Ingeniería Investigación y Tecnología, ISSN 2594 0732, I.1, 1-7, 1998
(artículo arbitrado)
DOI: http://dx.doi.org/10.22201/fi.25940732e.1998.01.001

Programación lineal mixta-lógica

M. A. Osorio Lama y J. N. Hooker
Facultad de Ciencias de la Computación
Benemérita Universidad Autónoma de Puebla y
Graduate School of Industrial Administration
Carnegie Mellon University
Email:aosorio@solarium.cs.buap.mx

(recibido: noviembre, 1997; aceptado: enero, 1998)

Resumen

Se muestran las ventajas que tiene el enfoque de la Programación Lineal Mixta-Lógica (PLML) sobre el enfoque de programación lineal mixta-entera tradicional en la solución de problemas de optimación que tienen elementos tanto discretos como continuos. La superioridad de la PLML sobre enfoques convencionales es mostrada mediante su aplicación en ejemplos de problemas de síntesis de procesos, calendarización de trabajos con transferencia cero, localización de almacenes y el "problema de la fiesta progresiva".

Abstract

The advantages of the Mixed logical/linear programming (MLLP) against traditional approaches to solve optimization problems that have both discrete and continues elements are shown by examples. The superiority of the MLLP's broader framework against the conventional ones are shown by the solution of processes synthesis problems, flow shop scheduling problems, warehouse location problems and the "progressive party problem".

Introducción

Los problemas mixtos con variables discretas y continuas se pueden concebir tradicionalmente como problemas continuos en los cuales algunas de las variables están restringidas a ser enteras. La Programación Lineal Mixta-Lógica (PLML) tiene un punto de vista completamente diferente. En vez de ajustarse a los aspectos discretos de un problema mixto-entero, mismos que en ocasiones tienen que ser artificialmente fijados, representa a los elementos discretos por medio de fórmulas lógicas y los continuos, por desigualdades lineales. Por lo tanto, tiene la opción de prescindir de las variables enteras a la hora de resolver el problema y, en lugar de requerir que una solución factible satisfaga un conjunto fijo de desigualdades, provee varios conjuntos alternativos de desigualdades, donde el papel de las fórmulas lógicas consiste en demostrar cuáles alternativas son aceptables. A su vez, puede afirmarse que todo problema expresado en forma de programa lineal mixto-lógico es equivalente a un problema de programación disyuntiva cuyo conjunto de restricciones es de sistemas lineales (*i e,* la solución debe satisfacer al menos a uno de los sistemas). Su conjunto factible es, por lo tanto, la unión de muchos poliedros finitos en el espacio de variables continuas.

El objetivo de la investigación cuyos resultados son aquí presentados, fue el de explorar la PLML como un enfoque general y práctico para resolver los problemas con elementos continuos y discretos. A su vez, se utilizaron los trabajos anteriores en el área, en un esfuerzo por exponer ordenadamente las ideas, unas antiguas y otras nuevas, asociadas con la PLML, en un esquema coherente donde se exponen y clarifican las ventajas potenciales de este enfoque, a través de la presentación de resultados computacionales obtenidos en diferentes campos de aplicación.

Antecedentes

Aunque la programación disyuntiva (Balas, 1979) puede ser considerada como el antecedente teórico más importante de la PLML, ya que sus problemas comparten las mismas propiedades matemáticas básicas

DOI: http://dx.doi.org/10.22201/fi.25940732e.1998.01.001

y se puede mostrar que cualquier modelo de PLML puede ser expresado como un modelo de programación disyuntiva, la PLML incluye diferentes formas de inferencia simbólica (Hooker,1998 a,b,c), así como de generación y expresión de planos de corte, además de cortes lógicos que no se han utilizado en aquélla; esto hace de la PLML una herramienta diferente y en ocasiones mucho más poderosa.

Cabe mencionar que mientras otros enfoques contemporáneos que utilizan la lógica como herramienta, como es el caso de la programación restringida y de algunos otros enfoques que se sitúan en el área de la inteligencia artificial, utilizan modelos basados en procedimientos implantados en PROLOG (McAloon y Tretkoff, 1995). La PLML se mantiene como un enfoque de programación matemática basado en modelos declarativos. Además, en algunos casos donde la programación lineal mixta-entera ha mostrado ser insuficiente para resolver ciertos problemas con modelos muy complejos, la PLML, gracias a la ayuda del procesamiento lógico, ha encontrado soluciones que sólo habían podido ser encontradas con procedimientos de inteligencia artificial.

Forma General

Un modelo de PLML tiene la forma siguiente:

$$\begin{aligned} &\min cx \\ \text{s.a. } &g_i(y,h),\ i \in I \\ &y_j \rightarrow (A^j x \geq a^j),\ j \in J \end{aligned} \qquad (1)$$

El modelo tiene una parte lógica y una parte continua. La parte lógica consiste en fórmulas $g_i(y,h)$ que involucran proposiciones atómicas $y=(y_1,\ldots,y_n)$, las cuales son verdaderas o falsas. Cada fórmula puede ser $y_1 \cup y_2$, lo cual significa que y_1 o y_2 (o ambas) deben ser veraderas. También debe haber algunas variables $h=(h_1,\ldots,h_m)$ que tomen varios valores discretos. La parte continua asocia cada proposición atómica con un sistema $A^j x \geq a^j$ de desigualdades lineales. El sistema se fuerza cuando y es verdadero. Así, la fórmula $y_1 \cup y_2$ requiere de cualquier solución x que satisfaga $A^1 x \geq a^1$ o $A^2 x \geq a^2$ (o ambas). En general, x es factible si satisface a los sistemas lineales forzados por al menos un valor (y,h) que cumpla las fórmulas lógicas.

El problema puede ser resuelto en un árbol de búsqueda de ramificación y acotamiento, extendiendo sobre los valores verdaderos de las y_j los valores discretos de las h_j. En cada nodo del árbol de búsqueda se resuelve un problema de programación lineal que contiene las restricciones que corresponden a las y_j que son veradaderas, más que cualquiera de los cortes agregados para reforzar la relajación. Un elemento clave de la PLML consiste en aplicar un algoritmo de inferencia lógica a las fórmulas lógicas antes de resolver el problema lineal. Esto puede generar futuras restricciones lógicas y puede fijar algunas y_j o h_j adicionales.

Es fácil mostrar que cualquier problema de PLML es equivalente a un problema de programación disyuntiva cuyo conjunto de restricciones es una alternativa de sistemas lineales (*i e*, la solución debe satisfacer al menos a uno de los sistemas). Su conjunto factible es, por lo tanto, una unión finita de muchos poliedros en el espacio de las variables continuas.

Aplicaciones

La finalidad de los experimentos computacionales descritos es mostrar el desempeño de diferentes herramientas de la PLML mediante su aplicación a problemas diversos, así como compararlo con el de los algoritmos más simples y el de programación lineal mixta-entera.

Enseguida se dan algunos datos y explicaciones sobre la forma en que fueron aplicadas las herramientas mencionadas. La regla de ramificación consiste en seleccionar la primera variable proposicional en la primera fórmula lógica no satisfecha. El algoritmo de procesamiento lógico es la resolución unitaria. La relajación de las fórmulas utilizadas varía en cada caso. El código se escribió en lenguaje C y se empleó en una estación de trabajo SPARC 330 con sistema SUN OS. Las relajaciones lineales de los problemas en cada nodo del árbol de búsqueda se resolvieron utilizando las rutinas de la librería de CPLEX. El algoritmo de programación lineal mixta-entera utilizado para efectuar comparaciones es el método clásico de ramificación y acotamiento que también utiliza la librería de CPLEX para resolver la relajación lineal en cada nodo del árbol.

Problemas en síntesis de procesos

En la tabla 1 se presentan resultados experimentales para cada uno de los dos problemas de cinco y seis componentes estudiados por Raman y Grossmann (1993) . Los problemas son resueltos primero mediante PLML usando sólo la regla disyuntiva en la ramificación, sin ninguna relajación de las restricciones disyuntivas. La siguiente columna ilustra el costo de generar cortes. Las siguientes tres columnas comparan el algoritmo de PLML con los de programación lineal mixta-entera y CPLEX. En la siguiente columna se adicionan cortes lógicos y se procesan en forma simbólica por medio de resolución. En las últimas tres columnas se agregan los cortes lógicos y se utilizan variables binarias en los modelos mixto-entero y CPLEX, así como sus relajaciones en el modelo mixto-lógico.

DOI: http://dx.doi.org/10.22201/fi.25940732e.1998.01.001

Tabla 1. Número de nodos y tiempo de CPU para redes de separación

Método:	Mixta-lógica *Sin cortes	Mixta-lógica *Cortes duales	Mixta-lógica *Cortes elementales	Mixta-entera	CPLEX	Mixta-lógica *Cortes elementales *Cortes lógicos *Resolución unitaria	Mixta-lógica *Cortes elementales *Cortes lógicos *Relajaciones lógicas	Mixta-entera	CPLEX
No. de Nodos									
5 componentes	61	21	15	17	11	9	3	3	7
*Rest. a 4 unidades			15	49	29	13	3	3	4
6 componentes	1 659	105	97	191	94	63	97	33	40
*Rest. a 5 unidades			9	163	56	5	3	3	15
Tiempo de CPU (s)									
5 componentes	0.91	3.39	0.41	0.31	0.33	0.35	0.4	0.18	0.4
*Rest. a 4 unidades			0.45	1.01	0.82	0.52	0.42	0.23	0.28
6 componentes	33.3	26.5	2.3	5.6	3.5	2.6	8.1	3.3	3.5
*Rest. a 5 unidades			0.8	5.9	2	0.6	0.9	0.4	1.4

Los experimentos de Raman y Grossmann (1993) proporcionan una indicación similar para el preprocesador de OSL. Los cortes lógicos reducen el número de nodos para el programa lineal mixto-lógico. La comparación de los métodos agrupados sugiere que la formulación tradicional 0-1 es al menos tan efectiva como la formulación lineal mixta-lógica.

Sin embargo, el uso de variables proposicionales es particularmente ventajoso cuando se agregan variables semicontinuas al modelo. Es ineficiente representar semicontinuidad con variables enteras por dos razones: la relajación continua, como cualquier relajación lineal de semicontinuidad, es muy ineficiente y la representación 0-1 no es integral.

La síntesis de una red de reactores, una de 10 y otra de 38 procesos, descritas por Sahinidis y Grossmann (1987) fueron resueltas. Los resultados se muestran en la tabla 2. El problema de 10 procesos tiene tres variables semicontinuas y el de 38 procesos, siete. Diferentes versiones del problema se obtuvieron variando el tiempo en el horizonte y los límites de los intervalos.

Los resultados muestran que una representación lógica general de la semicontinuidad reduce ampliamente el tiempo de cálculo, aun cuando el número de variables expresadas en forma semicontinua no es muy grande comparado con el número de variables discretas del problema. Un enfoque razonable para estos problemas puede, por lo tanto, usar una representación 0-1 tradicional de las variables discretas que representan los procesos involucrados y una representación basada en la lógica de la semicontinuidad. La PLML proporciona esta clase de flexibilidad. En esa tabla puede observarse la mejora en los resultados, columnas mixta-lógica *vs* la mixta-entera, para todos los casos.

Calendarización de trabajos con transferencia cero

Los problemas de calendarización de trabajos con transferencia cero ilustran dos ventajas de la PLML: a) puede resolver problemas con árboles de búsqueda mucho más pequeños que la programación lineal mixta-entera, y b) el tiempo de procesamiento en cada nodo es menor, ya que la eliminación de variables enteras hace que las relajaciones lineales por resolver en cada nodo sean más pequeñas. Si existen m trabajos y n máquinas, la PLML reduce el número de variables en relajación lineal de $2m + mn$ a sólo $2m$.

Más aún, el modelo de programación lineal mixta-entera crea siempre un árbol más grande, ya que su relajación continua no es integral.

En la tabla 3 se presentan resultados para tres ejemplos de calendarización de trabajos con tranferencia cero en una planta química (Sahinidis y Grossmann, 1987). La PLML genera cerca de 60% de los nodos de la programación lineal mixta-entera y usa menos de la mitad del tiempo por nodo. Por lo tanto, corre de tres a cuatro veces más rápido que la programación mixta-entera para estos problemas.

DOI: http://dx.doi.org/10.22201/fi.25940732e.1998.01.001

Tabla 2. Número de nodos y tiempo de CPU para síntesis de redes de reactores

Problema	Número de nodos			Tiempo de CPU (s)		
	Mixta-lógica	Mixta-entera	CPLEX	Mixta-lógica	Mixta-entera	CPLEX
10 procesos, versión 1	5	29	24	0.24	0.82	0.65
10 procesos, versión 2	13	35	52	0.41	0.88	1.47
38 procesos, versión 1	729	1 083	677	199	376	178
38 procesos, versión 2	1 907	3 237	868	559	1 173	271
38 procesos, versión 3	1 161	1 999	345	306	836	104
38 procesos, versión 4	1 901	2 861	747	514	1 093	229
38 procesos, versión 5	1 081	1 561	296	287	551	89

Tabla 3. Número de nodos, tiempo de CPU y tiempo/nodo para problemas de calendarización de trabajos con transferencia cero

Número de		Lineal mixta-lógica			Lineal mixta-entera			CPLEX		
Trabajos	Máquinas	Nodos	Tiempo (s)	Tiempo/nodo (s)	Nodos	Tiempo (s)	Tiempo/nodo (s)	Nodos	Tiempo (s)	Tiempo/nodo (s)
6	5	407	2.7	0.0066	689	10.1	0.0147	527	8.1	0.0154
7	5	1 951	15.7	0.0080	3 171	52.2	0.0165	2 647	51.0	0.0193
8	5	14 573	129.0	0.0089	24 181	546.4	0.0226	16 591	413.9	0.0249

Problemas de localización de almacenes

Los problemas de localización de almacenes ilustran la generación de cortes lógicos de una restricción tipo "maleta". La mejora real consiste en la utilización de los cortes lógicos.

Se resolvieron siete problemas de localización de almacenes con capacidades grandes de demanda (Beasley,1988); los resultados aparecen en la tabla 4. Cada problema tiene 50 puntos de demanda con un total de 58 268. Cada almacen tiene la misma capacidad; la relación total sobre la demanda se muestra en la columna 3.

Los problemas 1 a 7 de la tabla 4 fueron resueltos para probar la hipótesis de que los cortes contiguos tienen mayor efecto en problemas que están altamente restringidos, como se puede observar por la relación de la capacidad total de almacenamiento con la demanda total. Los problemas son idénticos, excepto por la capacidad de los almacenes. Existen siete puntos de demanda con valores de cuatro a diez. Los resultados obtenidos tendieron favorablemente a confirmar la hipótesis.

El problema de la fiesta progresiva

La PLML tiene ventajas sustanciales en la modelación y la obtención de resultados computacionales para este problema. La ventaja computacional más importante consiste en que del gran número de variables discretas necesarias en el problema, pocas de ellas corresponden a restricciones lineales. El programa lineal mixto-lógico puede procesarlas simbólicamente sin agrandar la relajación lineal en cada nodo del árbol de búsqueda, la cual contiene sólo unas cuantas restricciones por nodo. Los resultados computacionales son los presentados en la tabla 5.

Como puede observarse, el tiempo de CPU por nodo permanece casi constante para la PLML, mientras que crece considerablemente para el caso de la programación lineal mixta-entera. Esto se explica claramente con los tamaños de los modelos utilizados en cada caso. También, es interesante notar que para los casos de 10 yates, tres y cuatro periodos de tiempo, CPLEX no pudo siquiera encontrar una solución entera factible para el problema. Este problema muestra muy claramente las ventajas del entorno de la PLML sobre los métodos tradicionales para resolver este tipo de problemas.

DOI: http://dx.doi.org/10.22201/fi.25940732e.1998.01.001

Tabla 4. Número de nodos, tiempo de CPU y tiempo de CPU por nodo para el problema de localización de almacenes

Problema	No. de		Nodos			Tiempo de CPU (s)			Tiempo de CPU por nodo (s)		
	almacenes	Cap/Dem	Mixta-Log.	Mixta-Ent.	CPLEX	Mixta-Log.	Mixta-Ent.	CPLEX	Mixta-Log.	Mixta-Ent.	CPLEX
CA P41	16	1.37	57	81	62	8.6	8.8	5.5	0.15	0.11	0.09
CA P42	16	1.29	59	81	57	8.9	8.6	5.5	0.15	0.11	0.10
CA P43	16	1.29	61	83	42	9.1	8.9	4.4	0.15	0.11	0.10
CA P44	16	1.37	43	61	40	7.1	6.8	4.3	0.17	0.11	0.11
CA P51	16	2.75	1 239	1 429	1 134	172	135	92	0.14	0.09	0.08
CA P61	16	3.86	2 147	2 631	3 017	266	237	235	0.12	0.09	0.08
CA P71	16	16	3 481	4 495	8 830	409	398	658	0.12	0.09	0.07
1	10	6.12	61	147	31	1.21	0.70	0.57	0.020	0.015	0.018
2	10	5.1	63	45	25	1.34	0.67	0.52	0.021	0.015	0.021
3	10	4.08	71	73	59	1.54	1.14	1.17	0.022	0.016	0.020
4	10	3.06	49	173	138	1.11	2.61	2.50	0.023	0.015	0.018
5	10	2.04	19	31	27	0.45	0.45	0.55	0.024	0.015	0.020
6	10	1.02	3	21	20	0.16	0.30	0.32	0.053	0.014	0.016

Tabla 5. Número de nodos, tiempo de CPU y tiempo de CPU por nodo para el problema de la fiesta progresiva

No. de	Periodos de	Nodos		Tiempo de CPU (s)		Tiempo de CPU por nodo (s)	
yates	tiempo	Mixta-lógica	CPLEX	Mixta-lógica	CPLEX	Mixta-lógica	CPLEX
5	2	171	1 136	0.98	197	0.006	0.173
6	2	239	5 433	1.41	1 708	0.006	0.314
6	3	37	366	0.25	162.8	0.007	0.445
7	3	71	44	0.52	44.6	0.007	1.014
8	3	209	2 307	1.63	3113	0.008	1.349
8	4	167	582	1.35	887.5	0.008	1.525
10	3	24 142	*20 000	12 259	*54 150	0.011	2.708
10	4	28 923	*20 000	319.4	*43 223	0.011	2.161

Conclusiones

La PLML ha mostrado muchas ventajas en los problemas de síntesis de procesos en ingeniería química y los resultados computacionales para estos problemas confirman los trabajos publicados previamente. Con respecto a los demás casos explorados, algunos de los problemas fueron de creación propia (algunos problemas de localización de almacenes); otros se tomaron de algunas bases de datos (problemas de localización de almacenes con capacidades grandes) o se seleccionaron de la literatura en el área (calendarización de trabajos con transferencia cero y síntesis de procesos). En otros casos, se cambiaron algunas de las especificaciones (como utilizar variables semicontinuas en las redes de proceso).

Después de haber aplicado el conjunto de herramientas de la PLML, se puede concluir la importancia y el impacto de las ventajas de ésta sobre la programación lineal mixta-entera. Las ventajas potenciales del amplio entorno de herramientas y posibilidades que componen la PLML son evidentes.

DOI: http://dx.doi.org/10.22201/fi.25940732e.1998.01.001

Separación de la regla de ramificación y la relajación del problema

En la programación lineal mixta-entera las variables enteras sirven para fijar la regla de ramificación, designando alguna de ellas que contenga un valor fraccional en la solución relajada en el método de ramificación y acotamiento, que a su vez, sirva para relajar el problema, removiendo la restricción de integralidad en las variables. Sin embargo, existen otros esquemas de relajación y ramificación. La PLML permite el uso de cualquier esquema de ramificación en combinación con cualquiera de relajación porque ésta no necesita involucrar a las variables discretas.

Modelos de programación lineal más pequeños

Algunos problemas combinatorios están formulados con un gran número de variables enteras. La programación lineal mixta-entera permite a estas variables permanecer en los problemas lineales resueltos en cada nodo del árbol de ramificación y acotamiento, mientras que el esquema de la PLML incluye sólo las variables continuas en los problemas lineales.

Procesamiento más eficiente de las variables discretas

Las variables lógicas pueden ser fijadas frecuentemente en cada nodo mediante la aplicación de procedimientos de inferencia en las restricciones lógicas. Pero en estos casos, el procesamiento lógico es mucho más rápido, ya que puede ser realizado por medio de un procedimiento de resolución unitaria lineal en tiempo.

Las soluciones factibles se identifican más pronto

La relajación continua convencional de una formulación disyuntiva 0-1 puede tener soluciones fraccionales, aun cuando la disyunción esté satisfecha. Esto significa que la programación lineal mixta-entera puede seguir generando ramas en el árbol de ramificación y acotamiento, incluso cuando se haya encontrado una solución factible. La PLML evita este defecto y puede, por lo tanto, producir un árbol de búsqueda más pequeño.

Relajaciones más fuertes

La PLML puede proporcionar relajaciones más fuertes de dos maneras:

1. Algunas veces, los cortes válidos para las restricciones disyuntivas pueden ser más fuertes que la relajación continua de cualquier formulación obvia 0-1.
2. El procesamiento lógico puede generar restricciones lógicas (cortes lógicos) cuyas relajaciones lineales pueden ser agregadas al modelo lineal. A su vez, los mismos cortes lógicos pueden ser usados como planos en un algoritmo convencional de ramificación y acotamiento, pero el procesamiento lógico provee una forma sistemática de generarlos.

Un enfoque alternativo para identificar cortes

El punto de vista lógico puede sugerir cortes lógicos que pueden derivarse fácilmente de un entendimiento intuitivo del problema, pero que no son fácilmente revelados por el análisis poliédrico convencional.

Modelación más natural

Las variables enteras, particularmente 0-1, intentan expresar lo que originalmente fue concebido como restricciones lógicas. En dichos casos, el enfoque de la PLML permite una formulación más natural.

El hábito de escribir modelos pensando en la conveniencia del paquete computacional utilizado para su solución es común en investigación de operaciones y puede llevarnos a olvidar que el principal objetivo de la modelación en la ciencia es el explicativo. El proceso de modelación debería mejorar el entendimiento del problema y no supeditarse a la necesidad de escribir formulaciones compatibles con los paquetes de cómputo utilizados para resolverlos. De otra manera, si el modelo no es totalmente adecuado para la solución de un problema, debería formularse, ya sea por un procedimiento automático o por un experto humano. En algunos casos puede ser difícil relacionar la solución con la formulación original. Con la PLML, uno se puede situar a la mitad del camino, ya que permite que la modelación discreta-continua sea más natural, eliminando la necesidad de variables enteras e introduciendo restricciones lógicas y variables multivalentes, pero el enfoque de la solución está todavía relacionado con el problema original porque efectúa las ramificaciones en las variables proposicionales que aparecen en él. Generalmente, el modelo requiere una preparación adicional antes de su solución, pero la preparación involucra la adición de cortes y no la reformulación del mismo.

Finalmente, dado que la PLML es un enfoque general para resolver problemas mixtos-continuos/discretos comprende un conjunto de herramientas lógicas que pueden ser utilizadas de distinta manera para diferentes problemas; que, como en la programación lineal mixta-entera, su efectividad en ocasiones, depende de qué tan cuidadosamente se diseñan las relajaciones, los cortes, los

DOI: http://dx.doi.org/10.22201/fi.25940732e.1998.01.001

esquemas de ramificación y las ecuaciones lógicas para resolver cada problema específico.

Se ha proporcionado un amplio rango de opciones y mostrar cómo la mayoría de estas opciones son superiores a los métodos tradicionales para los problemas ejemplificados.

Referencias

Balas E. (1979). Disjunctive Programming. *Annals Discrete Mathematics 5*. 3-51.

Beasley J. E. (1988) An Algorithm for Solving Large Capacitated Warehous Location Problems. *European Journal of Operational Research 3*. 314-325.

Hooker J. N. (1988) (a). Resolution vs Cutting Plane Solution of Inference Problems: Some Computational Experience. *Operations Research Letters 7*. 1-7.

Hooker J. N. (1988)(b). Generalized Resolution and Cutting Planes. *Annals of Operations Reserach 12*. 217-239.

Hooker J. N. (1988)(c). A Quantitative Approach to Logical Inference. *Decision Support Systems 4*. 45-69.

Mc Aloon K. y Tretkooff C. (1995). *Linear Programming and Logic Programming*. P. Van Hentenryck y V. Saraswat.

Raman R. y Grossmann I. E. (1993). Symbolic Integration of Logic in Mixed-integer Linear Programming Techniques for Process Synthesis. *Computers and Chemical Engineering 17*. 909-927.

Raman R. y Grossmann I. E. (1994). Modeling and Computational Techniques for Logic Based Integer Programming. *Computer and Chemical Engineering 18*. 563-578.

Sahinidis N. V y Grossmann I. E. (1987). Multiperiod Investment Model for Processing Networks with Dedicated and Flexible Plants. *Computers and Chemical Engineering 30*. 1165-1171.

Semblanza de los autores

María Auxilio Osorio Lama. Obtuvo el grado de doctora en ingeniería en la Facultad de Ingeniería UNAM, junto con el premio al mejor desempeño (1996). Ingeniera química industrial con maestría en sistemas y planeación, así como ingeniera en investigación de operaciones, se ha dedicado fundamentalmente a la docencia y a la investigación; actualmente realiza una estancia de investigación en la Universidad de Carnegie Mellon trabajando con el profesor John Hooker con quien publicará un artículo sobre la programación lineal mixta-lógica para el número especial sobre métodos booleanos de la publicación Discrete Applied Mathematics. Se desempeña como investigadora de la Facultad de Ciencias de la Computación en la Universidad Autónoma de Puebla.

John N. Hooker. Profesor de la Escuela de Graduados de Administración Industrial de la Universidad de Carnegie Mellon desde 1985. Ha participado en numerosas conferencias internacionales en investigación de operaciones como organizador y ponente invitado. Fue presidente de la Sección Técnica de Computación del INFORMS (95-96), y es miembro del Consejo de Asesores en Investigación de Operaciones de la Kluwer Academic Publishers. Cuenta con más de 30 artículos en el área, en las revistas más importantes, y con capítulos en libros de programación matemática e inteligencia artificial, así como textos en preparación sobre métodos de optimación y lógica. Recibió el premio al mejor artículo en la Interface Investigación de Operaciones/Ciencias de la Computación en 1991. Fue editor del Management Science (1989-1994) y desde 1989 es editor del Journal on Computing, así como del Journal Heuristics, desde 1994.