11} \geq y_1 \\
& v_{12} - v_{22} \geq y_2 \\
& y_1 + y_2 = 1 \\
& 0 \leq v_{11} \leq y_1 \, v_1^{up} \\
& 0 \leq v_{12} \leq y_2 \, v_1^{up} \\
& 0 \leq v_{21} \leq y_1 \, v_2^{up} \\
& 0 \leq v_{22} \leq y_2 \, v_2^{up}
\end{aligned}
\qquad \text{(3-20)}
$$

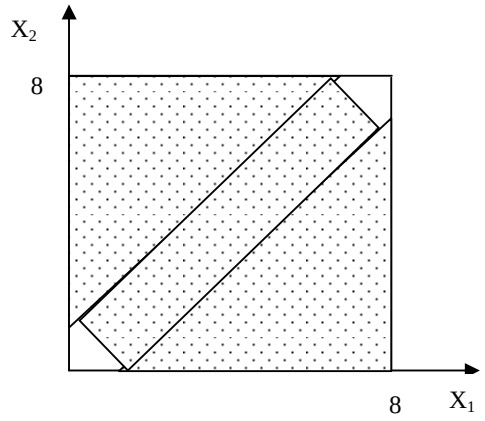


Figura 3.3: Representación geométrica para la cáscara convexa de (3-19)

Notar que para el ejemplo mostrado, la cáscara convexa y la relajación "Big-M" tienen casi la misma región factible. Este no es siempre el caso. Para mostrar esto, supongamos el siguiente ejemplo:

$$
\begin{bmatrix} Y_1 \\ 0 \leq x_1 \leq 2 \\ 0 \leq x_2 \leq 2 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ 3 \leq x_1 \leq 5 \\ 3 \leq x_2 \leq 5 \end{bmatrix} \qquad \text{(3-21)}
$$

La representación geométrica de la disyunción (3-21) para el conjunto disyuntivo original y las relajaciones "Big-M"y de la cáscara convexa se muestran en las Figuras 3.4, 3.5 y 3.6 respectivamente. A partir de las figuras es obvio que la región factible de la cáscara convexa es mas ajustada que la relajación "Big-M".

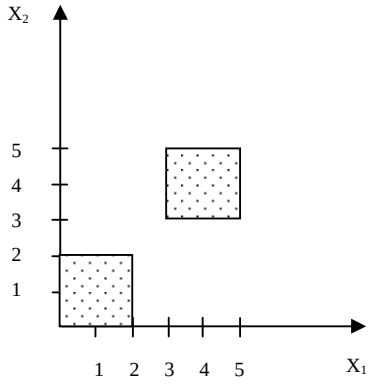


Figura 3.4: Conjunto disyuntivo (3-21)

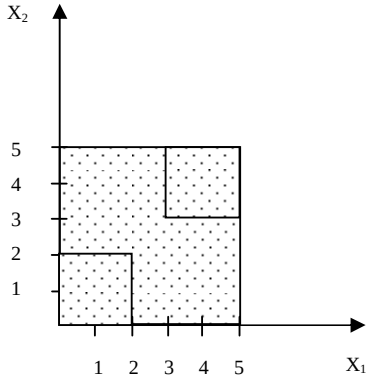


Figura 3.5: Relajación "Big-M" de (3-21)

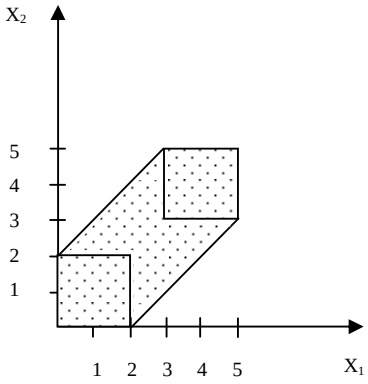


Figura 3.6: Cáscara convexa de (3-21)

Caso no-lineal

b) Relajación "Big-M"

La relajación "Big-M" para el caso no-lineal es similar al caso lineal. Dada la siguiente disyunción:

$$F = \bigvee_{i \in D} \left[h_i(x) \leq 0 \right] \qquad x \in R^n \tag{3-22}$$

se asume que $h_i(x) \leq 0$ es una desigualdad no-lineal continua y convexa, y que por simplicidad y sin por ello perder generalidad (3-22) tiene solo una desigualdad. A partir de estas suposiciones la formulación "Big-M" de (3-22) es :

$$\begin{aligned} & h_i(x) \le M_i(1-y_i) \\ & \sum_i y_i = 1 \end{aligned} \qquad (3\text{-}23)$$

Para este caso, el valor más ajustado que se puede calcular para M_i es:

$$M_i = max\{\, h_i(x) \mid x^{lo} \le x \le x^{up}\} \qquad (3\text{-}24)$$

b) Relajación de Beaumont

Una subrogada similar que en el caso lineal (3-17) se puede obtener para los términos no lineales en las disyunciones. En este caso la subrogada toma la siguiente forma:

$$\sum_i \frac{h_i(x)}{M_i} \le N - 1 \qquad (3\text{-}25)$$

el valor de M_i se puede calcular de manera semejante que en el caso lineal:

$$M_i = max\{\, h_i(x) \le 0 \mid x^{lo} \le x \le x^{up}\} \qquad (3\text{-}26)$$

c) Relajación por cáscara convexa

La cáscara convexa para un conjunto disyuntivo no-lineal disyuntivo se puede escribir como el siguiente conjunto de ecuaciones (Lee y Grossmann, 1999a):

$$\begin{aligned} & x - \sum_{i \in D} v_i = 0 \qquad x, v_i \in R^n \\ & y_i \, h_i(v_i / y_i) \le 0 \\ & \sum_{i \in D} y_i = 1 \quad , 0 \le y_i \le 1, \; i \in D \\ & 0 \le v^i \le v_i^{up} \, y_i \end{aligned} \qquad (3\text{-}27)$$

Las ecuaciones en (3-27) definen un conjunto convexo en el espacio *(x,v,y)*, v_i^{up} define una cota superior válida para las variables desagregadas.

Ejemplo:

$$[(x_1-1)^2+(x_2-1)^2 \leq 1] \vee [(x_1-4)^2+(x_2-2)^2 \leq 1] \vee [(x_1-2)^2+(x_2-4)^2 \leq 1] \quad (3\text{-}28)$$

donde $0 \leq x_1 \leq 5$, y $0 \leq x_2 \leq 5$, la representación geométrica para esta disyunción se muestra en la Figura 3.7. Las Figuras 3.8 y 3.9 muestran la representación geométrica de las relajaciones "Big-M" y la cáscara convexa respectivamente.

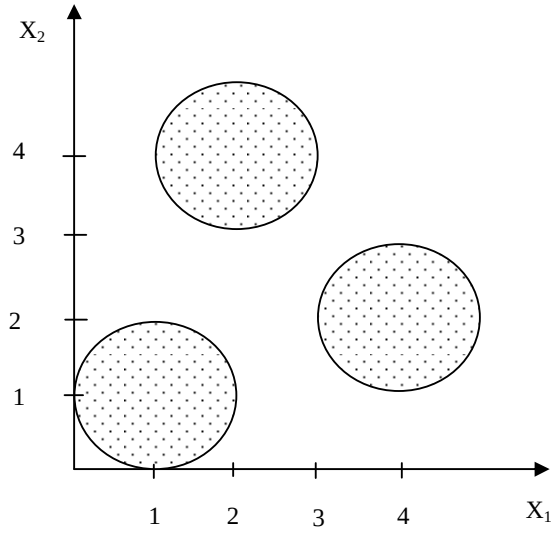


Figura 3.7: Representación geométrica de la disyunción (3-28)

Las ecuaciones para la relajación "Big-M" del ejemplo (3-28) son:

$$\begin{aligned} &(x_1-1)^2+(x_2-1)^2 \leq 1+31(1-y_1) \\ &(x_1-4)^2+(x_2-2)^2 \leq 1+24(1-y_2) \\ &(x_1-2)^2+(x_2-4)^2 \leq 1+24(1-y_3) \end{aligned} \quad (3\text{-}29)$$

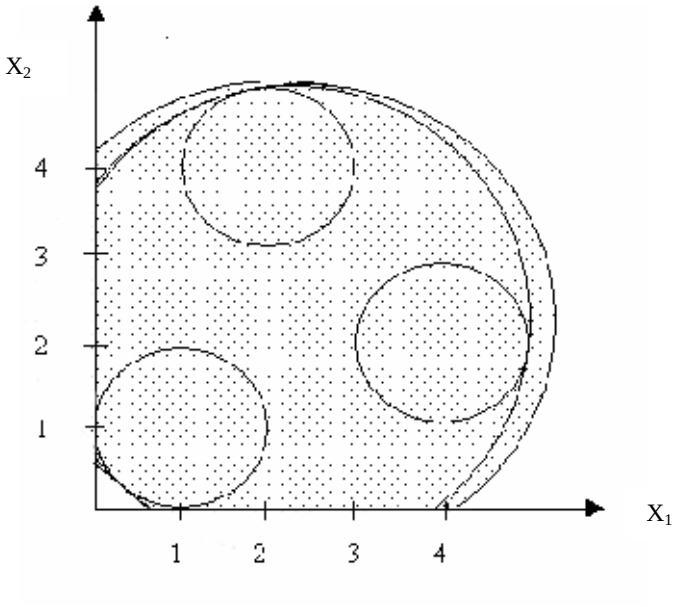


Figura 3.8: Representación geométrica de la relajación "Big-M" de (3-28)

Las ecuaciones para la cáscara convexa de la Figura 3.9 son las siguientes:

$$
\begin{aligned}
& x_1 = v_{11} + v_{12} + v_{13} \\
& x_2 = v_{21} + v_{22} + v_{23} \\
& (\lambda_1+\varepsilon)[(v_{11}/(\lambda_1+\varepsilon)-1)^2 + (v_{21}/(\lambda_1+\varepsilon)-1)^2 - 1] \le 0 \\
& (\lambda_2+\varepsilon)[(v_{12}/(\lambda_2+\varepsilon)-4)^2 + (v_{22}/(\lambda_2+\varepsilon)-2)^2 - 1] \le 0 \qquad (3\text{-}30) \\
& (\lambda_3+\varepsilon)[(v_{13}/(\lambda_3+\varepsilon)-2)^2 + (v_{23}/(\lambda_3+\varepsilon)-1)^2 - 1] \le 0 \\
& \lambda_1 + \lambda_2 + \lambda_3 = 1 \\
& v_{ij} = 5\lambda_j \qquad \forall i, \forall j
\end{aligned}
$$

Notar que para evitar la división por cero en las restricciones se introduce el escalar ε como una tolerancia pequeña. Los valores que usualmente se emplean para ε son 0.001-0.0001.

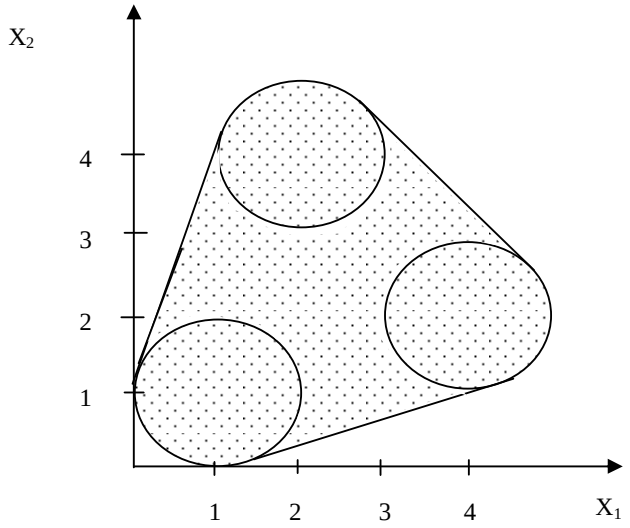


Figura 3.9: Representación geométrica de la cáscara convexa de (3-28)

De las figuras presentada previamente es claro que la cáscara convexa presenta una relajación más ajustada que la relajación "Big-M".

3.4. Propiedades de las relajaciones

La cáscara convexa brinda una relajación que es más ajustada ó igual que la relajación "Big-M". Turkay y Grossmann (1996b) probaron esta propiedad para el caso de disyunciones con términos lineales y Lee y Grossmann (1999a) lo hicieron para el caso no lineal. En las dos secciones que siguen se reproducen ambas demostraciones.

Comparación de las relajaciones "Big-M" y la cáscara convexa.

a) Caso lineal (Turkay and Grossmann, 1996b)

Dada la desigualdad(3-11) y considerando la expansión de los elementos de vector de variables x, (3-11) se puede escribir como:

$$\sum_j a_{ij}\ x_j \le b_i + M(1 - y_i) \qquad (3\text{-}31)$$

Multiplicando (3-31) por y_i nos queda:

$$\sum_j a_{ij}\ x_j\ y_i \le b_i\ y_i + M(1 - y_i)\ y_i \qquad (3\text{-}32)$$

Teniendo en cuenta el término bilineal $x_j y_i$ y definiendo:

$$x_j\ y_i = v_{ij} \qquad (3\text{-}33)$$

haciendo la sumatoria sobre los términos $i \in D$, nos queda:

$$\sum_i x_j\ y_i = \sum_i v_{ij} \qquad (3\text{-}34)$$

Considerando la condición de convexidad de y_i tal que:

$$\sum_i y_i = 1 \qquad (3\text{-}35)$$

implica

$$x_j = \sum_i v_{ij} \qquad (3\text{-}36)$$

reemplazando (33) en (32) nos queda :

$$\sum_i a_{ij}\ v_{ij} \le b_i\ y_i + M(1 - y_i)\ y_i \qquad (3\text{-}37)$$

el segundo término de la derecha de la desigualdad (3-37) es positivo para $0<y_i<1$ dando una relajación más débil que la segunda desigualdad de la cáscara convexa que se presenta en (3-18).

b) Caso no lineal (Lee y Grossmann, 1999a)

Una prueba similar se puede derivar para el caso no-lineal. Considerando la primera desigualdad de (3-27) y multiplicando ambos términos por y_i nos queda:

$$y_i\ h_i(x) \le M(1-y_i)\ y_i \qquad (3\text{-}38)$$

Definiendo una nueva variable desagregada $v_i = x y_i$, y sumando esta variable para todo $i \in D$, y considerando que se aplica la condición de convexidad $\sum_i y_i = 1$ se tiene la siguiente ecuación:

$$\sum_i x\, y_i = x = \sum_i v_i \qquad (3\text{-}39)$$

Introduciendo la variable desagregada v_i (3-39) se puede escribir como:

$$y_i\, h_i(v_i / y_i) \leq M(1\text{-}y_i)\, y_i \qquad (3\text{-}40)$$

Dado que el término de la derecha de (3-40) es un valor positivo para $0<y_i<1$, (3-40) da una relajación más débil comparado con la desigualdad en (3-27).

Si llamamos a la región factible de la cáscara convexa R_{CH}, a la región factible de la relajación "Big-M" R_{bM}, y a la región factible de la subrogada de Beaumont R_B, y de acuerdo con las demostraciones presentadas previamente, se pueden establecer las siguientes propiedades:

$$R_{CH} \subseteq R_{bM} \qquad (3\text{-}41)$$

Por otra parte, Beaumont (1990) demostró además que la región factible de la relajación "Big-M" (R_{bM}) es igual a la región factible de R_B, y por lo tanto se tiene la siguiente propiedad:

$$R_{bM} = R_B \qquad (3\text{-}42)$$

3.5. Caracterización de los conjuntos disyuntivos

Aún con las demostraciones previas, el propósito de esta sección es analizar si es conveniente o no transformar un conjunto disyuntivo en una formulación "Big-M" ó una subrogada de Beaumont. El objetivo básico es comparar la relajación de estas restricciones versus la relajación de la cáscara convexa. La relajación "Big-M" es igual de ajustada que la de Beaumont y es más frecuentemente usada, por lo tanto en las secciones siguientes se compararán ejemplos y resultados sólo con la relajación Big-M. Las conclusiones a las que se arribe serán similares que con la relajación de Beaumont. Se analizarán los siguientes casos: a) cuando el conjunto disyuntivo es *impropio*, b) cuando el conjunto disyuntivo es *propio*, dentro de este último caso se analizará cuando la intersección de la regiones factibles de los términos de la disyunción es no vacía y cuando es el conjunto vacío.

a) Disyunción impropia

Cuando el conjunto disyuntivo es *impropio*, se aplican las propiedades expresadas por (3-6) y (3-7), y por lo tanto, ambas relajaciones la cáscara convexa y la "Big-M" tienen el mismo valor. Los términos redundantes pueden ser eliminados y el conjunto disyuntivo puede ser representado por el término j-ésimo (R_j) con la región factible más grande. A efectos de ilustrar este caso, supongamos que tenemos el siguiente ejemplo:

$$\begin{aligned} & \min Z = (x_1 - 3.5)^2 + (x_2 - 4.5)^2 \\ & s.t. \\ & \begin{bmatrix} Y_1 \\ 1 \le x_1 \le 3 \\ 2 \le x_2 \le 4 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ 2 \le x_1 \le 3 \\ 3 \le x_2 \le 4 \end{bmatrix} \end{aligned} \qquad (3\text{-}43)$$

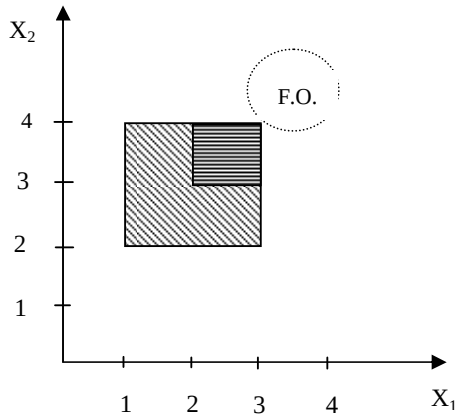


Figura 3.10: Representación geométrica de la disyunción (3-43)

La representación geométrica de la disyunción (3-43) se muestra en la Figura 3.10. Si se elige el término con la región factible más grande que en nuestro ejemplo es el primero, y se resuelve el problema como un NLP se obtiene la solución óptima *x=(3,4) y Z=0.5*. Si uno no se da cuenta que las regiones están incluidas unas en otras, la formulación "Big-M" resulta en el siguiente problema MINLP:

$$
\begin{aligned}
& min\, Z = (x_1 - 3.5)^2 + (x_2 - 4.5)^2 \\
& s.t. \\
& x_1 \geq 1 - M_1(1 - y_1) \\
& x_1 \leq 3 + M_2(1 - y_1) \\
& x_2 \geq 2 - M_3(1 - y_1) \\
& x_2 \leq 4 + M_4(1 - y_1) \\
& x_1 \geq 2 - M_5(1 - y_2) \\
& x_1 \leq 3 + M_6(1 - y_2) \\
& x_2 \geq 3 - M_7(1 - y_2) \\
& x_2 \leq 4 + M_8(1 - y_2) \\
& \\
& x_1, x_2 \in R^2;\ y_1, y_2 \in \{0,1\}
\end{aligned}
\qquad (3\text{-}44)
$$

Si se eligen valores arbitrarios de los parámetros M_i, por ejemplo $M_i=1\ \forall_i$ y se resuelve el MINLP relajado del problema (3-44), se obtiene la solución *x=(3.5,4.5), Z=0, y=(0.5,0.5)*. Si se eligen valores mayores de M_i, por ejemplo $M_i=5\ \forall_i$ y se resuelve nuevamente el problema MINLP (3-44) relajado la solución en este caso *es x=(3.5,4.5), Z=0, y=(0.1,0.9)*. Por lo tanto, en ambos casos, el método de resolución continuará la búsqueda hasta encontrar la solución óptima entera.

b) *Disyunción propia : regiones factibles con intersección no vacía*

El caso que las regiones factibles de los términos de la disyunción cuentan con un conjunto de puntos de intersección se debe analizar cuidadosamente. A priori, no es claro si la cáscara convexa ó la formulación "Big-M" tienen o no la misma relajación. Supongamos que tenemos disyunciones con términos cuyas representaciones geométricas son como las que se muestran en las Figuras 3.11 y 3.12.

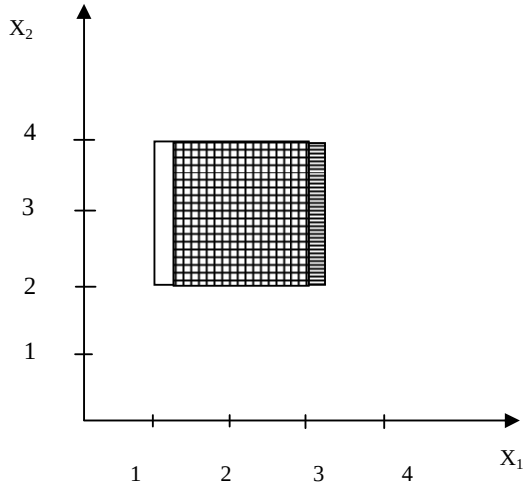


Figura 3.11: Disyunción con términos que se intersectan

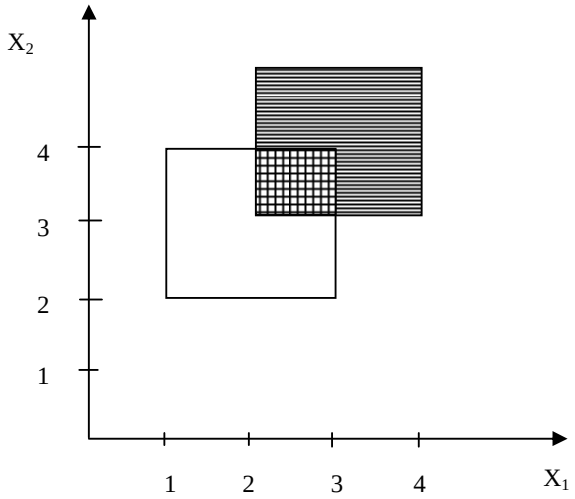


Figura 3.12: Disyunción con términos que se intersectan

En la Figura 3.11 se puede ver que la relajación "Big-M", con una selección apropiada de los valores de M_i puede dar la misma relajación que la cáscara convexa. En el caso de la Figura 3.12 (y en general para la mayoría de los casos prácticos) esta propiedad no es verdadera, y será necesario algún cálculo o etapa de preprocesamiento para determinar que relajación y por ende, que formulación es más apropiada. Supongamos el siguiente problema de optimización:

$$
\begin{aligned}
&\min \quad Z = f(x) \\
&s.a.: \\
&\qquad g(x) \le 0 \\
&\bigvee_{i \in D_k} \begin{bmatrix} Y_{ik} \\ h_{ik}(x) \le 0 \end{bmatrix} \quad k \in SD \qquad \textbf{(PDG)} \\
&\qquad \Omega(Y) = True \\
&x \in R^n \ , \ Y_{ik} \in \{True, False\}^m, \ c_k \ge 0,
\end{aligned}
$$

$\Omega(Y)$: conjunto de proposiciones lógicas en las variables Booleanas

Una estrategia posible es ejecutar el siguiente algoritmo de preprocesamiento:

1- Resolver el siguiente problema habiendo reemplazado la disyunción por la formulación "Big-M":

$$
\begin{aligned}
&\min Z = f(x) \\
&s.a.: \\
&\quad g(x) \le 0 \\
&\quad h_{ik}(x) \le M_{ik}(1 - y_{ik}); \quad i \in D_k, k \in SD \quad \textbf{(Big-M)} \\
&\quad Ay \ge a \\
&\quad \sum_i y_{ik} = 1 \qquad k \in SD \\
&\quad x \in R^n \ , 0 \le y_{ik} \le 1
\end{aligned}
$$

y de esta manera obtener la solución óptima relajada en las variables continuas x^*. Notar que en el problema *(Big-M)* el conjunto de proposiciones lógicas $\Omega(Y)$ ha sido transformado en el conjunto de desigualdades matemáticas $Ay \ge a$.

2- Resolver el siguiente problema de separación, donde las restricciones se corresponden con la cáscara convexa del problema original:

$$
\begin{aligned}
&min\ \ Z = (x - x^*)^T\ (x - x^*) \\
&s.a. \\
&x - \sum_{i \in D_k} v_{ik} = 0 \qquad x, v_{ik} \in R^n, k \in SD, i \in D_k \\
&y_{ik}\ h_{ik}(v_{ik} / y_{ik}) \geq 0 \qquad k \in SD, i \in D \qquad \textbf{(PS)} \\
&Ay \geq a \\
&\sum_{i \in D_k} y_{ik} = 1 \quad , 0 \leq y_i \leq 1,\ i \in D_k, k \in SD, \\
&0 \leq v_{ik} \leq v_{ik}^{up}\ y_{ik}
\end{aligned}
$$

donde x^* corresponde a la solución óptima obtenida en el problem (Big-M).

El objetivo de la solución del problema (PS) es encontrar cuán lejos está el punto obtenido con la relajación "Big-M" de la región factible definida por la cáscara convexa. Si el valor obtenido en la función objetivo *Z* del problema (PS) es 0, esto significa que la solución óptima del problema (Big-M) se encuentra dentro de la región factible de la cáscara convexa, y satisface ambos problemas. Se debe notar que éste puede no ser la mejor solución que uno puede obtener con la cáscara convexa como relajación. En este caso, será necesario un test adicional para determinar que relajación es la más ajustada y mejor. Un test adicional posible es resolver el problema original con la cáscara convexa como relajación del conjunto disyuntivo, y comparar la solución que se obtiene con la de la relajación "Big-M". Por otro lado, si este valor es positivo, la solución óptima x^* de la relajación "Big-M" se encuentra fuera de la región factible de la cáscara convexa. En este caso, es claro que si se obtiene un valor positivo *Z* de la solución del problema (PS) la cáscara convexa tiene una relajación más ajustada. Es importante remarcar algunos aspectos acerca del algoritmo de preprocesamiento presentado previamente:

- la obtención de la solución del problema relajado es frecuentemente menos costosa que la obtención de la solución entera, especialmente en problemas grandes.
- en la primera etapa del algoritmo en lugar de reemplazar las disyunciones por la formulación "Big-M" se pueden reemplazar por la la subrogada de Beaumont
- el algoritmo de preprocesamiento se puede aplicar a solo un subconjunto del conjunto original de disyunciones.

Ejemplo:

$$\min Z = (x_1 - 2)^2 + (x_2 - 5)^2 \qquad (3-45)$$

$$s.a.:$$

$$\begin{bmatrix} Y_1 \\ (x_1-4)^2 + (x_2-2)^2 \le 0.5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1-3)^2 + (x_2-2)^2 \le 1 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1-4)^2 + (x_2-3)^2 \le 1 \end{bmatrix}$$

$$0 \le x_1 \le 5, \quad 0 \le x_2 \le 5,$$

La representación geométrica del este problema se puede ver en la Figura 3.13.

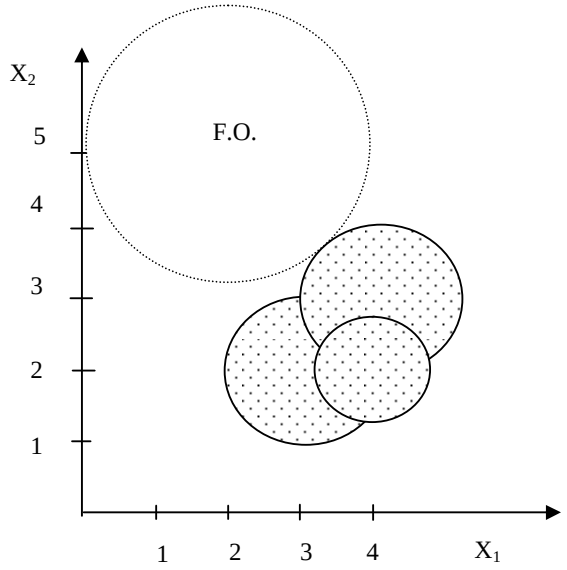


Figura 3.13: Representación geométrica de la disyunción (3-45)

La solución obtenida en la primera etapa del algoritmo de preprocesamiento (problema (Big-M)) con *M=(9.5,12.,4.)* es *x*=(2.469,4.371), Z=0.616*. Con la solución de la segunda etapa del algoritmo de preprocesamiento (problema PS) se obtienen los siguientes valores *x=(3.204,3.635), z=1.08*, donde *Z* es la desviación al cuadrado en el problema (PS). Lo que estos resultados sugieren es que la cáscara convexa tiene una mejor relajación. Para confirmarlo se resolvió el problema original con la relajación de la cáscara convexa como restricciones, se obtuvieron los siguientes resultados

$x^*=(3.271,3.701)$, $Z=3.302$, que son muy próximos a la solución óptima $x^{op}(3.290,3.711)$, $Z^{op}=3.325$.

Si se cambia la función objetivo del problema original, tal que el mínimo caiga dentro de la región de intersección de los términos de la disyunción, los resultados son diferentes. Supongamos el problema anterior pero que el mínimo de la función objetivo caiga en la zona de intersección de las tres regiones:

$$\min Z = (x_1 - 4)^2 + (x_2 - 2)^2 \qquad (3-46)$$

$$s.a.:$$

$$\begin{bmatrix} Y_1 \\ (x_1-4)^2 + (x_2-2)^2 \le 0.5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1-3)^2 + (x_2-2)^2 \le 1 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1-4)^2 + (x_2-3)^2 \le 1 \end{bmatrix}$$

$$0 \le x_1 \le 5, \;\; 0 \le x_2 \le 5$$

La solución obtenida en la primera etapa del algoritmo de preprocesamiento (problema (Big-M)) con valores de $M=(9.5,12.,4.)$ es $x^*=(4,2)$, $Z=0$, $y=(0,0,1)$, que es la solución óptima. Se resolvió la relajación con la cáscara convexa del mismo problema obteniéndose el mismo resultado. Por lo tanto, la conclusión para este caso es que ambas relajaciones tienen el mismo resultado. Si se eligen, en cambio, valores arbitrarios del parámetro M, por ejemplo $M=(7,7,7)$ la solución que se obtiene es la óptima para los valores de las variables continuas $x^*=(4,2)$, $Z=0$, pero la solución es no entera ya que los valores de las variables discretas son $y=(0.333, 0.333, 0.333)$. Por lo tanto, la selección del valor de M es crítico para obtener la mejor relajación.

c) Disyunción propia: intersección de las regiones factibles vacía (términos disyuntos)

De los resultados mostrados en los casos previos parecería que cuando se tiene un conjunto vacío en la intersección de las regiones factibles de los términos de la disyunción, la cáscara convexa daría la mejor relajación. De todos modos, en los casos en que esto no sea así, ó alguna duda exista acerca de la mejor relajación se puede aplicar el algoritmo de preprocesamiento presentado en la sección anterior. Para ilustrar este caso vamos a trabajar con el siguiente ejemplo:

Ejemplo:

$$\min Z = (x_1 - 1)^2 + (x_2 - 1)^2 \qquad (3-47)$$

$$s.a.:$$

$$\begin{bmatrix} Y_1 \\ (x_1 - 4)^2 + (x_2 - 2)^2 \le 0.5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1 - 3)^2 + (x_2 - 4)^2 \le 1 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1 - 1)^2 + (x_2 - 1)^2 \le 1.5 \end{bmatrix}$$

$$0 \le x_1 \le 5, \quad 0 \le x_2 \le 5$$

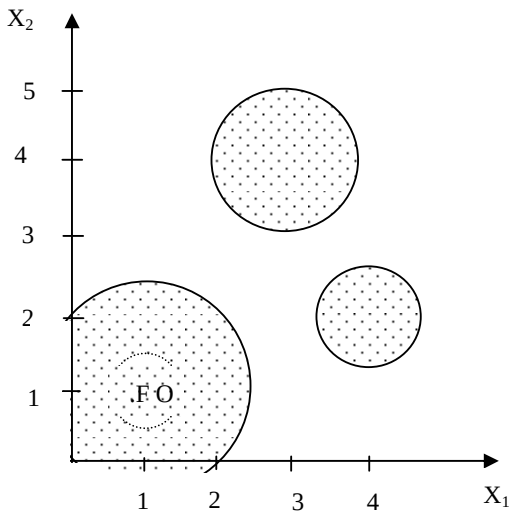


Figura 3.14: representación geométrica del problema (3-47)

En la Figura 3.14 se puede ver que el mínimo de la función objetivo de (3-47) se encuentra en el interior de la región factible de uno de los términos de la disyunción, que en nuestro caso es el tercer término. Con los valores de *M=(19.5,24,30.5)* se obtiene la siguiente solución *x=(1,1), Z=0, y=(0,0.161,0.839).* Con un pequeño cambio en los valores de *M=(19.5,25,30.5)* se obtiene la solución óptima entera *x=(1,1), z=0, y=(0,0,1).* El problema se resolvió también con valores arbitrarios de *M*, los resultados se pueden ver en la Tabla 3.1.

Tabla 3.1: Resultados del ejemplo con valores de M arbitrarios

Problema	M	x_1	x_2	y_1	y_2	y_3	z
Big-M (1)	1	Infactible					-
Big-M (2)	20	1	1	0.525	0.4	0.075	0
Big-M (3)	30	1	1	0	0	1	0

Los valores de la Tabla 3.1 reflejan nuevamente lo crítico que es la elección de los valores de *M* para obtener una buena relajación o aun más alcanzar la solución. Si el valor de *M* no es lo suficientemente grande se puede llegar a una solución del problema infactible porque no se puede encontrar una punto de solución que satisfaga a las tres regiones de la disyunción al mismo tiempo. El segundo valor de *M* (problema Big-M (2)) si bien obtiene la solución continua, los valores de las variables binarias son fraccionales, esto se debe a que el valor de *M* aún no es suficientemente grande como para relajar todas las regiones, con un valor apropiado (problema Big-M(3)) se obtiene la solución óptima.

Si se resuelve la cáscara convexa para el mismo problema se obtiene la solución óptima.

Si se cambia la función objetivo tal que el mínimo no caiga dentro de la región factible de uno de los términos de la disyunción, como por ejemplo en el siguiente problema:

$$\min Z = (x_1 - 2)^2 + (x_2 - 10)^2 \qquad (3-48)$$

$$s.a.:$$

$$\begin{bmatrix} Y_1 \\ (x_1-4)^2 + (x_2-2)^2 \le 0.5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1-3)^2 + (x_2-4)^2 \le 1 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1-1)^2 + (x_2-1)^2 \le 1.5 \end{bmatrix}$$

$$0 \le x_1 \le 5, \;\; 0 \le x_2 \le 5$$

En la Figura 3.15 se tiene la representación geométrica del problema (3-48).

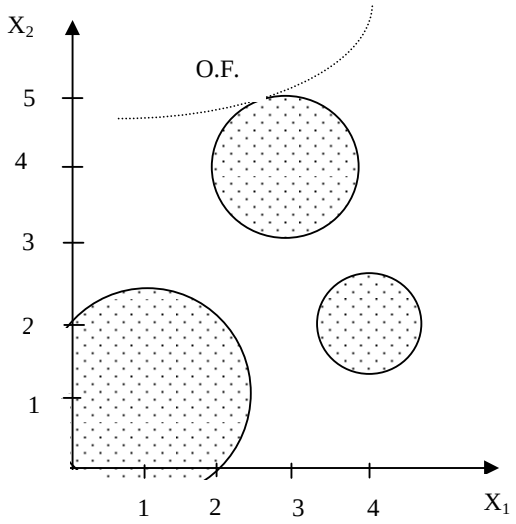


Figura 3.15: Representación geométrica de la disyunción (3-48)

Con la solución de la relajación "Big-M" con *M=(19.5,24,30.5)* se obtienen los siguientes valores *x=(2,5), Z=25.0, y=(0.359,0.328,0.313).* Se probaron varios valores arbitrarios del parámetro *M* pero no fue posible encontrar la solución óptima entera, mientras que resolviendo el problema con la cáscara convexa si es posible. La solución óptima entera para este ejemplo es *x=(2.836, 4.986), Z=25.83, y=(0,1,0).* Las soluciones previas muestran que, en general, la relajación por la cáscara convexa tiene un comportamiento mejor que la "Big-M" ya que, cuando los términos de la disyunción no tienen intersección, siempre se obtuvo la solución óptima entera en la solución del problema relajado. Sin embargo, la relajación "Big-M" en algunos casos también es competitiva ya que con una selección adecuada de los valores de *M* se puede obtener la solución entera en la solución del problema relajado.

d) El criterio "w-MIP"

Otro criterio para decidir que disyunciones se deben convertir a la forma mixta entera es encontrar si una restricción satisface el criterio "w-MIP" o no. Este criterio fue establecido por Raman y Grossmann (1994) y se puede definir como sigue:

Definición: La disyunción lineal (3-10) se puede representar con el criterio "w-MIP" si se satisfacen las siguientes condiciones:

i. Existe un $i^* \in D$ para el cual la cáscara convexa de (3-10) se puede reducir a la siguiente restricción:

$$a_i^T x \le b_i y_i, \qquad 0 \le y_i \le 1$$

ii. Toda solución factible:

$$x' \in F = \{ x \mid \bigvee_{i \in D} \left[a_i^T x \le b_i \right] \},$$

para la cual $a_i^T x' \ge b_i$, $a_{i^*}^T x' \le b_i$, $i \ne i^*$ implica que $y_{i^*}=1$ e $y_i=0 \ \forall i \ne i^*$.

El significado del criterio se mostrará por medio del siguiente ejemplo en el cual se satisface el criterio:

$$\begin{bmatrix} Y_j \\ c_j = f_j \\ 0 \le x_j \le U \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ c_j = 0 \\ x_j = 0 \end{bmatrix} \tag{3-49}$$

donde f_j es una constante fija y U la cota superior de x. Reemplazando la variable Booleana Y_j por la variable binaria correspondiente y_j y aplicando la cáscara convexa, la disyunción anterior se puede modelar como:

$$\begin{aligned} & x_j = v_{j1} + v_{j2} \\ & c_j = w_{j1} + w_{j2} \\ & w_{j1} = f_j y_{j1} \\ & w_{j2} = 0.y_{j2} \\ & y_{j1} + y_{j2} = 1 \\ & 0.y_{j1} \le v_{j1} \le U y_{j1} \\ & v_{j2} \le 0.y_{j2} \\ & v_{j1}, v_{j2} \ge 0 \end{aligned} \tag{3-50}$$

A partir de (3-50) se puede ver que $v_{j2}=0$, $w_{j2}=0$, y que $y_{j2}=1-y_{j1}$, de modo que se pueden expresar sin necesidad de desagregar las variables, por lo tanto (3-50) se puede reducir a la forma mixta-entera siguiente:

$$\begin{aligned} & c_j = f_j\, y_j \\ & x_j \le U y_j \\ & 0 \le y_j \le 1 \\ & x_j \ge 0 \end{aligned} \qquad (3\text{-}51)$$

que claramente satisface las condiciones (i) y (ii) del criterio "w-MIP".

3.6. Disyunciones con restricciones de igualdad

Existen muchas aplicaciones donde los términos disyuntivos contienen ecuaciones en lugar de desigualdades. Por ejemplo, esto es muy común en Ingeniería de Procesos donde los balances de masa y energía se deben satisfacer. Las ecuaciones no lineales son no convexas de modo que las condiciones para aplicar la cáscara convexa no son satisfechas. Para generar una relajación "Big-M" de (3-1) para el caso de ecuaciones, la restricción de igualdad se debe reemplazar por desigualdades dobles como sigue:

$$\begin{aligned} & h_i(x) \le M_i(1-y_i) \\ & h_i(x) \ge -M_i(1-y_i) \end{aligned} \qquad (3\text{-}52)$$

Debido al interés práctico que presentan las igualdades, el objetivo de esta sección es estudiar el caso a través de la solución de diversos ejemplos. La idea es comparar el comportamiento de la relajación "Big-M" y la relajación por cáscara convexa aún cuando las condiciones para aplicar la cáscara convexa no se satisfacen y el problema es de tipo no convexo.

Supongamos el siguiente ejemplo ilustrativo:

$$\begin{gathered} \min Z = (x_1 - 2)^2 + (x_2 - 2)^2 \\ s.a. \\ \begin{bmatrix} Y_1 \\ (x_1)^2 + (x_2)^2 = 1 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1)^2 + (x_2)^2 = 0.5 \end{bmatrix} \\ 0 \le x_1 \le 3, \quad 0 \le x_2 \le 3 \end{gathered} \qquad (3\text{-}53)$$

la representación geométrica de este ejemplo se puede ver en la Figura 3.16.

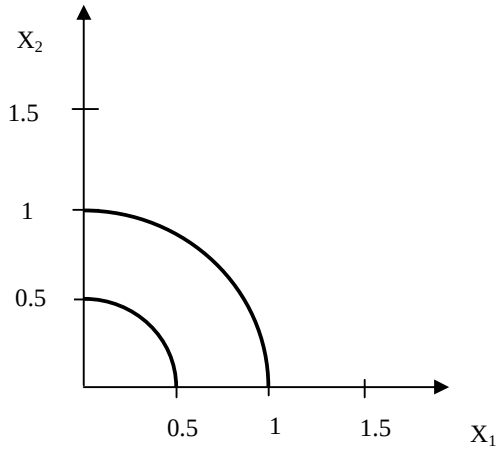


Figura 3.16: representación geométrica de (3-53)

La relajación "Big-M" empleando doble desigualdades para (3-53) es la siguiente:

$$
\begin{aligned}
& \min Z = (x_1 - 2)^2 + (x_2 - 2)^2 \\
& s.a. \\
& (x_1)^2 + (x_2)^2 \le 1 + M_1 (1 - y_1) \\
& (x_1)^2 + (x_2)^2 \ge 1 - M_2 (1 - y_1) \\
& (x_1)^2 + (x_2)^2 \le 0.5 + M_3 (1 - y_1) \\
& (x_1)^2 + (x_2)^2 \ge 0.5 - M_4 (1 - y_1) \\
& \sum_{i=1}^{2} y_i = 1 \\
& 0 \le x_1 \le 3, \qquad 0 \le x_2 \le 3
\end{aligned}
\tag{3-54}
$$

Notar que la primera dificultad que se presenta en la formulación "Big-M" para igualdades es que se deben calcular una mayor cantidad de valores *M*. Si se calculan los valores adecuados para (3-54) *M=(17,1,17.5,0.5)* la solución que se obtiene es la siguiente: *x=(2,2) y=(0.588,0.412), Z=0.* Para este problema si se eligen los siguientes valores de *M=(0,0,0.5,0.5)* se puede obtener la solución entera óptima *x=(0.707,0.707), y=(1,0), Z=3.43* en la solución del problema relajado. Se debe tener en claro que esto se puede hacer porque este ejemplo es pequeño y porque los valores de *M elegidos* se pueden deducir por la forma de la función objetivo y las restricciones de la Figura 3.16.

También se resolvió (3-54) suponiendo valores arbitrarios de *M*. Además se generó la cáscara convexa de (3-53), cuya representación es la siguiente:

$$
\begin{aligned}
& x_1 = v_{11} + v_{12} \\
& x_2 = v_{21} + v_{22} \\
& (\lambda_1+\varepsilon)[(v_{11}/(\lambda_1+\varepsilon))^2 + (v_{21}/(\lambda_1+\varepsilon))^2 - 1] = 0 \qquad (3\text{-}55) \\
& (\lambda_2+\varepsilon)[(v_{12}/(\lambda_2+\varepsilon))^2 + (v_{22}/(\lambda_2+\varepsilon))^2 - 0.5] = 0 \\
& \lambda_1 + \lambda_2 = 1 \\
& v_{ij} = 5\lambda_j \qquad \forall i, \forall j
\end{aligned}
$$

Los valores de ε debieron ser ajustados a 0.000001 para poder arribar a la solución. Los resultados obtenidos para estos casos se pueden ver en la Tabla 3.2.

Tabla 3.2. Resultados obtenidos con distintas relajaciones del ejemplo (3-53)

Problema	M_i	x_1	x_2	y_1	y_2	z
Big-M	2	0.935	0.935	0.625	0.375	2.27
Big-M	5	1.275	1.275	0.55	0.45	1.05
Cáscara convexa	---	0.707	0.707	1	0	3.43

De la tabla y los resultados obtenidos en este ejemplo se puede ver que la cáscara convexa en este caso tiene mejores resultados que la relajación "Big-M", aun cuando las condiciones de convexidad no se aplican a las disyunción original. Con la relajación "Big-M" se obtuvo la solución óptima entera pero con valores muy particulares de *M*. En el siguiente ejemplo una función objetivo y un conjunto disyuntivo diferente fueron seleccionados:

$$
\begin{gathered}
\min Z = (x_1 - 2)^2 + (x_2 - 2)^2 \\
s.a. \\
\begin{bmatrix} Y_1 \\ (x_1 - 0.5)^2 + (x_2)^2 = 0.25 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1 - 1)^2 + (x_2)^2 = 1 \end{bmatrix} \qquad (3\text{-}56) \\
0 \le x_1 \le 3, \qquad 0 \le x_2 \le 3
\end{gathered}
$$

la representación geométrica para este problema se puede ver en la Figura 3.17, donde se puede ver que en este caso las curvas no tienen el mismo centro.

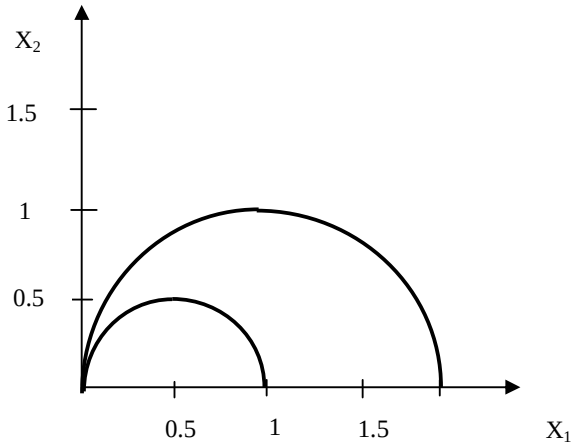


Figura 3.17: representación geométrica de (3-56)

Se generaron las relajaciones "Big-M" con doble desigualdades y de la cáscara convexa para este problema al igual que con el ejemplo anterior. No fue posible para este caso encontrar valores de *M* que pudieran obtener la solución óptima entera en el problema relajado. Con los valores *M=(15,0.25,12,1*) se obtuvieron los siguientes resultados *x=(2,2) y=(0.6,0.4), Z=0.* Otros resultados obtenidos con este ejemplo se muestran en la Tabla 3.3.

Tabla 3.3. Resultados obtenidos con la solución del ejemplo (3-56)

Problema	M_i	x_1	x_2	y_1	y_2	z
Big-M	0.75	0.75	0.968	0	1	2.630
Big-M	2	1.412	1.060	0.147	0.853	1.229
Cácara convexa	---	1.447	0.894	0	1	1.528

De la Tabla 3.3 se puede observar que con valores no apropiados de *M (M=0.75)* se puede obtener un resultado entero incorrecto. Observar que nuevamente la cáscara convexa obtiene el óptimo entero en la solución relajada.

Ejemplo de Síntesis de Procesos

Consideremos ahora la síntesis de redes de procesos que es modelada por medio de la siguiente disyunción:

$$\begin{bmatrix} Y_k \\ h_k(x) = 0 \end{bmatrix} \vee \begin{bmatrix} \neg Y_k \\ B_k x = 0 \end{bmatrix} \qquad k \in K \qquad (3\text{-}57)$$

El significado de (3-57) es que si la unidad de proceso se selecciona *(Y_k = true)* luego se debe satisfacer la restricción del primer término, si no es así ($\neg Y_i$) un subconjunto de las variables de procesos *x* o todas ellas se les asigna el valor 0.

En la Figura 3.18 se presenta el ejemplo visto al final del capítulo 1 de la superestructura de 8 procesos:

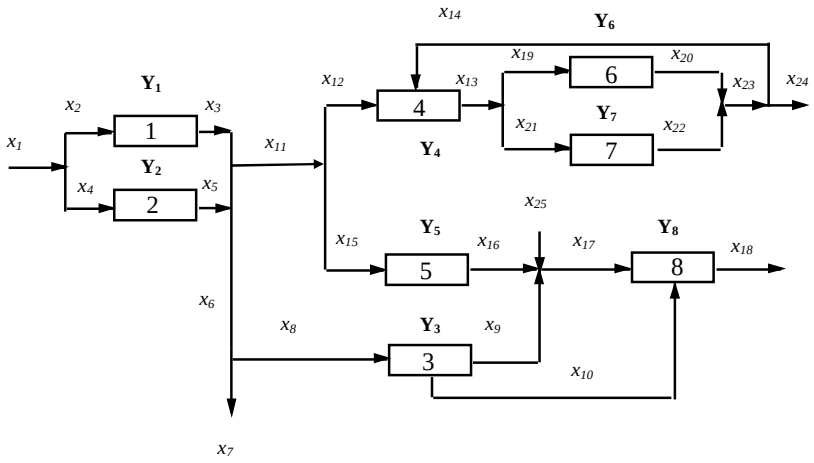


Figure 3.18 : Superestructura de Procesos

Datos del problema:
a^T= (a_1= 0, $a_{2=}$10, a_3=1, a_4=1, a5 =-15, a_6=0, a_7=0, a_8=0, a_9=-40, a_{10}=15, a_{11}=0, a_{12}=0, a_{13}=0, a_{14}=15, a_{15}=0, a_{16}=0, a_{17}=80, a_{18}=-65, a_{19}=25, a_{20}=-60, a_{21}=35, a_{22}=-80, a_{23}=0, a_{24}=0, a_{25}=-35)
x_i^{lo}=0 $\forall i$,

$$min\ \ Z = \sum_{k=1}^{8} c_k + a^T x + 122$$

sujeto a :

Balances de masa

$$x_1 = x_2 + x_4$$
$$x_6 = x_7 + x_8$$
$$x_3 + x_5 = x_6 + x_{11}$$
$$x_{11} = x_{12} + x_{15}$$
$$x_{13} = x_{19} + x_{21}$$
$$x_9 + x_{16} + x_{25} = x_{17}$$
$$x_{20} + x_{22} = x_{23}$$
$$x_{23} = x_{14} + x_{24}$$

Especificaciones :

$$x_{10} - 0.8\, x_{17} \le 0$$
$$x_{10} - 0.4\, x_{17} \ge 0$$
$$x_{12} - 5\, x_{14} \le 0$$
$$x_{12} - 5\, x_{14} \ge 0$$

Disyunciones

$$\begin{bmatrix} Y_1 \\ exp(x_3) - 1 - x_2 = 0 \\ c_1 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_3 = x_2 = 0 \\ c_1 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_2 \\ exp(x_5 / 1.2) - 1 - x_4 = 0 \\ c_2 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_4 = x_5 = 0 \\ c_2 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_3 \\ exp((x_9 + x_{10})/1.5) - 1 - x_8 = 0 \\ c_3 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_9 = 0 \\ c_3 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_4 \\ exp((x_{12}+x_{14})/1.5)-1-x_{13}=0 \\ c_4=10 \end{bmatrix} \vee \begin{bmatrix} \neg Y_4 \\ x_{12}=x_{13}=x_{14}=0 \\ c_4=0 \end{bmatrix} \quad (3\text{-}58)$$

$$\begin{bmatrix} Y_5 \\ exp(x_{15}/2.)-1-x_{16}=0 \\ c_5=6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_5 \\ x_{15}=x_{16}=0 \\ c_5=0 \end{bmatrix}$$

$$\begin{bmatrix} Y_6 \\ exp(x_{20}/1.5)-1-x_{19}=0 \\ c_6=7 \end{bmatrix} \vee \begin{bmatrix} \neg Y_6 \\ x_{19}=x_{20}=0 \\ c_6=0 \end{bmatrix}$$

$$\begin{bmatrix} Y_7 \\ exp(x_{22})-1-x_{21}=0 \\ c_7=4 \end{bmatrix} \vee \begin{bmatrix} \neg Y_7 \\ x_{21}=x_{22}=0 \\ c_7=0 \end{bmatrix}$$

$$\begin{bmatrix} Y_8 \\ exp(x_{18}/1.5)-1-x_{10}-x_{17}=0 \\ c_8=5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_8 \\ x_{10}=x_{17}=x_{18}=0 \\ c_8=0 \end{bmatrix}$$

Proposiciones lógicas:

$Y_1 \Rightarrow Y_3 \vee Y_4 \vee Y_5$
$Y_2 \Rightarrow Y_3 \vee Y_4 \vee Y_5$
$Y_3 \Rightarrow Y_1 \vee Y_2$
$Y_3 \Rightarrow Y_8$
$Y_4 \Rightarrow Y_1 \vee Y_2$
$Y_4 \Rightarrow Y_6 \vee Y_7$
$Y_5 \Rightarrow Y_1 \vee Y_2$
$Y_5 \Rightarrow Y_8$
$Y_6 \Rightarrow Y_4$
$Y_7 \Rightarrow Y_4$
$Y_8 \Rightarrow Y_3 \vee Y_5 \vee (\neg Y_3 \wedge \neg Y_5)$
$Y_1 \veebar Y_2$
$Y_4 \veebar Y_5$
$Y_6 \veebar Y_7$

Una relajación posible para este ejemplo es generar la desigualdades dobles como se hizo en los ejemplos previos pero, para este caso existe otra posibilidad. Notar que para las disyunciones de este ejemplo se tienen las siguientes propiedades:

i. las disyunciones son *impropias* dado que la región factible del segundo término de la disyunción se encuentra dentro de la región factible del primer término. Notar que para que se cumpla esta propiedad el ejemplo original fue modificado en las disyunciones 3, 4 y 5.

ii. las variables de Boole de cada disyunción pueden tomar el valor Verdadero o Falso.

iii. en el segundo término de las disyunciones las variables continuas se hacen iguales a cero.

Debido a que se satisfacen estas condiciones es posible redefinir el conjunto disyuntivo como sigue:

$$exp(x_3) - 1 - x_2 = 0$$
$$exp(x_5 / 1.2) - 1 - x_4 = 0$$
$$exp((x_9 + x_{10}) / 1.5) - 1 - x_8 = 0$$
$$exp((x_{12} + x_{14}) / 1.5) - 1 - x_{13} = 0$$
$$exp(x_{15} / 2.) - 1 - x_{16} = 0$$
$$exp(x_{20} / 1.5) - 1 - x_{19} = 0$$
$$exp(x_{22}) - 1 - x_{21} = 0$$
$$exp(x_{18} / 1.5) - 1 - x_{10} - x_{17} = 0$$

Disyunciones

$$\begin{bmatrix} Y_1 \\ 0 \le x_2 \le x_2^{up} \\ 0 \le x_3 \le x_3^{up} \\ c_1 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ x_3 = x_2 = 0 \\ c_1 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_2 \\ 0 \le x_4 \le x_4^{up} \\ 0 \le x_5 \le x_5^{up} \\ c_2 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_2 \\ x_4 = x_5 = 0 \\ c_2 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_3 \\ 0 \le x_9 \le x_9^{up} \\ c_3 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_3 \\ x_9 = 0 \\ c_3 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_4 \\ 0 \le x_{12} \le x_{12}^{up} \\ 0 \le x_{13} \le x_{13}^{up} \\ 0 \le x_{14} \le x_{14}^{up} \\ c_4 = 10 \end{bmatrix} \vee \begin{bmatrix} \neg Y_4 \\ x_{12} = x_{13} = x_{14} = 0 \\ c_4 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_5 \\ 0 \le x_{15} \le x_{15}^{up} \\ 0 \le x_{16} \le x_{16}^{up} \\ c_5 = 6 \end{bmatrix} \vee \begin{bmatrix} \neg Y_5 \\ x_{15} = x_{16} = 0 \\ c_5 = 0 \end{bmatrix} \qquad (3\text{-}59)$$

$$\begin{bmatrix} Y_6 \\ 0 \le x_{19} \le x_{19}^{up} \\ 0 \le x_{20} \le x_{20}^{up} \\ c_6 = 7 \end{bmatrix} \vee \begin{bmatrix} \neg Y_6 \\ x_{19} = x_{20} = 0 \\ c_6 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_7 \\ 0 \le x_{21} \le x_{21}^{up} \\ 0 \le x_{22} \le x_{22}^{up} \\ c_7 = 4 \end{bmatrix} \vee \begin{bmatrix} \neg Y_7 \\ x_{21} = x_{22} = 0 \\ c_7 = 0 \end{bmatrix}$$

$$\begin{bmatrix} Y_8 \\ 0 \le x_{10} \le x_{12}^{up} \\ 0 \le x_{17} \le x_{13}^{up} \\ 0 \le x_{18} \le x_{14}^{up} \\ c_8 = 5 \end{bmatrix} \vee \begin{bmatrix} \neg Y_8 \\ x_{10} = x_{17} = x_{18} = 0 \\ c_8 = 0 \end{bmatrix}$$

La relajación “Big-M” de este último conjunto de disyunciones son simplemente:

$$x_i \le x_i^{up} y_j \qquad i \in I, j \in J, y_i \in \{0,1\}$$

significando que si la disyunción es verdadera *(y_j=1)* luego las variables continuas x_j y c_k toman valores entre sus cotas para satisfacer las restricciones globales. Si *(y_i=0)* luego las variables continuas se hacen 0 y por lo tanto ocurre los mismo con las ecuaciones globales *(condición ii).*

Se resolvió el problema (3-58) con la relajación por medio de la cáscara convexa. Las cotas superiores adoptadas para las variables de proceso son las siguientes: $x_3^{up}=2$, $x_5^{up}=2$, $x_9^{up}=2$, $x_{10}^{up}=1$, $x_{14}^{up}=1$, $x_{17}^{up}=2$, $x_{19}^{up}=2$, $x_{21}^{up}=2$, $x_{25}^{up}=3$, y para el resto de las variables se adoptó una cota de 6.5 ($x_i^{up}=6.5$). Se obtuvo un valor de relajación de $Z=64.8$. Para la relajación "Big-M" se empleó la formulación (3-59) y las mismas cotas, se obtuvo un valor de $Z=49.9$ para la función objetivo relajada. Debido a que las disyunciones eran impropias se esperaba el mismo valor de relajación para ambas formulaciones. Una explicación posible para este comportamiento es que las no-convexidades hacen que la relajación "Big-M" quede atrapada en un punto sub-óptimo, mientras que la cáscara convexa no. El cambio de la cota de la variable x_{22} a $x_{22}^{up}=0.1$ y resolviendo la relajación "Big-M" de (3-59) se obtiene un valor de $Z=64.4$ muy cerca del valor obtenido por la cáscara convexa.

3.7. Conclusiones

El propósito de este capítulo ha sido analizar las diferentes alternativas de modelado de las decisiones discretas: como disyunciones o como una formulación mixto-entera (0-1). De acuerdo con este estudio, el modelo resultante puede caer en alguno de los tres tipos definidos en el capítulo 2 de esta tesis: modelo (MINLP), disyuntivo (PDG) o modelo híbrido (PH). Para realizar este análisis se han usado las diferentes relajaciones posibles de un conjunto disyuntivo: la cáscara convexa, "Big-M" y la subrogada de Beaumont. El análisis se hizo fundamentalmente usando las dos primera ya que la formulación "Big-M" esta incluida en la subrogada de Beaumont.

Si bien en la sección 3.4 se encuentran las demostraciones que la cáscara convexa tiene una relajación más ajustada que la formulación mixta entera "Big-M", existen algunos casos donde esta última compite con la cáscara convexa. Como una regla general, la formulación "Big-M" es competitiva cuando se tienen buenas cotas de las variables, y para problemas grandes en donde es importante mantener el número de variables y de ecuaciones lo más bajo posible. Para disyunciones convexas *impropias* ambas relajaciones la cáscara convexa y la "Big-M" dan la misma relajación . Para este caso el conjunto disyuntivo puede ser reemplazado por el término con la región factible más

grande. Para disyunciones *propias* donde las regiones factibles de los distintos términos tienen alguna intersección, la función objetivo tiene un rol importante, si el mínimo (ó máximo dependiendo del sentido de la optimización) se encuentra dentro de la región factible de alguno de los términos, es posible obtener el mismo valor de relajación en ambos la formulación "Big-M" o la cáscara convexa, en caso contrario la relajación por la cáscara convexa es mejor, sin embargo la formulación "Big-M" puede ser competitiva según el caso. Es por esto que se propuso un algoritmo de preprocesamiento para determinar en estos casos cuál de las relajaciones es mejor. Para el caso de las disyunciones *propias* con una región de intersección vacía la cáscara convexa es mejor que la "Big-M" aún cuando el mínimo (máximo) de la función objetivo se encuentra dentro de la región factible de uno de los términos. Cuando esto no sea claro, el algoritmo de preprocesamiento enunciado también se puede aplicar para este caso. Para las disyunciones lineales que cumplan con el criterio "w-MIP" se aplicar la formulación 0-1 que propone el criterio.

Para disyunciones no-convexas con restricciones de igualdad la conclusión arribada en los ejemplos analizados es que la cáscara convexa se comporta mejor que la relajación "Big-M".

Si la relajación "Big-M" es similar a la de la cáscara convexa , luego el conjunto disyuntivo puede ser reemplazado por la formulación 0-1 ya que involucra menos ecuaciones y variables, en este caso un modelo originalmente formulado como (PDG) puede ser transformado como un problema (MINLP) o uno (PH).

4

Algoritmos de solución: formulación, propiedades e implementación

4.1.Introducción

Los capítulos 2 y 3 se refirieron fundamentalmente al desarrollo de modelos discretos/continuos no lineales. En el capítulo 2, se propuso una representación híbrida, basada en disyunciones y variables binarias para las decisiones discretas, y se mostró la generalidad de esta representación, ya que de ella se pueden generar un Programa Disyuntivo General (PDG) o uno mixto-entero no lineal (MINLP). También se comparó esta representación con construcciones utilizadas por la Programación con Restricciones Lógicas (PRL), y se propusieron una serie de transformaciones de restricciones características de la PRL en la forma híbrida propuesta. En el capítulo 3, se hizo un análisis de las distintos tipos de disyunciones (*propias e impropias*) y se caracterizaron las mismas para determinar la conveniencia de modelarlas como disyunciones o transformarlas en restricciones mixto-enteras.

En el capítulo 1, se presentaron los algoritmos de resolución de los programas mixto-enteros no lineales(MINLP), pero allí no se incluyeron los métodos de resolución de los problemas disyuntivos e híbridos. Turkay and Grossmann (1996a) propusieron el método de Aproximación Exterior Basado en Lógica ("Logic-Based Outer Approximation") para resolver problemas se síntesis de procesos basados en una formulación (PDG). Este algoritmo, que se presentará mas adelante, está basado en la idea de extender el método de Aproximación Exterior para resolver los subproblemas NLP con un número reducido de ecuaciones, mientras que el suproblema maestro MILP corresponde a la cáscara convexa de las linealizaciones de las desigualdades no lineales. Una extensión de este método fue propuesta para los problemas híbridos y ambos fueron implementados en un código de computadora llamado LOGMIP (Vecchietti y Grossmann, 1999). LOGMIP es un programa de resolución de problemas continuos discretos no lineales formulados ya sea como híbridos, disyuntivos o mixto-enteros. En LOGMIP los métodos implementados para la resolución de los problemas son: el de Aproximación Exterior (AE), el de Descomposición Generalizada de Benders (DGB) y el de Planos Cortantes Extendido para los problemas (MINLP), el de Aproximación Exterior Basado en Lógica para los problemas disyuntivos generalizados(PDG) y una extensión de éste último para los problemas híbridos (PH). Más recientemente, Lee y Grossmann(1999b), basados en las propiedades de la cáscara convexa de un Programa Disyuntivo General (PDG)

propusieron la resolución de los problemas disyuntivos como problemas MINLP. La transformación se realiza por medio de la cáscara convexa o por medio de formulaciones "Big-M". Estos autores propusieron también un algoritmo de Ramificación y Acotamiento ("Branch and Bound") basados en un problema disyuntivo generalizado (PDG).

En este capítulo, el objetivo es presentar los algoritmos de resolución de los programas híbridos (PH) y los disyuntivos generalizados (PDG) existentes, y proponer nuevos métodos y técnicas para su resolución. También se muestran las características de la primera versión del programa de resolución (LOGMIP) desarrollado para problemas disyuntivos no lineales y un conjunto de ejemplos de Ingeniería de Procesos que se resolvieron con el programa.

4.2. Algoritmos

En la Figura 4.1 se presentan de modo general los distintas formulaciones posibles de un problema discreto/continuo no lineal y los algoritmos de resolución de acuerdo con su formulación. A continuación se describen las características y propiedades de los algoritmos que aún no se presentaron.

4.2.1. Aproximación Exterior Basado en Lógica

Este método se aplica para disyunciones que cuentan con solo dos términos que aplican fundamentalmente para problemas de síntesis de procesos. Este método se aplica a problemas que tiene la siguiente representación:

$$\min \; Z = \sum_i c_i + f(x)$$

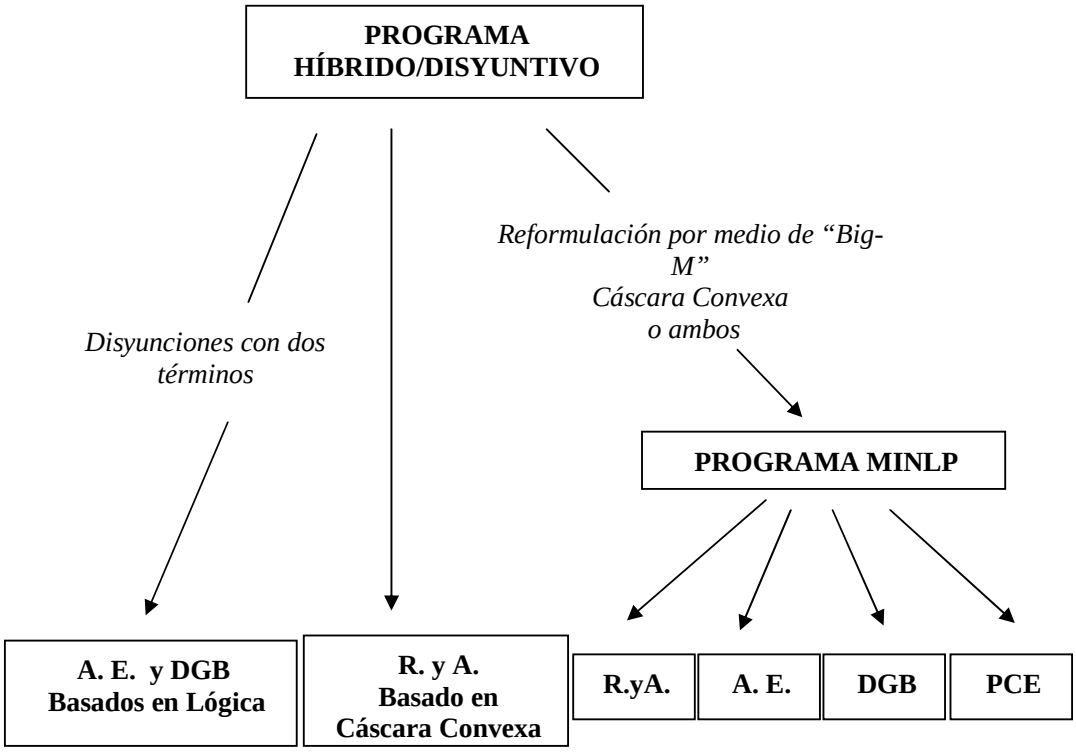
$$\text{sujeto a:}$$

$$g(x) \le 0$$

$$\begin{bmatrix} Y_i \\ h_i(x) \le 0 \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ B^i x = 0 \\ c_i = 0 \end{bmatrix} \quad i \in SD \qquad \text{(PDG-1)}$$

$$\Omega(Y) = \text{Verdadero}$$

$$x \in R^n \;,\; Y \in \{\text{Verdadero, Falso}\}^m,\; c_i \ge 0$$



Donde:
A. E. : Aproximación Exteriror
R. y A. : Ramificación y Acotamiento
DGB : Descomposición Generalizada de Benders
PCE : Planos de Corte Extendido

Figura 4.1: Gráfica general de las formulaciones y sus algoritmos de resolución

el significado de la disyunción presentada es: si Y_i es verdadero se aplican las ecuaciones y el costo del primer término de la disyunción, en cambio si Y_i es falso un subconjunto de las variables continuas x y el costo se hacen 0. El problema (PDG-1) se formuló para ser aplicado particularmente para problemas de síntesis de procesos, pero como se verá en los ejemplos resueltos con LOGMIP, que se presentan al final de este capítulo, también es factible su aplicación para problemas de otro tipo.

Este algoritmo, al igual que el de Aproximación Exterior, se basa en la descomposición del problema original en dos subproblemas uno NLP y un problema maestro MILP. Para este método se asume también que las restricciones no lineales son

convexas. El problema no lineal que se genera a partir del problema (PDG-1) en la iteración k es el siguiente:

$$\begin{aligned} &\min \quad Z = \sum c_i + f(x) \\ &\text{s.a.} \\ &\left.\begin{aligned} &g(x) \le 0 \\ &h_i(x) \le 0 \\ &c_i = \gamma_i \end{aligned}\right\} \quad \text{if } Y_i^k = \text{Verdadero} \\ &\left.\begin{aligned} &B^i x = 0 \\ &c_i = 0 \end{aligned}\right\} \quad \text{if } Y_i^k = \text{Falso} \end{aligned} \qquad \text{(NLP-1)}$$

La dimensión del problema (NLP-1) se reduce porque sólo se consideran en el momento de la solución las ecuaciones cuyas variables Booleanas son verdaderas. Con esto, se evitan incluir en el modelo las ecuaciones no lineales cuyas variables se hacen cero, reduciendo de esta forma las dificultades con singularidades numéricas.

El subproblema MILP maestro tiene la siguiente forma:

$$\begin{aligned} &\min \; Z_L^k = \alpha + \sum_i \gamma_i y \\ &\text{s.a.} \quad \left.\begin{aligned} &\alpha \ge f(x^k) + \nabla f(x^k)^T (x - x^k) \\ &g(x^k) + \nabla g(x^k)^T (x - x^k) \le 0 \end{aligned}\right\} \; k = 1..K \qquad \text{(MILP - 1)} \\ &\nabla h_i(x^k)^T x \le \left[- h_i(x^k) + \nabla h_i(x^k)^T x^k \right] y_i \quad k \in L_K^i, \; i \in SD \\ &Ay \le a \\ &x \in R^n, c \ge 0, y \in \{0,1\}^m \end{aligned}$$

en el cual se asume que todas las variables x en $h_i(x) \le 0$ toman el valor de cero para $y_i = 0$. Para el caso más general, cuando un subconjunto de variables no toman el valor de cero, véase Turkay y Grossmann (1996a) para la forma especifica del MILP.

Notar que el conjunto de proposiciones lógicas $\Omega(Y) = Verdadero$ se transformó en el conjunto de desigualdades matemáticas $Ay \le a$. Es importante hacer notar que, dependiendo del problema, se deben resolver más de un programa (NLP-1) inicial, para poder contar con linealizaciones de todas las ecuaciones no lineales que pertenecen a las disyunciones. En el caso de problemas de síntesis de procesos la determinación de la cantidad de problemas a resolver se puede hacer por medio de la resolución de un

problema de "set covering" como lo señalaron los autores del método (Turkay y Grossmann, 1996a). Otra alternativa es por medio de las proposiciones lógicas que describen las topologías que se pueden usar para determinar las estructuras iniciales y factibles que permiten generar todas las aproximaciones para el primer (MILP-1). Si se asume que se tienen problemas (NLP-1) factibles, el algoritmo puede ser establecido por medio de la ejecución de las siguientes etapas:

1- Modelar el problema discreto/continuo en la forma propuesta por (PDG-1).
2- Identificar los ***nf*** problemas iniciales a ser resueltos por medio del "set covering" ó por estimaciones iniciales (lógica proposicional u otra).
3- Resolver los ***nf*** problemas iniciales (NLP-1) para las topologías elegidas en la etapa 2. El costo más bajo de todos los problemas resueltos es la cota superior Z_S del problema. Obtener en cada problema resuelto en esta etapa las linealizaciones de la función objetivo, las restricciones globales y los términos disyuntivos no lineales.
4- Generar el problema maestro (MILP-1) y resolverlo. La solución de este problema es la cota inferior Z_I del problema.
5- Comparar la cota superior y la inferior. **Si** $|Z_S - Z_I| \leq \varepsilon$, donde ε es una tolerancia, entonces el algoritmo termina y la cota superior actual Z_S es el óptimo. **Si no** se continua en la etapa 6.
6- Resolver un nuevo problema (NLP-1) fijando las variables Booleanas predichas por la resolución del problema (MILP-1). La solución de este problema Z_{NLP-1} se compara con la cota superior actual Z_S. Si $Z_{NLP-1} < Z_S$, entonces $Z_S = Z_{NLP-1}$.
7- Comparar la cota superior y la inferior. **Si** $|Z_S - Z_I| \leq \varepsilon$, el algoritmo termina y la cota superior actual Z_S es el óptimo. **Si no** se continua en la etapa 4.

Cabe señalar que las cotas únicamente pueden ser garantizadas como rigurosas cuando las funciones $f(x)$, $g(x)$ y $h_i(x)$, $i \in SD$ son convexas.

Extensión para problemas híbridos

Este método fue extendido por Vecchietti y Grosmann (1999) para problemas híbridos. Partiendo de la siguiente formulación:

$$\begin{aligned}
&\min\ Z = \sum c_i + f(x) + d^T y \\
&\text{sujeto a:} \\
&\quad r(x) + D\,y \le 0 \\
&\quad g(x) \le 0 \\
&\quad \begin{bmatrix} Y_i \\ h_i(x) \le 0 \\ c_i = \gamma_i \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ B^i x = 0 \\ c_i = 0 \end{bmatrix} \quad i \in SD \qquad \text{(PH-1)} \\
&\quad A\,y \le a \\
&\quad \Omega(Y) = True \\
&\quad x \in R^n\ ,\ y \in \{0,1\},\ Y \in \{True, False\}^m,\ c_i \ge 0
\end{aligned}$$

Las restricciones $r(x)+D\,y \le 0$ pertenecen a disyunciones originales que fueron transformadas en ecuaciones mixto-enteras de acuerdo con los criterios planteados en el capítulo 3. Del mismo modo las restricciones $Ay \le a$ pertenecen a proposiciones lógicas originales transformadas a desigualdades matemáticas (Raman y Grossmann, 1991) .

En este caso también se descompone el problema original en dos subproblemas uno NLP y el otro MILP. El problema NLP para la iteración *k* tiene la siguiente forma:

$$\begin{aligned}
&\min\ Z_S = \sum c_i + f(x) + d^T y^k \\
&\text{s.a.} \\
&\quad r(x) + D\,y^k \le 0 \\
&\quad g(x) \le 0 \\
&\quad \left.\begin{aligned} h_i(x) &\le 0 \\ c_i &= \gamma_i \end{aligned}\right\} \quad si\ Y_i^k = Verdadero \\
&\quad \left.\begin{aligned} B^i x &= 0 \\ c_i &= 0 \end{aligned}\right\} \quad si\ Y_i^k = Falso
\end{aligned} \qquad \textit{(NLP-2)}$$

El problema maestro para el problema híbrido tiene la siguiente representación:

$$\begin{aligned} & \min\ Z_I^k = \alpha + \sum_i \gamma_i y + d^T y \\ s.a.\ & \left.\begin{array}{l} \alpha \geq f(x^k) + \nabla f(x^k)^T (x - x^k) \\ g(x^k) + \nabla g(x^k)^T (x - x^k) \leq 0 \end{array}\right\} \ k = 1..K \qquad (MILP - 2) \\ & r(x^k) + \nabla r(x^k)^T (x - x^k) + Dy \leq 0 \\ & \nabla h_i(x^k)^T x \leq \left[- h_i(x^k) + \nabla h_i(x^k)^T x^k \right] y_i \quad k \in L_K^i,\ i \in SD \\ & Ay \leq a \\ & Ey \leq e \\ & x \in R^n, c \geq 0, y \in \{0,1\}^{q+m} \end{aligned}$$

donde una vez más se supone que todas las variables *x* involucradas en la restricción $h_i(x) \leq 0$ toman el valor de cero para $y_i = 0$.

La resolución de un problema híbrido por el método de Aproximación Exterior Basado en Lógica difiere de los problemas disyuntivos (PDG-1) fundamentalmente en las primeras etapas, en la preparación e inicialización del problema, y en la generación de los subproblemas NLP y maestro. Este algoritmo se puede resumir en los siguientes pasos:

1- Modelar el problema discreto/continuo en la forma propuesta por (PH-1) ó modelarlo como en la forma (PDG-1) y transformar un subconjunto de las disyunciones en la forma mixta entera de acuerdo a lo propuesto en el capítulo 3.
2- Identificar los ***nf*** problemas iniciales a ser resueltos por medio del "set covering" ó por estimaciones iniciales (lógica proposicional u otra).
3- Asignar valores iniciales a las variables binarias de las disyunciones transformadas en mixtas enteras. Las opciones son fijarlas en 0 ó 1 para cada uno de los ***nf*** problemas iniciales ó dejarlas libres (relajadas). Si se decide dejar libres a las variables 0-1 asignar $Z_S = \infty$.
4- Resolver los ***nf*** problemas iniciales (NLP-2) para las topologías elegidas en la etapa 2. Si la inicialización es fija, entonces el costo más bajo de todos los determinados

por la función objetivo es la cota superior Z_S del problema. Obtener en cada problema resuelto en esta etapa las linealizaciones de: la función objetivo, las restricciones globales y los términos disyuntivos no lineales.

5- Generar el problema maestro (MILP-2) y resolverlo. La solución de este problema es la cota inferior Z_I del problema.

6- Comparar la cota superior y la inferior. **Si** $|Z_S - Z_I| \leq \varepsilon$, donde ε es una tolerancia entonces el algoritmo termina y la cota superior actual Z_S es el óptimo. **Si no** se continua en la etapa 7.

7- Resolver un nuevo problema (NLP-2) fijando las variables Booleanas y las binarias predichas por la resolución del problema (MILP-2). La solución de este problema Z_{NLP-1} se compara con la cota superior actual Z_S. Si $Z_{NLP-1} < Z_S$, entonces $Z_S = Z_{NLP-1}$.

8- Comparar la cota superior y la inferior. **Si** $|Z_S - Z_I| \leq \varepsilon$, el algoritmo termina y la cota superior actual Z_S es el óptimo. **Si no** se continua en la etapa 5.

4.2.2. Reformulación de problemas híbridos y disyuntivos a MINLP

Una alternativa para la resolución de un problema disyuntivo o híbrido es la reformulación del mismo como un problema mixto entero (MINLP) por medio de las relajaciones "Big-M" ó la cáscara convexa. Se debe notar aquí que si se reemplaza una disyunción por la subrogada de Beaumont no se genera un problema MINLP. Beaumont ha propuesto un algoritmo del tipo Ramificación y Acotamiento para la resolución de problemas por medio de su subrogada. Su estudio está fuera del alcance de esta tesis.

Sea el problema (PDG) propuesto en el capítulo 2:

$$\begin{aligned} &\min \quad Z = \Sigma_k c_k + f(x) \\ &\text{sujeto a:} \\ &\quad g(x) \leq 0 \\ &\quad \bigvee_{i \in D_k} \begin{bmatrix} Y_{ik} \\ h_{ik}(x) \leq 0 \\ c_k = \gamma_{ik} \end{bmatrix} \quad k \in SD \qquad \textbf{(PDG)} \\ &\quad \Omega(Y) = True \\ &x \in R^n \ , \ Y_{ik} \in \{True, False\}^m, \ c_k \geq 0 \end{aligned}$$

Por medio de las relajación “Big-M” que se estudió en el capítulo 3 y la transformación de la lógica proposicional en desigualdades matemáticas es posible generar un problema mixto entero con la siguiente formulación (Lee y Grossmann, 1999a):

$$\min Z = \sum_{k \in SD} \sum_{i \in D_k} \gamma_{ik}\, y_{ik} + f(x)$$

sujeto a:

$$g(x) \le 0$$

$$h_{ik}(x) \le M_{ik}(1 - y_{ik}), \quad i \in D_k, k \in SD$$

$$\sum_{i \in D_k} y_{ik} = 1, \quad k \in SD \qquad \textbf{(BM)}$$

$$Ay \le a$$

$$x \in R^n, \; y_{ik} \in \{0,1\}, \; c_k \ge 0$$

Del mismo modo se puede transformar el problema original en uno MINLP aplicando la relajación de la cáscara convexa:

$$\min Z = \sum_{k \in SD} \sum_{i \in D_k} \gamma_{ik}\, y_{ik} + f(x)$$

sujeto a:

$$g(x) \le 0$$

$$x = \sum_{i \in D_k} \nu_{ik}, \quad k \in SD$$

$$\lambda_{ik} h_{ik}(\nu_{ik}/\lambda_{ik}) \le 0, \quad i \in D_k, k \in SD$$

$$\sum_{i \in D_k} \lambda_{ik} = 1, \quad k \in SD \qquad \textbf{(CC)}$$

$$Ay \le a$$

$$0 \le \nu_{ik} \le \lambda_{ik} U_{ik}, \quad i \in D_k, k \in SD$$

$$x, \nu_{ik} \ge 0, \lambda_{ik} \in \{0,1\}$$

De acuerdo con lo expresado en el capítulo 3, una disyunción puede relajarse con una formulación "Big-M" o la cáscara convexa. Dependiendo del caso convendrá una u otra. En caso de generarse una formulación MINLP mixta haciendo uso de ambas relajaciones, la forma del problema MINLP será:

$$\min Z = \sum_{k \in SD} \sum_{i \in D_k} \gamma_{ik}\ y_{ik} + f(x)$$

sujeto a:

$$g(x) \leq 0$$

$$x = \sum_{i \in D_k} \nu_{ik}, \quad k \in SD1$$

$$y_{ik} h_{ik}(\nu_{ik} / y_{ik}) \leq 0, \quad i \in D_k, k \in SD1$$

$$h_{ik}(x) \leq M_{ik}(1 - y_{ik}), \quad i \in D_k, k \in SD2$$

$$\sum_{i \in D_k} y_{ik} = 1, \quad k \in SD \qquad \textbf{(BMCC)}$$

$$Ay \leq a$$

$$0 \leq \nu_{ik} \leq y_{ik} U_{ik}, \quad i \in D_k, k \in SD1$$

$$x, \nu_{ik} \geq 0, y_{ik} \in \{0,1\}, SD1, SD2 \in SD$$

En el problema (BMCC) *SD1* es un subconjunto del conjunto original *SD* en el que las disyunciones se relajaron con la cáscara convexa. En cambio *SD2* es un subconjunto del conjunto original *SD* en el que las disyunciones se relajaron con la formulación "Big-M". Todo lo establecido hasta ahora para un problema disyuntivo (PDG) se puede hacer de la misma forma para una formulación híbrida en donde las disyunciones de la formulación se pueden transformar por medio de relajaciones "Big-M" ó de la cáscara convexa.

En resumen, a partir de una formulación (PH) ó (PDG) se pueden transformar estos problemas por medio de relajaciones "Big-M", cáscara convexa ó ambas en problemas MINLP del tipo (BM), (CC) ó (BMCC) y resolver estas formulaciones con los

métodos para problemas mixto-enteros planteados en el capítulo 1: RyA , AE, DGB ó PCE.

4.2.3. Ramificación y Acotamiento (B&B) Basado en la Cáscara Convexa

Este método fue propuesto por Lee y Grossmann (1999b), está basado en la generación de la cáscara convexa para problemas formulados como PDG. En primer lugar se resuelve la relajación convexa del problema PDG. A partir de la solución de este nodo raíz se empieza a ramificar el árbol. Para ello se emplean las variables binaria λ_{ik} que miden la cercanía con la que cada término de la disyunción es satisfecho, esto es: $v_{ik} \rightarrow x$ si $\lambda_{ik} \rightarrow 1$. La solución del nodo raíz da generalmente valores fraccionales de λ_{ik}. La regla de ramificación es seleccionar la variable λ_{ik} que se encuentra más próxima a 1 dado que es el término de la disyunción *k-ésima* que se encuentra más próximo a ser factible. El NLP resultante está definido por el término *i-ésimo* de la disyunción *k-ésima* más la cáscara convexa de todas las disyunciones restantes $j \neq k$. La solución de este subproblema es una cota superior Z_S del problema. Después de ramificar un término de la disyunción con λ_{ik}=1, el nodo complementario del árbol es la cáscara convexa con todos los términos restantes de la disyunción *k-ésima* distintos de *i* más la cáscara convexa de todas las disyunciones restantes distintas de *k*.

El algoritmo de Ramificación y Acotamiento (B&B) basado en la cáscara convexa puede resumirse en las siguientes etapas:

1. Asignar $Z_S = \infty$.
2. Seleccionar ε.
3. Por cada disyunción $k \in SD$ formular la cáscara convexa correspondiente, resolver el problema resultante (CC) relajado y obtener la solución x_I y la cota inferior del problema Z_I. **Si** todas las variables binarias λ_{ik} son 0 ó 1, se encontró la solución óptima entera del problema y por lo tanto asignar $x^* = x_I$ y $Z^* = Z_I$. **Si no** continuar en la etapa siguiente.

4. Mientras no haya nodos sin examinar en el árbol:
   4.1. Seleccionar el mayor $\lambda_{ik} \neq 0,1$, $i \in D_k$, $k \in SD$, de la solución del problema (CC) relajado y fijarlo en 1.
   4.2. Fijar el término *i-ésimo* de la disyunción *k-ésima* cuyo valor λ_{ik} es el mayor, descartar el resto de los términos de la disyunción *k-ésima*, Generar la cáscara convexa del resto de las disyunciones diferentes de la *k-ésima* y generar de ésta forma el subproblema NLP.
   4.3. Resolver el subproblema NLP y obtener la solución x y Z.
   4.4. Si todos los λ_{ik} son 0 ó 1 se encontró una solución factible del problema y una cota superior del mismo. Si $Z \leq Z_S$ entonces hacer $Z_S=Z$ y $x_S=x$, luego continuar en la etapa 5. Si $Z \geq Z_S$ seguir en la etapa 5.
   4.5. Si la solución del problema en 4.3 es infactible continuar en la etapa 5.
   4.6. Si no todos los λ_{ik} son 0 ó 1 en la solución del problema 4.3. continuar en 4.1. haciendo una nueva ramificación.
5. Ramificar en la cáscara convexa del complemento
   5.1. Asignar $\lambda_{ik} = 0 \Rightarrow$ remover el término *i-ésimo* de la disyunción *k-ésima*
   5.2. Generar la cáscara convexa con todos los términos restantes de la disyunción *k-ésima* distintos del *i* más la cáscara convexa de todas las disyunciones restantes distintas de la *k*.
   5.3. Resolver el subproblema NLP y obtener la solución x y Z.
   5.4. Si todos los λ_{jk} son 0 ó 1 se encontró una solución factible del problema y una cota superior del mismo. Si $Z \leq Z_S$ entonces hacer $Z_S=Z$ y $x_S=x$, luego volver atrás en el árbol. Si $Z \geq Z_S$ volver atrás en el árbol.
   5.5. Si la solución del problema en 5.3 es infactible volver atrás en el árbol.
   5.6. Si no todos los λ_{ik} son 0 ó 1 en la solución del problema 5.3. continuar en 4.1. haciendo una nueva ramificación.

El algoritmo tiene una convergencia finita dado que el número de términos y las disyunciones son finitos. Dado que el algoritmo se basa en términos convexos de la disyunciones los subproblemas que se resuelven tienen una solución óptima única,

obviamente suponiendo que las restricciones no lineales son convexas. La solución óptima del problema corresponde a la cota superior $x^*=x_S$ y $Z^*=Z_S$ obtenida cuando en el árbol se han examinado todos los nodos y no es posible seguir ramificando ningún nodo.

Para extender este método partiendo de una formulación híbrida es necesario agregar el criterio de ramificación para las variables binarias de las restricciones mixto-enteras del problema original. Para ello hay que modificar fundamentalmente los pasos 4.1 y 5.1 incluyendo el criterio por el cual se fijarán las variables en 0 ó 1 y al resto se las dejará libres. Es evidente que para las restricciones mixto-enteras no es necesario generar la cáscara convexa.

Con la presentación de los métodos basado en lógica y aquellos presentados en el primer capítulo se completan los algoritmos de resolución de los problemas discretos-continuos no lineales formulados como problemas híbridos(PH), disyuntivos (PDG) ó mixtos-enteros no lineales (MINLP).

A continuación se introduce una extensión del algoritmo de Aproximación Exterior Basado en Lógica que está basado en las posibles relajaciones de un conjunto disyuntivo, la idea es evitar la solución de los ***nf*** subproblemas NLP iniciales haciendo uso de la relajación "Big-M", la subrogada de Beaumont ó la cáscara convexa. En lo que sigue se presenta esta modificación para la cáscara convexa, pero esta relajación puede reemplazarse por cualquiera de las otras cuando se crea necesario. A continuación se presentan las etapas de este algoritmo para un problema híbrido (PH).

Modificación del algoritmo de Aproximación Exterior Basado en Lógica

1- Modelar el problema discreto-continuo en la forma propuesta por (PH-1) ó modelarlo como en la forma (PDG-1) y transformar un subconjunto de las disyunciones en la forma mixta entera de acuerdo a lo propuesto en el capítulo 3. Asignar $Z_S=\infty$.
2- Generar la relajación del problema resultante por medio de la cáscara convexa, la subrogada de Beaumont ó la formulación "Big-M" para las disyunciones del problema (PH).
3- Resolver el subproblema NLP generado en la etapa segunda del algoritmo. Obtener en esta etapa las linealizaciones de: la función objetivo, las restricciones globales y los términos disyuntivos que son no lineales.

4- Generar el problema maestro (MILP-2) y resolverlo. La solución de este problema es la cota inferior Z_I del problema.

5- Comparar la cota superior y la inferior. **Si** $|Z_S - Z_I| \leq \varepsilon$, donde ε es una tolerancia entonces el algoritmo termina y la cota superior actual Z_S es el óptimo. **Si no** se continua en la etapa 6.

6- Resolver un nuevo problema (NLP-2) fijando las variables Booleanas y las binarias predichas por la resolución del problema (MILP-2). La solución de este problema Z_{NLP-1} se compara con la cota superior actual Z_S. **Si** $Z_{NLP-1} < Z_S$, **entonces** $Z_S = Z_{NLP-1}$.

7- Comparar la cota superior y la inferior. **Si** $|Z_S - Z_I| \leq \varepsilon$, el algoritmo termina y la cota superior actual Z_S es el óptimo. **Si no** se continua en la etapa 4.

4.3. LOGMIP Primera Versión

Vecchietti y Grossmann (1999) desarrollaron una programa de resolución (LOGMIP) de problemas discretos-continuos no lineales formulados en forma híbrida(PH), disyuntiva (PDG) ó mixto-entera (MINLP). LOGMIP fue el primer código de computadora para resolver problemas disyuntivos no lineales. Los métodos que se implementaron fueron: el de Aproximación Exterior Basado en Lógica para problemas disyuntivos y la extensión de ese mismo método para problemas híbridos. Para los modelos mixto-enteros se trabajó sobre la base del código DICOPT++ que tiene implementado el método de Aproximación Exterior modificado para administrar restricciones de igualdad y no convexas (Viswanathan y Grossmann, 1990). Usando este código como base se implementaron los métodos de Descomposición Generalizada de Benders (DGB) y de Planos de Corte Extendido (PCE).

La motivación principal para el desarrollo de LOGMIP fue la de proveer una herramienta flexible para el desarrollo, implementación y solución de problemas discretos-continuos no lineales permitiendo la formulación del problema con distintos modelos y la solución por varios métodos. LOGMIP fue incluido dentro del sistema comercial para la solución de programas matemáticos GAMS (Brooke y otros, 1996). Los motivos para conectar LOGMIP con GAMS fueron varios. Se tenía acceso a los códigos fuentes de DICOPT++ y de las funciones de entrada/salida de GAMS, se tenía

experiencia en el uso de este sistema y además, GAMS cuenta con un lenguaje muy completo de especificación de programas matemáticos algebraicos.

Para la especificación de las disyunciones se empleo la capacidad del lenguaje GAMS para controlar el dominio de la definición de una ecuación a través de la especificación de una condición. Para ello se emplea el signo $ y a continuación la definición de la condición. A modo de ejemplo y para mostrar como se especifica una disyunción, a continuación se muestra la definición de la primera disyunción del ejemplo de los 8 procesos presentado al final del capítulo 1. La definición de esta disyunción es la siguiente:

```
UNIT_1BAL   $(Boole('1') eq 1)..   exp(x('3')) – 1 – x('2') =e=0 ;
UNIT_1COS   $(Boole('1') eq 1)..   c('1') =e= 5;
UNIT_1NX2   $(Boole('1') eq 0)..   x('2') =e= 0;
UNIT_1NX3   $(Boole('1') eq 0)..   x('3') =e= 0;
UNIT_1NCO $(Boole('1') eq 0)..   c('1') =e= 0;
```

Como se muestra en el ejemplo, la disyunción está manejada por el parámetro Boole('1'). Las dos primeras ecuaciones definen al primer término de la ecuación y éstas formarán parte del subproblema NLP si Boole('1') toma el valor 1. Si Boole('1') toma el valor 0 formarán parte del modelo las tres últimas ecuaciones definidas en donde se asigna a las variables el valor 0. Esta expresión no fue pensada para la definición de una disyunción. Hacemos uso de ella porque es la única que nos permite trabajarla como si fuera una disyunción.

La especificación de la lógica proposicional se incluye dentro del archivo de entrada GAMS por medio de desigualdades matemáticas. Para ello se emplea un programa PROLOG desarrollado por Michel Tourn (1995) que hace la conversión automática, el archivo de salida del programa PROLOG con las desigualdades matemáticas equivalentes, se incluye en el archivo de entrada GAMS.

El código de LOGMIP fue desarrollado en lenguaje C para asegurar la portabilidad del programa a diversos sistemas operativos. En la Figura 4.2. se puede ver un diagrama de flujo general de LOGMIP. Después de la especificación de un problema, GAMS hace el chequeo de la sintaxis. Si está todo bien pasa el control a LOGMIP quien

detecta si hay disyunciones y/o variables binarias y determina por tanto, el algoritmo a aplicar.

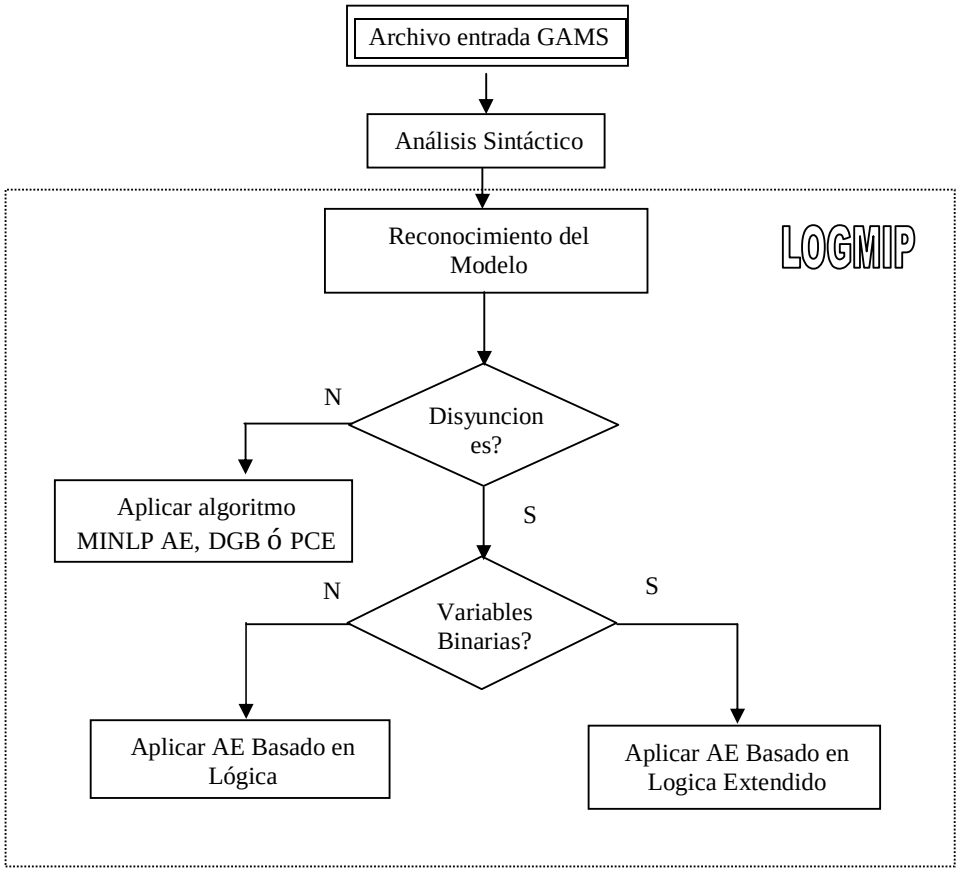


Figura 4.2: Diagrama de Flujo General de LOGMIP

A continuación se presentan un conjunto de ejemplos resueltos con esta primera versión de LOGMIP.

4.4. Ejemplos

4.4.1. Síntesis de una superestructura de 8 procesos.

El primer ejemplo que se presenta para la resolución es el ejemplo de los 8 procesos presentado al final del capítulo 1. Este ejemplo fue modelado en tres modos diferentes: algebraico, disyuntivo e híbrido. Los modelos algebraicos y disyuntivo ya

fueron presentados en el capítulo 1. Para el modelo híbrido se dejaron en forma mixta-entera las ecuaciones lineales de las disyunciones de correspondientes a los procesos 3, 4 y 5.

El modelo algebraico fue resuelto por tres métodos: AE(aproximación Exterior), DGB(Descomposición Generalizada de Benders) y PCE (Planos de Corte Extendido). El modelo disyuntivo fue resuelto por el método de AE Basado en Lógica y transformándolo en modelo mixto-entero (MINLP) por medio de la cáscara convexa, luego se resolvió por el método de AE. El modelo híbrido se resolvió por el método de AE Extendido.

En el Apéndice A se encuentran los archivos GAMS para el problema MINLP también se encuentran la formulación disyuntiva e híbrida para la primera versión de LOGMIP.

En la Tabla 4.1. se encuentra una comparación de los resultados obtenidos con el problema MINLP con los diferentes métodos. Este ejemplo consiste de 32 ecuaciones, 33 variables, 8 variables discretas (0-1).

Tabla 4.1. Resultados obtenidos en el ejemplo 4.4.1 para el modelo MINLP.

Método	Inicialización	Optimo relajado	Iteraciones	Tiempo ejecución
AE	Problema Relajado	15.08	1 NLP 4 itera.(**)	3.2 seg.
GBD	Problema Relajado	15.08	1 NLP 14 itera.(**)	10 seg.
PCE	Valores Iniciales	-25.13(*)	6 MILP	3.2 seg.
AE (cáscara convexa)	Problema relajado	62.57	1 NLP 2 itera(**)	1.95 seg.

(*) El valor obtenido corresponde a los valores iniciales planteados no a un problema relajado como en los casos anteriores.
(**) Se resuelve un subproblema NLP y uno MILP por cada iteración.

La solución que se obtiene corresponde a la siguiente estructura de procesos (2,4,6,8) y este resultado se muestra en la Figura 4.3. La solución óptima del problema es $Z^* = 68.08$ y los valores de los flujos son los siguientes:

x^T= *(*x_1= *0,* x_2=*0,* x_3=*0,* x_4=*4.29,* $x5$=*2,* x_6=*0.67,* x_7=*0.2,* x_8=*0.47,* x_9=*0,* x_{10}=*0.47,* x_{11}=*0.33,* x_{12}=*0.33,* x_{13}=*2,* x_{14}=*0.27,* x_{15}=*0,* x_{16}=*0,* x_{17}=*0.58,* x_{18}=*0.71,* x_{19}=*2.,* x_{20}=*1.65,* x_{21}=*0,* x_{22}=*0,* x_{23}=*1.65,* x_{24}=*1.38,* x_{25}=*0.58)*

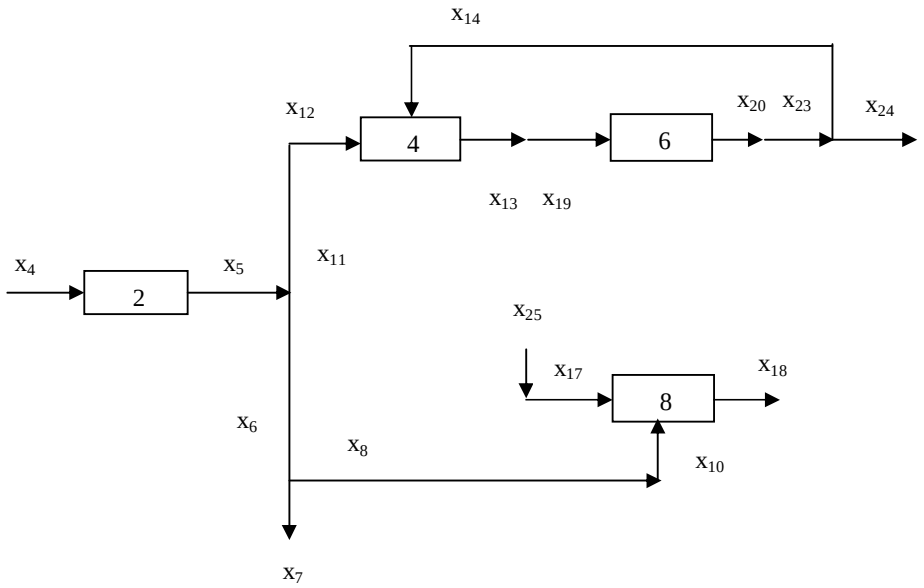


Figura 4.3: Solución óptima de la superestructura de los 8 procesos

El problema mixto entero generado por medio de la cáscara convexa partiendo de la formulación disyuntiva tiene las siguientes características: variables 41, ecuaciones 51, variables discretas 8. En la última fila de la Tabla 4.1 se pueden ver los resultados para este método. Notar la diferencia con los otros métodos que tiene el óptimo del problema relajado (62.57), esta muy cerca del óptimo del problema. La diferencia de integralidad ("integrality gap") es mucho más estrecha para este caso que con la formulación original. Es por ello que requiere menos iteraciones llegar a la solución. Dada la forma peculiar de las disyunciones para este problema de síntesis, la cáscara convexa tiene muchas simplificaciones y es por ello que el número de ecuaciones y de variables no se incrementa demasiado (ver el archivo correspondiente en el Apéndice A).

En la Tabla 4.2 se muestran los resultados obtenidos con la aplicación del método de AE basado en lógica para el modelo disyuntivo y el extendido para el híbrido. Para el problema disyuntivo el número de subproblemas inicial ***nf*** a resolver son tres y las

topologías iniciales seleccionadas fueron las siguientes: a) 1,3,4,7,8 - b) 2,3,4,6,8 - c) 1,3,5,8. Para el problema híbrido, dado que las ecuaciones lineales para los procesos 3, 4 y 5 se formularon como ecuaciones globales mixto-enteras del tipo "Big-M", se redujo en 1 los subproblemas iniciales ***nf*** a resolver. Las topologías seleccionadas para las disyunciones restantes fueron las siguientes: a) 1,7,8 - b) 2,6,8.

Los problemas (PDG-1) y (PH-1) de este ejemplo cuentan con 52 ecuaciones, 42 variables y 8 variables discretas. El número incrementado de ecuaciones se debe a que se agregan las ecuaciones de la lógica proposicional, el mayor número de variables es porque se agregaron variables para los coeficientes de costo.

Tabla 4.2. Resultados obtenidos en el ejemplo 4.4.1 para los modelos disyuntivo e híbrido

Modelo	Inicialización	Método	Iteraciones	Tiempo ejecución
Disyuntivo	1) 1-3-4-7-8 2) 2-3-4-6-8 3) 1-3-5-8	AE Lógico	3 NLP 1 itera.(**)	2 seg.
Híbrido	1) 1-7-8 2) 2-6-8	AE Lógico Extendido	2 NLP 1 itera.(**)	1 seg.

(**) Se resuelve un subproblema NLP y uno MILP por cada iteración.

4.4.2. Síntesis del proceso HDA

Este ejemplo corresponde al proceso de hidrodealquilación (HDA) de tolueno. Se propone una superestructura de unidades de procesos y se desea obtener el diagrama de flujos de operación que rinde mayores beneficios. La superestructura se presenta en la Figura 4.4. Se resolvieron dos modelos de este ejemplo, uno mixto-entero (MINLP) con el que se empleó el programa DICOPT^{++} para resolverlo, y uno disyuntivo (PDG) que se empleó LOGMIP con el método de AE Basado en Lógica. Para el modelo (MINLP) se emplearon dos inicializaciones: una completamente relajada y otra fijando las topologias iniciales. Para el modelo disyuntivo se necesitan resolver dos subproblemas iniciales ***nf***.

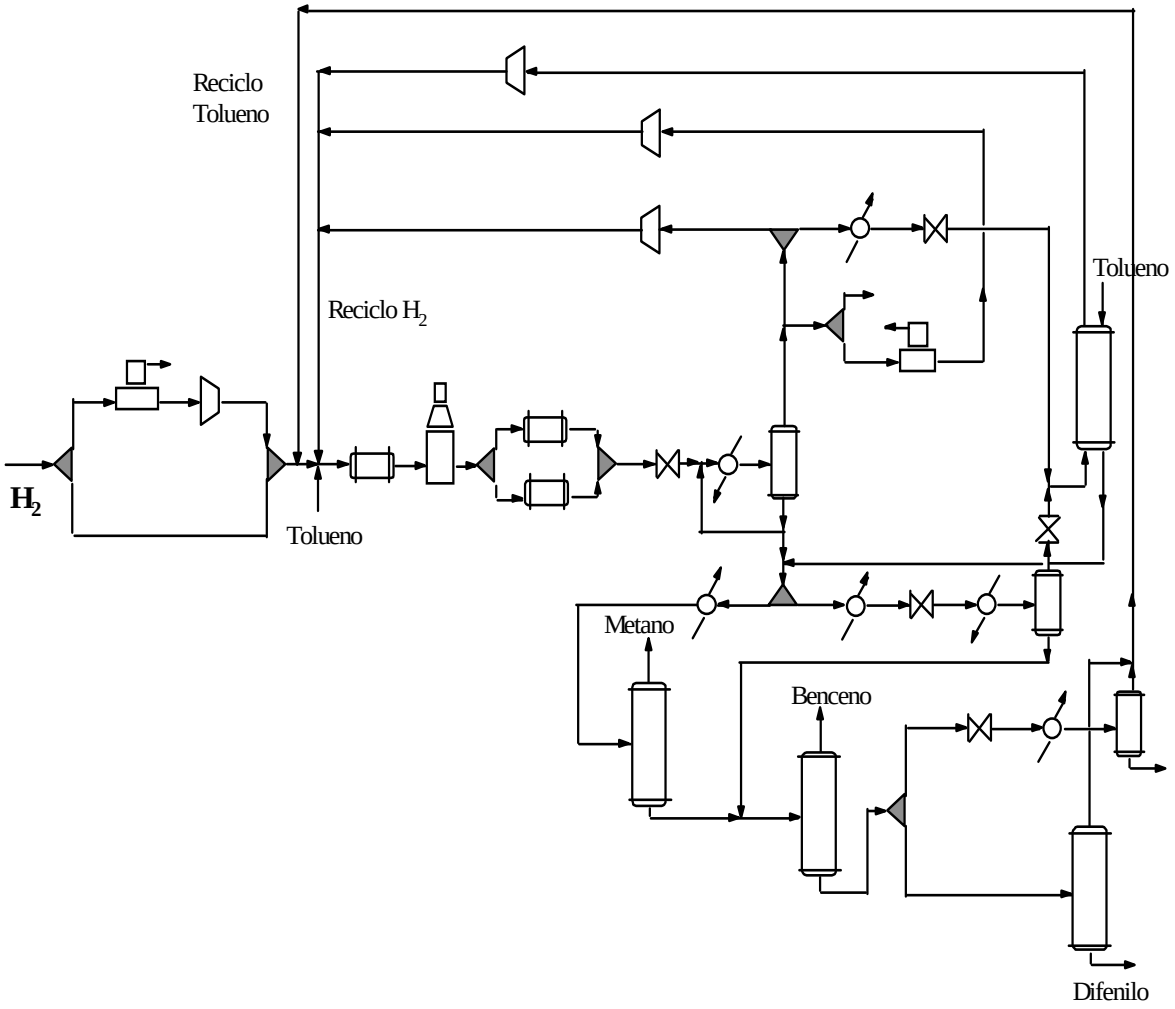


Figura 4.4: Superestructura del proceso HDA

En la Figura 4.5 se muestra como una secuencia de equipos que deben ir juntos en la superestructura porque forman un grupo (si se selecciona uno el resto de los equipos del grupo se debe seleccionar), son manejados por la misma variable Booleana. De esta forma se pueden ahorrar variables discretas en el problema. De esa figura se pueden ver las topologías seleccionadas para resolver los NLP iniciales ***(nf=2)*** para el problema disyuntivo. Estas topologías son: a) 1-3-4-5-6-8-10-12-13-14 y b) 1-2-4-5-7-9-11-12-13-14.

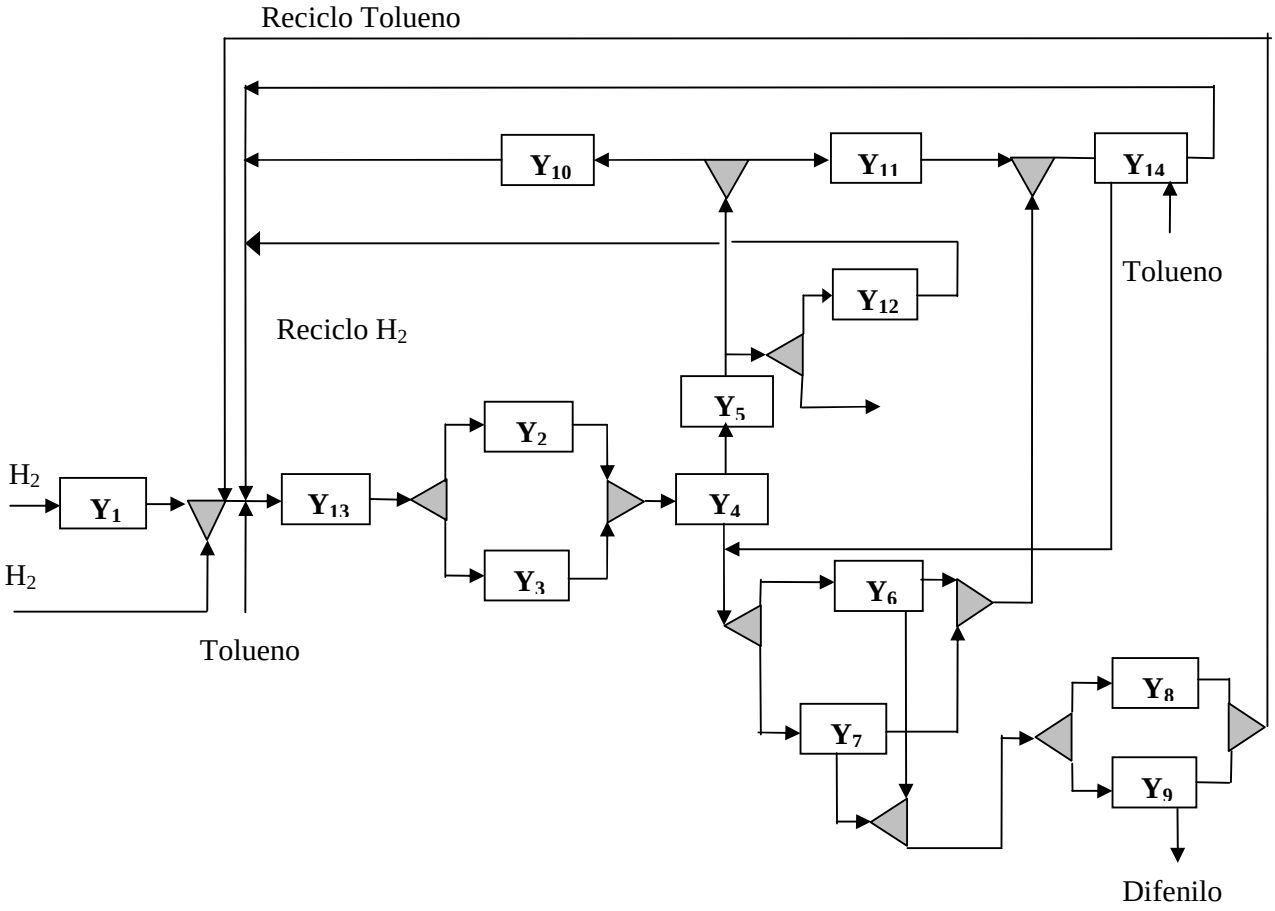


Figura 4.5. Superestructura del proceso HDA donde se muestra la agrupación de los equipos

Tabla 4.3. Resultados obtenidos en el ejemplo 4.4.2 para los modelos MINLP y disyuntivo

Modelo	MINLP	MINLP	PDG-1
Inicialización	Relajada	Fija	Fija
Restricciones	719	719	737
Variables	722	722	717
Variables discretas	13	13	14
Valor óptimo	5304.8	5671.4	5810.8
Tiempo ejecución	348 sec.	293 sec.	280 sec.
Iteraciones	1 NLP 3 itera. (**)	1 NLP 2 itera. (**)	2 NLP 1 itera.(**)

(**) Se resuelve un subproblema NLP y uno MILP por cada iteración

De la tabla 4.3. se puede ver que el modelo disyuntivo encontró una mejor solución (mayor beneficio) que los mixto-enteros con diferentes inicializaciones, probablemente porque en el modelo se evitan singularidades numéricas por medio de las disyunciones. Se debe tener en cuenta que este modelo es complejo y no convexo, el archivo de entrada GAMS posee 32 páginas para la especificación del modelo.

4.4.3. Espectroscopia infrarroja

Este ejemplo corresponde a la determinación de la estructura y estimación de los parámetros de una espectroscopia infrarroja propuesta por Brink y Westerlund (1995). El problema consiste en determinar el valor y número de parámetros en una regresión lineal de concentración *(c)* contra absorbancia *(a)* del espectro. La función objetivo está basada en el criterio de información de Akaike que involucra tanto la matriz de covariancia *(R)* así como el número de parámetros. El ejemplo original posee una función objetivo no lineal sujeto a restricciones lineales. Se generaron modelos mixto-enteros (MINLP) y modelos disyuntivos (PDG-1). Para ambos modelos la función objetivo no lineal se transformó en una restricción. Para el modelo disyuntivo (PDG-1) las restricciones lineales se transformaron en disyunciones.

Los datos de este problema son los siguientes:

i = número de onda. Se trabajó con 10 números de onda diferentes,
j= datos experimentales. Se hicieron 8 experiencias diferentes.

A (i,j) (absorbancia)

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
.0003	*.0764*	*.0318*	*.0007*	*.0534*	*.0773*	*.0536*	*.0320*
.0007	*.0003*	*.0004*	*.0009*	*.0005*	*.0009*	*.0005*	*.0003*
.0066	*.0789*	*.0275*	*.0043*	*.0704*	*.0683*	*.0842*	*.0309*
.0044	*.0186*	*.0180*	*.0179*	*.0351*	*.0024*	*.0108*	*.0052*
.0208	*.0605*	*.0601*	*.0601*	*.0981*	*.0025*	*.0394*	*.0221*
.0518	*.1656*	*.1491*	*.1385*	*.2389*	*.0248*	*.1122*	*.0633*
.0036	*.0035*	*.0032*	*.0051*	*.0015*	*.0094*	*.0015*	*.0024*
.0507	*.0361*	*.0433*	*.0635*	*.0048*	*.0891*	*.0213*	*.0310*
.0905	*.0600*	*.0754*	*.1091*	*.0038*	*.1443*	*.0420*	*.0574*
.0016	*.0209*	*.0063*	*.0010*	*.0132*	*.0203*	*.0139*	*.0057*

k = componente. Se trabajó con 3 compuestos diferentes

Matrix C(k,j) (composición)

502	*204*	*353*	*702*	*0*	*1016*	*104*	*204*
97	*351*	*351*	*351*	*351*	*700*	*201*	*97*
0	*22*	*8*	*0*	*14*	*22*	*14*	*8*

Definiendo w_j como el cuadrado del error entre la concentración experimental y la predicción lineal pesado por la matriz inversa de covariancia R, p_{ki} el parámetro para el componente *k* en el número de onda *i*, y_{ki} la variable binaria correspondiente para su selección el modelo está dado por:

Formulación MINLP

$$min \;\; Z = w_j + 2 \sum_k \sum_i y_{ki}$$

sujeto a:

$$w_j = \sum_k \{ [(c_{kj} - \sum_i p_{ki} a_{ij})^T * R^{-1}] * (\sum_k c_{kj} - \sum_i p_{ki} a_{ij})] \} \qquad \forall j$$

$$\left.\begin{array}{l} p_{ki} - p_{ki}^{max} \; y_{ki} \le 0 \\ p_{ki}^{min} \; y_{ki} - p_{ki} \le 0 \end{array}\right\} \quad \forall i,k$$

$$y_{ki} \in \{0,1\}, \; w_j \ge 0$$

donde el error está definido por la ecuación:

$$e_j = c_j - P a_j$$

y R = matriz de covariancia que en la primera iteración del problema se asume como la matriz identidad I.

Formulación Disyuntiva (PDG-1)

$$min \;\; Z = w_j + 2 \sum_k \sum_i y_{ki}$$

sujeto a:

$$w_j = \sum_k \{ [(c_{kj} - \sum_i p_{ki} a_{ij})^T * R^{-1}] * (\sum_k c_{kj} - \sum_i p_{ki} a_{ij})] \} \qquad \forall j$$

Disyunciones

$$\begin{bmatrix} Y_{ki} \\ P_{ki}^{min} \le P_{ki} \le P_{ki}^{max} \\ c_{ki} = 1 \end{bmatrix} \vee \begin{bmatrix} \neg Y_{ki} \\ P_{ki} = 0 \\ c_{ki} = 0 \end{bmatrix} \quad \forall i,k$$

$$Y_{ki} \in \{Verdadero, Falso\}$$

donde c_{ki} indica que se estima el parámetro del componente *k* en la longitud de onda *i*.

En la Tabla 4.4 se pueden observar los resultados obtenidos en la solución de este ejemplo.

Tabla 4.4. Resultados obtenidos en el ejemplo 4.4.3

Modelo	MINLP	MINLP	PDG-1
Inicialización	Relajada	Fija	Fija
Restricciones	39	39	102
Variables	69	69	99
Variables Discretas	30	30	30
Valor óptimo	13.98	13.98	13.98
Tiempo ejecución	55 sec	100 sec.	7 sec
Iteraciones	6 (**)	11 (**)	4 (**)

(**) Se resuelve un subproblema NLP y uno MILP por cada iteración.

Para los problemas MINLP se empleó DICOPT^{++} y para el modelo disyuntivo (PDG-1) se empleó LOGMIP con el algoritmo de AE Basado en Lógica. De los resultados de la Tabla 4.4 se puede ver que el modelo (PDG-1) tiene un comportamiento muy superior a los modelos MINLP, no solamente en un menor número de iteraciones, sino en el tiempo de ejecución necesario para llegar a la solución.

4.4.4. Diseño de una planta batch multiproducto

Este ejemplo corresponde al diseño de plantas batch multiproducto que pueden incluir múltiples unidades en paralelo y también tanques de almacenamiento intermedio. Los datos del ejemplo se obtuvieron de Ravemark (1995). El problema consiste en la determinación del tamaño de los equipos, la cantidad de unidades en paralelo, el volumen y ubicación de los tanques de almacenamiento intermedio. El objetivo es minimizar el costo de inversión de la planta. El problema se modeló de manera algebraica e híbrida.

Las restricciones, que en su forma original son no convexas, fueron convexificadas a través de las siguientes transformaciones de las variables originales:

$$b_{i,j} = log\ B_{i,j}$$
$$n_j = log\ N_j$$
$$m_j = log\ M_j$$
$$E_i = exp\ e_i$$
$$v_j = log\ V_j$$
$$v_{Tj} = log\ V_{Tj}$$

El significado de las distintas variables que se usan en este problema son las siguientes:

N_j: número de unidades en paralelo en-fase en la etapa j

Mj: número de unidades en paralelo fuera-de-fase en la etapa j

V_j: tamaño de la etapa batch j

V_{Tj}: volumen del tanque intermedio ubicado entre la etapa j y la j+1

S_j^*: factor de medida del tanque intermedio

Φ: factor de medida para las etapas

$coef_k$: coeficiente constante que determina la cantidad de unidades en paralelo

B_{ij}: Medida de la "batchada" para el producto i en la etapa j

E_i: inversa de la productividad para el producto i

Q_i: Requerimientos de producción para el producto i

Y_j: variable Booleana que es verdadera si se ubica un tanque en la posición j, sino es Falsa

ynk, ymk: variables binarias para seleccionar el número de unidades que operan en-fase y fuera-de-fase respectivamente.

Los datos del modelo son los siguientes:

Cantidad de productos i = 5, denominados A,B,C,D,E

Etapas j = 6

Horizonte de tiempo H = 6000 hs.

Requerimientos de producción de cada producto i: A=250000 , B=150000 , C=180000 D=160000 , E=120000

$S_{i,j}$ = factor de medida del producto i en la etapa j

	1	2	3	4	5	6
A	*7.9*	*2.*	*5.2*	*4.9*	*6.1*	*4.2*
B	*0.7*	*0.8*	*0.9*	*3.4*	*2.1*	*2.5*
C	*0.7*	*2.6*	*1.6*	*3.6*	*3.2*	*2.9*
D	*4.7*	*2.3*	*1.6*	*2.7*	*1.2*	*2.5*
E	*1.2*	*3.6*	*2.4*	*4.5*	*1.6*	*2.1*

$T_{i,j}$ = tiempo de procesamiento del producto i en la etapa j

	1	2	3	4	5	6
A	*6.4*	*4.7*	*8.3*	*3.9*	*2.1*	*1.2*
B	*6.8*	*6.4*	*6.5*	*4.4*	*2.3*	*3.2*
C	*1.0*	*6.3*	*5.4*	*11.9*	*5.7*	*6.2*
D	*3.2*	*3.0*	*3.5*	*3.3*	*2.8*	*3.4*
E	*2.1*	*2.5*	*4.2*	*3.6*	*3.7*	*2.2*

Formulación MINLP

$$Min\, C = \sum_{j=1}^{M} a_j exp(m_j + n_j + \alpha_j v_j) + \sum_{j=1}^{M-1} c_j exp(\gamma_j\, vt_j)$$

$$v_j \geq log(S_{ij}) + b_{ij} - n_j \qquad \forall i, \forall j$$

$$e_i \geq log(T_{ij}) - b_{ij} - m_j \qquad \forall i, \forall j$$

$$H \geq \sum_{i=1}^{P} Q_i exp(e_i)$$

$$n_j = \sum_k coef_k\, yn_{kj} \qquad \forall j$$

$$\sum_k yn_{kj} = 1 \qquad \forall j$$

$$m_j = \sum_k coef_k\, ym_{kj} \qquad \forall j$$

$$\sum_k ym_{kj} = 1 \qquad \forall j$$

$$vt_j \geq log(2S^*_{ij}) + b_{i,j+1} - K_{ij}(1 - y_j) \quad \forall i, \forall j = 1, M-1$$

$$vt_j \geq log(2S^*_{ij}) + b_{ij} - K_{ij}(1 - y_j) \quad \forall i, \forall j = 1, M-1$$

$$b_{ij} - b_{i,j+1} \leq \phi\, y_j \quad \forall i, \forall j = 1, M-1$$

$$b_{ij} - b_{i,j+1} \geq -\phi\, y_j \quad \forall i, \forall j = 1, M-1$$

Formulación Híbrida (PH-1)

$$Min\, C = \sum_{j=1}^{M} a_j exp(m_j + n_j + \alpha_j v_j) + \sum_{j=1}^{M-1} c_j exp(\gamma_j\, vt_j)$$

$$v_j \geq log(S_{ij}) + b_{ij} - n_j \qquad \forall i, \forall j$$

$$e_i \geq log(T_{ij}) - b_{ij} - m_j \qquad \forall i, \forall j$$

$$H \geq \sum_{i=1}^{P} Q_i exp(e_i)$$

$$n_j = \sum_k coef_k\, yn_{kj} \qquad \forall j$$

$$\sum_k yn_{kj} = 1 \qquad \forall j$$

$$m_j = \sum_k coef_k\, ym_{kj} \qquad \forall j$$

$$\sum_k ym_{kj} = 1 \qquad \forall j$$

$$\begin{bmatrix} Y_j \\ vt_j \geq log(2S_{ij}^*) + b_{i,j+1} \\ vt_j \geq log(2S_{ij}^*) + b_{i,j} \\ b_{ij} - b_{i,j+1} \leq \phi \\ b_{ij} - b_{i,j+1} \geq -\phi \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ vt_j = 0 \\ b_{ij} - b_{i,j+1} = 0 \end{bmatrix}$$

Los resultados que se obtuvieron se pueden observar en la Tabla 4.5.

Tabla 4.5 Resultados obtenidos en la solución del ejemplo 4.4.4

Modelo	MINLP	MINLP	PH-1
Inicialización	Relajado	Fija	Fija
Restricciones	186	186	187
Variables	112	112	113
Variables discretas	53	53	53
Valor óptimo	261883	261883	261883
Tiempo ejecución	287 sec.	616 sec.	80 sec.
Iteraciones	1 NLP 10 itera(**)	1 NLP 20 itera(**)	1 NLP 4 itera(**)

(**) Se resuelve un subproblema NLP y uno MILP por cada iteración.

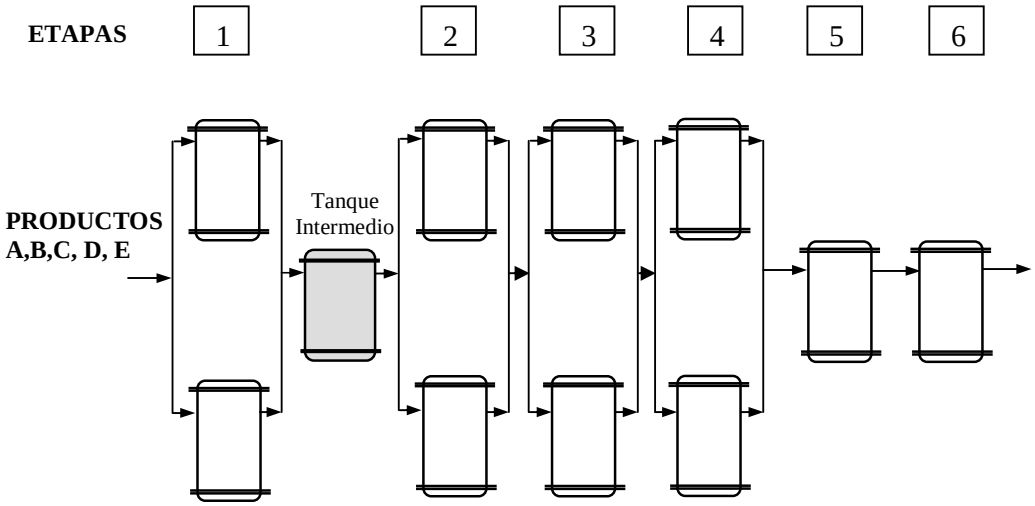


Figura 4.6:Resultados obtenidos en el diseño de la planta batch

En la Figura 4.6 se puede ver que el diseño de la planta tiene equipos duplicados fuera-de-fase para las etapas 1,2,3 y 4, no tiene equipos duplicados en las etapas 5 y 6. Los tamaños de cada etapa son: *V=(2492, 1030, 2127, 2807, 2495, 2291)* de los tanques intermedios V_T*=(0, 8637, 0, 0, 0)*. En el Apéndice A se incluyen los archivos de entrada GAMS para el modelo MINLP y el híbrido.

4.4.5. Síntesis de una superestructura de 9 unidades con ecuaciones de costo discontinuas.

Este ejemplo corresponde a la síntesis de un superestructura de 9 unidades de producción (Turkay y Grossmann, 1998). El diagrama de flujos es el siguiente:

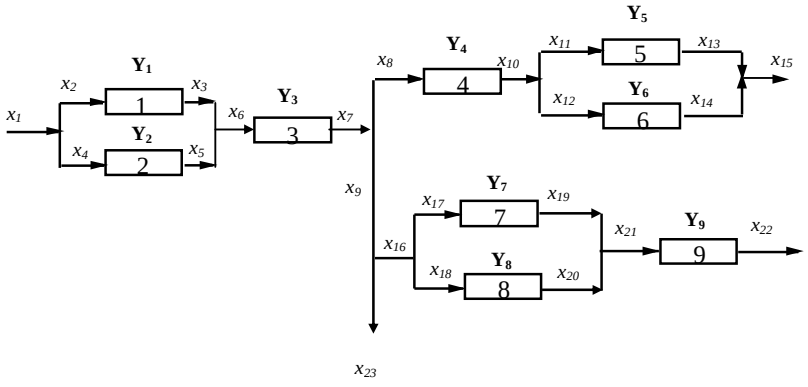


Figura 4.7 : Superestructura de 9 unidades de proceso

El ejemplo es una modificación del 4.4.1. previo. En este caso se agregó una nueva unidad, la distribución de las unidades no es la misma, y fundamentalmente, los costos no son fijos sino que son ecuaciones de costo de inversión discontinuas, en función del tamaño de la unidad. Este ejemplo fue propuesto por Turkay y Grossmann (1996b). Las ecuaciones de costo de inversión se basan en una componente de costo fija y una variable, en función de las variables de proceso. Guthrie (1974) encontró que las ecuaciones de costo tienen una forma discontinua en función de múltiples regiones de tamaño. Esta particularidad hace que las disyunciones sean del tipo disyunciones embebidas, la forma de las disyunciones para este ejemplo es:

$$\begin{bmatrix} Y_i \\ x_{i,sal} = \xi_i \, ln(1 + x_{i,ent}) \\ V_i = \sigma_i x_{i,sal} \\ \begin{bmatrix} c_i = \alpha_{i1} V_i^{\eta i1} + \beta_{i1} \\ V_{i1}^{lo} \le V_i \le V_{i1}^{up} \end{bmatrix} \vee \begin{bmatrix} c_i = \alpha_{i2} V_i^{\eta i2} + \beta_2 \\ V_{i1}^{up} \le V_i \le V_{i2}^{up} \end{bmatrix} \vee \begin{bmatrix} c_i = \alpha_{i3} V_i^{\eta i3} + \beta_{i3} \\ V_2^{up} \le V_i \le V_3^{up} \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ x_{i,ent} = 0 \\ x_{i,sal} = 0 \\ V_i = 0 \\ c_i = 0 \end{bmatrix}$$

donde β_i corresponde al coeficiente de costo fijo de la unidad *i*, $\alpha_i V_i^{\eta ik}$ es el término de costo variable de la unidad *i*. Notar que para nuestro ejemplo se han considerado sólo tres regiones posibles de costo 1, 2 y 3, esto puede variar según sea el caso. El significado de la disyunción previamente presentada es que si la unidad *i* se selecciona, luego se debe determinar que ecuación de costo, de las tres posibles, se aplica de acuerdo con el valor que toma la variable continua V_i. Para resolver este problema se generó un problema híbrido del tipo (PH-1). Para ello las disyunciones embebidas se transformaron en disyunciones de la forma del problema PDG de acuerdo con lo visto en el capítulo 2 en (2.42):

$$\begin{bmatrix} Y_i \\ x_{i,sal} = \xi_i \, ln(1 + x_{i,ent}) \\ V_i = \sigma_i x_{i,sal} \\ \begin{bmatrix} Z_{i1} \\ c_i = \alpha_{i1} V_i^{\eta i1} + \beta_{i1} \\ V_{i1}^{lo} \le V_i \le V_{i1}^{up} \end{bmatrix} \vee \begin{bmatrix} Z_{i2} \\ c_i = \alpha_{i2} V_i^{\eta i2} + \beta_2 \\ V_{i1}^{up} \le V_i \le V_{i2}^{up} \end{bmatrix} \vee \begin{bmatrix} Z_{i3} \\ c_i = \alpha_{i3} V_i^{\eta i3} + \beta_{i3} \\ V_2^{up} \le V_i \le V_3^{up} \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ x_{i,ent} = 0 \\ x_{i,sal} = 0 \\ V_i = 0 \\ c_i = 0 \end{bmatrix}$$

Las disyunciones correspondientes a las ecuaciones de costo se transformaron en ecuaciones mixto-enteras por medio de la relajación por la cáscara convexa. La proposición lógica (equivalencia) se transformó en una desigualdad matemáticas. Las disyunciones de las unidades de proceso se dejaron como disyunciones. Esta transformación tiene la siguiente forma:

$$c_{ik} = \alpha_{ik\,ik}\upsilon^{\eta ik} + \beta_{ik} z_{ik} \qquad i \in SD, k = 1..3$$
$$V_i = \sum_k \upsilon_{ik}$$
$$\sum_k z_{ik} = Y_i$$
$$\upsilon_{ik} \le \upsilon_{ik}^{up} z_{ik}$$
$$\upsilon_{ik} \ge \upsilon_{ik}^{lo} z_{ik}$$
$$Y_i, z_{ik} \in \{0,1\}$$

Al modelo lo completan la siguientes función objetivo y ecuaciones globales:

$$\min Z = 2x_1 - 60x_{15} - 55x_{22} - 23x_{23} + \sum_i c_i$$
$$x_7 = x_8 + x_9$$
$$x_8 \le 5x_9$$
$$x_8 \ge x_9$$
$$x_9 = x_{16} + x_{23}$$
$$x_{16} \le 5x_9$$
$$x_{16} \ge x_9$$
$$x_1 = x_2 + x_3$$
$$x_6 = x_4 + x_5$$
$$x_{10} = x_{11} + x_{12}$$
$$x_{15} = x_{13} + x_{14}$$
$$x_{16} = x_{17} + x_{18}$$
$$x_{21} = x_{19} + x_{20}$$

y las siguientes proposiciones lógicas:

$$Y_1 \Rightarrow Y_3$$
$$Y_2 \Rightarrow Y_3$$
$$Y_3 \Rightarrow Y_1 \vee Y_2$$
$$Y_3 \Rightarrow Y_4$$
$$Y_3 \Rightarrow Y_7 \vee Y_8$$
$$Y_4 \Rightarrow Y_5 \vee Y_6$$
$$Y_5 \Rightarrow Y_4$$
$$Y_6 \Rightarrow Y_4$$
$$Y_6 \Rightarrow Y_4$$
$$Y_7 \Rightarrow Y_9$$
$$Y_8 \Rightarrow Y_9$$
$$Y_9 \Rightarrow Y_7 \vee Y_8$$
$$Y_1 \veebar Y_2$$
$$Y_5 \veebar Y_6$$
$$Y_7 \veebar Y_8$$

Para resolver el problema se aplicó el método de AE Basado en Lógica Extendido para problemas híbridos.

Los datos para este problema se pueden ver en la Tabla 4.6.

Tabla 4.6. Datos para el ejemplo 4.4.5.

Unidad	ξ	σ	*Intervalo medida*	V_i^{lo}	V_i^{up}	α	β
1	1.1	0.7	1	0.	0.6	2.0	5.0
			2	0.6	1.0	3.0	6.0
			3	1.0	1.6	4.0	7.0
2	1.2	0.6	1	0.	0.6	3.0	2.0
			2	0.6	1.0	4.0	2.5
			3	1.0	1.6	5.0	3.0
3	1.4	0.9	1	0.	0.5	2.0	1.0
			2	0.5	1.2	3.0	2.0
			3	1.2	2.0	4.0	3.0
4	1.3	1.5	1	0.	0.4	3.0	2.0
			2	0.4	1.2	4.5	3.0
			3	1.2	1.8	6.0	4.0
5	1.2	1.5	1	0.	0.7	1.5	1.8
			2	0.7	1.3	2.6	3.2
			3	1.3	2.0	3.5	4.5
6	1.2	1.6	1	0.	0.7	4.5	4.0
			2	0.7	1.3	5.0	5.0
			3	1.3	2.0	5.5	6.0
7	1.2	2.0	1	0.0	0.5	2.0	2.1
			2	0.5	1.1	2.3	2.5
			3	1.1	1.6	2.7	3.0
8	1.2	1.9	1	0.0	0.5	1.8	1.6
			2	0.5	1.1	2.1	2.5
			3	1.1	1.6	2.6	3.7
9	1.1	2.1	1	0.0	0.5	3.0	1.0
			2	0.5	1.2	3.5	2.0
			3	1.2	1.9	4.0	3.0

Los valores de η_{ik} que se usaron fueron de 0.6 $\forall i$ y $\forall k$. Dos topologías iniciales se emplearon para inicializar las unidades de proceso que permanecen como disyunciones que son: (2,3,4,6,8,9) y (1,3,4,5,7,9), las variables binarias que corresponden a la cáscara convexa se han dejado libres (relajadas) para este problema. Dado que estas variables se dejan como relajadas, los valores de la función objetivo que se obtienen no son una cota

superior del problema. Los valores de la función objetivo para el primer valor de inicialización es Z_1 = -23.72 el cual corresponde a un beneficio de $23.720/día, mientras que para el segundo valor es Z_2 = -21.71 equivalente a un beneficio de $21.710/día.

La solución óptima corresponde a un valor de la función objetivo de $Z^* = 15.12$. La topología óptima corresponde a los equipos 2, 3, 4, 6, 8, 9. Después de resolver los dos primeros problemas iniciales se necesitaron dos iteraciones adicionales para arribar a la solución. Los resultados obtenidos son los siguientes en las variables continuas son los siguientes:

V=(0,0.47,0.725,0.66,0.,0.7,0,0.77,0.79)

x=(0.91,0,0.91,0.78,0.78,0.8,0.4,0.4,0.44,0.,0.44,0.,0.437,0.437,0.4,0.,0.4,0.,0.4,0.4,0.375,0.)

5

Lenguaje para la especificación de disyunciones y restricciones lógicas

5.1. Introducción

El desarrollo e implementación de la primera versión de LOGMIP fue importante porque nos permitió visualizar las dificultades y aspectos no resueltos en el desarrollo, implementación y solución de programas disyuntivos. Una de las primeras preguntas que surgió fue: cuál es el mejor modo de especificar una disyunción ó una proposición lógica?. Las herramientas disponibles masivamente para resolver problemas de optimización son sistemas comerciales para la resolución de programas matemáticos como GAMS (Brooke y otros, 1988) ó AMPL (Fourer y otros, 1993), que cuentan con un lenguaje declarativo de alto nivel para la especificación de expresiones matemáticas. Por medio de estos lenguajes es posible describir un problema de programación matemática mixto-entero lineal (MILP) ó mixto-entero no lineal (MINLP). Por lo general, estos sistemas no están adaptados para la incorporación de lógica en sus expresiones. Es por ello que, en nuestra primera implementación, adoptamos para expresar una disyunción una construcción del lenguaje GAMS que permite definir el dominio de aplicación de una restricción basado en el valor de una expresión constante. Si bien esto nos permitió momentáneamente resolver el problema de expresión, se necesita una expresión más natural que facilite la especificación de las expresiones lógicas y las disyunciones a quien modela un problema. Este aspecto fue resaltado en la introducción del primer capítulo cuando se expresó que la resolución de un problema de programación matemática o lógico, es un proceso iterativo donde el problema inicial se va depurando y refinando hasta obtener una solución que satisfaga a los objetivos de quien desarrolla el modelo. Para hacer más claro y efectivo este proceso es importante que el modelo sea compacto y claro para quien lo desarrolla. Compacto, para que en pocas líneas y con un análisis rápido se tenga una visión del modelo. Tiene que ser claro, para que quien busca mejorar el modelo pueda entender exactamente la estructura y composición del mismo. Para lograr estas dos cualidades el lenguaje y las construcciones con que se cuentan para la descripción de un modelo son de gran importancia.

En el pasado se han propuesto varias expresiones para la especificación de modelos condicionales similares a las disyunciones presentadas en los modelos (PH) y (PDG). Pantelides (1988) propuso un lenguaje para el simulador dinámico SpeedUp que se basaba en sentencias IF y los operadores lógicos “AND”, “OR” ó “NOT”, que

posteriormente fue ampliado para procesos continuos/discretos combinados (Barton y Pantelides, 1994, Oh y Pantelides, 1996). Más recientemente, Rico-Ramirez (1998) propuso varias sentencias basadas en construcciones como WHEN..CASE, SELECT..CASE, SWITCH..CASE, CONDITIONAL para la descripción de modelos condicionales en el sistema orientado a ecuaciones ASCEND. Algunas de estas construcciones también se pueden emplear para la descripción de disyunciones. Hooker (1998) propuso un lenguaje basado en operadores lógicos para la representación de restricciones lógicas en programas matemáticos.

Quienes más han evolucionado en la introducción de lógica en sus lenguajes son los sistemas basados en la Programación con Restricciones Lógicas (Darby-Dowman y otros, 1997; Bockmayr y Casper, 1998; Hajian y otros, 1995; Hajian y otros, 1996) (PRL). Esto se debe a que la PRL trabaja fundamentalmente con problemas que son altamente combinatorios y, por lo tanto, su formulación en términos de los lenguajes de programación matemática requiere de un gran número de variables y restricciones, generando programas difíciles de entender. Es por ello que en los sistemas PRL comerciales más populares como ECL^{i}PS^e ó ILOG, se pueden encontrar construcciones para describir expresiones lógicas, disyunciones, o sentencias especiales como "all-different". Es usual emplear los operadores lógicos "and", "or", "not", "equivalence", "implication" , "exclusive or" para relacionar ecuaciones, restricciones u otras expresiones lógicas. Las expresiones lógicas permiten expresar de manera compacta estos problemas combinatorios, que se emplean fundamentalmente en problemas de "scheduling" y de planeamiento. Estas expresiones se emplean luego a nivel de solución del problema, en el árbol de ramificación, para reducir el espacio de búsqueda de la solución. El lenguaje les permite combinar restricciones y/o definir restricciones sobre las restricciones (meta- retricciones). El uso de estas expresiones se pueden ver en el ejemplo presentado al final del capítulo 2. En este capítulo también se demostró que es posible transformar estas expresiones lógicas en disyunciones válidas para las representaciones propuestas en esta tesis.

En este capítulo el objetivo es presentar los componentes (operadores y operandos) y las expresiones que permitan especificar disyunciones y expresiones lógicas

que completen el lenguaje de programación matemática, que permitan la especificación de problemas con la formulación (PH) ó (PDG).

5.2. Lenguaje: operadores, operandos y expresiones

Para poder especificar un problema con la respresentación del problema (PH) ó (PDG) los elementos necesarios, además de aquellos que posee un lenguaje de programación matemática pero que no están disponibles en ellos, son los siguientes:

- Variables Booleanas que puedan tomar el valor Verdadero o Falso. Sin embargo, estas pueden ser reemplazadas por variables binarias, haciendo que el valor 0 sea Falso y el 1 el Verdadero
- Operadores Lógicos: $\wedge$ (“and”), $\vee$ (“or”), $\underline{\vee}$ (“or exclusivo”), ~(“not”,!,¬), ->(implicación),<->(equivalencia).
- Sentencias de selección (condicionales) para expresar disyunciones. Nuestra propuesta es el uso de sentencias IF..THEN..ELSE..ENDIF.
- Sentencias especiales para facilitar la expresión de restricciones lógicas y la lógica proposicional, sentencias tales como **atmost, atleast** ó **exactly**.

Las expresiones que se quieren especificar con estos elementos son:

- Las *sentencias de selección* de la forma *IF..THEN..ELSE..ENDIF* que se emplean para formular una disyunción. Estas seleccionan el conjunto de restricciones a aplicarse luego de evaluar una expresión lógica (ó una variable Booleana) en verdadero o falso. Un ejemplo simple sería:

  IF (Y_i) *THEN* $h_i(x) \leq 0$
  ELSE $g_i(x) \leq 0$
  ENDIF
- Los operadores lógicos que se emplean fundamentalmente para expresar la *lógica proposicional* que relaciona las distintas variables Booleanas que indican si los términos de las disyunciones son verdaderos o falsos, por ejemplo:

$$Y1 \wedge \neg Y2 \Rightarrow Y3 \vee Y4$$

- Los operadores lógicos relacionando expresiones matemáticas (restricciones, ecuaciones) que sirven para expresar relaciones lógicas entre las distintas expresiones. Por ejemplo:

$$[(x+3) \leq 0 \Rightarrow (y+4) \geq 0].$$

- Las sentencias especiales **atmost**, **atleast** ó **exactly** que sirven para especificar de manera compacta la lógica proposicional y restricciones lógicas. Su uso se explica al final del capítulo.

5.3. Expresión de las disyunciones

La propuesta del uso de las sentencias IF..THEN..ELSE..ENDIF para la expresión de las disyunciones obedece fundamentalmente a que es una sentencia muy conocida. La mayoría de los lenguajes de computación tienen alguna forma de implementación de la misma, ya que la sentencia es muy expresiva, no es ambigua, es fácil de entender, y sirve para la expresión de disyunciones de diferente tipo, como se muestra en las secciones siguientes.

Disyunciones de dos términos:

$$\begin{bmatrix} Verdadero \\ Restricciones\ 1 \end{bmatrix} \vee \begin{bmatrix} Falso \\ Restricciones\ 2 \end{bmatrix}$$

La descripción de una disyunción de dos términos (uno verdadero, el otro falso) como la anterior, que es frecuente encontrar en problemas de síntesis de procesos, se puede hacer por medio de la siguiente sentencia IF..THEN:

IF *(expresión lógica)* **THEN**

Restricciones 1 que aplican cuando expresión lógica es Verdadera

ELSE

Restricciones 2 que aplican cuando expresión lógica es Falsa

ENDIF

Supongamos la siguiente disyunción del ejemplo de diseño de la planta batch presentado en el capítulo 4:

$$\begin{bmatrix} Y_j \\ vt_j \geq log(2S_{ij}^*) + b_{i,j+1} \\ vt_j \geq log(2S_{ij}^*) + b_{i,j} \\ b_{ij} - b_{i,j+1} \leq \phi \\ b_{ij} - b_{i,j+1} \geq -\phi \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ vt_j = 0 \\ b_{ij} - b_{i,j+1} = 0 \end{bmatrix}$$

La disyunción de este ejemplo expresa que si el tanque de almacenamiento intermedio *j* va a ser instalado entre las etapas batch *j* y *j+1*, entonces se aplican las ecuaciones de diseño del volumen del tanque vt_j y las medidas de las etapas batch *j* y la *j+1* para el producto *i* tienen una diferencia que debe ser menor que ϕ, si no se instala un tanque, el tamaño del tanque es cero y no hay diferencia entre las medidas de las etapas batch *j* y la *j+1* para el producto *i*.

La expresión de esta disyunción con la expresión propuesta sería:

IF (Y_j) **THEN**

$$b_{ij} - b_{i,j+1} \leq -\phi$$

$$b_{ij} - b_{i,j+1} \leq \phi$$

$$vt_j \leq log(2S_{ij}^*) + b_{i,j}$$

$$vt_j \leq log(2S_{ij}^*) + b_{i,j+1}$$

ELSE

$$vt_j = 0$$

$$b_{ij} - b_{i,j+1} = 0$$

ENDIF

Disyunciones con varios términos

$$\begin{bmatrix} 1 \\ Restricciones\ 1 \end{bmatrix} \vee \begin{bmatrix} 2 \\ Restricciones\ 2 \end{bmatrix} \vee .. \vee \begin{bmatrix} N \\ Restricciones\ N \end{bmatrix}$$

Para una disyunción con varios términos como la anterior, la expresión IF..THEN es la siguiente:

IF *(expresión lógica1)* **THEN**

Restricciones 1 que aplican cuando la expresión lógica1 es Verdadera

ELSE IF *(expresión lógica2)* **THEN**

Restricciones 2 que aplican cuando la expresión lógica2 es Verdadera

ELSE IF *(expresión lógica3)* **THEN**

...

ELSE IF *(expresión lógicaN)* **THEN**

Restricciones N que aplican cuando la expresión lógicaN es Verdadera

ENDIF

Notar que la expresión anterior también se aplica cuando se tienen disyunciones con dos términos y cada una de ellas es activada por una expresión lógica diferente. Supongamos ahora el ejemplo (3-46) del capítulo 3 correspondiente a los tres círculos que no se interceptan:

$$\begin{bmatrix} Y_1 \\ (x_1-4)^2+(x_2-2)^2 \le 0.5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1-3)^2+(x_2-4)^2 \le 1 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1-1)^2+(x_2-1)^2 \le 1.5 \end{bmatrix}$$

Esta disyunción se expresa de la siguiente manera:

IF *(Y_1)* **THEN**

$(x_1-4)^2+(x_2-2)^2 \le 0.5$

ELSE IF *(Y_2)* **THEN**

$(x_1-3)^2+(x_2-4)^2 \le 1$

ELSE IF *(Y_3)* **THEN**

$(x_1-1)^2+(x_2-1)^2 \le 1.5$

ENDIF

Notar que la sentencia IF previa de múltiples alternativas podría parecer, en principio, que puede ser reemplazada por una sentencia SELECT..CASE o SWITCH..CASE del tipo:

SWITCH *(expresión)* {

CASE *valor_1*:

Restricciones a aplicar por valor_1

CASE *valor_2*:

Restricciones a aplicar por valor_2

...

CASE *valor_N*:

Restricciones a aplicar por valor_N

La principal diferencia de la sentencia SWITCH..CASE es que, en este último caso, se evalúa solo una expresión, que puede ser lógica o no, y que puede dar distintos valores, mientras que con las sentencias IF..ELSE IF ante cada decisión se evalúan diferentes expresiones, siendo por lo tanto la sentencia SWITCH un caso particular de la IF..ELSE IF. Por otra parte las sentencias IF presentan un modo más natural de especificar disyunciones.

Disyunciones embebidas

Caso a)

El primer caso el que involucra términos de la disyunción que se aplican por valores de Verdadero ó Falso. Una posible representación para este tipo de disyunciones es la siguiente:

$$\left[\begin{array}{l} \left[\begin{array}{l} \left[\begin{array}{l} Verdadero1 \\ \left[\begin{array}{l} Verdadero2 \\ \left[Verdadero3\right] \vee \left[Falso3\right] \end{array}\right] \end{array}\right] \vee \left[Falso2\right] \end{array}\right] \vee \left[Falso1\right] \end{array}\right]$$

IF *(expresión lógica$_1$)* **THEN**

Restricciones que aplican cuando expresión lógica$_1$ es Verdadera

IF *(expresión lógica$_2$)* **THEN**

Restricciones que aplican cuando expresión lógica$_2$ es Verdadera

IF *(expresión lógica$_3$)* **THEN**

Restricciones que aplican cuando expresión lógica$_3$ es Verdadera

ELSE

Restricciones que aplican cuando expresión lógica$_3$ es Falsa

ENDIF

ELSE

Restricciones que aplican cuando expresión lógica$_2$ es Falsa

ENDIF

ELSE

Restricciones que aplican cuando expresión lógica$_1$ es Falsa

ENDIF

Para ilustrar este caso con un ejemplo industrial, sea el siguiente ejemplo de optimización propuesto por Van der Heever y Grossmann (1999) para un diseño multiperíodo y de determinación de la capacidad de una red de unidades de proceso. Este problema involucra la selección de la topología *(Y_j)* de la red de unidades de proceso como también la posible expansión de la capacidad ante cada período de tiempo *(Z_{jt})*. Se propone una disyunción embebida porque tiene dos niveles de decisión jerárquicos, primero si la unidad de proceso es seleccionada, si es así, hay que determinar si la unidad se va a expandir en el período *t* con respecto al anterior *(t-1)*. La formulación del problema es la siguiente:

$$min\ Z = \sum_t \sum_j CE_{jt} + \sum_t \sum_i c_{it} x_{it}$$

sujeto a :

$$g_t(x_t, x_{t-1}) \le 0 \quad \forall t$$

$$\begin{bmatrix} Y_j \\ h_{jt}(Q_{jt}, x_t, x_{t-1}) \le d \quad \forall t \\ \begin{bmatrix} Z_{jt} \\ Q_{jt} = Q_{j,t-1} + QE_{jt} \\ CE_{jt} = \alpha_{jt} QE_{jt} + \beta_{jt} \end{bmatrix} \vee \begin{bmatrix} \neg Z_{jt} \\ Q_{jt} = Q_{j,t-1} \\ CE_{jt} = 0 \end{bmatrix} \forall t \end{bmatrix} \vee \begin{bmatrix} \neg Y_j \\ B^{jt} x_t = 0 \\ \forall t \end{bmatrix} \forall j$$

$$\Omega(Y, Z) = True$$

$$CE, Q, QE, x \ge 0 \quad ; \quad Y, Z \in \{True, False\}$$

donde $g_t(x_t,x_{t-1})$ son ecuaciones globales que son independientes de las decisiones discretas (por ejemplo, balances de masa) y dependen de las variables de estado en el período *t* (x_t) y en *t-1* (x_{t-1}); Y_j es una variable Booleana para decidir si se invierte en la unidad *j* en el período *t;* h_{jt} ecuaciones válidas para la unidad *j* en el período *t* y representan por lo general relaciones de entrada/salida para la unidad; Q_{jt} es la capacidad de la unidad *j* en el período *t*; QE_{jt} representa la expansión de capacidad de la unidad *j* en el período *t*; CE_{jt} el costo de la expansión de la unidad *j* en el período *t* (con dos coeficientes de costo α y β); Z_{jt} es una variable Booleana para decidir si se expande la capacidad de la unidad *j* en el período *t*; c_{it} es el costo asociado con la corriente de proceso *i* en el período *t.* El significado de la disyunción en este caso es: si la unidad de proceso se selecciona (Y_j=Verdadero) entonces aplicar la ecuación que relaciona las entradas con las salidas (h_{jt}) y aplicar el costo asociado ($c_{it}\ x_{it}$) si corresponde, si la unidad admite una expansión de capacidad en el período *t* (Z_{jt}=Verdadero) aplicar las ecuaciones de la expansión y del costo de expansión, sino (Z_{jt}=Falso) la expansión y su costo son cero, si la unidad no se instala (Y_j=Falso) un subconjunto de sus variables se hacen cero y no hay costo asociado con la instalación de la unidad ya que x_{it} es cero para esta unidad. La descripción de la disyunción por medio de la sentencia IF..THEN es la siguiente:

IF *(Y_j)* **THEN**

$$h_{jt}(Q_{jt}, x_t, x_{t-1}) \le d$$

IF *(Z_{jt})* **THEN**

$$Q_{jt} = Q_{j,t-1} + QE_{jt}$$

$$CE_{jt} = \alpha_{jt} QE_{jt} + \beta_{jt}$$

ELSE

$$Q_{jt} = Q_{j,t-1}$$

$$CE_{jt} = 0$$

ENDIF

ELSE

$$B^{jt} x_t = 0$$

ENDIF

Caso b)

En este caso los términos de la disyunción presentan selecciones múltiples. Un ejemplo para esta representación es la siguiente:

$$\left[\begin{array}{c} Verdadero1 \\ \left[Verdadero3 \right] \vee \left[Verdadero4 \right] \vee \left[Verdadero5 \right] \end{array} \right] \vee \left[Verdadero2 \right]$$

IF *(expresión lógica1)* **THEN**

Restricciones que aplican cuando la expresión lógica1 es Verdadera

IF *(expresión lógica3)* **THEN**

Restricciones que aplican cuando la expresión lógica3 es Verdadera

ELSE IF *(expresión lógica4)* **THEN**

Restricciones que aplican cuando la expresión lógica4 es Verdadera

ELSE IF *(expresión lógica5)* **THEN**

Restricciones que aplican cuando la expresión lógica5 es Verdadera

ENDIF

ELSE IF *(expresión lógica2)* **THEN**

Restricciones que aplican cuando la expresión lógica2 es Verdadera

ENDIF

Existen casos que son una combinación de los a) y b) para las disyunciones embebidas, en los cuales podemos tener disyunciones con dos términos (uno Verdadero y el otro Falso) e incluir disyunciones de selección múltiple, ó viceversa. Sea por ejemplo la disyunción correspondiente a la red de procesos de 9 unidades con ecuaciones de costos discontinuas presentado al final del capítulo 4:

$$\left[\begin{array}{c} Y_i \\ h_i(x) \le 0 \\ \left[\begin{array}{c} Z_1 \\ c_i = \alpha_{i1} x_i^{n_{i1}} + \beta_{i1} \\ x_1^{lo} \le x_i \le x_1^{up} \end{array}\right] \vee \left[\begin{array}{c} Z_2 \\ c_i = \alpha_{i2} x_i^{n_{i2}} + \beta_{i2} \\ x_1^{up} \le x_i \le x_2^{up} \end{array}\right] \vee \left[\begin{array}{c} Z_3 \\ c_i = \alpha_{i3} x_i^{n_{i3}} + \beta_{i3} \\ x_2^{up} \le x_i \le x_3^{up} \end{array}\right] \end{array}\right] \vee \left[\begin{array}{c} \neg Y_i \\ B^i x = 0 \\ c_i = 0 \end{array}\right]$$

En este ejemplo se cuenta con una disyunción de dos términos que encierra una disyunción de selección múltiple (las ecuaciones de costo). La expresión de esta disyunción embebida con sentencias IF es la siguiente:

IF *(Y_i)* **THEN**
$h_i(x) \le 0$

IF *(Z_1)* **THEN**
$c_i = \alpha_{i1} x_i^{n_{i1}} + \beta_{i1}$
$x_1^{lo} \le x_i \le x_1^{up}$

ELSE IF *(Z_2)* **THEN**
$c_i = \alpha_{i2} x_i^{n_{i2}} + \beta_{i2}$
$x_1^{up} \le x_i \le x_2^{up}$

ELSE IF *(Z_3)* **THEN**
$c_i = \alpha_{i3} x_i^{n_{i3}} + \beta_{i3}$
$x_2^{up} \le x_i \le x_3^{up}$

ENDIF

ELSE
$B_i x = 0$

$c_i = 0$

ENDIF

5.4. Expresión de la lógica proposicional y restricciones lógicas

La incorporación al lenguaje de los operadores lógicos: $\wedge$ (“and”), $\vee$ (“or”), $\underline{\vee}$ (“or exclusivo”), ~(“not”,!,¬), ->(implicación) y <->(equivalencia) permiten la descripción o definición de relaciones lógicas entre las restricciones, de manera similar a cómo lo hace actualmente la Programación con Restricciones Lógicas (PRL). Si bien el propósito de este trabajo es proponer un lenguaje para la Programación Híbrida (PH) y Disyuntiva (PDG), también es cierto, como ya se demostró en el capítulo 2 que es posible transformar restricciones lógicas en la forma disyuntiva. Por lo tanto, si se deja esta posibilidad, se brinda una mayor flexibilidad al modelador para definir el problema. No obstante se considera que la formulación por medio de disyunciones y de sentencias IF..THEN..ELSE..ENDIF dan mayor claridad y facilidad de interpretación de un modelo que la vinculación de ecuaciones y restricciones por medio de símbolos lógicos. Las transformaciones propuestas en el capítulo 2 entre expresiones de la (PRL) y la (PDG) establece un vínculo entre ambos campos de investigación, abriendo un espacio para trabajos futuros sobre teorías que vinculen ambas propuestas.

Si analizamos las restricciones lógicas del problema (2-35) del capítulo 2:

$$\{[\log(y + 2x+12) \le k+5] \vee [\, y \ge k^2 \,]\} \Rightarrow (x \le 0 \wedge y \le 1)$$

$$x \le 0 \Rightarrow k > 3$$

su expresión por medio de los operadores lógicos sería básicamente la misma. Si en cambio analizamos ahora su transformación a la disyunción de la forma (PDG) como en (2-36):

$$\begin{bmatrix} Y_1 \\ log(y+2x+12)-(k+5) \ge \varepsilon \\ y-k^2 \le \varepsilon \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ x \le 0 \\ y \le 1 \end{bmatrix}$$

$$\begin{bmatrix} W_1 \\ x \ge \varepsilon \end{bmatrix} \vee \begin{bmatrix} W_2 \\ k \ge 3 \end{bmatrix}$$

la expresión de ambas disyunciones por medio de sentencias IF..THEN sería:

IF *(Y_1)* **THEN**

$$log(y + 2x + 12) - (k + 5) \geq \varepsilon$$

$$y - k^2 \leq \varepsilon$$

ELSE IF *(Y_2)* **THEN**

$$x \leq 0$$

$$y \leq 1$$

ENDIF

IF *(W_1)* **THEN**

$$x \geq \varepsilon$$

ELSE IF *(W_2)* **THEN**

$$k \geq 3$$

ENDIF

interpretándose más fácilmente que la relación entre las restricciones por medio de símbolos.

Del mismo modo las proposiciones lógicas involucran relaciones por medio de símbolos, pero en lugar de tratarse de restricciones y ecuaciones del problema, en este caso se trata de las variables Booleanas. La descripción por medio de proposiciones lógicas no siempre puede ser clara y directa para un usuario, esto dependerá del caso y del usuario. Para síntesis de procesos y un usuario experimentado en estos problemas, escribir estas relaciones probablemente le resulte sencillo. Un ejemplo sería la proposición "seleccionar la unidad flash *(y_f)* implica seleccionar la columna de destilación *(y_d)* ó el absorbedor *(y_a)*", que se traduce en la forma de proposición lógica como sigue:

$$y_f \text{ -> } y_d \vee y_a$$

Para otros casos, la situación puede ser diferente. Considerando esto, y con el objeto de facilitar algunas construcciones léxicas comunes a los problemas lógicos, el lenguaje propuesto se completa con las siguientes sentencias especiales:

atmost *(lista de argumentos)*

atleast *(lista de argumentos)*

exactly *(lista de argumentos)*

la lista de argumentos de las sentencias anteriores pueden ser: una lista ó un arreglo indizado, donde los argumentos pueden ser variables discretas. El significado de cada una de las sentencias es el siguiente:

- **atmost** implica que como máximo (a lo sumo) uno de los componentes de la lista de argumentos debe ser verdadero ó satisfacerse,
- **atleast** implica que al menos uno de los argumentos de la lista debe ser verdadero ó cumplirse,
- **exactly** implica que sólo uno de los argumentos debe cumplirse ó ser verdadero.

La diferencia entre **atmost** y **exactly** radica que la primera puede no satisfacer/ser verdadero ninguno de los argumentos.

Ejemplos:

Supongamos que en un problema tengamos variables discretas (binarias ó Booleanas) indizadas Y(i), y queremos que sólo una o ninguna de ellas sea verdadera. Una posibilidad de especificarla es:

atmost *(Y(i))*

Recordar que otra posibilidad para este caso fue propuesta en el capítulo 2, por medio de la lógica proposicional (ecuación 2-2) o de una desigualdad matemática (ecuación 2-3). La sentencia **atmost** permite la especificación de lo mismo de una forma más compacta. Para una lista de argumentos, la construcción toma la forma:

atmost *(Y1, Y2, Y6, Y8)*

en donde uno o ninguno de la lista de argumentos debe ser verdadero. Una especificación mixta también puede ser factible:

atmost *(Y(i), Y2, Y6, Y8)*

Del mismo modo se pueden especificar sentencias **atleast** ó **exactly** con un significado diferente.

Las sentencias antes mencionadas se refieren sólo a un componente de los argumentos. Una variante que se puede plantear es el empleo de un número antes de la lista de argumentos que indique cuantos de los componentes de la lista de argumentos deben ser satisfechos. La sintaxis de estas sentencias para este caso es:

atmost (k) *(lista de argumentos)*

atleast (k) *(lista de argumentos)*

exactly (k) *(lista de argumentos)*

el valor **k** agrega el significado que a lo sumo, al menos y exactamente **k** elementos de la lista de argumentos deben ser verdaderos o satisfacerse. Estas sentencias tienen las propiedades de ser muy compactas, siempre y cuando no tengamos una lista muy larga de argumentos, y que son muy expresivas y completan la parte de expresiones lógicas de lenguajes para la expresión de problemas matemáticos/algebraicos. Un ejemplo concreto sería la restricción:

atmost (2) *(Y1, Y2, Y3, Y4)*

donde se permite que hasta dos variables tengan el valor de uno.

Por último cabe señalar que es posible desarrollar otras expresiones. Ejemplos son los siguientes:

adjacent *(lista de argumentos)*

que serviría para especificar que dos variables binarias (booleanas) adjuntas en una lista de argumentos deben ser verdaderas.

alldifferent*(lista de argumentos)*

en este caso serviría para especificar que todas las variables enteras de la lista tienen que tener valores distintos.

6

Implementación: LOGMIP segunda versión

6.1.Introducción

Las ideas y propuestas expresadas en los capítulos anteriores se pensaron siempre con el objetivo de llevarlas a cabo en la implementación de un sistema de solución de programas basados en lógica. El primer prototipo de LOGMIP fue una experiencia que nos permitió visualizar los fortalezas y debilidades de la implementación, pero que a su vez nos alentó a mejorarlo. La primera de estas últimas, era la falta de un lenguaje que nos permitiera describir un problema en términos de disyunciones y sus expresiones lógicas. Como consecuencia de esto, se propusieron los operadores, sentencias y construcciones léxicas del capítulo 5. Por otro lado, a nivel de los algoritmos de resolución, los métodos basados en lógica para problemas discretos/continuos no lineales son básicamente tres (de acuerdo con la gráfica 4.1 del capítulo 4): Aproximaciones Exterior y Descomposición Generalizada de Benders basados en lógica, Ramificación y Acotamiento basado en la cáscara convexa. Los otros métodos están basado en formulaciones mixto-enteras. Partiendo de formulaciones disyuntivas o híbridas, podemos llegar a representaciones mixto-enteras por medio de la transformación de las disyunciones en formulaciones algebraicas empleando las relajaciones "Big-M" o de la cáscara convexa como fue analizado en el capítulo 3. Por lo tanto, la programación disyuntiva no se contrapone a la programación matemática sino que la complementa en todos los aspectos de la resolución de un problema: formulación, implementación y resolución.

La incorporación de la lógica en los programas matemáticos expande las capacidades de este último, ampliando las posibilidades de modelado, haciéndolo más claros y compactos, sin sacrificar ventajas a la hora de resolverlos. Los aspectos discretos en lugar de formularlos como expresiones algebraicas se modelan por medio de expresiones lógicas, que en muchos casos presentan una forma más natural de describirlas y entenderlas. Con estos argumentos en mente se pensó, para desarrollo e implementación de lenguajes y algoritmos basados en lógica en ampliar la capacidades del sistema de programación matemática GAMS. La decisión de la implementación sobre la base de GAMS se basó fundamentalmente, como fue expresado en el capítulo 4, en que se tenía acceso a los códigos fuentes del programa DICOPT^{++} que tiene implementado el método AE. También se tenia acceso a los códigos fuentes de las funciones de entrada/salida de

GAMS, se tenía experiencia en el uso de este sistema y además, GAMS cuenta con un lenguaje muy completo de especificación de programas matemáticos algebraicos Los aspectos mas relevantes de esta implementación se explican en las siguientes secciones.

6.2. Implementación: visión general

En la Figura 6.1 se presenta un diagrama de flujo de los aspectos generales y sobresalientes de la implementación de la segunda versión de LOGMIP.

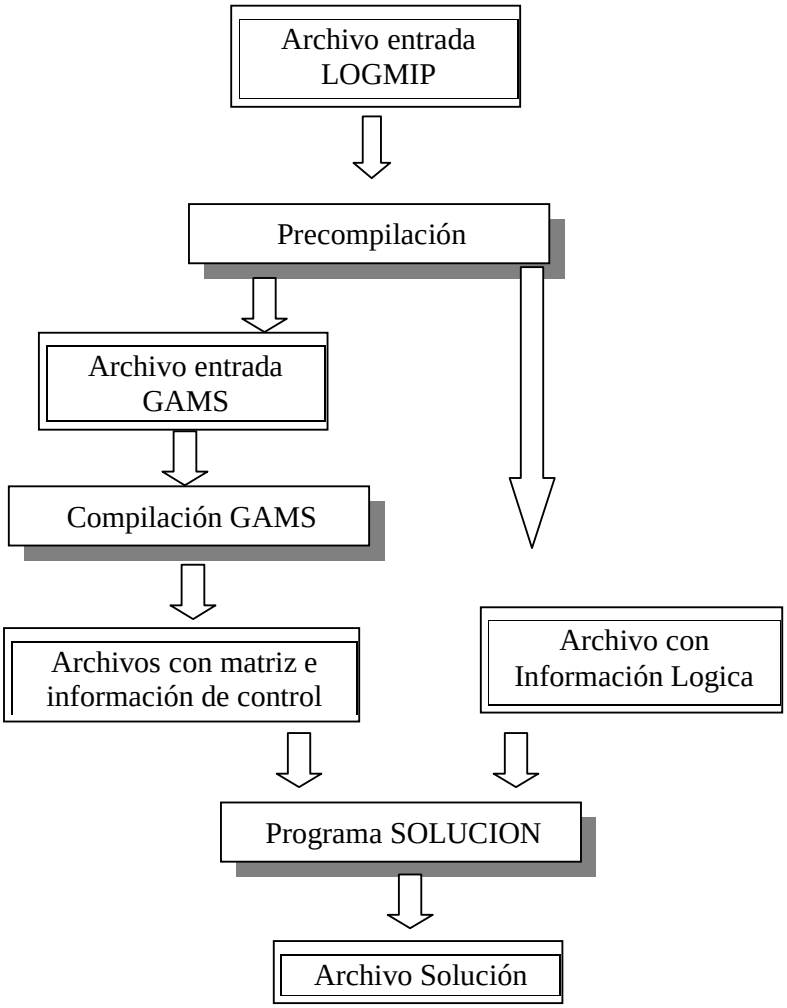


Figura 6.1: Diagrama de flujos general de LOGMIP

La explicación de la Figura 6.1 es la siguiente: la formulación de un problema disyuntivo ó híbrido parte de la especificación del problema con los elementos del lenguaje GAMS más los propuestos en el capítulo 5 para expresar disyunciones, proposiciones lógicas y expresiones lógicas. Es así que para especificar un problema en la forma PH ó PDG se tiene un superconjunto del lenguaje GAMS. El archivo de entrada pasa, en primer lugar, por una etapa de precompilación en donde se identifican y transforman los elementos nuevos del lenguaje. Esta técnica de precompilación es muy empleada en Tecnología de la Información en donde es posible encontrar, por ejemplo, sentencias especiales de acceso a Bases de Datos incluidas en lenguajes estándares de Programación como C^{++}, COBOL, etc., que se precompilan antes de pasar por los compiladores de los lenguajes. Al final de esta etapa de precompilado, si los chequeos de sintaxis y semánticos no han detectado ningún error, se generan dos archivos: uno en lenguaje GAMS con los elementos matemáticos-algebraicos del problema y otro con toda la información lógica (ecuaciones y variables de las disyunciones, tipos de disyunciones, cantidad, información de control, etc.). Del archivo GAMS se han eliminado por medio de comentarios las líneas con la declaración y/o definición de los elementos lógicos. Esto es así, para que estas expresiones no se detecten como errores de compilación en la etapa siguiente, donde este archivo es compilado para verificar la sintaxis y semántica de las expresiones del lenguaje GAMS (sin las extensiones lógicas obviamente). El resultado de esta última etapa, si todo esta bien, son los archivos con la información de las matrices del problema y la información de control que son necesarios para la resolución numérica del mismo. Si existieron errores sintácticos y/o semánticas en las etapas de precompilación ó de compilación GAMS se informan al final de esta última. Se espera entonces que el usuario las corrija y comience el ciclo nuevamente. Los archivos con la información de las matrices y de control, junto con el que tiene la información lógica, se combinan a nivel del programa de solución en donde ambos son necesarios para arribar a la solución óptima del problema. Como consecuencia de esta etapa de solución se genera el archivo que contiene los resultados numéricos.

Habiendo presentado de manera general como esta concebida la segunda versión de LOGMIP, en las siguientes secciones se dan detalles de cada uno de los componentes de esta implementación.

6.3. Precompilación

Esta etapa es una de las más importantes ya que está estrechamente relacionada con el lenguaje propuesto en el capítulo 5 y el reconocimiento de la sintaxis y semántica de las sentencias lógicas y su posterior transformación en información útil para los algoritmos de solución.

Varios aspectos deben resolverse antes de implementar un analizador sintáctico ("parser") de las diferentes sentencias IF..THEN..ELSE..ENDIF posibles para especificar disyunciones. En primer lugar, las reglas de nuestras sentencias no pueden contradecir las reglas generales ni particulares de GAMS referentes a identificadores, alcances, etc. Las disyunciones están relacionadas directamente con la definición de las restricciones (ecuaciones) que incluyen y con las variables Booleanas que la manejan. GAMS no tiene la posibilidad de definir variables Booleanas, pero si tiene variables binarias, para los propósitos prácticos de la implementación haremos uso de la variables binarias para reemplazar las Booleanas, asumiendo que 0 es Falso y 1 Verdadero. Las ecuaciones en GAMS tienen una identificación que debe ser única. Suponemos que la declaración (reserva) de nombres se hizo previamente, ya que ésta es la forma en que trabaja GAMS, para que un objeto pueda emplearse debe haberse declarado previamente. Esto abre dos posibilidades de definición de las restricciones en las disyunciones. En primer lugar, se puede hacer la definición de las restricción que definen las disyunciones incluidas en los bloques IF..THEN...ELSE..ENDIF, o se pueden definir fuera del bloque y dentro de la sentencia IF sólo mencionar las ecuaciones que las vinculan. Para visualizar mejor estas dos opciones, presentamos el siguiente ejemplo:

$$\begin{bmatrix} Y_1 \\ (x_1-4)^2+(x_2-2)^2 \le 0.5 \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ (x_1-3)^2+(x_2-4)^2 \le 1 \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ (x_1-1)^2+(x_2-1)^2 \le 1.5 \end{bmatrix}$$

Esta disyunción se expresa, incluyendo ya la sintaxis de GAMS para las ecuaciones, para la primera opción de la siguiente manera:

positive variables x1, x2;
binary variables Y1, Y2, Y3;
equations circle1,circle2,circle3;
IF *(Y_1)* **THEN**
circle1.. sqr(x1 - 4) + sqr(x2 - 2) =L= 0.5 ;
ELSE IF *(Y_2)* **THEN**
circle2.. sqr(x1 - 3) + sqr(x2 - 4) =L= 1 ;
ELSE IF *(Y_3)* **THEN**
circle3.. sqr(x1 - 1) + sqr(x2 - 1) =L= 1.5 ;
ENDIF

La otra posibilidad es hacer las definiciones fuera del bloque IF y luego construir un bloque IF empleando los nombre solamente es la siguiente:

positive variables x1, x2;
binary variables Y1, Y2, Y3;
equations circle1,circle2,circle3;
circle1.. sqr(x1 - 4) + sqr(x2 - 2) =L= 0.5 ;
circle2.. sqr(x1 - 3) + sqr(x2 - 4) =L= 1 ;
circle3.. sqr(x1 - 1) + sqr(x2 - 1) =L= 1.5 ;
IF *(Y_1)* **THEN**
circle1
ELSE IF *(Y_2)* **THEN**
circle2
ELSE IF *(Y_3)* **THEN**
circle3
ENDIF

Nos inclinamos por la primera implementación fundamentalmente porque se puede extraer fácilmente la información contenida en cada disyunción, ya que la definición de las restricciones que la componen están incluidas en ella. En la otra opción,

en cambio, si bien es posible conocer la identificación de las ecuaciones que componen una disyunción, para conocer su forma y su tipo habrá que mirar en otro lugar del archivo lo que no resulta conveniente. Por otro lado, esto tiene la ventaja que si una ecuación se repite en más de un término de una disyunción se repetirá su identificación, pero no su definición (como sería con el primer caso presentado) haciendo un modelo más compacto. De todos modos, se pierde la idea del contenido de las distintas restricciones.

Otro aspecto a analizar aquí, relacionado con la implementación en GAMS, es que la identificación de las ecuaciones permite definir en un mismo archivo diferentes modelos de optimización a través de la sentencia MODEL. Si se define una identificación para las disyunciones permitirá, del mismo modo, definir diferentes modelos e incluir en ellos distintas disyunciones, por lo que es apropiado contar con un identificador para las disyunciones. Para identificar la definición de una disyunción se pueden emplear los dos puntos seguidos (..) que se emplean para la definición de las ecuaciones. De ésta forma la gramática de la definición de disyunciones no será libre de contexto, sino que dependerá del mismo. Esto significa que si un identificador tiene los dos puntos seguidos, para determinar si se trata de la definición de una disyunción o de una ecuación deberemos ver si existe una sentencia IF..THEN..ELSE..ENDIF después de ellos o no. Que dependa del contexto dificulta los chequeos de sintaxis y semántica. Por otra parte si se elige otro símbolo para que la definición de las disyunciones sea libre del contexto, se facilita el chequeo de sintaxis y semántica pero se introducen nuevos símbolos que pueden complicar los elementos del lenguaje. Es difícil tomar una definición en este aspecto, y va mas allá del alcance de esta tesis. Contar con la posibilidad de definir distintos modelos con un mismo conjunto de disyunciones suena atractivo y da flexibilidad a la hora de resolver problemas. Para este caso nos inclinamos por emplear una gramática que no es libre del contexto para no complicar con más elementos el lenguaje. De esta forma, la definición del ejemplo de ILOG presentado en el capítulo 2 (2-36) sería de la siguiente manera:

```
* Declaración de variables
*
variables X, obje;
positive variables  Y;
integer variable K;
binary variables Y1, Y2, Z, W1, W2;

* Declaración de ecuaciones
*
equations con1, con2, con3, cost, first11, first21, first22, first31
          secnd11, secnd21, secnd22, secnd31, prop1;

*Declaración de disyunciones
*
disjunctions disy1, disy2;

* Ecuaciones que deben satisfacerse independientemente de las decisiones discretas
*
con1.. POWER(x,3) + 10.*X - EXP(X*LOG(Y)) + EXP(K*LOG(2)) =E= 0.0    ;
con2.. K*X + 7.7*Y - 2.4 =E=0.0         ;
con3.. EXP((Y+1)*LOG(K-1)) =L= 10  ;

*  DISJUNCTIONS
*
disy1..   IF(Y1) THEN
                first11.. LOG( Y + 2*X + 12) – (k-5) =G= EP ;
                first12.. Y - K**2 =L= EP;
          ELSE IF(Y2) THEN
                secnd21.. X =L= 0;
                secnd22.. Y =L= 1;
          ENDIF

disy2..   IF(W1) THEN
                first31.. X =G= EP;
          ELSE IF(W2) THEN
                secnd31.. K =G= 3;
          ENDIF
cost.. obje =E= x;

MODEL   A1  / con1, con2, disy2/;
MODEL   A2  /con1, con2, con3, disy1/;
MODEL   Logic  /all/;

SOLVE Logic USING DISJUNCTIVE minimizing obje;
```

Como se observa en el ejemplo se han definido tres modelos a partir de la declaración de la identificación de las disyunciones y de las ecuaciones, dando mayor flexibilidad a quien modela. La etapa de precompilación si no detecta errores produce un archivo listo para ser compilado por GAMS. Las líneas de la definición de la lógica se comentan con un asterisco (*) en la primera columna, no así la declaración de las disyunciones que se deben mantener para que sigan siendo válidas las sentencias MODEL. El archivo luego de la etapa de precompilación es el siguiente:

```
* Declaración de variables
*
variables X, obje;
positive variables  Y;
integer variable K;
binary variables Y1, Y2, Z, W1, W2;

* Declaración de ecuaciones
*
equations con1, con2, con3, cost, first11, first21, first22, first31
          secnd11, secnd21, secnd22, secnd31, prop1;

*Declaración de disyunciones
*
 disjunctions disy1, disy2;

* Ecuaciones que deben satisfacerse independientemente de las decisiones discretas
*
con1.. POWER(x,3) + 10.*X - EXP(X*LOG(Y)) + EXP(K*LOG(2)) =E= 0.0     ;
con2.. K*X + 7.7*Y - 2.4 =E=0.0          ;
con3.. EXP((Y+1)*LOG(K-1)) =L= 10  ;

*  DISJUNCTIONS
*
* disy1..   IF(Y1) THEN
                first11.. LOG( Y + 2*X + 12) – (k-5) =G= EP ;
                first12.. Y - K**2 =L= EP;
*           ELSE IF(Y2) THEN
                secnd21.. X =L= 0;
                secnd22.. Y =L= 1;
*           ENDIF
```

```
* disy2..   IF(W1) THEN
                first31.. X =G= EP;
*          ELSE IF(W2) THEN
                secnd31.. K =G= 3;
*          ENDIF

cost.. obje =E= x;

MODEL   A1  / con1, con2, disy2/;
MODEL   A2  /con1, con2, con3, disy1/;
MODEL   Logic  /all/;

SOLVE Logic USING DISJUNCTIVE minimizing obje;
```

Observar que la única diferencia de este archivo luego de la etapa de precompilación son los asteriscos que "comentan" las instrucciones IF..THEN.. ELSE..ENDIF.

El archivo con la información lógica no se muestra aquí porque tiene formato binario. Pero la información que contiene, tomando en cuenta nuestro ejemplo, es que la disyunción *disjy1*, tiene dos términos, el primer término se aplica cuando la variable *Y1* es verdadera (1) y las ecuaciones que se deben satisfacer son *first11* y *first12*, el segundo término se activa cuando *Y2* es verdadera (1) y las ecuaciones que se deben satisfacer son *secnd21* y *secnd22* y de la misma forma para la otra disyunción.

Un comentario aparte merece la definición de proposiciones lógicas, si bien los operadores lógicos deben estar disponibles para que sean empleados por el usuario dentro de la definición de un programa disyuntivo ó híbrido, creemos necesario mantener un programa aparte denominado "LOG2MAT" que realiza una transformación de un archivo de entrada con proposiciones lógicas en desigualdades matemáticas de modo tal que el usuario tenga la oportunidad de experimentar el mismo e identificar las restricciones enteras más comunes que surgen de una proposición lógica. El archivo de desigualdades matemáticas puede luego ser incluido en un archivo LOGMIP. Recordemos en este punto que debido a la dificultad que puede presentar para algunos usuarios el empleo de la lógica es que se han definido sentencias especiales para su manejo como: *atmost, atleast, exactly, adjacent* y *alldifferent.*

6.4. Implementación de los algoritmos de solución

La estructura de resolución de LOGMIP versión 2 relacionada con los algoritmos de solución a aplicar de acuerdo con la formulación presentada es la siguiente:

a) si el problema se modela como mixto-entero el programa por omisión que se utiliza es el de AE extendido y el programa de resolución DICOPT++ se emplea para la solución. Se puede elegir también entre los métodos de DGB ó PCE.

b) si el problema se modela como PH ó PDG, LOGMIP por omisión genera un programa mixto-entero (MINLP) por medio de la relajación por la cáscara convexa y las proposiciones lógicas se transforman en desigualdades matemáticas. El problema se resuelve como un MINLP con cualquiera de los métodos enunciados en a). Si la formulación es una con disyunciones de dos términos (uno Verdadero y el otro Falso) tal que pueda aplicarse el método de AE basado en lógica el usuario puede optar por este método para problemas PDG o el extendido para problemas híbridos.

Implementación de algortimos MINLP

Como se expresó en el capítulo 4, para la implementación de los métodos DGB y PCE de solución de problemas MINLP se empleó como base el código de DICOPT^{++} que tiene implementado el método de AE extendido para el tratamiento de ecuaciones de igualdad y restricciones no-convexas (Viswanathan y Grossmann, 1990). El código fue escrito en lenguaje de programación C. La descomposición en módulos de este programa facilitó el desarrollo de esta implementación. Como se vió en el capítulo 1, la diferencia entre los métodos de AE y el de DGB está en el modo que se genera el problema maestro, más específicamente en la manera que se generan las linealizaciones de las ecuaciones no-lineales. Como ésto se encuentra en DICOPT^{++} encapsulado en una función de lenguaje C, para la implementación se desarrolló una nueva función con las características de la linealización para DGB. El resto del código en general no se alteró. En la Figura 6.2 se muestra el diagrama de flujo de la lógica de los métodos de resolución AE y DGB tal como están implementados actualmente.

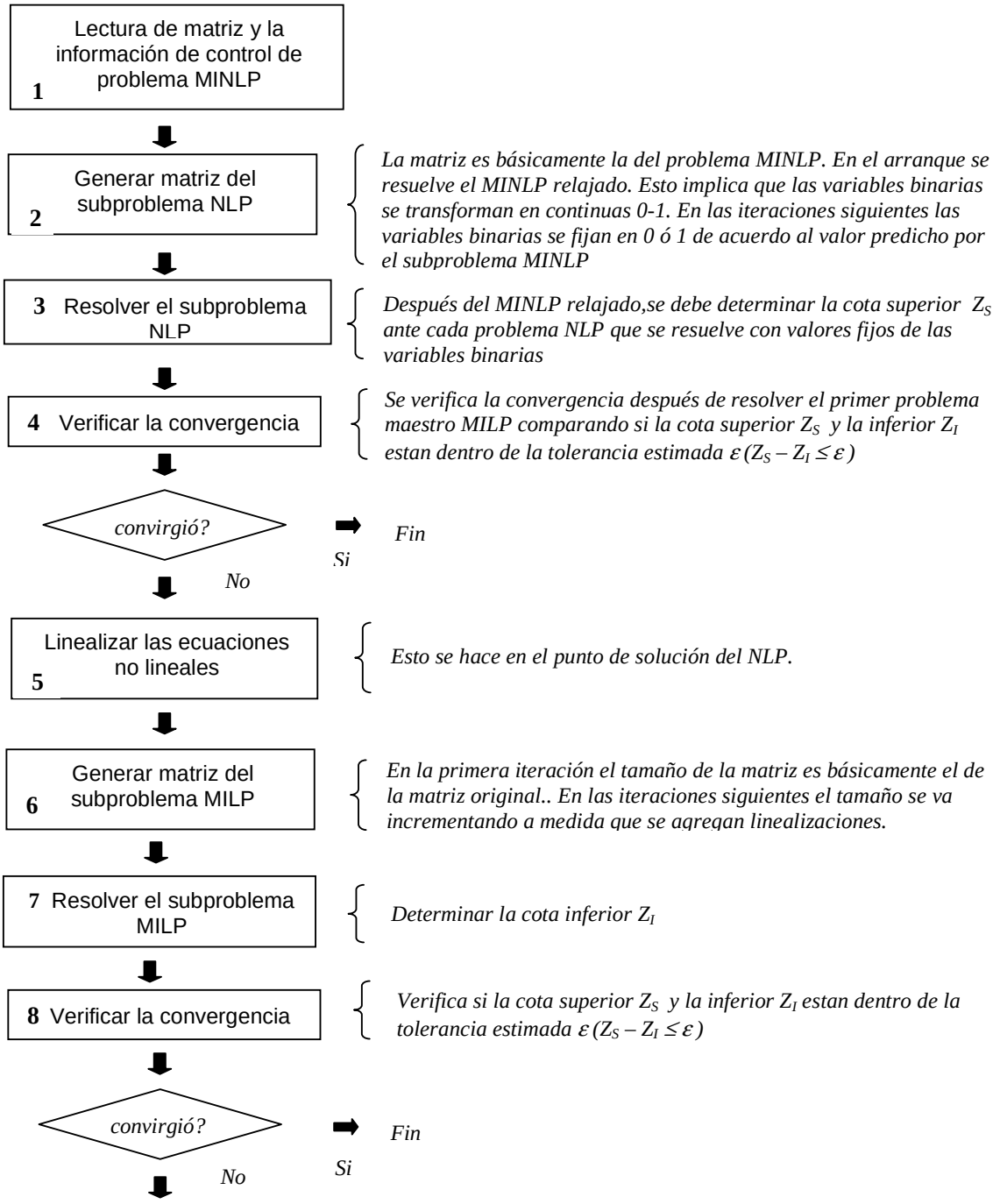


Figura 6.2: Diagrama de flujos de los métodos AE y GBD

Como se puede observar en la Figura 6.2 la lógica de los algoritmos de AE y DGB es básicamente la misma. En donde difieren fundamentalmente es en la implementación de las funciones para la ejecución de las etapas **5** y **6**.

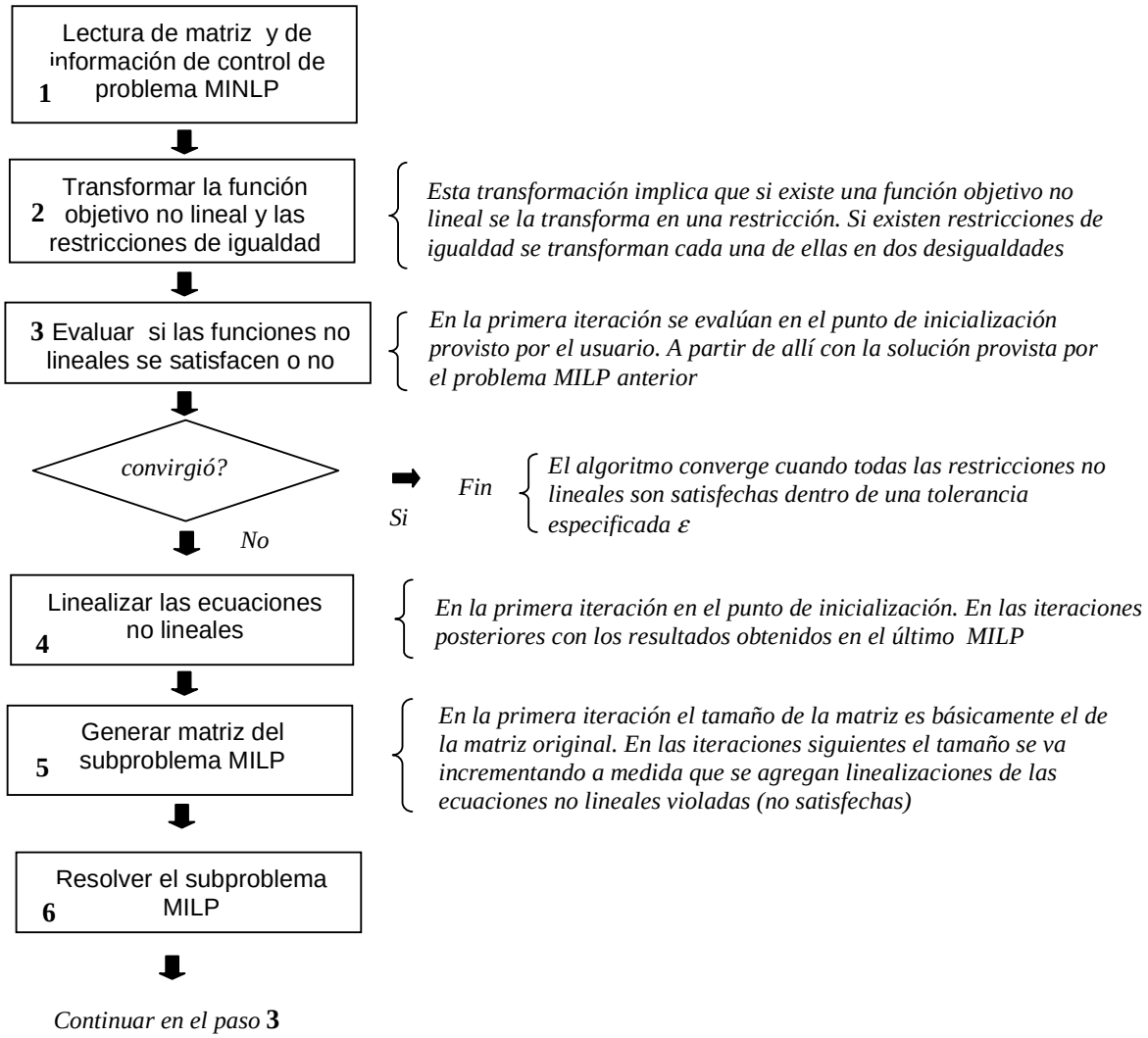


Figura 6.3: Diagrama de flujos de la implementación del método PCE

Para la implementación del algoritmo PCE se debe recordar que no se resuelven subproblemas NLP. Por lo tanto, los módulos de DICOPT^{++} relacionados con la

resolución de estos subproblema se quitaron. Otros aspectos a tener cuenta son que si se tiene una función objetivo no lineal ésta se debe transformar en una restricción, que si se tienen restricciones de igualdad se deben transformar en dos desigualdades una por mayor-igual y otra por menor-igual. Además, la manera de generar las linealizaciones de las restricciones no lineales se hacen de igual modo que para el método de AE. Para nuestra implementación del método PCE se linealizan todas las ecuaciones no lineales violadas, en lugar de sólo la más violada como lo pide el algoritmo. En la Figura 6.3 se muestra el diagrama de flujos de la implementación de este algoritmo de solución.

También se ha trabajado en la mejora del código de DICOPT^{++}, incorporándole algunas características adicionales. Las mejoras que se introdujeron son :

- Inicialización con valores fijos. Hasta este momento el método de AE empleaba la solución del MINLP relajado como método de arranque, a partir de esta incorporación, el usuario con buenas estimaciones puede incorporar los valores iniciales de las variables binarias como método de arranque.
- Posibilidad de definir variables enteras para las decisiones discretas. En la versión previa de DICOPT^{++} no era posible incorporar decisiones discretas sino a través de las variables binarias. Si se necesitaban variables enteras se empleaban variables binarias para ello, aumentando el tamaño del modelo, de esta forma los usuarios que necesiten variables enteras pueden hacerlo directamente.
- Impresión de los resultados intermedios de las variables discretas. Antes de esta mejora no era posible conocer las soluciones enteras obtenidos en las iteraciones intermedias.
- Con el propósito de analizar resultados el usuario puede imprimir el archivo de entrada GAMS de cada uno de los problemas maestros MILP que se resuelven en cada una de las etapas intermedias.

Métodos AE basados en lógica

En la primera versión de LOGMIP se implementaron los métodos Aproximación Exterior basados en lógica para problemas disyuntivos e híbridos. La principal dificultad que presentaba la implementación de estos métodos es la naturaleza dinámica de las

matrices, debido a que cambia la dimensión del subproblema NLP según las disyunciones que se están incluyendo y excluyendo del modelo. Esta dimensión a veces se incrementa, a veces disminuye. Esto contrasta con la naturaleza estática que la mayoría de los algoritmos MINLP anteriores. Es por esto que en la primera versión de LOGMIP, la implementación se basó en escribir, en cada una de las iteraciones, archivos GAMS para los subproblemas NLP y MILP. Estos son compilados para verificar la sintaxis y semántica de los mismos y luego son resueltos por los programas de resolución NLP y MILP correspondientes. La dificultad que tiene esta implementación es el tiempo extra necesarios para la compilación y generación de las matrices de cada subproblema ante cada iteración. Es por ello que se pensó para esta segunda versión mejorar esto, trabajando directamente con las matrices de los subproblemas en memoria incrementando y disminuyendo la medida de la matriz NLP de acuerdo a las necesidades. La incorporación del lenguaje más la estructura de archivos con la información lógica nos permitió realizar esta implementación. Desde que el método AE basado en lógica fue propuesto por primera vez por Turkay y Grossmann (1996a) nuevas extensiones, y fundamentalmente nuevas inicializaciones fueron propuestas, el lenguaje para la descripción de estos problemas debe incluir sentencias que especifiquen como va a arrancar el método. Básicamente las inicializaciones podrían ser cuatro:

fija: por medio del “set covering” ó provista por el usuario

relajada: por medio de la cáscara convexa ó “Big-M”

Por razones de simplicidad y eficacia en la implementación de la incialización se trabajará con la fija provista por el usuario y la cáscara convexa para la relajada. En la Figura 6.4 y 6.5 se encuentran los diagramas de flujo de estas implementaciones respectivamente.

Métodos basados en la cáscara convexa

En esta sección se aborda el método que es el más general para resolver problemas disyuntivos o híbridos que es la transformación de los problemas PH y PDG en problemas mixto-enteros MINLP por medio de la relajación de las disyunciones por la cáscara convexa. La manera de implementar las transformaciones es haciendo una

transformación del archivo de entrada GAMS con la descripción de un problema PH ó PDG en uno con la descripción del problema MINLP generado por la cáscara convexa.

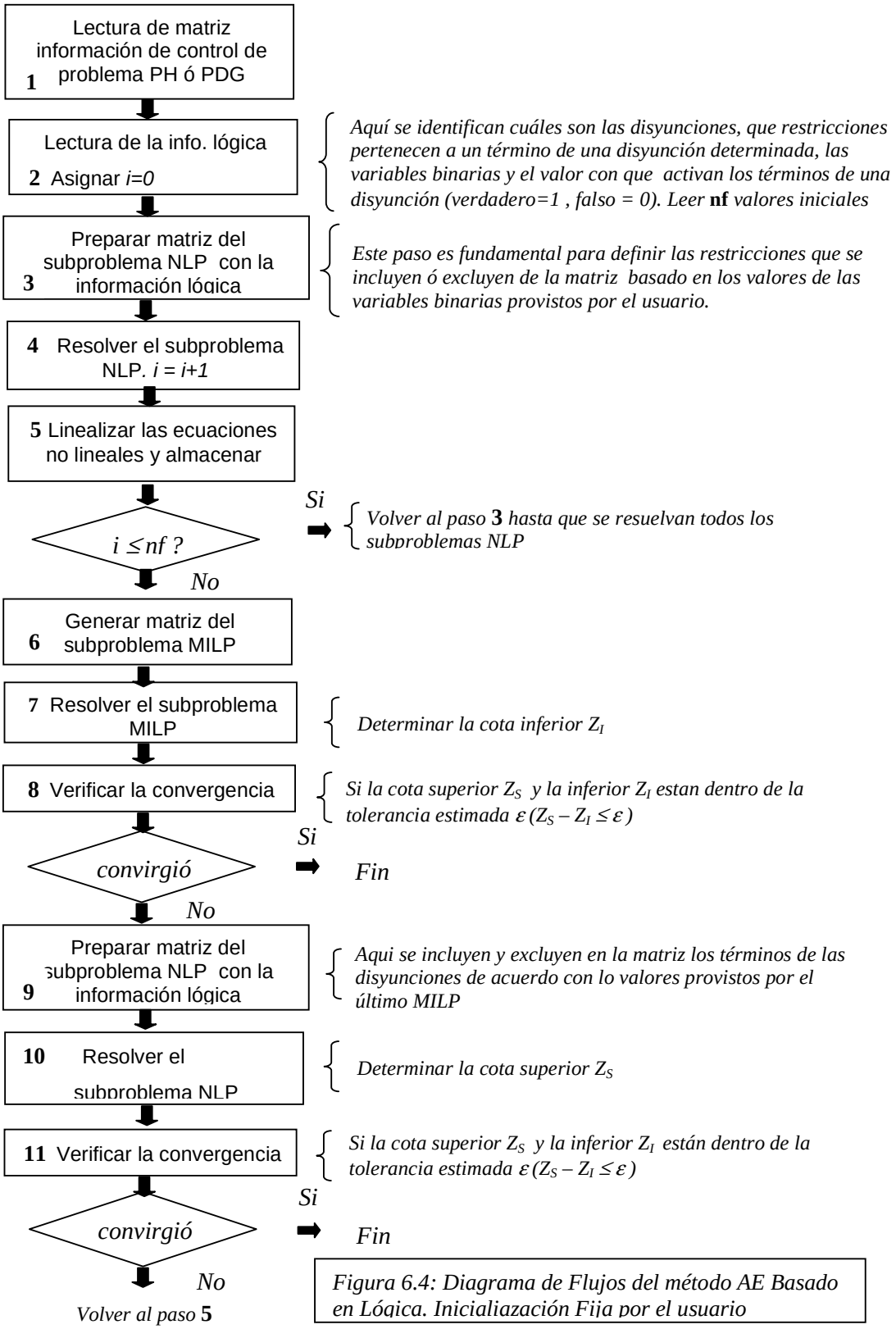


Figura 6.4: Diagrama de Flujos del método AE Basado en Lógica. Inicialiazación Fija por el usuario

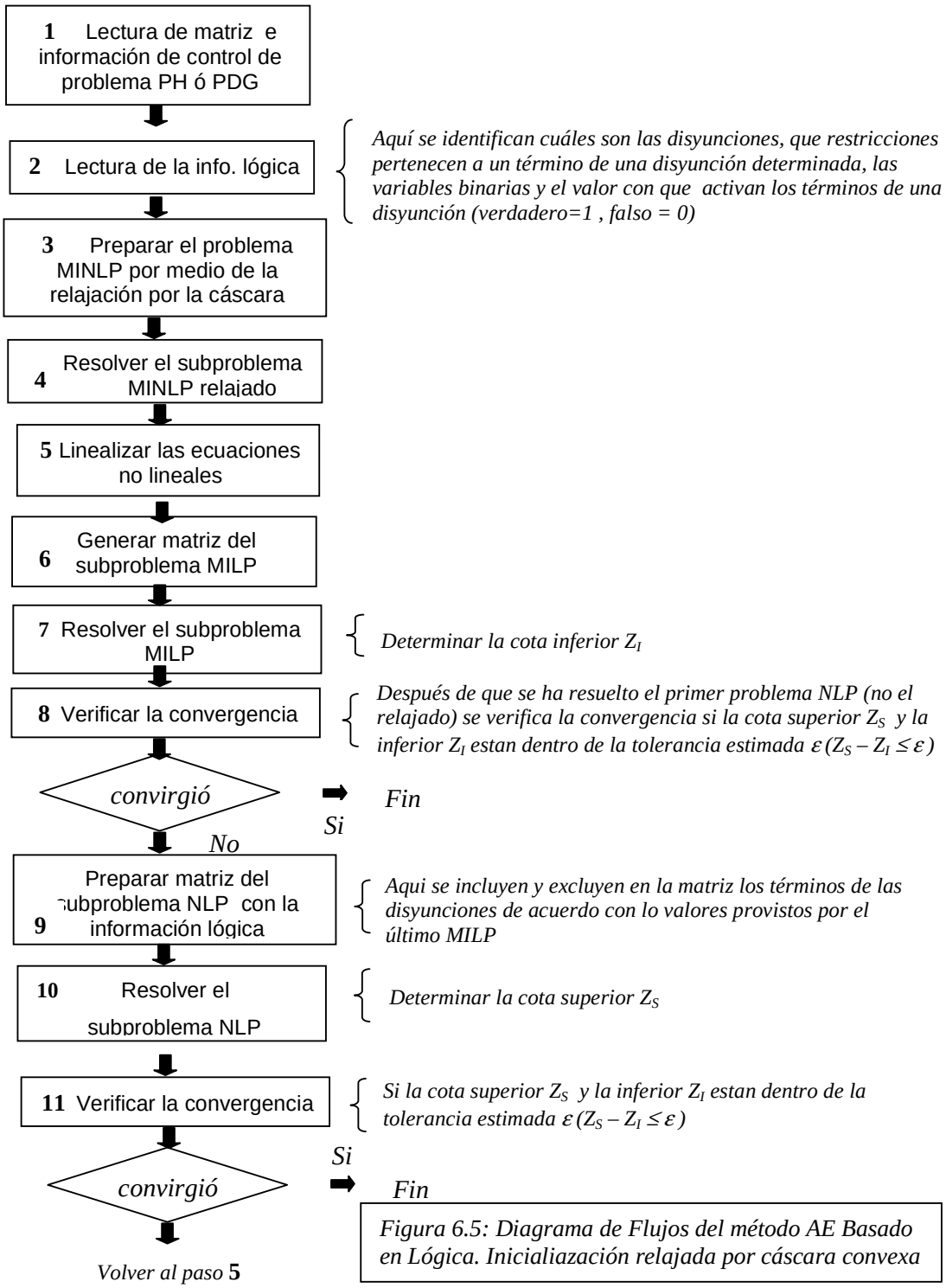


Figura 6.5: Diagrama de Flujos del método AE Basado en Lógica. Inicialiazación relajada por cáscara convexa

Se creyó conveniente este modo por dos motivos. En primer lugar generar la cáscara convexa a nivel de la matriz del problema original es excesivamente complicado, por que implica cambiar las ecuaciones de las restricciones originales agregando variables y cambiando la forma de la ecuación, lo que implica también realizar cambios a nivel de sus derivadas, etc.. En segundo lugar, si se genera un archivo con la cáscara convexa del problema original, el usuario antes de resolver el problema puede visualizar el archivo generado e introducir cambios si es necesario. Esta transformación a nivel de archivos de entrada es posible porque la generación de la cáscara convexa para una disyunción puede hacerse de manera sistemática. La generación de la relajación "Big-M" también es posible hacerla de modo sistemático pero el problema que tiene es precisamente el cálculo de los valores de "*M*", fundamentalmente cuando no se tienen cotas sobre las variables. Por otra parte a un usuario le es más fácil generar una relajación "Big-M" que una cáscara convexa. El diagrama de flujos de esta transformación puede verse en la Figura 6.6.

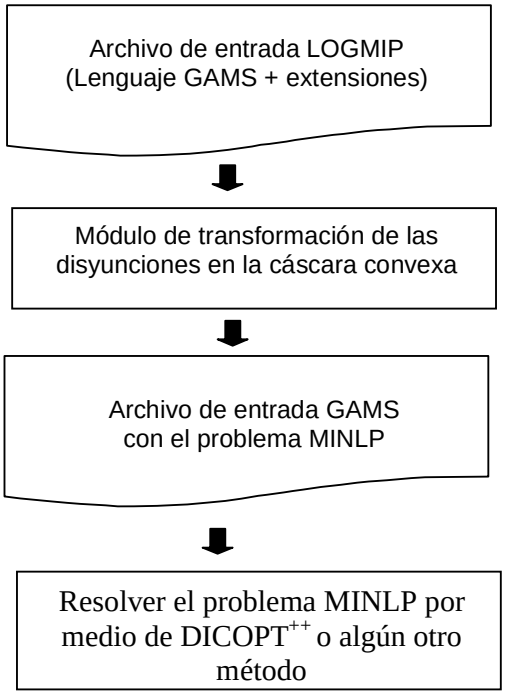


Figura 6.6: Diagrama de flujo para la transformación de un programa PH ó PDG en un MINLP por la cáscara convexa

Un ejemplo de cómo es esta implementación se puede ver en la transformación del ejemplo de ILOG presentado previamente.

```
$TITLE CLP Problem from ILOG
*
*  CONVEX HULL of  ORIGINAL PROBLEM
*
set  i / 1 * 4 /, l /1*2/, n /1*3/, j /1*4/;

parameters    COEFF(i) / 1 1, 2 2, 3 3, 4 4/;
scalar EP / 0.0000001/;

*Declaración de variables
*
variables x, v(l,n,l), dummy;
positive variables y, k, lamda(j);
binary variables y1(i), y2(j);

*Declaración de ecuaciones
*
equations con1, con2, con3, first11, first12, secnd21, secnd22, first31,
          secnd31, sumx1, sumy1, sumk1,  sumx2, sumk2;
equations sumlamda1, sumlamda2, inteq, inteq1,binfx(j), obj;
equations   boundk11, boundk12, boundk13, boundk14, boundy11, boundy12,
            boundx11, boundx12, boundx21, boundx22, boundx31, boundx32,
            boundx33, boundx34, boundk31, boundk32, boundk33, boundk34;

* restricciones globales
*
con1.. power(x,3) + 10.* x - exp(x*log(y)) + exp(k*log(2)) =e= 0.0    ;
con2.. k*x + 7.7 * y - 2.4 =e=0.0       ;
con3.. exp((y+1)*log(k-1)) =l= 10        ;
*
*** DISYUNCION 1

first11.. (lamda('1')+ep)* [log((v('1','2','1')/(lamda('1')+ep))
          +(2 * (v('1','1','1')/(lamda('1')+ep)) + 12))
          (v('1','3','1')/(lamda('1')+ep)) - 5] =g= ep ;
first12.. (lamda('1')+ep)*[(v('1','2','1') / (lamda('1')+ep))-
          ((v('1','3','1')/(lamda('1')+ep))**2)] =l= ep;

secnd21..  v('1','1','2') =l= 0      ;
secnd22..  v('1','2','2') =l= lamda('2')    ;
```

```
*   cotas DISYUNCION 1

  boundx11.. v('1','1','1')=l= x.up*lamda('1');
  boundx12.. v('1','1','1')=g= -5*lamda('1');
  boundx21.. v('1','1','2')=l= x.up*lamda('2');
  boundx22.. v('1','1','2')=g= -5*lamda('2');
  boundk11.. v('1','3','1')=l= k.up*lamda('1');
  boundk12.. v('1','3','2')=l= k.up*lamda('2');
  boundk13.. v('1','3','1')=g= 0;
  boundk14.. v('1','3','2')=g= 0;
  boundy11.. v('1','2','1')=l= y.up*lamda('1');
  boundy12.. v('1','2','2')=l= y.up*lamda('2');

  sumx1.. v('1','1','1')+ v('1','1','2') =e= x  ;
  sumy1.. v('1','2','1')+ v('1','2','2') =e= y  ;
  sumk1.. v('1','3','1')+ v('1','3','2') =e= k  ;
  sumlamda1.. lamda('1')+lamda('2') =e= 1         ;

*
*** DISYUNCION 2
  first31..  v('2','1','1') =g= ep *lamda('3');
secnd31..  v('2','3','2') =g= (3+ep) * lamda('4')      ;
*
*cotas DISYUNCION 2

  boundx31.. v('2','1','1')=l= lamda('3');
  boundx32.. v('2','1','1')=g= -5*lamda('3');
  boundx33.. v('2','1','2')=l= lamda('4');
  boundx34.. v('2','1','2')=g= -5*lamda('4');
  boundk31.. v('2','3','1')=l= k.up*lamda('3');
  boundk32.. v('2','3','2')=l= k.up*lamda('4');
  boundk33.. v('2','3','1')=g= 0;
  boundk34.. v('2','3','2')=g= 0;

  sumx2.. v('2','1','1')+ v('2','1','2')=e= x;
  sumk2.. v('2','3','1')+ v('2','3','2')=e= k;
  sumlamda2.. lamda('3')+lamda('4') =e= 1 ;

*
* lamda toma el valor 0 ó 1 de acuerdo con y2
*
   binfx(j).. y2(j) =e= lamda(j)   ;
```

```
*  inteq e inteq1 asignan valores enteros 0,1,2,3,4 a k ( variable entera)
*
  inteq..   k  =e= sum (I , COEFF(I) * Y1(I)) ;
  inteq1..  sum(i,y1(i))=e=1;

*
* función objetivo dummy
obj..   dummy =e= x;

v.lo('1','2','1')=0;
v.lo('1','3','1')=0;
k.up=4;
y.up=100;
x.up=500;
x.lo=-5;
y.l = 2 ;
x.l=-3 ;
k.l= 2   ;

MODEL   ilog   / ALL /          ;
option nlp= conopt;
SOLVE ilog USING MINLP MINIMIZING dummy;
```

Se debe notar que existe un prototipo implementado con la segunda versión de LOGMIP. Con este prototipo se han resuelto los problemas 4.4.1, 4.4.3, 4.4.4, y 4.4.5 presentados en el capítulo 4. En el Apéndice B se encuentran los archivos de entrada LOGMIP para los ejemplos 4.4.1, 4.4.3 y 4.4.5.

7

Conclusiones

7.1. Introducción

Los principales objetivos de esta tesis han sido en primer lugar, presentar alternativas válidas para el modelado de problemas discretos/continuos no lineales por medio de disyunciones, proposiciones lógicas y variables binarias y, por otra parte, en proponer métodos y herramientas que simplifiquen las tareas de implementación y solución de estos problemas. Esta tesis contribuye en los aspectos del modelado fundamentalmente en lo referente a la formulación de las decisiones discretas, por medio de disyunciones y/o en forma mixto-entera. También se ha indicado como transformar o trasladar restricciones lógicas generales a la forma disyuntiva y de proposiciones lógicas. En cuanto a los métodos, técnicas y herramientas, se ha propuesto un lenguaje para la expresión y descripción de problemas basados en disyunciones y proposiciones lógicas, que complementa y agrega posibilidades a los lenguajes matemáticos actuales que se emplean para describir problemas mixto-entero no lineales. También se han desarrollado extensiones a los algoritmos actuales para tratar problemas híbridos y se han propuesto guías para la implementación de ambos, el lenguaje y los algoritmos. Estas propuestas se implementaron en un software denominado LOGMIP que tiene la particularidad de ser el primer programa de resolución de problemas discretos/continuos no lineales basados en disyunciones, proposiciones lógicas y variables binarias.

7.2. Resumen de la tesis

7.2.1. Modelo discreto/continuo basado en disyunciones y variables binarias

En el capítulo 2 se presentó la base sobre la que se soporta nuestra tesis. Allí se propone un modelo híbrido (PH), basado en disyunciones, proposiciones lógicas y variables binarias (0-1) para la representación de las decisiones discretas en problemas no lineales. La representación es general por varios motivos. En primer lugar porque la representación disyuntiva generalizada (PDG) y la mixto-entera no lineal (MINLP) están contenidas dentro de la propuesta (PH). Por lo tanto, ésta última incluye a las representaciones anteriores. En segundo lugar, para mostrar la generalidad de estas representaciones se han desarrollado transformaciones de restricciones lógicas de otras áreas, como por ejemplo la Programación con Restricciones Lógicas (“Contraint Logic

Programming"- CLP), en la forma disyuntiva/híbrida presentada. También se mostraron transformaciones para disyunciones embebidas como las que se pueden encontrar en problemas de ingeniería de procesos, para establecer decisiones del tipo jerárquicas. La importancia de estas transformaciones son dos: por un lado, porque podemos traducir otros modelos en la representación híbrida propuesta, y resolverlos con métodos generales en lugar de algoritmos particulares; por otro lado se establece un vínculo entre la programación disyuntiva/híbrida y las otras propuestas basadas en lógica.

La pregunta que resta contestar es si es necesario contar con una representación híbrida ya que estos modelos también pueden ser expresados como MINLP ó como completamente disyuntivos. Las ideas detrás de los modelos híbridos son la naturalidad y conveniencia a nivel de modelado y de resolución de un problema. No siempre una decisión discreta es posible expresarla de manera fácil como disyunción ó como mixta-entera. La conveniencia está relacionada con dos aspectos, por un lado la habilidad de quien desarrolla el modelo y los datos que posee, y por otro, el tamaño del modelo y los métodos de resolución disponibles. Una mejor caracterización de cuando conviene una u otra representación se analiza en el capítulo 3 de la tesis. Por otra parte un modelo híbrido da mayor flexibilidad a la hora de modelar el problema, fundamentalmente porque completa a las otras representaciones.

7.2.2.Caracterización de las disyunciones y sus relajaciones

El capítulo 3 es clave para el modelado de los problemas híbridos, porque allí se trató de responder cuando es conveniente expresar una decisión como disyunción ó como una restricción mixto-entera.

El análisis de cuando conviene una u otra representación se hizo sobre la base de la caracterización de las disyunciones y las distintas relajaciones posibles de un conjunto disyuntivo: "Big-M", subrogada de Beaumont y cáscara convexa. Se presentaron las formulaciones de las distintas relajaciones y sus propiedades. La formulación "Big-M" y la subrogada de Beaumont están relacionadas con una formulación mixto-entera de un conjunto disyuntivo, la cáscara convexa con una formulación a través de disyunciones. La medida a tener en cuenta en este caso, es la diferencia entre la solución relajada y la solución entera ("integrality gap") de un problema. El otro aspecto importante es que las

relajaciones aumentan, en distinto grado, el tamaño de un problema, incrementando el número de variables y restricciones del mismo. Esto, en algunos casos, puede hacer que el problema sea insoluble.

Se compararon fundamentalmente la relajación "Big-M" y la cáscara convexa. Las conclusiones sobre la relajación "Big-M" son válidas para la subrogada de Beaumont.

Las disyunciones se caracterizaron de acuerdo con la unión de las regiones factibles de los términos que las componen en *propias e impropias.*

Las disyunciones *impropias* son aquellas en donde las regiones factibles de sus términos están incluidas en el término que tiene la región factible mayor. En este caso, se mostró analíticamente y a través de la solución de un ejemplo que ambas relajaciones son iguales. En realidad la disyunción *impropia* puede ser reemplazada por el término que posee la región factible mayor.

En cuanto a la disyunción *propia* puede ser de dos tipos. Una donde la unión de las regiones factibles de los términos tiene alguna intersección, y el otro caso, en donde la unión de los distintos regiones factibles de los términos es vacía. En el primer caso se demostró que no es posible sacar una conclusión general, y se propuso un algoritmo de preprocesamiento para determinar cuando es conveniente una relajación "Big-M" ó la cáscara convexa. Para el segundo tipo se demostró por medio de la solución de ejemplos que la cáscara convexa presenta un mejor comportamiento. En caso de existir dudas, ya que para disyunciones con gran cantidad de términos o muy complejas, no es posible conocer a priori a que tipo corresponden, se puede aplicar el algoritmo de preprocesamiento propuesto.

Como regla general, para considerar la aplicación de las relajaciones "Big-M" o la subrogada de Beaumont es importante tener buenas cotas ó buenas estimaciones de las cotas de las variables que componen el conjunto disyuntivo.

También se analizó el caso en donde las restricciones que componen los términos de las disyunciones son ecuaciones de igualdad. Estas restricciones son muy comunes en ingeniería química para satisfacer los balances de materia y energía de los procesos, de allí la importancia de su análisis. El análisis se hizo sobre la base de un ejemplo de síntesis de procesos modelado con diferentes formulaciones. La cáscara convexa tiene mejor comportamiento en este caso que la formulación "Big-M". Una explicación posible

es que la cáscara convexa maneja mejor las no convexidades que la formulación "Big-M".

7.2.3. Algoritmos de solución: formulación, propiedades e implementación

En el capítulo 4 se trataron los algoritmos que existen para resolver problemas discretos/continuos no lineales y su implementación en LOGMIP.

Habiendo presentado en el capítulo 1 de introducción los métodos para problemas MINLP, aquí se desarrollaron aquellos que están basados en lógica. Además, se propusieron dos extensiones del algoritmo de AE (Aproximación Exterior) basado en lógica, para ser aplicado a problemas híbridos. Las diferencias se encuentran en la formulación inicial y de los subproblemas NLP y MILP, y en el modo de inicializar los problemas. La extensión del algoritmo para problemas híbridos permite mayor flexibilidad a la hora de asignar valores iniciales a las variables.

También se presentan los algoritmos de transformación de un problema disyuntivo o híbrido a una formulación MINLP, y el de Ramificación y Acotamiento basado en lógica propuestos por Lee y Grossmann (1999b). La mayor contribución de esta tesis para este caso no está en los algoritmos, sino las conclusiones del capítulo 3 en cuanto a la conveniencia de aplicar las distintas relajaciones para transformar un problema disyuntivo/híbrido en uno MINLP.

En este capítulo se destacan la implementación de la primera versión de LOGMIP, un software desarrollado en lenguaje de programación C que permitió no solo la formulación, sino también la resolución de problemas mixto-enteros, disyuntivos e híbridos. Para la descripción de las disyunciones se empleó una adaptación del lenguaje para problemas matemáticos GAMS.

Se resolvieron varios problemas relacionados con la ingeniería de procesos. Dos de síntesis de procesos, uno pequeño de 8 unidades de procesos, que fue modelado y resuelto como MINLP, disyuntivo e híbrido, resultando para este caso la formulación híbrida más conveniente.

El otro problema de síntesis que se resolvió fue el del proceso HDA. El problema se modeló tanto con formulación disyuntiva como MINLP. La formulación disyuntiva presentó ventajas no sólo en tiempo de ejecución, sino también en el óptimo encontrado.

Otro de los problemas resueltos con LOGMIP es el diseño de una planta batch multiproducto, que permite la selecciones de unidades en paralelo en fase y fuera de fase y tanques de almacenamiento intermedio. Este problema se resolvió con formulación MINLP con varios métodos y como híbrido por el método de AE extendido, presentando esta última formulación claras ventajas frente a la formulación MINLP.

Se resolvió también un problema de determinación de parámetros en una espectroscopia infrarroja. Se modeló como MINLP y como disyuntivo. El tiempo de ejecución del algoritmo de AE basado en lógica fue mucho más rápido que el presentado por los algoritmos MINLP.

Por último, a los efectos de demostrar las posibilidades de modelado y resolución de un problema híbrido, se resolvió con LOGMIP un problema cuyo modelo original presentaba disyunciones embebidas. El problema corresponde a uno de síntesis de procesos con ecuaciones de costo discontinuas. Por medio de las transformaciones propuestas en el capítulo 2 de esta tesis se transformó el problema original en uno híbrido, en donde se dejaron como disyunciones las decisiones discretas correspondientes a la estructura del proceso, mientras que las ecuaciones de costo discontinuas se relajaron por medio de la cáscara convexa. El problema se resolvió por medio del método de AE extendido a problemas híbridos.

La resolución de estos problemas permitió mostrar las ventajas de contar con una herramienta que permita la formulación y resolución de un problema por varios modelos y métodos respectivamente.

7.2.4. Lenguaje para la especificación de disyunciones y restricciones lógicas

En el capítulo 5 se propuso un lenguaje para la especificación de disyunciones, de las proposiciones y restricciones lógicas. La propuesta se hizo con el sentido de completar o extender los lenguajes matemáticos existentes de modo que puedan introducirse expresiones lógicas en la descripción de un problema.

Para la especificación de las disyunciones se propusieron sentencias IF..THEN..ELSE..ENDIF simples y anidadas para poder especificar disyunciones de distinto tipo: con dos términos, varios términos, embebidas, etc. Se mostró el modo como se emplean estas sentencias a través de ejemplos de distinto tipo. Si bien podrían existir

variantes y sentencias de otro tipo que pueden ser equivalentes, se escogió este tipo de sentencias por varios motivos: es ampliamente conocida, es muy expresiva, no es ambigua y en general cubre todas las posibilidades de expresión.

Para las proposiciones y otras expresiones lógicas se han incluido los operadores lógicos: $\wedge$("and"), $\vee$("or"), $\underline{\vee}$("or exclusivo"), ~("not",!,¬), ->(implicación), <-> (equivalencia). Dado que el uso de estos operadores puede no resultar fácil ni claro para muchos usuarios, es que se han propuestos sentencias especiales más expresivas y fáciles de usar que permiten cubrir la mayoría de los casos más comunes que se pueden presentar con los operadores lógicos. Estas sentencias especiales son: **atmost(), atleast(), exactly(), adjacent() y alldifferent()** y las variantes **atmost k (), atleast k (), exactly k ()**.

Estas expresiones, junto un lenguaje de expresión de programas matemáticos, facilitan al descripción de un problema disyuntivo ó híbrido.

7.2.5. Implementación: LOGMIP segunda versión

En el capítulo 6 se propusieron ideas para la implementación del lenguaje y los algoritmos descritos en LOGMIP en su segunda versión. En la primera versión no se había propuesto un lenguaje, ni tampoco se disponían las técnicas y algoritmos que surgieron más adelante. Por otra parte la propuesta de implementación por medio de un analizador sintáctico y semántico del lenguaje en una etapa de precompilación de un lenguaje matemático, permite trabajar de manera independiente la parte lógica de la matemática, y juntarlos nuevamente en la etapa de solución del problema, para poder allí administrar la generación de las matrices de los distintos subproblemas. Se propuso un identificador para las disyunciones de modo que esto permita generar distintos modelos con una misma descripción de un problema. Se mostraron ejemplos que permiten visualizar el modo en que se lleva a cabo esta implementación. También se presentaron distintos diagramas de flujos con detalles acerca de cómo se llevan a cabo las distintas implementaciones a nivel de LOGMIP y de los métodos de resolución de los problemas disyuntivos e híbridos. Se trabajó no solamente en los algoritmos basados en lógica, sino también en aquellos algoritmos mixto-enteros. Se generaron códigos de los métodos PCE (Planos de Corte Extendido), DGB (Descomposición Generalizada de Benders) y se realizaron mejoras del método de AE (Aproximación Exterior) implementado en el

código DICOPT^{++}. Se generaron nuevos programas en lenguaje C para implementar el método de AE basados en lógica para problemas disyuntivos e híbridos. Se propuso también generar de manera automática el problema MINLP a partir de un modelo disyuntivo ó híbrido por medio de la relajación por la cáscara convexa. Se realiza esta implementación por medio de una traducción a nivel de un archivo de entrada LOGMIP en uno MINLP. El sentido de estos desarrollos y su implementación es el proveer mayor flexibilidad a la hora de generar los modelos y de escoger un método de solución para los mismos. De esta manera LOGMIP en su segunda versión constituye una herramienta sin precedentes para el desarrollo e implementación de problemas discretos/continuos no lineales. Aunque es claro que LOGMIP no se ha completado en todos sus detalles, existe un prototipo avanzado donde se han implementado las ideas de esta tesis.

7.3. Contribuciones del trabajo de investigación

Esta tesis ha producido las siguientes contribuciones y resultados:

1- Se desarrolló una representación híbrida para la formulación de problemas discretos/continuos no lineales para proveer una mayor flexibilidad y un nuevo marco de referencia a la hora de modelar problemas de este tipo.

2- Se desarrollaron transformaciones de restricciones lógicas y disyunciones embebidas en la forma de disyunciones y proposiciones lógicas como las del modelo híbrido propuesto a efectos de mostrar la validez de ésta representación.

3- Se estudiaron las disyunciones de acuerdo con las propiedades que presentan las regiones factibles de sus términos (propias e impropias). Con este estudio se dieron guías para modelar las decisiones discretas, si se hace en forma mixto-entera o en forma de disyunciones.

4- Se estudiaron las disyunciones con restricciones de igualdad en sus términos para poder entender las implicaciones de reformularlas como cáscara convexa o restricción “Big-M”.

5- Se propuso un método de preprocesamiento que permite determinar que relajación de una disyunción es mas ajustada, si la "Big-M" ó la cáscara convexa.

6- Se desarrollaron modificaciones en la definición de los subproblemas NLP y el maestro MILP del algoritmo de Aproximación Exterior basado en lógica para problemas híbridos.

7- Se desarrolló la implementación de los algoritmos de Aproximación Exterior basado en lógica para problemas disyuntivos e híbridos en la primera versión de LOGMIP. Con esta implementación se resolvieron diversos ejemplos de ingeniería de procesos para mostrar la capacidad y eficiencia de los métodos propuestos.

8- Se propuso un lenguaje de modelado para la especificación de disyunciones, proposiciones lógicas y restricciones lógicas, con el objetivo de completar los lenguajes de programación matemática y poder introducir modelos basados en lógica en programas de aplicación general.

9- Se implementaron mejoras al código DICOPT^{++} para introducir nuevas inicializaciones, producir reportes diferentes y poder modelar decisiones discretas por medio de variables enteras, además de las binarias existentes.

10- Usando el código de DICOPT^{++} se realizaron implementaciones de los métodos de Descomposición Generalizada de Benders (DGB) y de Planos de Corte Extendido (PCE). Estos métodos se emplean para resolver problemas MINLP. Son importantes para dar mayor flexibilidad para elegir el método de solución de un problema.

11- Se desarrolló la implementación del lenguaje y de los nuevos algoritmos en la segunda versión del código LOGMIP. Esta implementación esta basada en las ideas de propuestas de completar un lenguaje de programación matemática, y en la de

incorporar una etapa de precompilación de la lógica, para independizar la parte matemática de la lógica y hacer más flexible el software.

7.4. Trabajos futuros

Existen diversos aspectos que aún necesitan ser estudiados e investigados para tener una mejor comprensión y aprovechamiento de la incorporación de la lógica en los modelos, algoritmos y las implementación de problemas discretos/continuos no lineales. También se requiere una implementación completa de LOGMIP para que incluya todos los aspectos que se han discutido en esta tesis. Los puntos específicos que se pueden investigar en el futuro incluyen los siguientes:

1- A nivel de los modelos aún no se tienen respuestas suficientes para los modelos en donde intervienen restricciones no convexas en los términos de las disyunciones. Es necesario estudiar los modelos basados en lógica con fuertes restricciones no convexas.

2- Se necesitan establecer vínculos entre los modelos y técnicas de otras áreas basados en la incorporación de lógica en sus modelos, como la Programación con Restricciones Lógicas y los conceptos y técnicas de la Programación Disyuntiva/ Híbrida. Un posible vínculo son los métodos de Ramificación y Acotamiento y las técnicas empleadas para acotar las búsquedas en el árbol. En la programación disyuntiva e híbrida la lógica proposicional podría tener un rol importante que aún no ha sido explotado. Del mismo modo no se ha experimentado a nivel del programa maestro MILP (en el que se emplea fundamentalmente el algoritmo de Ramificación y Acotamiento) el rol que la lógica proposicional podría tener por la incorporación de cortes del tipo lógico para acotar la búsqueda de la solución entera.

3- No se ha investigado aún el impacto que pueden tener los modelos basados en lógica en modelos industriales de gran escala en donde la eficiencia para alcanzar la solución es de suma importancia. Los indicios con los problemas resueltos son

alentadores, la incorporación de disyunciones y de proposiciones lógicas permiten obtener soluciones más eficientes, pero esta es un área que necesita exploración.

4- Otra área de investigación es el desarrollo de restricciones subrogadas que puedan reemplazar a los conjuntos disyuntivos, y produzcan una relajación ajustada, que permita generar modelos más compactos a partir de los disyuntivos.

5- También se debe avanzar en la implementación de estas técnicas, a efectos de contar con herramientas robustas que permitan experimentar, investigar y desarrollar nuevos conocimientos en el área de los modelos discretos/continuos. La incorporación de variables Booleanas en los lenguajes ayudaría a la claridad de la interpretación de los modelos con lógica.

R

Referencias

Balas, E.
"Disjunctive programming". Discrete Optimizations II, Annals of Discrete Mathematics, 5, North Holland, Amsterdam, **1979**.

Balas, E.
"*Disjunctive Programming and a hierarchy of relaxations for discrete optimization problems"*, SIAM J. Alg. Dis. Meth., 6 (3), 466-485, **1985**.

Balas, E.
"*Disjunctive Programming: Properties of the convex hull of feasible points"*, Discrete Applied Mathematics 89 (1), 3-44, **1998**.

Barton P.I. y Pantelides C.C.
"Modeling of Combined Discrete/Continuous Processes". AICHE Journal, 40(6), 966-979, **1994.**

Beaumont Nicholas.
"An algorithm for disjunctive programs". European Journal of Operational Research, 48, 362-371, **1990**.

Bjorkqvist, J. y Westerlund T.
"Automated Reformulation of Disjunctive Constraints in MINLP Optimization". Comp. Chem. Eng., 23,Supp., S11-S14, **1999**.

Bockmayr, A. y Kasper T.
"Branch and Infer: A Unifying Framework for Integer and Finite Domain Constraint Programming". INFORMS Journal of Computing, 10 (3), **1998**.

Borchers, B. y Mitchell J. E.
"An Improved Branch and Bound Algorithm for Mixed Integer Nonlinear Programming", Computers and Operations Research, 21, 359-367, **1994**.

Brink A. y Westerlund T.
"The joint problem of model structure determination and parameter estimation on quntitative IR-spectroscopy". Chemomet. and Intell. Lab. Sys., 29 (1), 29-36, **1995**.

Brooke A., Kendrick D. y Meeraus A.
"GAMS a User's Guide". Gams Development Corporation, **1996**.

Clocksin W. F. y Mellish C.S.
"Programming in Prolog". Springer-Verlag, New York, **1981.**

Dakin, R. J.
"A Tree Search Algorithm for Mixed-Integer Programming Problems", *Computer Journal*, 8, 250-255, **1965**.

Darby-Dowman K., Little J., Mitra G. y Zaffalon M.
"Constraint Logic Programming and Integer Programming Approaches and Their Collaboration in Solving an Assignment Scheduling Problem". Constraints, 1, 245-264, **1997**.

Duran M. y Grossmann I.E.
"*An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs"*. Mathematical Programming, 36, 307-339, **1986**.

Fletcher R. y Leyffer S.
"Solving Mixed Integer Nonlinear Programs by Outer Approximation", Math Programming 66, 327 , **1994**.

Flippo O.E. y Rinnoy Kan A.H.G.
"Decomposition in General Mathematical Programming", Mathematical Programming, 60, 361-382, **1993**.

Fourer R., Gay D. y Kernighan D.
"*AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press / Brooks/Cole Publishing Company, **1993**.

Geoffrion A. M.
"Generalized Benders Decomposition". Journal of Optimization Theory and Application, 10 (4), 237-260, **1972**.

Gil, J. J.
"LOG2MAT: un programa de transformación de proposiciones lógicas en desigualdades matemáticas". Informe interno. INGAR, Instituto de Desarrollo y Diseño, Santa Fe, Argentina, **2000.**

Grossmann I.E.
"Mixed-Integer Optimization Techniques for Algorithmic Process Synthesis", Advances in Chemical Engineering, Vol. 23, Process Synthesis, pp.171-246, **1996a**.

Grossmann I.E. y Daichendt M.M..
"New Trends in Optimization-based Approaches for Process Synthesis", Computers and Chemical Engineering, 20, 665-683, **1996**.

Grossmann I.E., J. Quesada, R. Raman y V. Voudouris.
"Mixed Integer Optimization Techniques for the Design and Scheduling of Batch Processes" Batch Processing Systems Engineering (Eds. G.V. Reklaitis, A.K. Sunol, D.W.T. Rippin, O. Hortacsu), 451-494, Springer-Verlag, Berlin, **1996**.

Grossmann I.E., J.A. Caballero y H. Yeomans.
"Advances in Mathematical Programming for Automated Design, Integration and Operation of Chemical Processes". Proceedngs of the International Conference on Process Integration (PI'99), Copenhagen, Denmark, **1999**.

Grossmann, I.E.
"Mixed-Integer Non-Linear Programming Techniques for Process System Engineering". Trabajo presentado en la Academia Mexicana de Ingenería, **1999**.

Gupta O.K. y Ravindran V.
"Branch and Bound Experiments in Convex Nonlinear Integer Programming". Management Science, 31 (12), 1533-1546, **1985**.

Guthrie, K. M.
"Process plant estimating, evaluation and control". Craftsman Book, Co., Los Angeles, **1974.**

Hajian M. T., El-Sakkout H., Wallace M., Lever J. M. y Richards E.B.
"Towards a Closer Integration of Finite Domain Propagation and Simplex-Based Algorithms". IC-Parc Publications (UK), **1995**.

Hajian M. T., Rodosek R. y Richards E.B.
"Introduction of a New Class of Variables to Discrete and Integer Programming Problems". IC-Parc Publications (UK), **1996**.

Hooker J.
"Logic-Based Methods for Optimization". Wiley, New York, **2000**.

Hooker J. y Kim H.
"A Declarative Modeling Framework that Integrates Solution Methods". Enviado para publicación. **1998.**

ILOG Solver 4.3
User's Manuals. ILOG, 1998.

Kelley J.
"The Cutting Plane Method for Solving Convex Programs". Journal of SIAM, Vol. VIII (4), 703-712, **1960.**

Lee S. y Grossmann I.E.
"Generalized Disjunctive Programming: Nonlinear Convex Hull Relaxation and Algorithms".Enviado para publicación, **1999a**.

Lee S. y Grossmann I.E.
"New algorithm for Nonlinear Generalized Disjunctive Programming". Enviado para publicación, **1999b**.

Leyffer S.
"Integrating SQP and Branch and Bound for Mixed Integer Noninear Programming". Enviado para publicación, **1998**.

Marquardt W., von Wedel L. y Bayer B.
"Perspectives on lifecycle process modeling". Trabajo I17 presentado en FOCAPD '99. Breckenridge (Colorado). July **1999**.

Nabar S. y Schrage, L.
"Modeling and Solving Nonlinear Integer Programming Problems". Presentado en el Annual AIChE Meeting, Chicago, **1991**.

Nemhauser G. L. y Wolsey L. A.
"Integer and Combinatorial Optimization". Wiley, New York, **1988**.

Oh M. y Pantelides C.C.
"A Modelling and Simulation Language for Combined Lumped and Distributed Parameter Systems". Comp. Chem. Eng., 20 (6-7), 611-633, **1996.**

Pantelides, C.C.
"SpeedUp recent advances in process simulation". Comp. Chem. Eng., 12 (7), 745-755, **1988**.

Pinto J. y Grossmann I.E.

"Assignment and Sequencing Models for the Scheduling of Chemical Processes", Annals of Operations Research 81 433-466, **1998**.

Quesada, I. and I.E. Grossmann
"An LP/NLP Based Branch and Bound Algorithm for Convex MINLP Optimization Problems," Computers and Chemical Engineering **16**, 937-947, **1992**.

Raman R.y Grossmann I.E.
"Relation Between MILP Modelling and Logical Inference for Chemical Process Synthesis". Comp. Chem. Eng., 15 (2), 73-84, **1991**.

Raman R.y Grossmann I.E.
"Symbolic Integration for Logic in Mixed-Integer Linear Programming Techniques for Process Synthesis". Comp. Chem. Eng., 17 (9), 909-927, **1993**.

Raman R. y Grossmann I.E.
"Modeling and Computational Techniques for Logic Based Integer Programming". Comp. Chem. Eng., 18 (7), 563-578, **1994**.

Ravemark D.
"Optimization Model for Design and Operation of Chemical Batch Processes". Tesis de Doctorado, Swiss Federal Institute of Technology, Zurich, Suiza, **1995.**

Rico-Ramirez V.
"Representation, Analysis and Solution of Conditional Models in an Equation-Based Environment". Tesis de doctorado. Depto. de Ingeniería Química, Carnegie Mellon University, Pittsburgh, PA, **1998**.

Ryoo H. S., y Sahinidis N.V.
"Global Optimization of Nonconvex NLPs and MINLPs with Applications in Process Design". Comp. Chem. Eng., 19 (5), 551-566, **1995**.

Sahinidis, N.V. y Grossmann I.E.
"Convergence Properties of Generalized Benders Decomposition". Comp. Chem. Eng., 15, 481, **1991**.

Shah, N.
"*Single and Multisite Planning and Scheduling: Current Status and Future Challenges,"* FOCAPO Proceedings , **1998**.

Stubbs R. y Mehrotra S.
"Branch and Cut Methods for Mixed 0-1 Convex Nonlinear Programming". Trabajo presentado en Fifth SIAM Conference on Optimization, Victoria, British Columbia, Canada, **1996**.

Tourn, Michel.
"PROLOG Program for converting propositions into linear inequalities". Chemical Engineering Department , Carnegie Mellon University, Pittsburgh, PA, **1995**.

Tsang, E.P.
"Formulations at Constraint Satisfaction". Academic Press, **1993**.

Turkay M. y Grossmann I.E.

"Logic-Based Algorithms for the Optimal Synthesis of Process Networks". Comp. Chem. Eng., 20 (8), 959-978, **1996a**.

Turkay M. y Grossmann I.E.
"Disjunctive Programming Techniques for the Optimization of Process Systems with Discontinuous Investment Costs-Multiple Size Regions". I&EC Research, 35 (8), 2611-2623, **1996b**.

Turkay M. y Grossmann I.E.
"Structural flowsheet optimization with complex investment cost functions". Comp. Chem. Eng., 22 (4-5), 673-686, **1998**.

Van der Heever S. y Grossmann I.E.
"Disjunctive multiperiod Optimization Methods for Design and Planning at Process Systems". Comp. Chem. Eng., 23, 1075-1095, **1999**.

Van Hentenryck, P.
"Constraint Satisfaction in Logic Programming'. MIT Press, **1989**.

Van Hentenryck P.
"*The OPL Optimization Programming Language"*. MIT Press, **1999**.

Van Hentenryck P. y Saraswat V.
"*Strategic Directions in Constraint Programming"*. ACM Computin Surveys, 28, 4, 701-726, **1996**.

Vecchietti A. y Grossmann I.E.
"*LOGMIP: A Disjunctive 0-1 Nonlinear Optimizer for Process System Models"*. Comp. Chem. Eng., 23,. 555-565, **1999**.

Viswanathan J.V. y Grossmann I.E.
"A Combined Penalty Function and Outer-Approximation method for MINLP Optimization". Comp. Chem. Eng., 14 (7), 769-778, **1990**.

Wallace M., Novello S. y Schimpf J.
"Eclipse: A platform for Constraint Logic Programming". Technical Report, IC-Parc, Imperial College, London, **1997.**

Westerlund T y Pettersson R.
"An Extended Cutting Plane Method for Solving Convex MINLP Problems". Comp. Chem. Eng., 19, S131-36, **1995**.

Westerlund T. y Lundqvist K.
"ALPHAECP 4.0. An Interactive MINLP-Solver Based on the Extended Cutting Plane Method". Report 00-170-A, Process Design Laboratory at Abo Akademi University, Finlandia, **2000**.

A

Apéndice A

A.1. Archivo de entrada GAMS del ejemplo 4.4.1. Modelo MINLP.

```
$OFFSYMXREF
$OFFSYMLIST
*  SELECT OPTIMAL PROCESS FROM WITHIN GIVEN SUPERSTRUCTURE.
*

SETS   I  PROCESS STREAMS                    / 2*25 /
       J  PROCESS UNITS                      / 1*8 /

 PARAMETERS CF(J)     FIXED COST INVESTMENT COEFF FOR PROCESS UNITS  /
                        1 = 5 , 2 = 8 , 3 = 6 , 4 = 10
                        5 = 6 , 6 = 7 , 7 = 4 , 8 = 5          /

            CV(I)     VARIABLE COST COEFF FOR PROCESS UNITS - STREAMS/
                        3 = -10  ,  5 = -15  ,  9 = -40
                       19 =  25  , 21 =  35  , 25 = -35
                       17 =  80  , 14 =  15  , 10 =  15
                        2 =  1   ,  4 =  1   , 18 = -65
                       20 = -60  , 22 = -80                    /

 Scalar CONSTANT OBJECTIVE FUNCTION CONSTANT TERM ;

 CONSTANT= 122.0 ;

 VARIABLES  Y(J)     BINARY VARIABLE DENOTING EXISTENCE-NONEXISTENCE
            X(I)     FLOWRATES OF PROCESS STREAMS

            PROF     OVERALL PROCESS PROFIT

 BINARY VARIABLES    Y(J)  ;
 POSITIVE VARIABLES  X(I)  ;

 EQUATIONS  INOUT1       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 1
            INOUT2       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 2
            INOUT3       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 3
            INOUT4       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 4
            INOUT5       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 5
            INOUT6       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 6
            INOUT7       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 7
            INOUT8       INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 8
            MASSBAL1     MASS BALANCE #1
            MASSBAL2     MASS BALANCE #2
            MASSBAL3     MASS BALANCE #3
            MASSBAL4     MASS BALANCE #4
            MASSBAL5     MASS BALANCE #5
            MASSBAL6     MASS BALANCE #6
            MASSBAL7     MASS BALANCE #7
            SPECS1       DESIGN SPECSIFICATION 1
            SPECS2       DESIGN SPECSIFICATION 2
            SPECS3       DESIGN SPECSIFICATION 3
            SPECS4       DESIGN SPECSIFICATION 4
            LOGICAL1    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 1 EXISTS
            LOGICAL2    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 2 EXISTS
            LOGICAL3    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 3 EXISTS
            LOGICAL4    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 4 EXISTS
            LOGICAL5    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 5 EXISTS
            LOGICAL6    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 6 EXISTS
            LOGICAL7    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 7 EXISTS
```

```
          LOGICAL8    CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 8 EXISTS
          PUREINT1     PURELY INTEGER CONSTRAINT #1
          PUREINT2     PURELY INTEGER CONSTRAINT #2
          PUREINT3     PURELY INTEGER CONSTRAINT #3
          PUREINT4     PURELY INTEGER CONSTRAINT #4
          OBJ          OBJECTIVE FUNCTION DEFINITION ;

INOUT1..       EXP(X('3'))      -1.  =E=  X('2')                ;
INOUT2..       EXP(X('5')/1.2)  -1.  =E=  X('4')                ;
INOUT3..       1.5 * X('9') + X('10')  =E=  X('8')              ;
INOUT4..       1.25 * ( X('12')+X('14') )  =E=  X('13')         ;
INOUT5..       X('15')               =E=  2. * X('16')          ;
INOUT6..       EXP(X('20')/1.5) -1.  =E=  X('19')               ;
INOUT7..       EXP(X('22'))     -1.  =E=  X('21')               ;
INOUT8..       EXP(X('18'))     -1.  =E=  X('10') + X('17')     ;

MASSBAL1..       X('13')         =E=  X('19') + X('21')         ;
MASSBAL2..       X('17')         =E=  X('9') + X('16') + X('25') ;
MASSBAL3..       X('11')         =E=  X('12') + X('15')         ;
MASSBAL4..       X('3') + X('5') =E=  X('6') + X('11')          ;
MASSBAL5..       X('6')          =E=  X('7') + X('8')           ;
MASSBAL6..       X('23')         =E=  X('20') + X('22')         ;
MASSBAL7..       X('23')         =E=  X('14') + X('24')         ;

SPECS1..         X('10')  =L=  0.8 * X('17')                    ;
SPECS2..         X('10')  =G=  0.4 * X('17')                    ;
SPECS3..         X('12')  =L=  5.0 * X('14')                    ;
SPECS4..         X('12')  =G=  2.0 * X('14')                    ;

LOGICAL1..       X('2')   =L=  10. * Y('1')                     ;
LOGICAL2..       X('4')   =L=  10. * Y('2')                     ;
LOGICAL3..       X('9')   =L=  10. * Y('3')                     ;
LOGICAL4..       X('12') + X('14')  =L=  10. * Y('4')           ;
LOGICAL5..       X('15')  =L=  10. * Y('5')                     ;
LOGICAL6..       X('19')  =L=  10. * Y('6')                     ;
LOGICAL7..       X('21')  =L=  10. * Y('7')                     ;
LOGICAL8..       X('10') + X('17')  =L=  10. * Y('8')           ;

PUREINT1..       Y('1') + Y('2')  =E=  1.                       ;
PUREINT2..       Y('4') + Y('5')  =L=  1.                       ;
PUREINT3..       Y('6') + Y('7') - Y('4')  =E= 0.               ;
PUREINT4..       Y('3') - Y('8')  =L=  0.                       ;

OBJ.. PROF  =E=  SUM(J , Y(J)*CF(J)) +  SUM(I , X(I)*CV(I)) + CONSTANT;

X.L('2') =  2.    ;
X.L('3') =  1.5   ;
X.L('6') =  0.75  ;
X.L('7') =  0.5   ;
X.L('8') =  0.5   ;
X.L('9') =  0.75  ;
X.L('11') = 1.5   ;
X.L('12') = 1.34  ;
X.L('13') = 2.    ;
X.L('14') = 2.5   ;
X.L('17') = 2.    ;
X.L('18') = 0.75  ;
X.L('19') = 2.    ;
X.L('20') = 1.5   ;
```

```
 X.L('23') = 1.7   ;
 X.L('24') = 1.5   ;
 X.L('25') = 0.5   ;

*  BOUNDS SECTION
 X.LO(I) =   0.0   ;
 X.UP('3') = 2.    ;
 X.UP('5') = 2.    ;
 X.UP('9') = 2.    ;
 X.UP('10') = 1.   ;
 X.UP('14') = 1.   ;
 X.UP('17') = 2.   ;
 X.UP('19') = 2.   ;
 X.UP('21') = 2.   ;
 X.UP('25') = 3.   ;

 MODEL EXAMPLE3  / ALL / ;
 EXAMPLE3.optfile=1 ;
 OPTION OPTCR = 0.0, iterlim =20000, reslim = 2000 ;

 SOLVE EXAMPLE3 USING MINLP MINIMIZING PROF   ;
```

A.2. Archivo de entrada GAMS del ejemplo 4.4.1. Modelo PDG.

```
$OFFSYMXREF
$OFFSYMLIST
*  SELECT OPTIMAL PROCESS FROM WITHIN GIVEN SUPERSTRUCTURE.
*

SETS   I      PROCESS STREAMS                / 1*25 /
       J      PROCESS UNITS                  / 1*8 /


PARAMETERS   CV(I)     VARIABLE COST COEFF FOR PROCESS UNITS - STREAMS/
                            3 = -10  ,  5 = -15  ,  9 = -40
                           19 =  25  , 21 =  35  , 25 = -35
                           17 =  80  , 14 =  15  , 10 =  15
                            2 =  1   ,  4 =  1   , 18 = -65
                           20 = -60  , 22 = -80                    /;

PARAMETERS YMT(J)     BINARY VARIABLES FOR DISJUNCTIONS                 ;

 VARIABLES  PROF     PROFIT ;

 BINARY VARIABLES    Y(J)  ;
 POSITIVE VARIABLES  X(I) , CF(J) ;

 EQUATIONS

* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* -------------------------------------------------------
            MASSBAL1     MASS BALANCE #1
            MASSBAL2     MASS BALANCE #2
            MASSBAL3     MASS BALANCE #3
            MASSBAL4     MASS BALANCE #4
            MASSBAL5     MASS BALANCE #5
            MASSBAL6     MASS BALANCE #6
            MASSBAL7     MASS BALANCE #7
            MASSBAL8     MASS BALANCE #8
            SPECS1       DESIGN SPECIFICATION 1
            SPECS2       DESIGN SPECIFICATION 2
            SPECS3       DESIGN SPECIFICATION 3
            SPECS4       DESIGN SPECIFICATION 4

* EQUATIONS FOR THE MASTER PROBLEMS ONLY:
* ---------------------------------------
           PUREINT1     PURELY INTEGER CONSTRAINT #1
            PUREINT2     PURELY INTEGER CONSTRAINT #2
            PUREINT3     PURELY INTEGER CONSTRAINT #3
            PUREINT4     PURELY INTEGER CONSTRAINT #4
            PUREINT5     PURELY INTEGER CONSTRAINT #5
            PUREINT6     PURELY INTEGER CONSTRAINT #6
            PUREINT7     PURELY INTEGER CONSTRAINT #7
            PUREINT8     PURELY INTEGER CONSTRAINT #8
            PUREINT9     PURELY INTEGER CONSTRAINT #9
            PUREINT10    PURELY INTEGER CONSTRAINT #10
            PUREINT11    PURELY INTEGER CONSTRAINT #11
            PUREINT12    PURELY INTEGER CONSTRAINT #12
            PUREINT13    PURELY INTEGER CONSTRAINT #13
            PUREINT14    PURELY INTEGER CONSTRAINT #14
            PUREINT15    PURELY INTEGER CONSTRAINT #15
```

```
* EQUATIONS FOR THE NLP SUBPROBLEMS ONLY:
* -------------------------------------
           INOUT11      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 1
           INOUT12
           INOUT13
           INOUT14
           INOUT21      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 2
           INOUT22
           INOUT23
           INOUT24
           INOUT31      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 3
           INOUT32
           INOUT34
           INOUT41      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 4
           INOUT42
           INOUT43
           INOUT44
           INOUT45
           INOUT51      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 5
           INOUT52
           INOUT53
           INOUT54
           INOUT61      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 6
           INOUT62
           INOUT63
           INOUT64
           INOUT71      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 7
           INOUT72
           INOUT73
           INOUT74
           INOUT81      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 8
           INOUT82
           INOUT83
           INOUT84
           INOUT85
           INOUT86
           OBJETIVO          OBJECTIVE FUNCTION DEFINITION ;

* BOUNDS SECTION:
* ---------------
 X.UP('3')  =  2.0 ;
 X.UP('5')  =  2.0 ;
 X.UP('9')  =  2.0 ;
 X.UP('10') =  1.0 ;
 X.UP('14') =  1.0 ;
 X.UP('17') =  2.0 ;
 X.UP('19') =  2.0 ;
 X.UP('21') =  2.0 ;
 X.UP('25') =  3.0 ;

 OPTIONS LIMCOL = 0   ;
 OPTION LIMROW = 0    ;
 OPTION OPTCR = 0     ;
 OPTION MINLP = LOGMIP;

* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* ------------------------------------------------------
 MASSBAL1    ..   X('13')         =E=  X('19') + X('21')          ;
```

```
MASSBAL2    ..   X('17')         =E=  X('9') + X('16') + X('25') ;
MASSBAL3    ..   X('11')         =E=  X('12') + X('15')          ;
MASSBAL4    ..   X('3') + X('5') =E=  X('6') + X('11')           ;
MASSBAL5    ..   X('6')          =E=  X('7') + X('8')            ;
MASSBAL6    ..   X('23')         =E=  X('20') + X('22')          ;
MASSBAL7    ..   X('23')         =E=  X('14') + X('24')          ;
MASSBAL8    ..   X('1')          =E=  X('2') + X('4')            ;

SPECS1      ..   X('10')  =L=  0.8 * X('17')                     ;
SPECS2      ..   X('10')  =G=  0.4 * X('17')                     ;
SPECS3      ..   X('12')  =L=  5.0 * X('14')                     ;
SPECS4      ..   X('12')  =G=  2.0 * X('14')                     ;


LOGICAL1    ..   X('2') + X('3')   =L=  10. * Y('1')          ;
LOGICAL2    ..   X('4') + X('5')   =L=  10. * Y('2')          ;
LOGICAL3    ..   X('9')   =L=  10. * Y('3')                   ;
LOGICAL4    ..   X('12') + X('14')  =L=  10. * Y('4')         ;
LOGICAL5    ..   X('15')  =L=  10. * Y('5')                   ;
LOGICAL6    ..   X('19')  =L=  10. * Y('6')                   ;
LOGICAL7    ..   X('21')  =L=  10. * Y('7')                   ;
LOGICAL8    ..   X('10') + X('17')  =L=  10. * Y('8')         ;

PUREINT1    ..   Y('1') + Y('2')  =L=  1.                     ;
PUREINT2    .. - Y('1') + Y('3') + Y('4') + Y('5')  =G=  0    ;
PUREINT3    .. - Y('2') + Y('3') + Y('4') + Y('5')  =G=  0    ;
PUREINT4    .. - Y('3') + Y('8')  =G=  0                      ;
PUREINT5    ..   Y('1') + Y('2') - Y('3')  =G= 0              ;
PUREINT6    ..   Y('1') + Y('2') - Y('4')  =G= 0              ;
PUREINT7    .. - Y('4') + Y('6') + Y('7')  =G= 0              ;
PUREINT8    ..   Y('4') + Y('5')  =L=  1                      ;
PUREINT9    ..   Y('1') + Y('2') - Y('5')  =G= 0              ;
PUREINT10   .. - Y('5') + Y('8')  =G=  0                      ;
PUREINT11   ..   Y('4') - Y('6')  =G=  0                      ;
PUREINT12   ..   Y('6') + Y('7')  =L=  1                      ;
PUREINT13   ..   Y('4') - Y('7')  =G=  0                      ;
PUREINT14   .. - Y('5') + Y('8')  =L=  1                      ;
PUREINT15   .. - Y('3') + Y('8')  =L=  1                      ;

* EQUATIONS FOR THE NLP SUBPROBLEMS ONLY:
* ---------------------------------------
INOUT11 $ (YMT('1') EQ 1)  ..   EXP(X('3')) -1. =E= X('2')           ;
INOUT12 $ (YMT('1') EQ 0)  ..   X('2') =E= 0                         ;
INOUT13 $ (YMT('1') EQ 0)  ..   X('3') =E= 0                         ;
INOUT14 $ (YMT('1') EQ 1)  ..   CF('1') =E= 5                        ;
INOUT21 $ (YMT('2') EQ 1)  ..   EXP(X('5')/1.2) -1. =E= X('4')       ;
INOUT22 $ (YMT('2') EQ 0)  ..   X('4') =E= 0                         ;
INOUT23 $ (YMT('2') EQ 0)  ..   X('5') =E= 0                         ;
INOUT24 $ (YMT('2') EQ 1)  ..   CF('2') =E= 8                        ;
INOUT31 $ (YMT('3') EQ 1)  ..   1.5 * X('9') + X('10') =E= X('8')    ;
INOUT32 $ (YMT('3') EQ 0)  ..   X('9') =E= 0                         ;
INOUT34 $ (YMT('3') EQ 1)  ..   CF('3') =E= 6                        ;
INOUT41 $ (YMT('4') EQ 1)  ..   1.25 * (X('12')+X('14')) =E= X('13') ;
INOUT42 $ (YMT('4') EQ 0)  ..   X('12') =E= 0                        ;
INOUT43 $ (YMT('4') EQ 0)  ..   X('13') =E= 0                        ;
INOUT44 $ (YMT('4') EQ 0)  ..   X('14') =E= 0                        ;
INOUT45 $ (YMT('4') EQ 1)  ..   CF('4') =E= 10                       ;
INOUT51 $ (YMT('5') EQ 1)  ..   X('15') =E= 2. * X('16')             ;
INOUT52 $ (YMT('5') EQ 0)  ..   X('15') =E= 0                        ;
```

```
 INOUT53 $ (YMT('5') EQ 0)  ..   X('16') =E= 0                          ;
 INOUT54 $ (YMT('5') EQ 1)  ..   CF('5') =E= 6                          ;
 INOUT61 $ (YMT('6') EQ 1)  ..   EXP(X('20')/1.5) -1. =E= X('19')       ;
 INOUT62 $ (YMT('6') EQ 0)  ..   X('19') =E= 0                          ;
 INOUT63 $ (YMT('6') EQ 0)  ..   X('20') =E= 0                          ;
 INOUT64 $ (YMT('6') EQ 1)  ..   CF('6') =E= 7                          ;
 INOUT71 $ (YMT('7') EQ 1)  ..   EXP(X('22')) -1. =E= X('21')           ;
 INOUT72 $ (YMT('7') EQ 0)  ..   X('21') =E= 0                          ;
 INOUT73 $ (YMT('7') EQ 0)  ..   X('22') =E= 0                          ;
 INOUT74 $ (YMT('7') EQ 1)  ..   CF('7') =E= 4                          ;
 INOUT81 $ (YMT('8') EQ 1)  ..   EXP(X('18')) -1. =E= X('10') + X('17');
 INOUT82 $ (YMT('8') EQ 0)  ..   X('10') =E= 0                          ;
 INOUT83 $ (YMT('8') EQ 0)  ..   X('17') =E= 0                          ;
 INOUT84 $ (YMT('8') EQ 0)  ..   X('18') =E= 0                          ;
 INOUT85 $ (YMT('8') EQ 0)  ..   X('25') =E= 0                          ;
 INOUT86 $ (YMT('8') EQ 1)  ..   CF('8') =E= 5                          ;
 OBJETIVO  .. PROF  =E= SUM(J,CF(J)) + SUM(I , X(I)*CV(I)) + 122 ;

* SUBPROBLEM EQUATIONS:
* ---------------------
 MODEL NLPX / INOUT11, INOUT12, INOUT13, INOUT14
   INOUT21, INOUT22, INOUT23, INOUT24
   INOUT31, INOUT32, INOUT34
   INOUT41, INOUT42, INOUT43, INOUT44, INOUT45
   INOUT51, INOUT52, INOUT53, INOUT54
   INOUT61, INOUT62, INOUT63, INOUT64
   INOUT71, INOUT72, INOUT73, INOUT74
   INOUT81, INOUT82, INOUT83, INOUT84, INOUT85, INOUT86
   MASSBAL1, MASSBAL2, MASSBAL3, MASSBAL4
   MASSBAL5, MASSBAL6, MASSBAL7, MASSBAL8
   SPECS1, SPECS2, SPECS3, SPECS4
   OBJETIVO /;

* MASTER PROBLEM EQUATIONS:
* -------------------------
 MODEL MASTER  / MASSBAL1, MASSBAL2, MASSBAL3, MASSBAL4
   MASSBAL5, MASSBAL6, MASSBAL7, MASSBAL8
   SPECS1, SPECS2, SPECS3, SPECS4
   PUREINT1, PUREINT2, PUREINT3, PUREINT4
   PUREINT5, PUREINT6, PUREINT7, PUREINT8
   PUREINT9, PUREINT10, PUREINT11, PUREINT12
   PUREINT13, PUREINT14, PUREINT15 /;

  MODEL LOGIC /ALL/;
  SOLVE LOGIC USING MINLP MINIMIZING PROF;
```

A.3. Archivo de entrada GAMS del ejemplo 4.4.1. Modelo híbrido (PH).

```
$TITLE APPLICATION OF THE LOGIC-BASED MINLP ALGORITHM IN EXAMPLE #3
* FOR THIS CASE THE PROBLEM FORMULATION IS HYBRID (disjunctions y 0-1
variables)
$OFFSYMXREF
$OFFSYMLIST
*  SELECT OPTIMAL PROCESS FROM WITHIN GIVEN SUPERSTRUCTURE.
*
SETS   I      PROCESS STREAMS                / 1*25 /
       J      PROCESS UNITS                  / 1*8 /
       K      DISJUNCTION SUBSET          / 1*5 /
       M      BINARY VAR. SUBSET             / 1*3 /

 PARAMETERS   CV(I)     VARIABLE COST COEFF FOR PROCESS UNITS - STREAMS/
                            3 = -10  ,  5 = -15  ,  9 = -40
                           19 =  25  , 21 =  35  , 25 = -35
                           17 =  80  , 14 =  15  , 10 =  15
                            2 =  1   ,  4 =  1   , 18 = -65
                           20 = -60  , 22 = -80                    /;

 PARAMETERS YMT(K)     BINARY VARIABLES FOR DISJUNCTIONS
;

 VARIABLES  PROF     PROFIT ;

*Y are binary variables to handle disjunctions,Y2 binary variables for
* process that are not in disjunctions (linear equations)

 BINARY VARIABLES    Y(K), Y2(M)  ;
 POSITIVE VARIABLES  X(I) , CF(J) ;

 EQUATIONS
* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* ---------------------------------------------------------
            MASSBAL1     MASS BALANCE #1
            MASSBAL2     MASS BALANCE #2
            MASSBAL3     MASS BALANCE #3
            MASSBAL4     MASS BALANCE #4
            MASSBAL5     MASS BALANCE #5
            MASSBAL6     MASS BALANCE #6
            MASSBAL7     MASS BALANCE #7
            MASSBAL8     MASS BALANCE #8
            SPECS1       DESIGN SPECIFICATION 1
            SPECS2       DESIGN SPECIFICATION 2
            SPECS3       DESIGN SPECIFICATION 3
            SPECS4       DESIGN SPECIFICATION 4

* EQUATIONS FOR THE MASTER PROBLEMS ONLY:
* ---------------------------------------
           PUREINT1     PURELY INTEGER CONSTRAINT #1
            PUREINT2     PURELY INTEGER CONSTRAINT #2
            PUREINT3     PURELY INTEGER CONSTRAINT #3
            PUREINT4     PURELY INTEGER CONSTRAINT #4
            PUREINT5     PURELY INTEGER CONSTRAINT #5
            PUREINT6     PURELY INTEGER CONSTRAINT #6
            PUREINT7     PURELY INTEGER CONSTRAINT #7
            PUREINT8     PURELY INTEGER CONSTRAINT #8
            PUREINT9     PURELY INTEGER CONSTRAINT #9
```

```
            PUREINT10    PURELY INTEGER CONSTRAINT #10
            PUREINT11    PURELY INTEGER CONSTRAINT #11
            PUREINT12    PURELY INTEGER CONSTRAINT #12
            PUREINT13    PURELY INTEGER CONSTRAINT #13
            PUREINT14    PURELY INTEGER CONSTRAINT #14
            PUREINT15    PURELY INTEGER CONSTRAINT #15

* EQUATIONS FOR THE NLP SUBPROBLEMS ONLY:
* ------------------------------------
            INOUT11      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 1
            INOUT12
            INOUT13
            INOUT14
            INOUT21      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 2
            INOUT22
            INOUT23
            INOUT24
            INOUT31      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 3
            INOUT34
            INOUT41      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 4
            INOUT45
            INOUT51      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 5
            INOUT54
            INOUT61      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 6
            INOUT62
            INOUT63
            INOUT64
            INOUT71      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 7
            INOUT72
            INOUT73
            INOUT74
            INOUT81      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 8
            INOUT82
            INOUT83
            INOUT84
            INOUT85
            INOUT86
            OBJETIVO          OBJECTIVE FUNCTION DEFINITION ;

* BOUNDS SECTION:
* ---------------
 X.UP('3')  =  2.0 ;
 X.UP('5')  =  2.0 ;
 X.UP('9')  =  2.0 ;
 X.UP('10') =  1.0 ;
 X.UP('14') =  1.0 ;
 X.UP('17') =  2.0 ;
 X.UP('19') =  2.0 ;
 X.UP('21') =  2.0 ;
 X.UP('25') =  3.0 ;

* INITIAL VALUES FOR BINARY VARIABLES:
* ------------------------------------
 Y.L('1')   =  1   ;
 Y.L('2')   =  1   ;
 Y.L('3')   =  1   ;
 Y.L('4')   =  1   ;
 Y.L('5')   =  1   ;
 YMT(K) = Y.L(K);
```

```
 Y2.L('2') = 1.;
 Y2.L('1') = 1.;
* INITIAL VALUES FOR CONTINUOUS VARIABLES:
* ----------------------------------------

 OPTIONS LIMCOL = 0   ;
 OPTION LIMROW  = 0   ;
 OPTION OPTCR   = 0   ;
 OPTION MINLP   = LOGMIP;

* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* -------------------------------------------------------
 MASSBAL1    ..   X('13')         =E=  X('19') + X('21')          ;
 MASSBAL2    ..   X('17')         =E=  X('9') + X('16') + X('25') ;
 MASSBAL3    ..   X('11')         =E=  X('12') + X('15')          ;
 MASSBAL4    ..   X('3') + X('5') =E=  X('6') + X('11')           ;
 MASSBAL5    ..   X('6')          =E=  X('7') + X('8')            ;
 MASSBAL6    ..   X('23')         =E=  X('20') + X('22')          ;
 MASSBAL7    ..   X('23')         =E=  X('14') + X('24')          ;
 MASSBAL8    ..   X('1')          =E=  X('2') + X('4')            ;

 SPECS1      ..   X('10')  =L=  0.8 * X('17')                     ;
 SPECS2      ..   X('10')  =G=  0.4 * X('17')                     ;
 SPECS3      ..   X('12')  =L=  5.0 * X('14')                     ;
 SPECS4      ..   X('12')  =G=  2.0 * X('14')                     ;

 PUREINT1    ..   Y('1') + Y('2')  =L=  1.                      ;
 PUREINT2    .. - Y('1') + Y2('1') + Y2('2') + Y2('3')  =G=  0    ;
 PUREINT3    .. - Y('2') + Y2('1') + Y2('2') + Y2('3')  =G=  0    ;
 PUREINT4    .. - Y2('1') + Y('5')  =G=  0                       ;
 PUREINT5    ..   Y('1') + Y('2') - Y2('1')  =G= 0               ;
 PUREINT6    ..   Y('1') + Y('2') - Y2('2')  =G= 0               ;
 PUREINT7    .. - Y2('2') + Y('3') + Y('4')  =G= 0               ;
 PUREINT8    ..   Y2('1') + Y2('2')  =L=  1                       ;
 PUREINT9    ..   Y('1') + Y('2') - Y2('3')  =G= 0               ;
 PUREINT10   .. - Y2('3') + Y('5')  =G=  0                       ;
 PUREINT11   ..   Y2('2') - Y('3')  =G=  0                       ;
 PUREINT12   ..   Y('3') + Y('4')  =L=  1                      ;
 PUREINT13   ..   Y2('2') - Y('4')  =G=  0                       ;
 PUREINT14   .. - Y2('3') + Y('5')  =L=  1                       ;
 PUREINT15   .. - Y2('1') + Y('5')  =L=  1                       ;

* EQUATIONS FOR THE NLP SUBPROBLEMS ONLY:
* ---------------------------------------
 INOUT11 $ (YMT('1') EQ 1)  ..   EXP(X('3')) -1. =E= X('2')           ;
 INOUT12 $ (YMT('1') EQ 0)  ..   X('2') =E= 0                         ;
 INOUT13 $ (YMT('1') EQ 0)  ..   X('3') =E= 0                         ;
 INOUT14 $ (YMT('1') EQ 1)  ..   CF('1') =E= 5                        ;
 INOUT21 $ (YMT('2') EQ 1)  ..   EXP(X('5')/1.2) -1. =E= X('4')       ;
 INOUT22 $ (YMT('2') EQ 0)  ..   X('4') =E= 0                         ;
 INOUT23 $ (YMT('2') EQ 0)  ..   X('5') =E= 0                         ;
 INOUT24 $ (YMT('2') EQ 1)  ..   CF('2') =E= 8                        ;
 INOUT31..   1.5 * X('9') + X('10') =E= X('8')     ;
 INOUT34..   CF('3') =E= 6. * Y2('1')               ;
 INOUT41..   1.25 * (X('12')+X('14')) =E= X('13')  ;
 INOUT45..   CF('4') =E= 10 * Y2('2')              ;
 INOUT51..   X('15') =E= 2. * X('16')              ;
 INOUT54..   CF('5') =E= 6. * Y2('3')               ;
 INOUT61 $ (YMT('3') EQ 1)  ..   EXP(X('20')/1.5) -1. =E= X('19')     ;
```

```
 INOUT62 $ (YMT('3') EQ 0)  ..   X('19') =E= 0                          ;
 INOUT63 $ (YMT('3') EQ 0)  ..   X('20') =E= 0                          ;
 INOUT64 $ (YMT('3') EQ 1)  ..   CF('6') =E= 7                          ;
 INOUT71 $ (YMT('4') EQ 1)  ..   EXP(X('22')) -1. =E= X('21')           ;
 INOUT72 $ (YMT('4') EQ 0)  ..   X('21') =E= 0                          ;
 INOUT73 $ (YMT('4') EQ 0)  ..   X('22') =E= 0                          ;
 INOUT74 $ (YMT('4') EQ 1)  ..   CF('7') =E= 4                          ;
 INOUT81 $ (YMT('5') EQ 1)  ..   EXP(X('18')) -1. =E= X('10') + X('17');
 INOUT82 $ (YMT('5') EQ 0)  ..   X('10') =E= 0                          ;
 INOUT83 $ (YMT('5') EQ 0)  ..   X('17') =E= 0                          ;
 INOUT84 $ (YMT('5') EQ 0)  ..   X('18') =E= 0                          ;
 INOUT85 $ (YMT('5') EQ 0)  ..   X('25') =E= 0                          ;
 INOUT86 $ (YMT('5') EQ 1)  ..   CF('8') =E= 5                          ;
 OBJETIVO  .. PROF  =E= SUM(J,CF(J)) + SUM(I , X(I)*CV(I)) + 122 ;

* SUBPROBLEM EQUATIONS:
* ---------------------
 MODEL NLPX / INOUT11, INOUT12, INOUT13, INOUT14
   INOUT21, INOUT22, INOUT23, INOUT24
   INOUT31, INOUT34
   INOUT41, INOUT45
   INOUT51, INOUT54
   INOUT61, INOUT62, INOUT63, INOUT64
   INOUT71, INOUT72, INOUT73, INOUT74
   INOUT81, INOUT82, INOUT83, INOUT84, INOUT85, INOUT86
   MASSBAL1, MASSBAL2, MASSBAL3, MASSBAL4
   MASSBAL5, MASSBAL6, MASSBAL7, MASSBAL8
   SPECS1, SPECS2, SPECS3, SPECS4
   OBJETIVO /;

* MASTER PROBLEM EQUATIONS:
* -------------------------
 MODEL MASTER  /
   INOUT31, INOUT34
   INOUT41, INOUT45
   INOUT51, INOUT54
   MASSBAL1, MASSBAL2, MASSBAL3, MASSBAL4
   MASSBAL5, MASSBAL6, MASSBAL7, MASSBAL8
   SPECS1, SPECS2, SPECS3, SPECS4
   LOGICAL1, LOGICAL2, LOGICAL3, LOGICAL4
   LOGICAL5, LOGICAL6, LOGICAL7, LOGICAL8
   PUREINT1, PUREINT2, PUREINT3, PUREINT4
   PUREINT5, PUREINT6, PUREINT7, PUREINT8
   PUREINT9, PUREINT10, PUREINT11, PUREINT12
   PUREINT13, PUREINT14, PUREINT15 /;

  MODEL LOGIC /ALL/;
  SOLVE LOGIC USING MINLP MINIMIZING PROF;
```

A.4. Archivo de entrada GAMS del ejemplo 4.4.4. Modelo MINLP.

```
$TITLE BATCH PROCESSING PROBLEM with intermediate storage tanks (Ravemark
form)
$OFFSYMXREF
$OFFSYMLIST
*
*  MINLP FORMULATION OF BATCH PROCESSING PROBLEM

 SETS          I         PRODUCTS                / A , B , C , D , E /
               J         POTENTIAL STAGES                    / 1 * 6 /
               M(J)         POTENTIAL STORAGE TANKS             / 1 * 5 /
               K         BINARIES NEEDED TO REPRESENT N(J)   / 1 * 4 /

 SCALAR        H         HORIZON TIME (available time hrs)   / 6000. /
               ALPHA     COST COEFFICIENT FOR BATCH UNITS    / 250.  /
               COSST     COST COEFF. FOR STORAGE TANK        /150./
               GAMMA     COST COEFF. FOR STORAGE TANK        /0.5 /

 PARAMETERS    Q(I)      PRODUCTION RATE OF PRODUCT i     / A=250000 ,
                          B=150000 , C=180000 , D=160000 , E=120000 /
               PHI
               ACTTL(I,J)
               ACTV(J)   ACTUAL VOLUME
               ACTB(I,J) ACTUAL BATCH SIZES
               ACTN(J)   ACTUAL NUMBER OF UNITS IN PARALLEL  (in phase)
               COEFF(K)
               COEFF('1') = LOG(1.)  ;
               COEFF('2') = LOG(2.)  ;
               COEFF('3') = LOG(3.)  ;
               COEFF('4') = LOG(4.)  ;


 TABLE         S(I,J)    SIZE FACTOR FOR PRODUCT i IN STAGE j
                 1         2         3         4         5         6
    A          7.9       2.        5.2       4.9       6.1       4.2
    B          0.7       0.8       0.9       3.4       2.1       2.5
    C          0.7       2.6       1.6       3.6       3.2       2.9
    D          4.7       2.3       1.6       2.7       1.2       2.5
    E          1.2       3.6       2.4       4.5       1.6       2.1

 TABLE         T(I,J)    PROCESSING TIME OF PRODUCT I IN BATCH J
                 1         2         3         4         5         6
    A          6.4       4.7       8.3       3.9       2.1       1.2
    B          6.8       6.4       6.5       4.4       2.3       3.2
    C          1.0       6.3       5.4      11.9       5.7       6.2
    D          3.2       3.0       3.5       3.3       2.8       3.4
    E          2.1       2.5       4.2       3.6       3.7       2.2

 VARIABLES     V(J)      VOLUME OF STAGE J
               B(I,J)    BATCH SIZE OF PRODUCT i STAGE(j)
               TL(I,J)     CYCLE TIME OF PRODUCT i
               N(J)      NUMBER OF UNIT IN PARALLEL STAGE j
               MI(J)
               COST      TOTAL COST OF BATCH PROCESSING UNITS
               VST(J)    VOLUME OF STORAGE TANK BETEWN STAGE J AND J+1
               E(I) ;
```

```
* Y1, Y2 Binary Variable, Y3 binary for the storage tanks
 BINARY   VARIABLES     Y1(K,J), Y2(K,J), Y3(J)                  ;
 POSITIVE VARIABLES     V(J),B(I,J), TL(I,J),N(J),MI(J),VST(J)  ;

 EQUATIONS  VOL(I,J)     CALCULATE VOLUME OF STAGE j
            TI(I,J)
            TIME         TIME CONSTRAINT
            VOLUM1(I,J) STORAGE TANK SIZE EQUATION for PRODUCT I STAGE J
            VOLUM2(I,J) STORAGE TANK SIZE EQUATION for PRODUCT I STAGE J
            UNIT1(J) CALCULATE NUMBER PROCESSING UNITS PER STAGE(in-phase)
            UNIT2(J) CALCULATE NUMBER PROCESSING UNITS PER STAGE(out-phase)
            LIM1(J)    LIMIT SELECTION TO ONE NUMBER for N(J)
            LIM2(J)
            BTSZE1(I,J) BATCH SIZE EQUAT. PRODUCT I stage J
            BTSZE2(I,J) BATCH SIZE EQUAT. PRODUCT i stage J
            OBJ       OBJECTIVE FUNCTION DEFINITION               ;

*************************************************************************
*  BOUNDS SECTION
 V.LO(J)   =  LOG(300.) ;
 V.UP(J)   =  LOG(3500.);
 VST.LO(J) =  LOG(100.) ;
 VST.UP(J) =  LOG(15000.);
 N.UP(J)   =  LOG(4.)   ;
 MI.UP(J)  =  LOG(4.)   ;
 PARAMETERS   TLLOW(I)       LOWER BOUND ON TL(I)
              TLUPP(I)       UPPER BOUND ON TL(I)                  ;
              TLLOW(I)  =  SMAX(J, T(I,J) / EXP(N.UP(J)))          ;
              TLUPP(I)  =  SMAX(J, T(I,J) )                        ;
 TL.LO(I,J)  =  LOG(TLLOW(I)) ;
 TL.UP(I,J)  =  LOG(TLUPP(I)) ;
 PARAMETERS   BLOW(I)        LOWER BOUND ON B
              BUPP(I)        UPPER BOUND ON B                   ;

              BLOW(I)  =  Q(I) * ( SMAX(J, T(I,J) / EXP(N.UP(J))))/ H;

              BUPP(I)  =  MIN( Q(I) , SMIN(J , EXP(V.UP(J))/S(I,J))) ;

 B.LO(I,J)   =  LOG(BLOW(I)) ;
 B.UP(I,J)   =  LOG(BUPP(I)) ;
 PARAMETERS ACTVST(J), ACTMI(J);
*  INITIAL POINT
 Y3.L(J) = 1. ;
 Y1.L('1','1') = 1.      ;
 Y1.L('1','2') = 1.      ;
 Y1.L('1','3') = 1.      ;
 Y1.L('1','4') = 1.      ;
 Y1.L('1','5') = 1.      ;
 Y1.L('1','6') = 1.      ;
 Y2.L('2','1') = 1.      ;
 Y2.L('2','2') = 1.      ;
 Y2.L('2','3') = 1.      ;
 Y2.L('2','4') = 1.      ;
 Y2.L('2','5') = 1.      ;
 Y2.L('2','6') = 1.      ;

 N.L(J)    = SUM( K , COEFF(K)*  Y1.L(K,J) )                      ;
 MI.L(J)   = SUM( K , COEFF(K)*  Y2.L(K,J) )                      ;
 B.L(I,J)  = (B.UP(I,J) + B.LO(I,J))/2.                           ;
```

```
TL.L(I,J) = (TL.UP(I,J) + TL.LO(I,J))/2.                          ;
V.L(J)    = SMAX(I ,  LOG(S(I,J)) + B.L(I,J))                     ;
VST.L(J)  = SMAX(I , 2*S(I,J)*B.L(I,J))                           ;
PHI       = SMAX(I,SMAX(J $(ORD(J) LE 5), B.L(I,J)/B.L(I,J+1)))   ;
*********************************************************************

OPTION OPTCR = 0.0               ;
OPTIONS LIMCOL   = 0             ;
OPTIONS ITERLIM  = 10000000      ;

VOL(I,J)..     V(J) =G= LOG(S(I,J)) + B(I,J) - N(J);
TI(I,J)..  E(I) =G= LOG(T(I,J))- B(I,J)- MI(J)     ;
TIME..         SUM(I , Q(I) * EXP(E(I))) =L= H     ;
VOLUM1(I,J)$(ORD(J)LT CARD(J)).. VST(J)=G=LOG(2*5)+ B(I,J+1)-10*(1-Y3(J));
VOLUM2(I,J)$(ORD(J)LT CARD(J)).. VST(J)=G=LOG(2*5)+ B(I,J) - 10*(1-Y3(J));
BTSZE1(I,J)$(ORD(J)LT CARD(J)).. B(I,J) - B(I,J+1) =L= LOG(3.) * Y3(J);
BTSZE2(I,J)$(ORD(J)LT CARD(J)).. B(I,J) - B(I,J+1) =G= -LOG(3.)* Y3(J);
UNIT1(J)..  N(J) =E= SUM(K , COEFF(K) * Y1(K,J))        ;
LIM1(J) ..  SUM(K , Y1(K,J)) =E= 1                      ;
UNIT2(J)..  MI(J)  =E=       SUM(K , COEFF(K) * Y2(K,J));
LIM2(J) ..  SUM(K , Y2(K,J)) =E= 1                      ;
OBJ..       COST =E= ALPHA * SUM(J ,EXP(N(J) + MI(J) + 0.6*V(J))) +
                150.  * SUM(J$(ORD(J) LT CARD(J)),EXP(0.5 * VST(J)))
              - 150.  * SUM(J$(ORD(J) LT CARD(J)),EXP(0.5 * VST.LO(J)));

MODEL   BATCH   / ALL / ;
BATCH.optfile = 1      ;
SOLVE BATCH USING MINLP  MINIMIZING  COST ;

ACTV(J) = EXP(V.L(J))      ;
ACTB(I,J) = EXP(B.L(I,J))  ;
ACTTL(I,J) = EXP(TL.L(I,J))    ;
ACTN(J) = EXP(N.L(J))  ;
ACTMI(J) = EXP(MI.L(J));
ACTVST(J) = EXP(VST.L(J));
DISPLAY ACTV       ;
DISPLAY ACTVST     ;
DISPLAY ACTB       ;
DISPLAY ACTTL      ;
DISPLAY ACTN       ;
DISPLAY ACTMI      ;
DISPLAY PHI;
```

A.5. Archivo de entrada GAMS del ejemplo 4.4.4. Modelo híbrido (PH).

```
$TITLE BATCH PROCESSING PROBLEM with intermediate storage tanks (Ravemark
form.)
* HYBRID FORMULATION PROBLEM
$OFFSYMXREF
$OFFSYMLIST
*
*  MINLP FORMULATION OF BATCH PROCESSING PROBLEM

 SETS         I         PRODUCTS                / A , B , C , D , E /
              J         POTENTIAL STAGES                  / 1 * 6 /
              M(J)      POTENTIAL STORAGE TANKS           / 1 * 5 /
              K         BINARIES NEEDED TO REPRESENT N(J)  / 1 * 4 /

 SCALAR       H         HORIZON TIME (available time hrs)  / 6000. /
              ALPHA     COST COEFFICIENT FOR BATCH UNITS   / 250.  /
              COSST     COST COEFF. FOR STORAGE TANK       /150./
              GAMMA     COST COEFF. FOR STORAGE TANK       /0.5 /

 PARAMETERS   Q(I)      PRODUCTION RATE OF PRODUCT i    / A=250000 ,
                         B=150000 , C=180000 , D=160000 , E=120000 /
              PHI
              ACTTL(I,J)
              ACTV(J)   ACTUAL VOLUME
              ACTB(I,J) ACTUAL BATCH SIZES
              ACTN(J)   ACTUAL NUMBER OF UNITS IN PARALLEL  (in phase)
              COEFF(K)
              YMT(J) ;
              COEFF('1') = LOG(1.)  ;
              COEFF('2') = LOG(2.)  ;
              COEFF('3') = LOG(3.)  ;
              COEFF('4') = LOG(4.)  ;


 TABLE        S(I,J)    SIZE FACTOR FOR PRODUCT i IN STAGE j
                1         2         3         4         5         6
    A         7.9       2.        5.2       4.9       6.1       4.2
    B         0.7       0.8       0.9       3.4       2.1       2.5
    C         0.7       2.6       1.6       3.6       3.2       2.9
    D         4.7       2.3       1.6       2.7       1.2       2.5
    E         1.2       3.6       2.4       4.5       1.6       2.1

 TABLE        T(I,J)    PROCESSING TIME OF PRODUCT I IN BATCH J
                1         2         3         4         5         6
    A         6.4       4.7       8.3       3.9       2.1       1.2
    B         6.8       6.4       6.5       4.4       2.3       3.2
    C         1.0       6.3       5.4      11.9       5.7       6.2
    D         3.2       3.0       3.5       3.3       2.8       3.4
    E         2.1       2.5       4.2       3.6       3.7       2.2

 VARIABLES    V(J)      VOLUME OF STAGE J
              B(I,J)    BATCH SIZE OF PRODUCT i STAGE(j)
              TL(I,J)     CYCLE TIME OF PRODUCT i
              N(J)      NUMBER OF UNIT IN PARALLEL STAGE j
              MI(J)
              COST      TOTAL COST OF BATCH PROCESSING UNITS
              VST(J)    VOLUME OF STORAGE TANK BETEWN STAGE J AND J+1
              E(I) ;
```

```
* Y1, Y2 Binary Variable, Y3 Binary Variables to handle disjunctions

 BINARY   VARIABLES     Y1(K,J), Y2(K,J), Y3(J)               ;
 POSITIVE VARIABLES     V(J),B(I,J),TL(I,J),N(J),MI(J),VST(J) ;

 EQUATIONS VOL(I,J)     CALCULATE VOLUME OF STAGE j
           TI(I,J)
           TIME         TIME CONSTRAINT
           VOLUM1(I,J) STORAGE TANK SIZE EQUATION for PRODUCT I STAGE J
           VOLUM2(I,J) STORAGE TANK SIZE EQUATION for PRODUCT I STAGE J
           VOLUMNO(J)  Size = 0
           UNIT1(J)  CALCULATE NUMBER PROCESSING UNITS PER STAGE(in phase)
           UNIT2(J)  CALCULATE NUMBER PROCESSING UNITS PER STAGE(out phase)
           LIM1(J)    LIMIT SELECTION TO ONE NUMBER for N(J)
           LIM2(J)
           BTSZE1(I,J) BATCH SIZE EQUAT. PRODUCT I stage J
           BTSZE2(I,J) BATCH SIZE EQUAT. PRODUCT i stage J
           BTSZNO(I,J) B(i,j)=B(i,j+1)
           FICT      EQUATION VALID JUST TO SET UP VARIABLE Y3
           OBJ       OBJECTIVE FUNCTION DEFINITION               ;

*************************************************************************
*  BOUNDS SECTION
 V.LO(J)   =  LOG(300.) ;
 V.UP(J)   =  LOG(3500.);
 VST.LO(J) =  LOG(100.) ;
 VST.UP(J) =  LOG(15000.);
 N.UP(J)   =  LOG(4.)   ;
 MI.UP(J)  =  LOG(4.)   ;

 PARAMETERS   TLLOW(I)       LOWER BOUND ON TL(I)
              TLUPP(I)       UPPER BOUND ON TL(I)                  ;
              TLLOW(I)  =  SMAX(J, T(I,J) / EXP(N.UP(J)))          ;
              TLUPP(I)  =  SMAX(J, T(I,J) )                        ;
 TL.LO(I,J)  =  LOG(TLLOW(I)) ;
 TL.UP(I,J)  =  LOG(TLUPP(I)) ;
 PARAMETERS   BLOW(I)        LOWER BOUND ON B
              BUPP(I)        UPPER BOUND ON B                   ;

              BLOW(I)  =  Q(I) * ( SMAX(J, T(I,J) / EXP(N.UP(J))))/ H;

              BUPP(I)  =  MIN( Q(I) , SMIN(J , EXP(V.UP(J))/S(I,J))) ;

 B.LO(I,J)   =  LOG(BLOW(I)) ;
 B.UP(I,J)   =  LOG(BUPP(I)) ;
 PARAMETERS ACTVST(J), ACTMI(J);
*  INITIAL POINT
 Y3.L(J) = 1. ;
 Y1.L('1','1') = 1.      ;
 Y1.L('1','2') = 1.      ;
 Y1.L('1','3') = 1.      ;
 Y1.L('1','4') = 1.      ;
 Y1.L('1','5') = 1.      ;
 Y1.L('1','6') = 1.      ;
 Y2.L('2','1') = 1.      ;
 Y2.L('2','2') = 1.      ;
 Y2.L('2','3') = 1.      ;
 Y2.L('2','4') = 1.      ;
```

```
Y2.L('2','5') = 1.      ;
Y2.L('2','6') = 1.      ;
YMT(J) = Y3.L(J)       ;

N.L(J)    = SUM( K , COEFF(K)*  Y1.L(K,J) )                    ;
MI.L(J)   = SUM( K , COEFF(K)*  Y2.L(K,J) )                    ;
B.L(I,J)  = (B.UP(I,J) + B.LO(I,J))/2.                         ;
TL.L(I,J) = (TL.UP(I,J) + TL.LO(I,J))/2.                       ;
V.L(J)    = SMAX(I ,  LOG(S(I,J)) + B.L(I,J))                  ;
VST.L(J)  = SMAX(I , 2*S(I,J)*B.L(I,J))                        ;
PHI       = SMAX(I,SMAX(J $(ORD(J) LE 5), B.L(I,J)/B.L(I,J+1)))   ;
*************************************************************************

OPTION OPTCR = 0.0               ;
OPTIONS LIMCOL   = 0             ;
OPTIONS ITERLIM  = 10000000      ;
OPTION MINLP = LOGMIP            ;

VOL(I,J)..     V(J) =G= LOG(S(I,J)) + B(I,J)  - N(J);
TI(I,J)..  E(I) =G= LOG(T(I,J))- B(I,J)- MI(J)      ;
TIME..         SUM(I , Q(I) * EXP(E(I))) =L= H      ;
VOLUM1(I,J)$(YMT(J) EQ 1 AND ORD(J) LT CARD(J))..
                    VST(J)=G= LOG(2*5)+B(I,J+1)   ;
VOLUM2(I,J)$(YMT(J) EQ 1 AND ORD(J) LT CARD(J))..
                    VST(J) =G= LOG(2*5)+B(I,J)    ;
VOLUMNO(J)$(YMT(J)EQ 0 AND ORD(J) LT CARD(J)).. VST(J) =E= LOG(VST(J).LO);
BTSZE1(I,J)$(YMT(J)EQ 1 AND ORD(J)LT CARD(J)).. B(I,J)-B(I,J+1)=L= LOG(3);
BTSZE2(I,J)$(YMT(J)EQ 1 AND ORD(J)LT CARD(J))..B(I,J)-B(I,J+1)=G=-LOG(3.);
BTSZNO(I,J)$(YMT(J)EQ 0 AND ORD(J) LT CARD(J)).. B(I,J)-B(I,J+1)=E=0     ;
UNIT1(J)..  N(J) =E= SUM(K , COEFF(K) * Y1(K,J))          ;
LIM1(J) ..  SUM(K , Y1(K,J)) =E= 1                        ;
UNIT2(J)..  MI(J)  =E=       SUM(K , COEFF(K) * Y2(K,J))  ;
LIM2(J) ..  SUM(K , Y2(K,J)) =E= 1                        ;
FICT..      SUM(J,Y3(J)) =G= 0;
OBJ..       COST =E= ALPHA * SUM(J ,EXP(N(J) + MI(J) + 0.6*V(J))) +
                150.  * SUM(J$(ORD(J) LT CARD(J)),EXP(0.5 * VST(J)))
              - 150.  * SUM(J$(ORD(J) LT CARD(J)),EXP(0.5 * VST.LO(J)));

MODEL NLPX/VOL, TI, TIME, VOLUM1, VOLUM2, VOLUMNO, BTSZE1, BTSZE2, BTSZNO,
          UNIT1, LIM1, UNIT2, LIM2, OBJ/;

MODEL MASTER / VOL, TI, UNIT1, LIM1, UNIT2, LIM2, FICT /;
 MODEL   BATCH   / ALL / ;
BATCH.optfile = 1       ;
SOLVE BATCH USING MINLP  MINIMIZING  COST ;

ACTV(J) = EXP(V.L(J))      ;
ACTB(I,J) = EXP(B.L(I,J))  ;
ACTTL(I,J) = EXP(TL.L(I,J))    ;
ACTN(J) = EXP(N.L(J))  ;
ACTMI(J) = EXP(MI.L(J));
ACTVST(J) = EXP(VST.L(J));
DISPLAY ACTV       ;
DISPLAY ACTVST     ;
DISPLAY ACTB       ;
DISPLAY ACTTL      ;
DISPLAY ACTN       ;
DISPLAY ACTMI      ;
DISPLAY PHI        ;
```

B

Apéndice B

B.1. Archivo de entrada LOGMIP (versión 2) para el ejemplo 4.4.1.

```
$TITLE APPLICATION OF THE LOGIC-BASED MINLP ALGORITHM IN EXAMPLE #3
* FOR THIS PROBLEM THE FORMULATION IS DISJUNCTIVE
$OFFSYMXREF
$OFFSYMLIST
*  SELECT OPTIMAL PROCESS FROM WITHIN GIVEN SUPERSTRUCTURE.
*
*  REFERENCE: MARCO DURAN , PH.D. THESIS , 1984.
*             CARNEGIE-MELLON UNIVERSITY , PITTSBURGH , PA.

SETS   I      PROCESS STREAMS                / 1*25 /
       J      PROCESS UNITS                  / 1*8 /


 PARAMETERS   CV(I)     VARIABLE COST COEFF FOR PROCESS UNITS -
STREAMS/
                            3 = -10  ,  5 = -15  ,  9 = -40
                           19 =  25  , 21 =  35  , 25 = -35
                           17 =  80  , 14 =  15  , 10 =  15
                            2 =  1   ,  4 =  1   , 18 = -65
                           20 = -60  , 22 = -80
/;

 VARIABLES  PROF     PROFIT ;

 BINARY VARIABLES    Y(J)  ;
 POSITIVE VARIABLES  X(I) , CF(J) ;

 EQUATIONS

* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* -------------------------------------------------------
            MASSBAL1     MASS BALANCE #1
            MASSBAL2     MASS BALANCE #2
            MASSBAL3     MASS BALANCE #3
            MASSBAL4     MASS BALANCE #4
            MASSBAL5     MASS BALANCE #5
            MASSBAL6     MASS BALANCE #6
            MASSBAL7     MASS BALANCE #7
            MASSBAL8     MASS BALANCE #8
            SPECS1       DESIGN SPECIFICATION 1
            SPECS2       DESIGN SPECIFICATION 2
            SPECS3       DESIGN SPECIFICATION 3
            SPECS4       DESIGN SPECIFICATION 4

* EQUATIONS FOR THE MASTER PROBLEMS ONLY:
* ---------------------------------------
           LOGICAL1     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 1 EXISTS
           LOGICAL2     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 2 EXISTS
           LOGICAL3     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 3 EXISTS
           LOGICAL4     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 4 EXISTS
           LOGICAL5     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 5 EXISTS
           LOGICAL6     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 6 EXISTS
           LOGICAL7     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 7 EXISTS
           LOGICAL8     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 8 EXISTS
           PUREINT1     PURELY INTEGER CONSTRAINT #1
           PUREINT2     PURELY INTEGER CONSTRAINT #2
           PUREINT3     PURELY INTEGER CONSTRAINT #3
```

```
           PUREINT4     PURELY INTEGER CONSTRAINT #4
           PUREINT5     PURELY INTEGER CONSTRAINT #5
           PUREINT6     PURELY INTEGER CONSTRAINT #6
           PUREINT7     PURELY INTEGER CONSTRAINT #7
           PUREINT8     PURELY INTEGER CONSTRAINT #8
           PUREINT9     PURELY INTEGER CONSTRAINT #9
           PUREINT10    PURELY INTEGER CONSTRAINT #10
           PUREINT11    PURELY INTEGER CONSTRAINT #11
           PUREINT12    PURELY INTEGER CONSTRAINT #12
           PUREINT13    PURELY INTEGER CONSTRAINT #13
           PUREINT14    PURELY INTEGER CONSTRAINT #14
           PUREINT15    PURELY INTEGER CONSTRAINT #15

* EQUATIONS FOR THE NLP SUBPROBLEMS ONLY:
* ---------------------------------------
            INOUT11      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 1
            INOUT12
            INOUT13
            INOUT14
            INOUT21      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 2
            INOUT22
            INOUT23
            INOUT24
            INOUT31      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 3
            INOUT32
            INOUT34
            INOUT41      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 4
            INOUT42
            INOUT43
            INOUT44
            INOUT45
            INOUT51      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 5
            INOUT52
            INOUT53
            INOUT54
            INOUT61      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 6
            INOUT62
            INOUT63
            INOUT64
            INOUT71      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 7
            INOUT72
            INOUT73
            INOUT74
            INOUT81      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 8
            INOUT82
            INOUT83
            INOUT84
            INOUT85
            INOUT86
            OBJETIVO          OBJECTIVE FUNCTION DEFINITION ;

* BOUNDS SECTION:
* ---------------
 X.UP('3')  =  2.0 ;
 X.UP('5')  =  2.0 ;
 X.UP('9')  =  2.0 ;
 X.UP('10') =  1.0 ;
 X.UP('14') =  1.0 ;
 X.UP('17') =  2.0 ;
```

```
 X.UP('19') =  2.0 ;
 X.UP('21') =  2.0 ;
 X.UP('25') =  3.0 ;

INITIAL BU 3;

TRUE Y('1') Y('3') Y('4') Y('7') Y('8');
TRUE Y('2') Y('3') Y('4') Y('6') Y('8');
TRUE Y('1') Y('3') Y('5') Y('8');

 OPTION LIMCOL = 0   ;
 OPTION LIMROW = 0   ;
 OPTION OPTCR  = 0   ;

 OPTION MINLP = LOGMIPV2;

* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* -------------------------------------------------------
 MASSBAL1    ..   X('13')         =E=  X('19') + X('21')          ;
 MASSBAL2    ..   X('17')         =E=  X('9') + X('16') + X('25') ;
 MASSBAL3    ..   X('11')         =E=  X('12') + X('15')          ;
 MASSBAL4    ..   X('3') + X('5') =E=  X('6') + X('11')           ;
 MASSBAL5    ..   X('6')          =E=  X('7') + X('8')            ;
 MASSBAL6    ..   X('23')         =E=  X('20') + X('22')          ;
 MASSBAL7    ..   X('23')         =E=  X('14') + X('24')          ;
 MASSBAL8    ..   X('1')          =E=  X('2') + X('4')            ;

 SPECS1      ..   X('10')  =L=  0.8 * X('17')                     ;
 SPECS2      ..   X('10')  =G=  0.4 * X('17')                     ;
 SPECS3      ..   X('12')  =L=  5.0 * X('14')                     ;
 SPECS4      ..   X('12')  =G=  2.0 * X('14')                     ;

 PUREINT1    ..   Y('1') + Y('2')  =L=  1.                    ;
 PUREINT2    .. - Y('1') + Y('3') + Y('4') + Y('5')  =G=  0    ;
 PUREINT3    .. - Y('2') + Y('3') + Y('4') + Y('5')  =G=  0    ;
 PUREINT4    .. - Y('3') + Y('8')  =G=  0                      ;
 PUREINT5    ..   Y('1') + Y('2') - Y('3')  =G= 0              ;
 PUREINT6    ..   Y('1') + Y('2') - Y('4')  =G= 0              ;
 PUREINT7    .. - Y('4') + Y('6') + Y('7')  =G= 0              ;
 PUREINT8    ..   Y('4') + Y('5')  =L=  1                      ;
 PUREINT9    ..   Y('1') + Y('2') - Y('5')  =G= 0              ;
 PUREINT10   .. - Y('5') + Y('8')  =G=  0                      ;
 PUREINT11   ..   Y('4') - Y('6')  =G=  0                      ;
 PUREINT12   ..   Y('6') + Y('7')  =L=  1                      ;
 PUREINT13   ..   Y('4') - Y('7')  =G=  0                      ;
 PUREINT14   .. - Y('5') + Y('8')  =L=  1                      ;
 PUREINT15   .. - Y('3') + Y('8')  =L=  1                      ;

* EQUATIONS FOR THE NLP SUBPROBLEMS ONLY:
* ---------------------------------------
IF (Y('1'))THEN
 INOUT11..   EXP(X('3')) -1. =E= X('2')           ;
 INOUT14..   CF('1') =E= 5                        ;
ELSE
 INOUT12..   X('2') =E= 0                         ;
 INOUT13..   X('3') =E= 0                         ;
ENDIF

IF (Y('2')) THEN
```

```
 INOUT21..   EXP(X('5')/1.2) -1. =E= X('4')        ;
 INOUT24..   CF('2') =E= 8                         ;
ELSE
 INOUT22..   X('4') =E= 0                          ;
 INOUT23..   X('5') =E= 0                          ;
ENDIF

IF(Y('3')) THEN
 INOUT31..   1.5 * X('9') + X('10') =E= X('8')     ;
 INOUT34..   CF('3') =E= 6                         ;
ELSE
 INOUT32..   X('9') =E= 0                          ;
ENDIF

IF(Y('4')) THEN
 INOUT41..   1.25 * (X('12')+X('14')) =E= X('13')  ;
 INOUT45..   CF('4') =E= 10                        ;
ELSE
 INOUT42..   X('12') =E= 0                         ;
 INOUT43..   X('13') =E= 0                         ;
 INOUT44..   X('14') =E= 0                         ;
ENDIF

IF(Y('5'))THEN
 INOUT51..   X('15') =E= 2. * X('16')              ;
 INOUT54..   CF('5') =E= 6                         ;
ELSE
 INOUT52..   X('15') =E= 0                         ;
 INOUT53..   X('16') =E= 0                         ;
ENDIF

IF(Y('6')) THEN
 INOUT61..   EXP(X('20')/1.5) -1. =E= X('19')      ;
 INOUT64..   CF('6') =E= 7                         ;
ELSE
 INOUT62..   X('19') =E= 0                         ;
 INOUT63..   X('20') =E= 0                         ;
ENDIF

IF(Y('7'))THEN
 INOUT71..   EXP(X('22')) -1. =E= X('21')          ;
 INOUT74..   CF('7') =E= 4                         ;
ELSE
 INOUT72..   X('21') =E= 0                         ;
 INOUT73..   X('22') =E= 0                         ;
ENDIF

IF(Y('8'))THEN
 INOUT81..   EXP(X('18')) -1. =E= X('10') + X('17');
 INOUT86..   CF('8') =E= 5                         ;
ELSE
 INOUT82..   X('10') =E= 0                         ;
 INOUT83..   X('17') =E= 0                         ;
 INOUT84..   X('18') =E= 0                         ;
 INOUT85..   X('25') =E= 0                         ;
ENDIF

 OBJETIVO  .. PROF  =E= SUM(J,CF(J)) + SUM(I , X(I)*CV(I)) + 122 ;
```

```
MODEL LOGIC /ALL/;
LOGIC.optfile=1;
SOLVE LOGIC USING MINLP MINIMIZING PROF;
```

B.2. Archivo de entrada LOGMIP (versión 2) para el ejemplo 4.4.3.

```
* PARAMETER ESTIMATION IN QUANTITATIVE IR SPECTROSCOPY
*
$offsymlist
$offsymxref

SET     I       "wave number"         /1*10/
        J       spectra data          /1*8 /
        K        compounds            /1*3/;
ALIAS(M,K);

TABLE A spectroscopic data
          1      2      3       4        5       6       7       8
1     0.0003  0.0764  0.0318  0.0007  0.0534  0.0773  0.0536  0.0320
2     0.0007  0.0003  0.0004  0.0009  0.0005  0.0009  0.0005  0.0003
3     0.0066  0.0789  0.0275  0.0043  0.0704  0.0683  0.0842  0.0309
4     0.0044  0.0186  0.0180  0.0179  0.0351  0.0024  0.0108  0.0052
5     0.0208  0.0605  0.0601  0.0604  0.0981  0.0025  0.0394  0.0221
6     0.0518  0.1656  0.1491  0.1385  0.2389  0.0248  0.1122  0.0633
7     0.0036  0.0035  0.0032  0.0051  0.0015  0.0094  0.0015  0.0024
8     0.0507  0.0361  0.0433  0.0635  0.0048  0.0891  0.0213  0.0310
9     0.0905  0.0600  0.0754  0.1098  0.0038  0.1443  0.0420  0.0574
10    0.0016  0.0209  0.0063  0.0010  0.0132  0.0203  0.0139  0.0057;

TABLE C(K,J)
         1       2       3       4       5        6       7       8
1       502     204     353     702     0       1016    104     204
2        97     351     351     351     700     0       201      97
3        0      22      8       0       14      22      14       8 ;

TABLE R(K,K)
      1     2     3
1     1     0     0
2     0     1     0
3     0     0     1;

VARIABLES  VAL(J), PROFIT, ENT(K,I);
BINARY VARIABLES Y(K,I);
POSITIVE VARIABLES P(K,I);
EQUATIONS EQUAT1(J), FICT(K), EQUAT2(K,I), EQUAT3(K,I), EQUAT4(K,I),
EQUAT5(K,I)
EQUAT6(K,I), OBJ ;
INITIAL BU 1
TRUE ALL

IF (Y(K,I)) THEN
  EQUAT2(K,I).. P(K,I)-1000 =L= 0;
  EQUAT3(K,I).. P(K,I) =G= 0;
  EQUAT5(K,I).. ENT(K,I) =E= 1 ;
ELSE
  EQUAT4(K,I).. P(K,I) =E= 0;
  EQUAT6(K,I).. ENT(K,I) =E= 0 ;
ENDIF

EQUAT1(J)..  VAL(J)=E= SUM(K, SUM(M,(C(M,J)/100.
SUM(I,P(M,I)*A(I,J)))*R(M,K))*(C(K,J)/100. - SUM(I,P(K,I)*A(I,J))));
```

```
FICT(K).. SUM(I, Y(K,I)) =G= 0;

OBJ.. PROFIT =E= SUM(J,VAL(J))+ 2*SUM(K,SUM(I,ENT(K,I)));

 OPTION INTEGER1  = 10                        ;
 OPTION ITERLIM   = 50000                     ;
 OPTION WORK      = 10000                     ;
 OPTION LIMCOL    = 0                         ;
 OPTION LIMROW    = 0                         ;
 OPTION MINLP     = LOGMIPV2                  ;


MODEL SPECTRA/ALL/;
spectra.workspace=0.5;
SOLVE SPECTRA MINIMIZING PROFIT USING MINLP;
```

B.3. Archivo de entrada LOGMIP (versión 2) para el ejemplo 4.4.5.

```
$TITLE APPLICATION OF THE LOGIC-BASED MINLP ALGORITHM IN EXAMPLE #3
* FOR THIS PROBLEM THE FORMULATION IS HYBRID
$OFFSYMXREF
$OFFSYMLIST
*  SELECT OPTIMAL PROCESS FROM WITHIN GIVEN SUPERSTRUCTURE.

SETS   I      PROCESS STREAMS                / 1*25 /
       J      PROCESS UNITS                  / 1*9 /
       K  DISJUNCTIONS                       / 1*3  /
       IN(J,I)  / 1.2  , 2.3  , 3.6  , 4.8 , 5.11
                  6.12 , 7.17 , 8.18 , 9.21         /
       OU(J,I)  / 1.4  , 2.5  , 3.7  , 4.10 , 5.13
                  6.14 , 7.19 , 8.20 , 9.22         / ;
 PARAMETERS CV(I)     VARIABLE COST COEFF FOR PROCESS STREAMS       /
                            1 = 2 , 15 = -60 , 22 = -55 , 23 = -25 /

 S(J)  EQUIPMENT SIZE COEFFICIENT
  / 1   0.7
    2   0.6
    3   0.9
    4   1.5
    5   1.5
    6   1.6
    7   2.0
    8   1.9
    9   2.1  /;

 SCALAR     EP         / 1e-2 /;

TABLE AL(J,K)  VARIABLE COST COEFF FOR INVESTMENT COST
         1      2      3
    1   4.0    5.5    6.0
    2   2.0    2.7    3.5
    3   2.0    3.0    4.0
    4   3.0    4.5    6.0
    5   2.0    3.0    4.0
    6   4.0    5.0    6.0
    7   2.0    3.0    4.0
    8   2.2    3.5    4.3
    9   3.0    3.5    4.0

 TABLE BE(J,K)  FIXED COST FOR INVESTMENT COST
         1      2      3
    1   2.0    3.0    4.0
    2   3.0    4.0    5.0
    3   1.0    2.0    3.0
    4   2.0    3.0    4.0
    5   1.0    1.5    2.0
    6   2.5    3.5    4.5
    7   1.0    1.5    2.0
    8   1.5    2.0    2.5
    9   1.0    2.0    3.0
```

```
 TABLE LB(J,K)  LOWER BOUNDS FOR SIZE INTERVALS
        1      2      3
    1   0.0    0.6    1.0
    2   0.0    0.6    1.0
    3   0.0    0.5    1.2
    4   0.0    0.4    1.2
    5   0.0    0.7    1.3
    6   0.0    0.7    1.3
    7   0.0    0.5    1.1
    8   0.0    0.5    1.1
    9   0.0    0.5    1.2

 TABLE UB(J,K)  UPPER BOUNDS FOR SIZE INTERVALS
        1      2      3
    1   0.6    1.0    1.6
    2   0.6    1.0    1.6
    3   0.5    1.2    2.0
    4   0.4    1.2    1.8
    5   0.7    1.3    2.0
    6   0.7    1.3    2.0
    7   0.5    1.1    1.6
    8   0.5    1.1    1.6
    9   0.5    1.2    1.9  ;


 VARIABLES           X(I)  FLOWRATES OF PROCESS STREAMS
                     V(J)  SIZES OF THE UNIT
                     VD(J,K)
                     CF(J) FIXED COST FOR UNITS
                     CFD(J,K)
                     PROF  PROFIT ;
 BINARY VARIABLES    Y(J), Z(J,K) ;
 POSITIVE VARIABLES  X(I),V(J),VD(J,K),CF(J),CFD(J,K) ;



 EQUATIONS  C1(J,K), C2(J,K), C3(J,K), C4(J), C5(J),

* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* ---------------------------------------------------------
            MASSBAL1     MASS BALANCE #1
            MASSBAL2     MASS BALANCE #2
            MASSBAL3     MASS BALANCE #3
            MASSBAL4     MASS BALANCE #4
            MASSBAL5     MASS BALANCE #5
            MASSBAL6     MASS BALANCE #6
            MASSBAL7     MASS BALANCE #7
            MASSBAL8     MASS BALANCE #8

* EQUATIONS
* ---------------------------------------
            LOG1     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 1 EXISTS
            LOG2     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 2 EXISTS
            LOG3     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 3 EXISTS
            LOG4     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 4 EXISTS
            LOG5     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 5 EXISTS
            LOG6     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 6 EXISTS
            LOG7     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 7 EXISTS
            LOG8     CONSTRAINTS WHICH ALLOW FLOW IFF UNIT 8 EXISTS
```

```
            LOG9
            LOG10
            LOG11
            LOG12
            LOG13
            LOG14
            L1(J), L2(J)
            MBS1, MBS2, MBS3, MBS4,
            ION1(J), ION2(J), ION3(J), ION4(J)

* EQUATIONS :
* ------------------------------------
            INOUT11      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 1
            INOUT12
            INOUT13
            INOUT14
            INOUT15
            INOUT161
            INOUT162
            INOUT163
            INOUT21      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 2
            INOUT22
            INOUT23
            INOUT24
            INOUT25
            INOUT261
            INOUT262
            INOUT263
            INOUT31      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 3
            INOUT32
            INOUT33
            INOUT34
            INOUT35
            INOUT361
            INOUT362
            INOUT363
            INOUT41      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 4
            INOUT42
            INOUT43
            INOUT44
            INOUT45
            INOUT461
            INOUT462
            INOUT463
            INOUT51      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 5
            INOUT52
            INOUT53
            INOUT54
            INOUT55
            INOUT561
            INOUT562
            INOUT563
            INOUT61      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 6
            INOUT62
            INOUT63
            INOUT64
            INOUT65
            INOUT661
            INOUT662
```

```
            INOUT663
            INOUT71      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 7
            INOUT72
            INOUT73
            INOUT74
            INOUT75
            INOUT761
            INOUT762
            INOUT763
            INOUT81      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 8
            INOUT82
            INOUT83
            INOUT84
            INOUT85
            INOUT861
            INOUT862
            INOUT863
            INOUT91      INPUT-OUTPUT RELATIONS FOR PROCESS UNIT 9
            INOUT92
            INOUT93
            INOUT94
            INOUT95
            INOUT961
            INOUT962
            INOUT963
            OBJETIVO          OBJECTIVE FUNCTION DEFINITION ;

* INITIAL VALUES
* ---------------

INITIAL BU 2;

TRUE Y('2') Y('3') Y('4') Y('6') Y('8') Y('9');
TRUE Y('1') Y('3') Y('4') Y('5') Y('7') Y('9');

 OPTIONS LIMCOL = 0   ;
 OPTION LIMROW = 0    ;
 OPTION OPTCR = 0     ;

 OPTION MINLP = LOGMIPV2;

* EQUATIONS COMMON TO NLP SUBPROBLEMS AND MASTER PROBLEMS:
* -------------------------------------------------------
 MASSBAL1    ..   X('16')         =E=  X('17') + X('18')          ;
 MASSBAL2    ..   X('19')+X('20') =E=  X('21')                   ;
 MASSBAL3    ..   X('9')          =E=  X('16') + X('23')          ;
 MASSBAL4    ..   X('4') + X('5') =E=  X('6')                     ;
 MASSBAL5    ..   X('7')          =E=  X('8')  + X('9')           ;
 MASSBAL6    ..   X('10')         =E=  X('11') + X('12')          ;
 MASSBAL7    ..   X('13')+X('14') =E=  X('15')                    ;
 MASSBAL8    ..   X('1')          =E=  X('2') + X('3')            ;

 C1(J,K).. CFD(J,K)     =E= AL(J,K)*(VD(J,K)+EP)**0.6
                            + BE(J,K)*Z(J,K)-AL(J,K)*(EP)**0.6 ;
 C2(J,K).. VD(J,K)      =G= LB(J,K) * Z(J,K)            ;
 C3(J,K).. VD(J,K)      =L= UB(J,K) * Z(J,K)            ;
 C4(J)  .. V(J)         =E= SUM(K, VD(J,K))             ;
 C5(J)  .. CF(J)        =E= SUM(K, CFD(J,K))            ;
```

```
 MBS1   .. X('8')       =L= 5*X('9')                    ;
 MBS2   .. X('8')       =G= 1*X('9')                    ;
 MBS3   .. X('16')      =L= 5*X('9')                    ;
 MBS4   .. X('16')      =G= 1*X('9')                    ;

 LOG1               .. Y('1') + Y('2')          =L= 1          ;
 LOG2               .. Y('1') - Y('3')          =L= 0          ;
 LOG3               .. Y('2') - Y('3')          =L= 0          ;
 LOG4               .. Y('1') + Y('2') - Y('3') =G= 0          ;
 LOG5               .. Y('3') - Y('4')          =L= 0          ;
 LOG6               .. Y('3') - Y('7') - Y('8') =L= 0          ;
 LOG7               .. Y('4') - Y('5') - Y('6') =L= 0          ;
 LOG8               .. Y('5') + Y('6')          =L= 1          ;
 LOG9               .. Y('4') - Y('5')          =G= 0          ;
 LOG10              .. Y('4') - Y('6')          =G= 0          ;
 LOG11              .. Y('7') + Y('8')          =L= 1          ;
 LOG12              .. Y('7') - Y('9')          =L= 0          ;
 LOG13              .. Y('8') - Y('9')          =L= 0          ;
 LOG14              .. Y('7') + Y('8') - Y('9') =G= 0          ;

 L1(j)              .. (1-y(j))+sum(k,z(j,k)) =g=1   ;
 L2(j)              .. y(j)+ sum(k,(1-z(j,k)))=g=1   ;

 ION1(J).. SUM(I$IN(J,I), X(I)) =L= 10.0 * Y(J)         ;
 ION2(J).. SUM(I$OU(J,I), X(I)) =L= 10.0 * Y(J)         ;
 ION3(J).. V(J)         =L=  5.0 * Y(J)                 ;
 ION4(J).. CF(J)        =L= 15.0 * Y(J)                 ;

* EQUATIONS FOR THE NLP SUBPROBLEMS ONLY:
* ---------------------------------------
IF (Y('1'))THEN
 INOUT11..   X('4') =E= 1.1*LOG(1+X('2'))          ;
 INOUT14..   V('1') =E= S('1')*X('4')              ;
ELSE
 INOUT12..   X('2') =E= 0                          ;
 INOUT13..   X('4') =E= 0                          ;
 INOUT15..   V('1') =E= 0                    ;
 INOUT161..   VD('1','1') =E=0                ;
 INOUT162..   VD('1','2') =E=0                ;
 INOUT163..   VD('1','3') =E=0                ;
ENDIF

IF (Y('2')) THEN
 INOUT21..   X('5') =E= 1.2*LOG(1+X('3'))          ;
 INOUT24..   V('2') =E= S('2')*X('5')              ;
ELSE
 INOUT22..   X('3') =E= 0                          ;
 INOUT23..   X('5') =E= 0                          ;
 INOUT25..   V('2') =E= 0                    ;
 INOUT261..   VD('2','1') =E=0                ;
 INOUT262..   VD('2','2') =E=0                ;
 INOUT263..   VD('2','3') =E=0                ;
ENDIF

IF(Y('3')) THEN
 INOUT31..   X('7') =E= 1.4*LOG(1+X('6'))          ;
 INOUT34..   V('3') =E= S('3')*X('7')              ;
ELSE
 INOUT32..   X('6') =E= 0                          ;
```

```
 INOUT33..   X('7') =E= 0                          ;
 INOUT35..   V('3') =E= 0                    ;
 INOUT361..   VD('3','1') =E=0                    ;
 INOUT362..   VD('3','2') =E=0                    ;
 INOUT363..   VD('3','3') =E=0                    ;
ENDIF

IF(Y('4')) THEN
 INOUT41..   X('10') =E= 1.3*LOG(1+X('8'))         ;
 INOUT44..   V('4') =E= S('4')*X('10')             ;
ELSE
 INOUT42..   X('8') =E= 0                          ;
 INOUT43..   X('10') =E= 0                         ;
 INOUT45..   V('4') =E= 0                         ;
 INOUT461..   VD('4','1') =E=0                    ;
 INOUT462..   VD('4','2') =E=0                    ;
 INOUT463..   VD('4','3') =E=0                    ;
ENDIF

IF(Y('5'))THEN
 INOUT51..   X('13') =E= 1.2*LOG(1+X('11'))        ;
 INOUT54..   V('5') =E= S('5')*X('13')             ;
ELSE
 INOUT52..   X('13') =E= 0               ;
 INOUT53..   X('11') =E= 0               ;
 INOUT55..   V('5') =E= 0                ;
 INOUT561..   VD('5','1') =E=0           ;
 INOUT562..   VD('5','2') =E=0           ;
 INOUT563..   VD('5','3') =E=0           ;
ENDIF

IF(Y('6')) THEN
 INOUT61 ..   X('14') =E= 1.2*LOG(1+X('12'))     ;
 INOUT64 ..   V('6') =E= S('6')*X('14')          ;
ELSE
 INOUT62 ..   X('12') =E= 0                       ;
 INOUT63 ..   X('14') =E= 0                       ;
 INOUT65 ..   V('6') =E= 0                        ;
 INOUT661..   VD('6','1') =E=0                    ;
 INOUT662..   VD('6','2') =E=0                    ;
 INOUT663..   VD('6','3') =E=0                    ;
ENDIF

IF(Y('7'))THEN
 INOUT71 ..   X('19') =E= 1.0*LOG(1+X('17'))        ;
 INOUT74 ..   V('7') =E= S('7')*X('19')             ;
ELSE
 INOUT72 ..   X('19') =E= 0                         ;
 INOUT73 ..   X('17') =E= 0                         ;
 INOUT75 ..   V('7') =E= 0                        ;
 INOUT761..   VD('7','1') =E=0                    ;
 INOUT762..   VD('7','2') =E=0                    ;
 INOUT763..   VD('7','3') =E=0                    ;
ENDIF

IF(Y('8'))THEN
 INOUT81..    X('20') =E= 1.2*LOG(1+X('18'))       ;
 INOUT84 ..   V('8') =E= S('8')*X('20')             ;
ELSE
```

```
 INOUT82..   X('20') =E= 0                        ;
 INOUT83..   X('18') =E= 0                        ;
 INOUT85..   V('8') =E= 0                         ;
 INOUT861..   VD('8','1') =E=0                    ;
 INOUT862..   VD('8','2') =E=0                    ;
 INOUT863..   VD('8','3') =E=0                    ;
ENDIF

IF(Y('9'))THEN
 INOUT91..    X('22') =E= 1.1*LOG(1+X('21'))    ;
 INOUT94 ..   V('9') =E= S('9')*X('22')         ;
ELSE
 INOUT92..   X('22') =E= 0          ;
 INOUT93..   X('21') =E= 0          ;
 INOUT95..   V('9') =E= 0           ;
 INOUT961..   VD('9','1') =E=0      ;
 INOUT962..   VD('9','2') =E=0      ;
 INOUT963..   VD('9','3') =E=0      ;
ENDIF
*  BOUNDS SECTION
 X.LO(I)                  =   0       ;
 X.UP(I)                  =  25       ;
 X.LO('15')               =   0.4     ;
 V.UP(J)     = SMAX(K, UB(J,K)) ;

 X.UP('1')  =   25      ;
 X.UP('2')  = X.UP('1') ;
 X.UP('4')  = X.UP('1') ;
 X.UP('3')  = X.UP('1') ;
 X.UP('5')  = X.UP('1') ;
 X.UP('15') =   10      ;
 X.UP('11') = X.UP('15');
 X.UP('13') = X.UP('15');
 X.UP('12') = X.UP('15');
 X.UP('14') = X.UP('15');
 X.L('16') =  X.L('17') + X.L('18') ;
 X.L('21') =  X.L('19') + X.L('20') ;
 X.L('9')  =  X.L('16') + X.L('23') ;
 X.L('6')  =  X.L('4')  + X.L('5')  ;
 X.L('7')  =  X.L('8')  + X.L('9')  ;
 X.L('10') =  X.L('11') + X.L('12') ;
 X.L('15') =  X.L('13') + X.L('14') ;
 X.L('1')  =  X.L('2')  + X.L('3')  ;


*  INITIAL POINT
 X.L('2')   =  2.5      ;
 X.L('3')   =  2.5      ;
 X.L('13')  =  1.0      ;
 X.L('14')  =  1.0      ;
 V.L(J)     =  SMAX(K, LB(J,K)) +0.1 ;
 Z.L(J,K)     =   0    ;
 Z.L('1','1') =   0    ;
 Z.L('2','2') =   1    ;
 Z.L('3','2') =   1    ;
 Z.L('4','2') =   1    ;
 Z.L('5','1') =   0    ;
 Z.L('6','2') =   1    ;
 Z.L('7','2') =   0    ;
```

```
Z.L('8','2') =   1    ;
Z.L('9','2') =   1    ;
VD.L(J,K)    = V.L(J)*Z.L(J,K);

OBJETIVO .. PROF =E= SUM(J, CF(J))+SUM(I, X(I)*CV(I))  ;


MODEL LOGIC /ALL/;
LOGIC.optfile=1;
SOLVE LOGIC USING MINLP MINIMIZING PROF;
```