METODOLOGÍA ALTERNATIVA PARA EL TRATAMIENTO DE PROBLEMAS DE PROGRAMACIÓN ENTERA USANDO PROGRAMACIÓN NO LINEAL

RESUMEN

Este trabajo propone una alternativa de solución a los problemas de Programación Entera, dejando de lado la idea de la enumeración implícita y sus consecuentes dificultades a la vez que aprovecha información matemática de la función objetivo y de las funciones que determinan el espacio de soluciones. Tal alternativa de solución es posible gracias a una sencilla transformación de las restricciones de integralidad usando un polinomio que permite tratar el problema con Programación No Lineal (PNL).

PALABRAS CLAVE: Programación No Lineal, Programación Entera.

DIEGO A. MEJÍA GIRALDO
Ingeniero Electricista, M.Sc.
Profesor Aspirante
Universidad de Antioquia
diegomej@udea.edu.co

ALEXANDER MOLINA C.
Ingeniero Electricista, MSc.
Profesor Auxiliar
Universidad Tecnológica de Pereira
almo@utp.edu.co

FRANCISCO F. FRANCO A.
Ingeniero Electricista
Profesor Transitorio
Universidad Tecnológica de Pereira
ffranco@ohm.utp.edu.co

ABSTRACT

This paper shows an alternative methodology for Integer Problems which avoids implicit enumeration and its difficulties; it also takes advantage of the mathematical information from the objective function and the functions that determine the solutions space. Such alternative is possible due to a transformation of the integrality constraints using a polynomial function that allows to solve the problem as a non-linear optimization one problem.

KEYWORDS: *Nonlinear Programming, Integer Programming.*

1. INTRODUCCIÓN

La programación matemática es una herramienta de gran utilidad en diferentes ramas de la ciencia dado que a través de ella se pueden optimizar diferentes procesos que se presentan en la realidad, con la previa definición de un criterio de desempeño denominado función objetivo. En tales procesos se debe establecer claramente cuáles han de ser las variables de decisión y establecer el conjunto al cual éstas deben pertenecer una vez el proceso de optimización haya culminado. Estos conjuntos pueden ser convexos o no convexos, acotados o no acotados, continuos o discretos. Sobre estos últimos recae el interés de este trabajo, donde las variables están sujetas a ser enteras. En estos casos los problemas de optimización se denominan de Programación Entera. [1].

Los problemas cuyas variables solo pueden asumir valores enteros representan una familia importante. Algunos de ellos pueden ser los problemas que incluyen como variables el número de unidades a fabricar en una empresa, el número de fábricas a instalar en un país, el número de vehículos necesarios para atender entrega de pedidos, entre otros [2].

Para resolver los anteriores problemas, de manera que se encuentre los óptimos globales, se han usado técnicas de programación matemática de planos de corte y enumeración implícita como Branch and bound, entre otros. Por otro lado métodos como Danzig-Wolfe o descomposición de Benders son usados para solucionar problemas que incluyen variables enteras y continuas denominados problemas de Programación Entera Mixta (PEM) [3]. Estos métodos, dado que efectúan búsquedas exhaustivas sufren dificultades con los tiempos computacionales empleados y la memoria usada en el proceso de búsqueda debido la naturaleza combinatorial de los mismos [2].

Para superar las dificultades descritas anteriormente en los problemas de PE, se han usado técnicas heurísticas que a pesar de rechazar la posibilidad de encontrar óptimos globales encuentra varias soluciones de muy buena calidad. La búsqueda efectuada por este tipo de algoritmos heurísticos no usa información matemática de las restricciones del problema. Por otro lado, éstas mismas técnicas sólo usan la función objetivo para evaluar las soluciones que se encuentran en el proceso y guardar las mejores respuestas [4].

Este trabajo propone una alternativa de solución a los problemas de PE que abandona la idea de la enumeración implícita y sus consecuentes dificultades, a la vez que aprovecha información matemática de la función objetivo y de las funciones que determinan el espacio de soluciones. Tal alternativa de solución es posible gracias a una sencilla transformación de las restricciones de integralidad usando un polinomio que permite tratar el problema con Programación No Lineal (PNL).

El artículo trata en la segunda sección, generalidades de la Programación No lineal. En la tercera sección se hace una breve descripción de los problemas de programación entera así como de la estructura y la manera de

Fecha de Recepción: 09 Febrero de 2007
Fecha de Aceptación: 28 Marzo de 2007

solucionarlos. En la cuarta sección se menciona la metodología y se describe analítica y de manera gráfica la transformación aplicada para el tratamiento de los problemas de PE.

2. PROGRAMACIÓN NO LINEAL (PNL)

El campo de la Programación No Lineal ha tenido un drástico desarrollo en los últimos tiempos gracias al avance de las capacidades de los computadores modernos que son capaces de efectuar complejos cálculos matemáticos en poco tiempo. Tales avances han permitido resolver problemas de PL de una manera un tanto diferente debido al tratamiento novedoso que se le da a las restricciones. Por otro lado, el uso de métodos iterativos que hacen búsquedas irrestrictas locales son los potenciadores del uso de la PNL [5].

La programación no lineal generalmente resuelve problemas cuya estructura es como se muestra en la ecuación (1).

$$\begin{aligned} &\min \quad f(X) \\ &\text{s.a} \quad g_i(X) \le 0, \forall i = 1,....,m \\ &\qquad h_j(X) = 0, \forall j = 1,....,l \\ &\text{Donde, } X = \left[x_1\ x_2.....x_n\right]^T \end{aligned} \tag{1}$$

Donde tanto $f(X)$, como $g_i(X)$ y $h_j(X)$ generalmente son no lineales y forman conjuntos, a través de igualdades y desigualdades, que no siempre son convexos. Un ejemplo de tal tipo de problemas se presenta gráficamente en la figura 1. Aquí las curvas punteadas circulares representan la función objetivo de características no lineales. Las líneas curvas continuas representan las restricciones que dan forma al conjunto de posibles soluciones. Tal conjunto es usualmente llamado región factible.

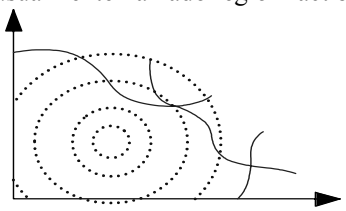

Figura1. Representación de un problema de PNL

Las metodologías de solución usadas en PNL son muy variadas. Es preciso decir que existen metodologías para problemas irrestrictos, los cuales en muchos casos constituyen una base para problemas restrictos, como se menciona en párrafos anteriores [2]. Entre los algoritmos empleados para problemas irrestrictos se encuentran:

- Búsqueda Unidimensional (en una variable)
- Método del Gradiente
- Método de Newton y sus variaciones
- Método de las Direcciones Conjugadas
- Métodos Quasi-Newton y métodos no derivativos

En estos el espacio de búsqueda es ilimitado tal como se muestra en la Figura 2.

Figura 2. Espacio de búsqueda PNL Irrestricto

De la misma forma existen algoritmos para resolver problemas restrictos. En estos, el espacio de búsqueda es acotado. Algunos de estos algoritmos son:

- Método de las Direcciones Factibles
- Método de Proyección de Gradiente
- Métodos de suboptimización múltiple
- Métodos de Penalidad y Barrera
- Método del Lagrangeano Aumentado
- Programación Cuadrática Secuencial

2.1. Método de Penalidad.

El método de penalidad convierte un problema restricto en un problema irrestricto equivalente, haciendo uso de una función que penaliza la violación de las restricciones. El valor entregado por la función de penalización se agrega a la función objetivo del problema original generando un problema equivalente [5]. La transformación mencionada anteriormente se muestra en la ecuación 2.

$$\begin{aligned} &\min \quad f(X) \\ &\text{s.a} \quad g_i(X) \le 0, \forall i = 1,....,m \\ &\qquad h_j(X) = 0, \forall j = 1,....,l \\ &\qquad\qquad \Downarrow \\ &\min \quad f(X) + \mu P(X) \\ &\text{s.a} \quad X \in \Re^n \\ &P(X) \text{ Función de Penalidad} \end{aligned} \tag{2}$$

Es de la anterior manera como se enfrenta problemas de PNL restrictos usando métodos irrestrictos. El valor de μ que acompaña la función de penalidad aumenta a medida que avanza el proceso iterativo. Tal estrategia permite que el problema avance por el óptimo de cada subproblema (problema para cada valor de μ) hasta que la penalización sea tan alta que se consigue la factibilidad. [5].

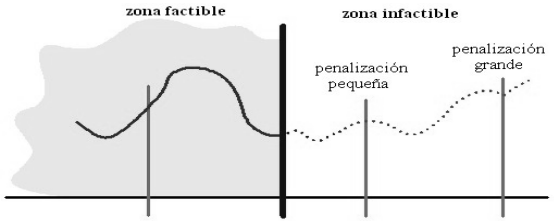

Figura 3. Penalización de las restricciones

Una forma de construir $P(X)$ es la siguiente:

$$P(X) = \sum \langle \Phi_i (g_i(X)) \rangle + \sum \langle \Psi_j (h_j(X)) \rangle \quad (3)$$

donde $\Phi_i, \Psi_j : \Re \to \Re$ continuas y diferenciables.

La idea principal es penalizar a un punto X e incluir éste valor dentro de la función objetivo siempre y cuando se viole una igualdad o una desigualdad, de la siguiente forma [5]:

Φ_i Penaliza cada desigualdad cuando $g_i(X) > 0$

Ψ_j Penaliza cada igualdad cuando $h_j(X) \neq 0$

El valor que cada uno de estos parámetros toma, se obtiene siguiendo las estrategias mostradas continuación:

- Estrategia para Φ_i :

$$\text{Condiciones} = \begin{cases} \Phi(g_i(X)) > 0 \text{ y creciente, si } g_i(X) > 0 \\ \Phi(g_i(X)) = 0, \text{ si } g_i(X) \leq 0 \end{cases}$$

Una función que satisface los requisitos puede ser

$$\Phi(g_i(X)) = \left(\max\{0, g_i(X)\}\right)^k, \; k \geq 1 \quad (4)$$

- Estrategia para Ψ_j :

$$\text{Condiciones} = \begin{cases} \Psi(h_j(X)) > 0 \text{ y creciente, si } h_j(X) \neq 0 \\ \Psi(h_j(X)) = 0, \text{ si } h_j(X) = 0 \end{cases}$$

La penalización de las restricciones de igualdad puede ser de la forma que se presenta en la ecuación (5).

$$\Psi(h_j(X)) = \left|h_j(X)\right|^k, \; k \geq 1 \quad (5)$$

En este trabajo se usaó k=2 en 4 y 5 para garantizar diferenciabilidad.

Algoritmo:

datos $X^0, \varepsilon, \mu_0 > 0, \beta > 1$, Hacer $\tau = 0$

Mientras $\mu_\tau P(X^\tau) > \varepsilon$

partiendo de X^τ encontrar

$X^{\tau+1} = \arg\min\{F(X) = f(X) + \mu_\tau P(X)\}$

$\mu_{\tau+1} = \beta \mu_\tau$

$\tau = \tau + 1$

Fin

$X^* = X^\tau$

En el anterior algoritmo X^0 es el punto de partida (no tiene que ser factible), ε es el criterio de parada, μ_0 es el nivel de penalización inicial y β es la tasa de crecimiento de la penalización.

Esta metodología requiere resolver en cada iteración un problema de optimización irrestricta por cada valor de μ (o por cada iteración) y así encontrar el nuevo valor $X^{(\tau+1)}$ de las variables de decisión. El éxito de la metodología radica en el desempeño del optimizador usado. Este debe garantizar la optimalidad para el parámetro μ de una determinada iteración de manera que cada nuevo punto sirva de partida para el siguiente problema que tiene mayor nivel de penalidad.

Según experimentaciones previas en cuanto a la comparación del desempeño de los optimizadotes irrestrictos, se observa que el método de Newton y el del gradiente conjugado son los que ofrecen mejor desempeño. Sin embargo, éste último requiere mayor número de iteraciones pero no de cálculo de segundas derivadas ni de inversión de matrices. Por estas razones el método del gradiente conjugado mostrado en [5] se implementa con apoyo del método de búsqueda secuencial para la búsqueda unidimensional.

Finalmente, iteración tras iteración, se va logrando que cada sub-óptimo (óptimo de cada iteración, encontrado por el optimizador irrestricto) se aproxime al óptimo (o al menos a un óptimo local) del problema original irrestricto. Cuando el producto μ^*P sea menor que el criterio de parada, se puede decir que se ha alcanzado la factibilidad y que el modelo ha convergido a un mínimo local.

3. PROGRAMACIÓN ENTERA

La Programación Entera es la rama de la optimización matemática que permite hacer optimización en problemas donde las variables de decisión son de naturaleza entera. Cuando las variables asumen valores enteros y continuos los problemas reciben el nombre de problemas de Programación Entera Mixta. Aplicaciones como el problema de la asignación generalizada, el planeamiento de sistemas de transmisión de energía eléctrica, entre otros, forman parte de esta familia de problemas.

La representación matemática de los problemas de PE se ilustra en la ecuación (6).

$$\begin{aligned} &\min \quad f(X) \\ &\text{s.a} \\ &g_i(X) \leq 0, \; \forall\, i = 1, \ldots\ldots, m \\ &X \in Z \\ &\text{donde } X = [x_1 \; x_2 \; \ldots\ldots x_n]^T \end{aligned} \quad (6)$$

Un caso de problemas de programación entera de dos variables se presenta en la figura 4. En la misma figura es posible apreciar las restricciones de tipo lineal que forman un conjunto convexo.

Estos problemas tienen una complejidad mayor que la de los problemas de PL. Tal complejidad radica en dos situaciones a saber. Una de ellas es la imposibilidad de encontrar el óptimo en alguno de los vértices del conjunto convexo formado por las restricciones. La anterior condición evita la aplicación de las metodologías de PL para encontrar la solución de manera directa. La

segunda situación; el conjunto de soluciones posibles es discreto y finito, en contra de los problemas de PL cuyo espacio de solución es infinito, pero tal discretización hace que el número de soluciones tenga un crecimiento de tipo combinatorial con respecto a las variables.

Una prueba de lo anterior se puede observar en la figura 4. El número de vértices del conjunto generado por las rectas es de 5, así, si se tratara de un problema de PL el método simplex encontraría el vértice óptimo de manera rápida. Ahora, considerando el problema de PE se tendrían 13 posibles soluciones señaladas por los puntos en el interior del poliedro convexo generado por las rectas.

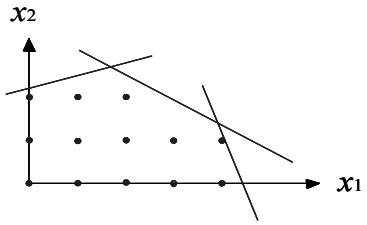

Figura 4 Espacio de soluciones en Programación Entera.

Las situaciones antes planteadas obligan a que la búsqueda de la solución global se deba realizar usando métodos de enumeración o dividiendo el problema en una serie de subproblemas lineales de manera que en algún momento alguna solución coincida con un punto entero de inmejorable función objetivo.

Las metodologías exactas usadas para los problemas de PE han sido tradicionalmente

- Descomposición de Benders
- Branch and Bound (B&B)
- Cortes de Gomory
- Enumeración implícita de Balas

Estas se caracterizan por exigir grandes capacidades de memoria, requerir tiempos grandes (debido a la gran cantidad de subproblemas de PL) y ser de difícil implementación computacional.

Las técnicas heurísticas (la mayoría de estas se inspiran en el comportamiento de fenómenos físicos y biológicos) tienen ventajas que las técnicas anteriores no presentan, con el inconveniente de que es imposible garantizar la convergencia hacia el óptimo global del problema. De todas formas estas convergen hacia puntos de buena calidad en términos de función objetivo

Entre las heurísticas y metaheurísticas se tiene:

- Algoritmos genéticos.
- Búsqueda Tabú.
- Simulated Annealing.
- Colonia de Hormigas.
- GRASP.

4. ESTRATEGIA EMPLEADA

Como se mencionó en las secciones 2 y 3, es posible resolver los problemas de PE usando técnicas exactas ó técnicas heurísticas. El uso de técnicas exactas apunta a la búsqueda de la solución global a través de la enumeración o del uso de planos de corte ya sea global o sucesivo. Esto implica tiempos altos para la evaluación de soluciones así como grandes capacidades de memoria. Por otro lado las técnicas heurísticas, aunque encuentran buenas respuestas, realizan una búsqueda que es preciso orientar a través de la variación de los parámetros de funcionamiento de la misma. Tal ajuste de parámetros es posible solo en casos donde el investigador conoce a profundidad tanto la técnica de búsqueda como el problema a resolver.

En resumen, los métodos de enumeración solamente consideran la función objetivo para establecer direcciones de búsqueda en los subproblemas relajados y no en el problema global, además, las restricciones son consideradas solamente si son violadas y rara vez se camina por la infactibilidad. A su vez, las técnicas heurísticas prescinden de toda información de la función objetivo y de las restricciones. Se hace simplemente una búsqueda seudoaleatoria controlada.

Para tratar de solucionar algunos de los anteriores problemas presentados se propone una alternativa donde se obliga a que las variables, en la respuesta definitiva del problema, tomen valores enteros y que la búsqueda se realice de una manera más justificada matemáticamente sin los inconvenientes de la enumeración aprovechando la información del problema en sí (curvas de nivel de la función objetivo, conjunto de factibilidad). Es decir, el problema original de PE o de PEM se transforma, resultando en uno de PNL.

Tal alternativa plantea que las restricciones de integralidad de las variables, en problemas de Programación Entera, se pueden transformar en restricciones de tipo no lineal. Más exactamente se propone transformar las restricciones de integralidad en polinomios de un grado igual al número de valores enteros que ha de tomar cada variable.

Se plantea entonces que si se tiene un problema que obligue a sus variables a tomar un valor entero máximo *p,* tal como aparece en (7), es posible generar un polinomio *h(x)* que tome la forma presentada en (8).

$$x \in \{0,1,2,\ldots, p\} \quad (7)$$

$$h(x_i)=(x_i-0)(x_i-1)(x_i-2)\ldots(x_i-p) =0 \quad (8)$$

Para luego obligar a que h_i sea igual a cero, de manera que se cumpla la restricción del problema original. La representación se muestra en la figura 6 y se nota que

cuando el valor de h_i se hace cero se tiene el problema original.

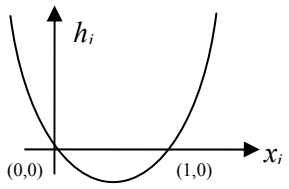

Figura 6. Representación de la transformación.

La transformación así propuesta tiene tal alcance que se puede usar la misma formulación para valores no enteros y discretos, es decir que siempre es posible plantear un polinomio cuyas raíces pueden tomar valores discretos arbitrarios.

La otra ventaja de tal formulación es que se pueden resolver problemas de PEM sin necesidad de complicar la programación del método iterativo. Basta con usar el algoritmo de PNL y trabajar con tantas restricciones como se quiera. Por otro lado, el hecho que se use PNL hace que el método pueda ser aplicado a problemas cuya función objetivo puede ser no lineal así como sus restricciones.

Por tanto, para un problema de PE como se muestra en (9) es necesario su transformación en uno no lineal como se muestra en (10).

$$\begin{aligned} &\min \quad f(X) \\ &\text{s.a} \\ &g_i(X) \le 0, \quad \forall\, i = 1,.....,m \\ &x_j \in \left\{0,1,2,...,p_j\right\} \\ &\forall\, j = 1,....n \end{aligned} \tag{9}$$

$$\begin{aligned} &\min \quad f(X) \\ &\text{s.a} \\ &g_i(X) \le 0, \quad \forall\ i = 1,.....,m \\ &h_j(x_j) = (x_j)(x_j - 1)...(x_j - p_j) = 0 \\ &\forall\ j = 1,....n \quad \text{con} \quad x \in \Re^n \end{aligned} \tag{10}$$

Nótese que en (10) la restricción de integralidad del problema que aparece en (9) se reemplaza por los polinomios $h_j(x_j)$. Este nuevo problema es resuelto usando el método de Penalidad mostrado en la sección 2.1.

De esta manera se cuenta con el problema planteado de manera No-Lineal que junto con la técnica planteada es posible ser solucionado. En la sección 5 se muestra mediante un ejemplo, la aplicabilidad de la metodología.

5. PROBLEMA DE LA MOCHILA

Existen una amplia variedad de problemas que obligan a la variables a tomar valores enteros, entre ellos está el clásico "problema del morral" o "problema de la mochila".

Tal problema se enuncia de la siguiente manera: Una persona planea un campamento. Dada la exigencia de la actividad y su fuerza, en su morral es capaz de cargar hasta b kilos. Según sus necesidades, ha escogido n tipos de alimentos candidatos para llevar $A_1,\ldots,A_n$. Cada alimento A_i viene en paquetes indivisibles con peso p_i. Según el contenido nutricional cada alimento tiene un beneficio c_i. La formulación matemática se presenta en (11).

$$\begin{aligned} &\max \ \sum c_i x_i \\ &\text{s.a.} \ \ \sum p_i x_i \le b \\ &\qquad x_i \in [0\ 1] \end{aligned} \tag{11}$$

Este es un problema combinatorial en donde a medida que el número de productos aumenta, el número de alternativas o espacio de solución crece exponencialmente. Esta característica conlleva a usar casi siempre técnicas heurísticas que resuelvan parcial o totalmente el problema. Con la propuesta presentada en las secciones anteriores, el problema de la mochila se puede plantear como:

$$\begin{aligned} &\max f(\mathbf{x}) = \sum_{i=1}^{n} c_i x_i \\ &\text{s.a.} \\ &\qquad g(\mathbf{x}) = \sum_{i=1}^{n} p_i x_i - b \le 0 \\ &\qquad h_i(\mathbf{x}) = x_i(x_i - 1) = 0 \ \ \forall\, i = 1,\ldots,n \end{aligned} \tag{12}$$

Este modelo es resuelto con el método de penalidad y usando las funciones de penalización mostradas anteriormente. Se resolvieron dos problemas, el primero de 6 productos y el segundo de 20. En ambos casos se utilizó un β=10 y ε=0.001 (criterio de parada).

En el Tabla 1 se observa la información correspondiente al modelo mostrado en (12) para el problema de la mochila de 6 productos con b=12. En la fila de x_i^* se muestra la solución óptima del problema. Esta respuesta ha sido comprobada mediante el software de optimización GAMS.

c_i	6	4	6	10	5	8
a_i	6	3	5	2	1	6
x_i^*	0	1	0	1	1	1

Tabla 1. Datos para el problema de 20 productos

Para encontrar esta misma solución el algoritmo de penalidad tomó 8 iteraciones y un tiempo computacional de 1.8s. En la figura 7 se presenta el comportamiento del algoritmo cuando se propone como solución inicial x_i=1, $\forall i$=1,.., 6. Allí se nota que al inicio del proceso prima la optimalidad (ver comportamiento de $f(\mathbf{x})$) a pesar de que la solución en esos instantes viole las restricciones. A medida que el nivel de penalidad se incrementa la función $P(\boldsymbol{X})$ tiende a cero, es decir, se alcanza

factibilidad (se cumple con la disponibilidad de peso de la mochila y las variables de decisión finales son binarias) y la calidad de la función objetivo disminuye. Cuando el algoritmo ha convergido $F(\boldsymbol{X})$ se aproxima a $f(\boldsymbol{X})$=27, es decir, $\mu P(\boldsymbol{X})\approx 0$.

La respuesta encontrada por el algoritmo planteado es insensible ante la solución inicial propuesta, sin embargo, el tiempo computacional si resulta ser afectado.

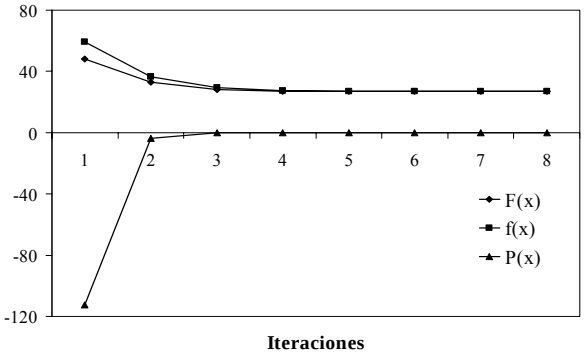

Figura 7. Convergencia del problema de 6 productos

Vale la pena mencionar que la respuesta obtenida por la metodología aquí planteada tiene un componente diferente de cero en el octavo o noveno decimal, lo anterior dado a que ésta metodología usa un proceso iterativo que tiene un criterio de parada. Cuanto menor sea este criterio de parada menor será la cifra decimal obtenida. Aún cuando se presenta ésta situación es posible mencionar que tal algoritmo funciona de manera satisfactoria.

Otro elemento a considerar es la sencilla manera en la que tal metodología se implementa, dejando de lado la gran cantidad de reglas y condiciones que han de ser implementadas en los algoritmos de descomposición y enumeración. Por otro lado, ésta metodología no requiere almacenar incumbentes, por lo que para problemas de gran tamaño no es indispensable capacidades altas de memoria, contrario a los requerimientos en otras técnicas.

6. CONCLUSIONES

Se presentó una metodología alterna para resolver problemas de PE basada en una transformación matemática del problema original. El algoritmo presentado para resolver el nuevo problema de PNL muestra ser eficiente cuando es apoyado por un minimizador irrestricto de buen desempeño como el gradiente conjugado. Inclusive muestra que el tiempo computacional empleado no es elevado aún cuando se tratan problemas de gran tamaño. Sin embargo, los algoritmos de PNL no garantizan la convergencia hacia el optimizador global, sino que resultan en un punto que satisface condiciones de primer orden de Karush Kuhn Tucker, es decir, optimizadores locales. Se puede afirmar, que problemas de pocas variables convergen rápidamente al optimizador global.

Es interesante mencionar que la propuesta es válida solo para variables acotadas o inclusive para variables que pueden tomar cualquier valor arbitrario discreto. Teniendo en cuenta esta característica se puede afirmar que el modelo propuesto altamente aplicable en problemas de optimización en ingeniería y/o como estrategia que se puede fusionar con otro tipo de metodologías que intervienen en dichos problemas. En ese sentido, por ejemplo, este modelo puede entregar soluciones de alta calidad a algoritmos de intensa búsqueda local para tratar de refinar y mejorar la solución y a la vez disminuyendo esfuerzo computacional.

7. AGRADECIMIENTOS

A la Universidad Tecnológica de Pereira y al grupo de investigación GIMEL de la Universidad de Antioquia por el valioso apoyo académico y logístico prestado.

8. BIBLIOGRAFÍA

[1] MEJIA Diego, GIRALDO Dídier, MOLINA Alexander. "*Control Optimo Usando Redes Neuronales Dinámicas*". Tesis de grado. Universidad Tecnológica de Pereira. 2004.

[2] BERTSEKAS, Dimitri. "Nonlinear Programming". Second Edition. Athena Scientific Print. Belmont. 2003.

[3] GALLEGO Ramón, ESCOBAR Antonio y GRANADA Mauricio. "*Programación Entera y Métodos de Descomposición*". Grupo de Planeamiento de Sistemas Eléctricos. UTP.

[4] GALLEGO Ramón, ESCOBAR Antonio y ROMERO Rubén. "*Técnicas de Optimización Combinatorial*". Grupo de Planeamiento de Sistemas Eléctricos. UTP.

[5] MORA, Hector M. "Optimización No Lineal y Dinámica". Segunda Edición. Universidad Nacional de Colombia. 2001.

[6] OLIVEIRA, G.G. and SOARES, Secundino. *A Second Order Network Flow Algorithm for Hydrothermal Scheduling*. IEEE Transactions on Power Systems, vol 10, No. 3, pp. 1635-1641. 1995.

[7] CARNEIRO, A.A.F.M. and SOARES, Secundino. *A Large Scale Application of an Optimal Deterministic Hydrothermal Scheduling Algorithm*. IEEE Transactions on Power Systems, vol 10, No. 3, pp. 1635-1641. 1995.