Un algoritmo Stackelberg-Evolutivo para resolver el problema binivel de localización de plantas con preferencias de los clientes.

Fernando Camacho [1], Eduardo Cordero[1] and Rosa González[2]

[1] Facultad de Ciencias Físico-Matemáticas
Universidad Autónoma de Nuevo León
Pedro de Alba S/N, Ciudad Universitaria
San Nicolás de los Garza, Nuevo León, 66450, México
jose.camachovl@uanl.edu.mx, alvaro.corderofr@uanl.edu.mx

[2]Escuela de Ingeniería Industrial
Pontífica Universidad Católica de Valparaíso
Valparaíso, Chile
rosa.gonzalez@ucv.cl

Resumen: Nosotros estudiamos el problema de localización de plantas sin capacidades agregando la suposición de que los clientes van a establecer una lista a priori con sus preferencias para ser servidos por alguna de las plantas. Este problema se modela como un problema de programación binivel donde en el nivel superior, el líder minimiza el costo total de instalación (costo de apertura y distribución) decidiendo donde instalar las plantas considerando las preferencias de los clientes (seguidor) para abastecerse por alguna de ellas. En el nivel inferior, los clientes minimizan sus preferencias basados en una lista dada y ellos son libres de elegir la planta a la que se asignarán. Nosotros proponemos una reformulación del problema lineal binivel reduciéndolo a un problema de programación mixta entera. Además diseñamos e implementamos un algoritmo Stackelberg-Evolutivo para resolver el problema binivel; este algoritmo se valida con experimentaciones numéricas comparándose contra valores existentes en la literatura.

Palabras clave: programación binivel, problemas de localización de plantas, algoritmos evolutivos, equilibrio de Stackelberg.

Abstract. We study the bilevel uncapacitated facility location problem with the assumption that the customers are going to establish a list with their preferences to be served by a particular facility. This problem is modeled as a bilevel programming problem where in the upper level, the leader minimize the total cost of the facility (setup and distribution) deciding the locations of the facilities considering the preferences of the customers (followers). In the lower level, the customers minimize their preferences considering that they are allowed to select the facility is going to serve them. We propose a reformulation of the bilevel problem reducing it to a single level mixt-integer programming problem. Also we designed and implemented a Stackelberg-Evolutionary algorithm in order to solve the bilevel problem; the efficiency of the proposed algorithm is validated with numerical experimentations and the obtained results are compared against some values existing in the literature.

Keywords: bilevel programming, facility location problems, evolutionary heuristics, Stackelberg's equilibrium.

1. Introducción

Los problemas de localización de instalaciones han sido ampliamente estudiados en la literatura para desarrollar modelos, algoritmos y para análisis de complejidad. El problema consiste tanto en determinar los sitios para realizar una instalación, ya sea de plantas, almacenes y/o centros de distribución, como en asignar los clientes para que sean servidos por esas instalaciones y por último, identificar como están conectadas entre sí dichas instalaciones. El primer artículo en teoría de localización es el trabajo presentado por Weber quien consideró el problema de instalar una planta tal que se minimice el total de las distancias de viaje entre la planta y los clientes. Después de este trabajo muchos autores han prestado atención a este problema ampliando el rango de aplicaciones con muchas variaciones del modelo, algoritmos y aplicaciones.

El problema de localización de plantas sin capacidades es uno de los problemas más estudiados en Teoría de Localización, en donde se asume que las plantas que se van a instalar no tienen capacidad y son capaces de producir o servir lo que se requiera sin límite. En este trabajo nosotros retomamos este problema considerando una función objetivo del tipo *minisum*, pero le agregamos preferencias a los clientes sobre las plantas que los van a servir. Considerar las preferencias de los clientes con respecto a las plantas que lo van a servir es una característica muy importante debido a la competencia tan grande que existe actualmente en los mercados, es por esto, que tiene mucha coherencia tomar en cuenta las opiniones de los usuarios.

El primer problema que consideró las preferencias en los clientes fue propuesto por Hanjoul y Peeters (1987), quienes extendieron el Problema de Localización de Una Planta considerando las preferencias para llamarlo El Problema de Localización de Una Planta con Orden. También en Krarup y Pruzan (1983) y Gorbachevskaya y Kochetov (1992), se presentan resultados de complejidad de algunos casos polinomialmente resolvibles, así como algunas reducciones y reformulaciones del Problema de Localización de Plantas sin Capacidades con Preferencias de los Usuarios y en Hansen, Kochetov y Madlenovic (2004) se presenta una nueva formulación que contiene a las formulaciones previas y dieron algunos límites inferiores para la solución. En Cánovas, García, Labbé y Marín (2007) se muestran algunas desigualdades válidas para el SPLPO propuesto en Hanjoul y Peeters (1987) así como los resultados numéricos resultantes de haber experimentado con las reformulaciones y fortalecimiento del modelo.

En Ishii, Lee y Yeh (2007) se presentó una modelación difusa para el Problema de Localización de Plantas con preferencias de los sitios candidatos considerando el grado de satisfacción con respecto a la distancia desde la planta para cada cliente y la preferencia del sitio en un área urbana, donde la meta era optimizar dos criterios: encontrar el sitio que maximice el grado de satisfacción mínimo entre todos los puntos de demanda y que maximice la preferencia del sitio. En Vasil'ev, Klimentova y Kochetov (2009) se presentan nuevos límites inferiores para el problema introduciendo una familia de nuevas desigualdades válidas, mostrando que la formulación propuesta es más fuerte que las formulaciones previas con respecto a la relajación lineal y la holgura de integralidad. Después en Vasil'ev y Klimentova (2010), consideran este problema binivel e introducen unas desigualdades relacionadas con las preferencias y trabajan con ese problema de un sólo nivel. Relajan el problema y encuentran un límite inferior, luego proponen un heurístico de recocido simulado donde encuentran un límite superior. Después con esos límites aplican un algoritmo basado en ramificación y corte al problema de un sólo nivel con más restricciones, pero con el procesamiento dado por los valores obtenidos en los límites y encuentran muy buenos resultados numéricos.

En este trabajo nosotros reformulamos el problema original y propusimos una técnica eficiente para resolver el problema binivel directamente. La diferencia con los artículos existentes es que ellos proponen técnicas de solución de alguna versión del problema reformulado y nosotros creamos un buen algoritmo para encontrar soluciones óptimas o en algunos casos soluciones iniciales de buena calidad.

La organización del trabajo es la siguiente: la segunda sección presenta la formulación matemática del modelo y la reformulación del problema binivel aquí considerado. La tercera sección describe el algoritmo Stackelberg-Evolutivo. La cuarta sección muestra los resultados numéricos y una breve discusión sobre éstos. Por último, al final se presentan las conclusiones, recomendaciones y trabajo futuro.

2. Descripción del problema.

En esta sección vamos a describir el enunciado del problema y la formulación matemática del problema de localización de plantas sin capacidades añadiendo la restricción que los clientes van a establecer una lista ordenada de sus preferencias que indique cuales plantas desea que lo sirvan. En el nivel superior, el líder minimiza el costo total de localización (de apertura y de distribución) decidiendo donde abrir una planta considerando las preferencias que tienen los usuarios (seguidores) para usar alguna planta. En el nivel inferior, los clientes minimizan sus preferencias basados en una lista ordenada determinada a priori en donde ellos son libres de elegir la planta a la cual serán asignados.

Durante toda la investigación consideramos las siguientes suposiciones sin perder generalidad:

- Los clientes establecen de antemano sus preferencias con respecto a cada planta mediante una lista ordenada de números enteros del 1 al n donde 1 corresponde a la planta más preferida y n es la menos preferida.
- No se consideran restricciones de capacidad para las plantas.

Basado en esto, se desea contestar las siguientes preguntas: ¿Dónde se instalarán las plantas? y ¿Cuáles clientes va a ser servidos por cada planta?. El enunciado del problema puede verse como: decidir la localización de las plantas que minimicen el costo total tomando en cuenta las preferencias de los clientes.

2.1 Modelo Matemático

Sea i quien denote a las plantas y j a los clientes, donde $i \in I$ y $j \in J$. Los parámetros c_{ij} representan los costos de enviar toda la demanda del cliente j desde la planta i. Además, definamos f_i como el costo fijo de abrir la planta i. Cada cliente j tiene una preferencia p_{ij} de ser servido por la planta i, en donde un valor de $p_{ij} = 1$ corresponde al sitio más preferido. Las variables de decisión del problema binivel son las variables binarias x_{ij} que denotan si la planta i satisface la demanda del cliente j y las variables binarias y_i que representan si se abre o no la planta i.

El modelo matemático es el siguiente:

$$\min_{y,x} \sum_i \sum_j c_{ij} x_{ij} + \sum_i f_i y_i \qquad (1)$$

$$\text{sujeto a:} \quad y_i \in \{0,1\} \qquad \forall i \in I \qquad (2)$$

$$x \in \operatorname{Arg\,min} \sum_i \sum_j p_{ij} x_{ij} \qquad (3)$$

$$\text{sujeto a:} \quad \sum_i x_{ij} = 1 \qquad \forall j \in J \qquad (4)$$

$$x_{ij} \leq y_i \qquad \forall i \in I, j \in J \qquad (5)$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \qquad (6)$$

Este modelo fue propuesto en el trabajo de Vasil'ev, Klimentova y Kochetov (2009), es un problema de localización de plantas sin capacidades, es decir, una planta puede surtir tantos clientes como sea conveniente, pero los clientes se abastecen por una sola planta y los clientes brindan una lista ordenada con las preferencias que tiene de ser servidos por las plantas potenciales. En Ausiello, Crescenzi, Gambosi, et al (1999) se muestra que debido a la estructura de este problema se le considera como NP-Hard.

Vamos a reformular el problema para reducirlo a un solo nivel utilizando las relaciones primal-duales, pero para encontrar estas relaciones del problema (1)-(6) necesitamos que la restricción (6) sea remplazada por la restricción de no negatividad $x_{ij} \geq 0, \forall i \in I, j \in J$, es fácil apreciar que está estrechamente relacionada con la ecuación (5) y a su vez con la ecuación (4). No obstante, después de obtener dichas relaciones volveremos a considerar la ecuación (6) teniendo en cuenta sus implicaciones en la reformulación.

El problema reformulado queda como sigue:

$$\min_{y,x,\alpha,\beta} \sum_i \sum_j c_{ij} x_{ij} + \sum_i f_i y_i \quad (7)$$

$$\text{sujeto a:} \quad y_i \in \{0,1\} \quad \forall i \in I \quad (8)$$

$$\sum_i x_{ij} = 1 \quad \forall j \in J \quad (9)$$

$$x_{ij} \leq y_i \quad \forall i \in I, j \in J \quad (10)$$

$$\alpha_j + \beta_{ij} \leq p_{ij} \quad \forall i \in I, j \in J \quad (11)$$

$$\sum_i \sum_j p_{ij} x_{ij} = \sum_j \alpha_j + \sum_i \sum_j \beta_{ij} y_i \quad (12)$$

$$x_{ij} \in \{0,1\} \ \forall i \in I, j \in J \quad (13)$$

$$\beta_{ij} \leq 0 \quad \forall i \in I \quad (14)$$

Puede verse fácilmente que el problema descrito en (7)-(14) se obtiene tomando las variables del líder y_i como parámetros para el problema lineal del nivel inferior. A pesar de que el problema (7)-(14) es un problema de un solo nivel no es tan fácil de resolver debido a la restricción no lineal (12).

Ahora vamos a analizar la ecuación (12). Vamos a hacer el cambio de variable $\pi_{ij} = \beta_{ij} y_i$ asegurándonos de que cuando $y_i = 0$ entonces $\pi_{ij} = 0$; y si $y_i = 1$ entonces $\pi_{ij} = \beta_{ij}$. Esto se logra introduciendo las siguientes desigualdades:

$$\pi_{ij} \leq 0 \quad \forall i \in I, j \in J \quad (15)$$

$$\pi_{ij} \geq -\mathrm{M} y_i \quad \forall i \in I, j \in J \quad (16)$$

$$\pi_{ij} \geq \beta_{ij} \quad \forall i \in I, j \in J \quad (17)$$

$$\pi_{ij} + \mathrm{M} y_i \leq \beta_{ij} + \mathrm{M} \quad \forall i \in I, j \in J \quad (18)$$

y remplazando la ecuación (12) por la siguiente ecuación:

$$\sum_i \sum_j p_{ij} x_{ij} = \sum_j \alpha_j + \sum_i \sum_j \pi_{ij} \quad (19)$$

Lo que nos conduce a un problema de programación mixta-entera conformado por las ecuaciones (7)-(11) y (13)-(19).

Cabe mencionar que basados en la reducción a un solo nivel utilizando las relaciones primal-duales, de manera alternativa podemos sustituir la igualdad de las funciones objetivo de ambos problemas, esto es la ecuación (12), por la restricción de complementariedad de la forma:

$$x_{ij}\left(\alpha_j + \beta_{ij} - p_{ij}\right) = 0 \qquad \forall i \in I, j \in J \tag{20}$$

Aquí podemos linealizar la ecuación (20) con la siguiente expresión:

$$\alpha_j + \beta_{ij} - p_{ij} \geq -M(1 - x_{ij}) \qquad \forall i \in I, j \in J \tag{21}$$

entonces el problema de programación mixta entera estaría definido por (7)-(11), (13)-(14), (21).

A esta modelación se le puede añadir la consideración de un problema de la P-mediana, es decir, si consideramos en el nivel inferior que se deben de abrir un número P de plantas. La restricción que se debe añadir al problema es:

$$\sum_i y_i \geq P \tag{22}$$

Es claro que esto no afecta el análisis descrito anteriormente puesto que las relaciones primal-dual son del problema del nivel inferior y la restricción (2) se añade en el nivel superior. Entonces basta con resolver el problema definido por las ecuaciones (7)-(11), (13)-(19) considerando además la restricción (22).

3. Descripción del algoritmo Stackelberg-Evolutivo.

En esta sección vamos a presentar el algoritmo propuesto basado en los principios del equilibrio de Stackelberg bajo un esquema evolutivo. Proponemos utilizar un conjunto de soluciones para el líder como parámetros para el nivel inferior y resolver este problema con el fin de obtener la respuesta óptima. Después evaluamos la función objetivo del líder y comenzamos a mejorar las soluciones.

3.1. Equilibrio de Stackelberg

El concepto del Equilibrio de Stackelberg para teoría de juegos no cooperativa se propuso en Stackelberg (1952) es muy conocido y puede verse como un juego con dos participantes en un mercado donde hay una firma dominante que optimiza su propio criterio considerando la reacción de la firma dominada. Cuando la firma dominante optimiza su decisión considerando la mejor respuesta del seguidor se dice que se ha encontrado un equilibrio de Stackelberg. Este esquema es el mismo que se considera en programación binivel.

3.2 Programación Evolutiva

Los algoritmos basados en Programación Evolutiva (PE) tienen su fundamento en el comportamiento de los seres vivos en la naturaleza. Es por esto, que existen muchas variaciones de este tipo de algoritmos. El esquema general es el siguiente: existe un conjunto de individuos (soluciones) los cuales forman una población para una generación determinada. Después, se eligen dos individuos para hacer un cruzamiento, o bien, cada individuo puede sufrir una mutación. Estos cruces y mutaciones se realizan de manera aleatoria basados en un factor prestablecido lo que garantiza diversidad; aunque también puede producir abortos (soluciones infactibles, a las cuales habrá que desecharlas). Los individuos más fuertes (los que tengan mejor valor en la medida de desempeño que estemos considerando, esto es, mejor función objetivo) son los que sobreviven (basados en un criterio de selección) y permanecen en la siguiente generación. Este proceso continua hasta cumplir con un criterio de paro.

Los componentes principales del algoritmo son:

- Representación: se refiere a la manera en que se van a caracterizar los individuos de una población.
- Función de evaluación: se entenderá como la función objetivo del problema que se está optimizando.
- Población: se refiere a la forma en que se crearán los conjuntos de individuos y el tamaño de dicho conjunto.
- Operadores de variación: son aquellos que definen la forma en que se llevará a cabo el cruzamiento entre individuos y la forma de la mutación.
- Mecanismo de selección: debe especificar la forma en que se va a decidir cuales individuos son los que prevalecen para la siguiente generación. El proceso de selección debe asegurarse de mantener la diversidad de la población.

3.3. Descripción del algoritmo

En esta subsección se describirán detalladamente todos los componentes del algoritmo Stackelberg-Evolutivo aquí propuesto para resolver el problema binivel de localización de plantas sin capacidades con preferencias de los clientes.

Primero vamos a definir como se van a representar los individuos de la población. Un individuo y_k^t consiste en el $k-ésimo$ vector que indica las plantas que el líder ha decidido abrir en la generación t. La función de evaluación que usaremos está definida por

$$\varphi\left(y_i, x_{ij}(y_i)\right) = \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij}(y_i) + \sum_{i\in I} f_i y_i \qquad (23)$$

donde $x(y)$ es la respuesta óptima del seguidor después de resolver el problema del nivel inferior definido por (3)-(6) considerando la y como fija. Además, entiéndase por $\varphi_k^t\left(y_k^t, x(y_k^t)\right)$ como el valor de la función objetivo para el individuo k en la generación t cuando el líder toma decisión y_k^t y el seguidor optimiza encontrando $x(y_k^t)$.

Inicialización. $(t = 0)$ Se determina el tamaño n de la población inicial P^t. De forma aleatoria se generan n vectores factibles de localización de plantas, donde cada componente del vector y_k^t es un 0 o 1 elegido de forma aleatoria. Después, para todos los $k = 1, 2, \dots, n$ se minimiza la función objetivo del seguidor (las preferencias) utilizando los vectores y_k^t de la población, obteniendo así las asignaciones $x(y_k^t)$ correspondientes para cada y_k^t. Por último, para $k = 1, 2, \dots, n$ se evalúa la función objetivo del líder $\varphi_k^t\left(y_k^t, x(y_k^t)\right)$, donde se busca minimizar el costo de instalación y distribución.

Mutación. Para cada vector y_k^t se debe decidir aleatoriamente una de las siguientes tres opciones: a) mantener su número de plantas abiertas, es decir, de manera aleatoria a un elemento que sea 0 convertirlo en 1 y otro elemento que sea 1 convertirlo en 0; b) disminuir el número de plantas abiertas, esto es, únicamente elegir al azar un elemento que sea 1 y convertirlo en 0; c) aumentar el número de plantas abiertas, es decir, elegir aleatoriamente un elemento que sea 0 y convertirlo en 1. Con esto se generan n nuevos individuos que les llamaremos P_{new}^t, los cuales se añaden a P^t para conformar $P_{aum}^t = [P^t; P_{new}^t,]$. Es fácil ver que la población P_{aum}^t tiene $2n$ individuos. Luego, para cada nuevo individuo y_k^t, con $k = n+1, n+2, \dots, 2n$ se minimiza la función objetivo del seguidor, obteniendo así las asignaciones $x(y_k^t)$; posteriormente, se calcula la función objetivo del líder $\varphi_k^t\left(y_k^t, x(y_k^t)\right)$ correspondiente a para $k = n+1, n+2, \dots, 2n$.

Selección. Para cada vector y_k^t, $k = 1, 2, \ldots, 2n$, se elige de manera aleatoria otro vector y_l^t, donde $k \neq l$, y se realiza una comparación entre los valores de las respectivas funciones objetivo, si $\varphi_k^t \leq \varphi_l^t$; consideramos que φ_k^t ha tenido una victoria y lo guardamos en una función ganadora, sino le asignamos el triunfo a φ_l^t. A este proceso se le llama *torneo* y se realizará un número predeterminado de veces (Max_Tor); de modo que cada individuo $y_k^t \in P_{aum}^t$ irá acumulando victorias a lo largo de los torneos. Una vez concluidos los torneos, se obtendrán los n individuos que más victorias hayan obtenido y formarán la nueva población P^{t+1}. Después se regresa al paso de *mutación* para continuar el ciclo hasta que se completa el número de generaciones deseadas (Max_Gen), es decir, se continuará iterando mientras que $t \leq$ Max_Gen.

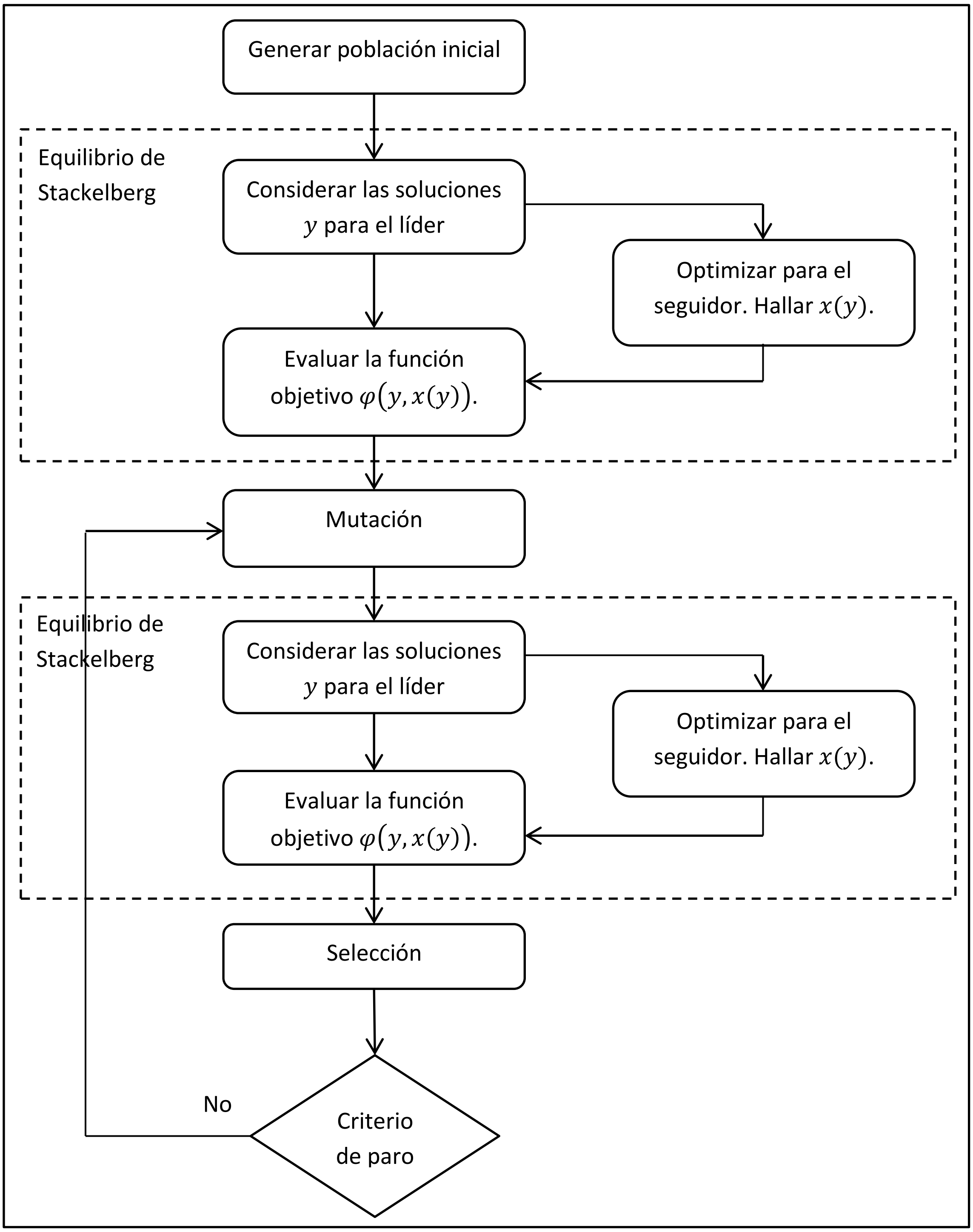

Figura 2. Diagrama del algoritmo Stackelberg-Evolutivo.

Es importante notar que en el algoritmo aquí propuesto no estamos considerando el cruzamiento entre dos individuos de una población. Solamente se considerarán mutaciones para que cada individuo evolucione hacia una mejor solución. Los valores predeterminados como el tamaño de la población (n), el número de generaciones buscadas (Max_Gen), el número de torneos (Max_Tor) y los valores para los tres casos posibles en la mutación están detallados en la siguiente sección.

4. Experimentación numérica

Esta sección podemos considerarla en dos partes: La primer parte consistió en generar instancias aleatorias de tamaño pequeño y mediano para realizar la experimentación inicial; en dicha experimentación inicial implementamos tres códigos en el software Matlab 7.9.0 utilizando el optimizador CPLEX 12.1, el primer código fue para resolver directamente la reformulación definida por (7)-(11), (13)-(19), el segundo código fue un algoritmo exhaustivo para obtener la solución óptima del problema binivel y el tercer código fue la implementación del algoritmo Stackelberg Evolutivo.

Es claro ver que dada la naturaleza del problema, un algoritmo exhaustivo consiste en explorar todas las soluciones posibles para el líder, optimizando el problema del nivel inferior para cada solución. El número de posibles soluciones para el líder está dado por $\sum_{j=1}^{|J|} {}_{|J|}C_j$, por lo que vemos que conforme el tamaño del problema crezca el número de posibles soluciones aumenta en una forma no polinomial. Es por esto que surge la necesidad de diseñar un algoritmo heurístico.

Aquí obtuvimos resultados satisfactorios en lo cuales la reducción del problema a un solo nivel obtiene la misma solución que el método exhaustivo y el algoritmo heurístico aquí propuesto. Es importante señalar que el tamaño máximo de problemas que resolvió exactamente la reformulación (7)-(11), (13)-(19) fueron de 25 plantas y 30 clientes.

Motivados por los resultados obtenidos hicimos la segunda parte de la experimentación numérica, la cual consideramos que es la más relevante. Para esto, conseguimos los parámetros de las instancias probadas en Cánovas, et al (2007) y realizamos las pruebas computacionales. Cabe mencionar que nosotros utilizamos una computadora personal HP con un procesador dual core corriendo a 2.8 GHz y con 2GB de RAM.

4.1 Descripción de las instancias y los parámetros propuestos.

Como ya mencionamos, en Cánovas, et al (2007) se describe cómo generaron las preferencias y manipularon los datos para generar las instancias para sus pruebas; estas instancias utilizadas para la experimentación fueron obtenidas parcialmente de la Beasley's OR-Library. Nosotros consideramos dos diferentes tamaños de problemas, estos problemas consisten en un problema de 50 plantas y 50 clientes (Instancia 1) y otro de 50 plantas con 75 clientes (Instancia 2).

Para cada una de las instancias nosotros consideramos 3 problemas diferentes y para cada problema se experimentó con 4 cambios en las preferencias dejando los demás parámetros (costos de distribución c_{ij} y costos de instalación f_i) de la misma manera ese problema. Esto quiere decir que nosotros consideramos 12 problemas para cada instancia.

4.2 Resultados numéricos.

Ahora vamos a presentar los resultados numéricos obtenidos. Los parámetros para el algoritmo Stackelberg-Evolutivo los fijamos tal que el tamaño de la población será $n = 100$, el número de generaciones será Max_Gen=100 y decidimos que el número de torneos sea Max_Tor=5, con el objetivo de mantener diversidad y de que es el número más utilizado en este tipo de algoritmos.

Estos valores resultaron ser los que mostraban mejor comportamiento después de una serie de experimentos con diferentes valores de entrada. Después realizamos 15 corridas para cada problema y presentamos el promedio de sus resultados, el promedio del tiempo requerido y el porcentaje de las veces que obtuvo el óptimo.

En las Tabla 1 y 2 se muestran los resultados obtenidos por nuestro algoritmo y las holguras existentes en la literatura. La columna "Problema" indica el problema que estamos experimentando, la columna "Valor Opt" representa el valor óptimo para el problema. La columna "Gap M1" es el porcentaje de holgura de optimalidad que llegaron en Cánovas et al (2007) por el Método 1, ese método consistía en utilizar un optimizador con el método simplex dual aplicado a la formulación original (No era una formulación binivel). La columna "Gap M2" representa el porcentaje de holgura al que llegaron en ese mismo trabajo después de aplicar un preprocesamiento a una reformulación del problema y aplicarle un algoritmo de ranch & bound. Las siguientes 4 columnas se refieren a nuestros resultados. Hay que recordar que son promedios de las 15 corridas realizadas. La columna "Gap" indica el valor de la holgura de optimalidad que obtuvimos, lo calculamos con la fórmula:

$$\%\text{Gap} = \left(\frac{\varphi_{SE} - \varphi_{opt}}{\varphi_{opt}}\right) \times 100\%$$

"Valor obtenido" significa el promedio de los valores que encontramos después de aplicar el algoritmo Stackelberg-Evolutivo, "% de opt" el porcentaje de precisión, es decir, de cuántas veces obtiene el óptimo del problema y "Tiempo" es el tiempo promedio (en segundos) que tarda nuestro algoritmo en ejecutarse.

En el Apéndice A se muestran dos gráficas obtenidas de una de nuestras 15 corridas para un problema en particular. El objetivo es mostrar el comportamiento de nuestro algoritmo conforme avanzaban las generaciones. Decidimos incluir esas dos gráficas por las siguientes razones: la Figura 2 muestra resultados cercanos al óptimo donde se aprecia que el algoritmo podía seguir mejorando la solución si continuarán las generaciones; y la Figura 3 se refiere a un problema donde se alcanzó el óptimo mucho antes de concluir con el número de generaciones propuesto. Lo que queremos notar es la importancia de la selección de los parámetros de entrada al algoritmo propuesto.

Problema	Valor Opt	Gap M1	Gap M2	Gap (%)	Valor obtenido	% de opt	Tiempo (seg)
132_1	1122750	10.27	8.64	4.10	1168748	20	579.79
132_2	1157722	14.44	11.82	3.18	1194568	27	748.30
132_3	1146301	11.97	10.09	7.32	1230178	13	550.23
132_4	1036779	6.80	6.06	1.56	1052944	53	717.39
133_1	1103272	9.52	7.38	4.08	1148244	13	591.28
133_2	1035443	6.15	5.23	6.06	1098162	7	665.20
133_3	1171331	12.61	11.72	4.39	1222739	27	503.58
133_4	1083636	7.60	6.27	0.48	1088794	80	665.75
134_1	1179639	12.12	7.78	3.59	1221944	33	499.93
134_2	1121633	7.12	5.43	3.62	1162207	13	543.36
134_3	1171409	12.66	12.20	2.80	1204179	47	580.92
134_4	1210465	13.25	11.63	2.70	1243165	47	576.02

Tabla 1. Instancia 1. Problema 50x50.

Problema	Valor Opt	Gap M1	Gap M2	Gap	Valor obtenido	% de opt	Tiempo (seg)
a75_50_1	1661269	27.67	24.38	1.05	1678773	73	668.01
a75_50_2	1632907	27.22	23.51	3.47	1689493	20	715.31
a75_50_3	1632213	27.14	23.59	1.10	1650176	60	513.97
a75_50_4	1585028	24.29	21.69	1.98	1616455	33	822.15
b75_50_1	1252804	28.11	24.48	4.40	1307881	7	1057.77
b75_50_2	1337446	31.06	26.93	1.97	1363789	40	903.66
b75_50_3	1249750	28.22	24.33	4.02	1299982	13	984.20
b75_50_4	1217508	27.28	22.08	1.63	1237324	80	1011.29
c75_50_1	1310193	31.85	26.77	1.79	1333694	40	1131.98
c75_50_2	1244255	29.04	24.76	1.98	1268965	53	1127.96
c75_50_3	1201706	28.47	22.09	2.69	1234099	47	1366.68
c75_50_4	1334782	30.47	26.30	1.78	1358600	33	1014.75

Tabla 2. Instancia 2. Problema 50x75.

5. Conclusiones y trabajo futuro

En este trabajo nosotros proponemos un algoritmo binivel y una reformulación que atacamos directamente. En cuanto a la reducción del problema binivel a uno MIP, no analizamos métodos de solución o relajación sino que encontramos el óptimo al resolverlo directamente (para cierto tamaño de problemas). Ahora bien, para nuestro algoritmo Stackelberg-Evolutivo, observamos de los resultados que arroja buenos resultados, la mayoría de los casos reduce las holguras de optimalidad encontradas en la literatura. Además en muchos casos también encuentra el óptimo. Aunque es evidente que no podemos comparar el tiempo requerido para obtener la solución porque la naturaleza del algoritmo diseñado es diferente a la relación y obtención de cotas existentes en la literatura. Es importante hacer énfasis que estamos proponemos un algoritmo para resolver un problema binivel y dicho enfoque no se ha abordado como tal hasta la fecha.

Además, en Vasilyev y Klimentova (2010) proponen encontrar unos límites inferiores con planos cortantes y luego encuentran unos límites superiores con un heurístico basado en recocido simulado, después con esos valores pre-procesan un método exacto y lo resuelven. Tanto ellos como Cánovas et. al. (2007) no resuelven el problema binivel sino una reformulación relajada del problema. Será interesante para nosotros comparar nuestro algoritmo contra el de recocido simulado para comparar tiempos y resultados, siempre y cuando tengamos los límites superiores e inferiores para pre-procesar nuestro algoritmo.

Actualmente estamos realizando experimentación con problemas de tamaño de 75 plantas y 100 clientes para hacer un esquema similar al que presentamos en la sección anterior. Parece ser que nuestro método es significativamente más eficiente para resolver problemas de gran escala. Además vamos a realizar un diseño de experimentos para identificar los parámetros del algoritmo (tamaño de población y número de generaciones) para tener un sustento estadístico sobre la selección de dichos valores.

Finalmente las siguientes direcciones de esta investigación son: diseñar e implementar un algoritmo Stackelberg-Búsqueda Dispersa, el cual creemos que tendrá mejores resultado que el Stackelberg-Evolutivo debido a que no es completamente aleatorio el proceso de combinación y mejora. La segunda dirección es considerar la extensión de este problema a la versión con capacidades, es decir, donde se consideren demandas de los clientes y capacidades en las plantas manteniendo en cuenta las preferencias de los clientes.

Agradecimientos

Esta investigación ha sido parcialmente apoyada por la Universidad Autónoma de Nuevo León con el programa de apoyo para investigación científica y tecnológica (PAICyT) con clave de proyecto CE407-10. Los autores también agradecen amablemente a las estudiantes Selene Casas y Yesenia Martínez por su ayuda con la experimentación computacional.

Apéndice A

En esta sección vamos a presentar dos gráficas específicas de corridas que realizamos durante la experimentación para mostrar algunos aspectos interesantes de la investigación.

La Figura 2 muestra los resultados obtenidos en la 6ta corrida durante la resolución del problema 132_3. Aquí se obtuvo un valor de la función objetivo de 1,271,100 con una holgura de optimalidad de 10.89%. Podemos apreciar que en las últimas generaciones mejoró el valor objetivo y ya no hubo oportunidad de disminuirlo otro poco debido al número máximo de generaciones pactadas. Si le hubiéramos permitido realizar más generaciones podríamos mejorar ese valor.

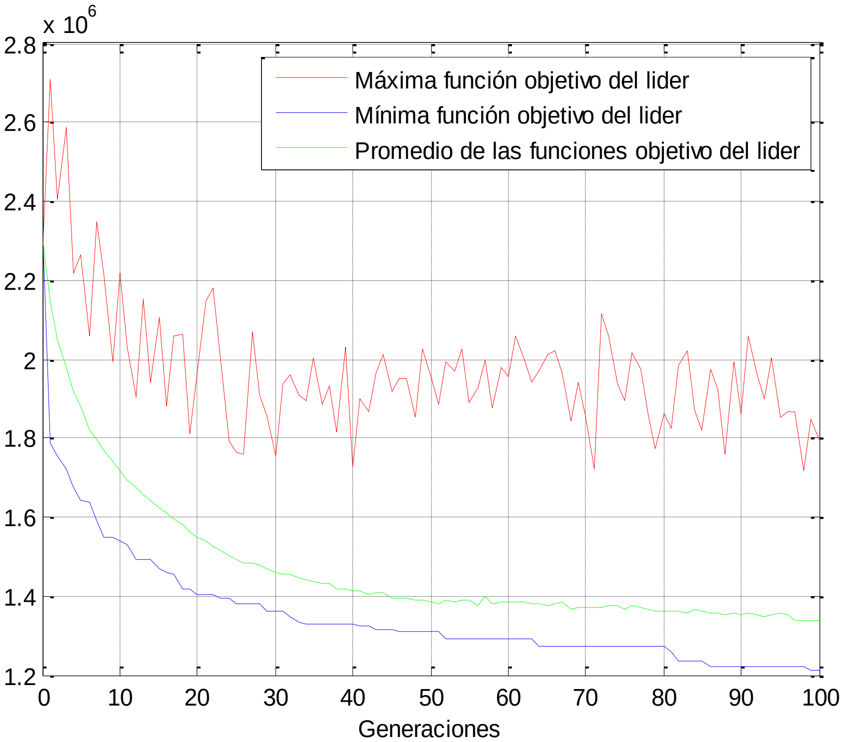

Figura 2. Problema 132_3. Corrida 6.

En la Figura 3 se aprecia el comportamiento del algoritmo Stackelberg-Evolutivo aplicado al problema 132_4 durante su 1er corrida. En este caso el algoritmo encontró el valor de 1,036,779, el cual es el óptimo para ese problema. Podemos ver que se obtuvo el óptimo en la generación 66 aproximadamente y a partir de ahí puede verse como tiempo muerto, aunque mejoró otras soluciones ya había encontrado la mejor posible. Es por esto la importancia de decidir adecuadamente el tamaño de población y principalmente el número de generaciones.

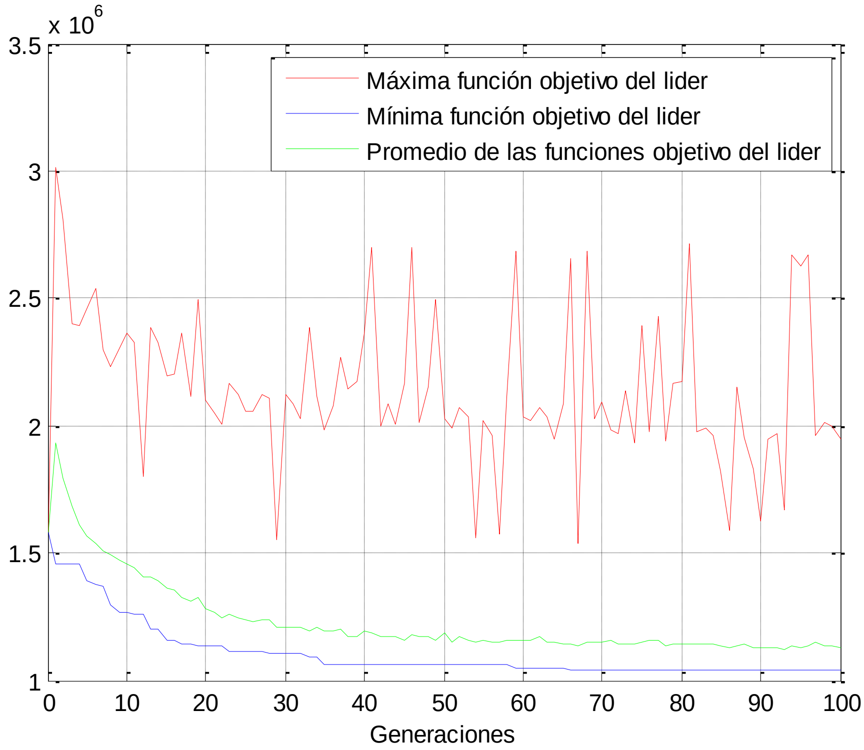

Figura 3. Problema 132_4. Corrida 1.

Referencias

Ausiello, G., Crescenzi, P., & Gambosi, G. (1999). *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*, Berlin, Germany, Springer.
Cánovas, L., García, S., Labbé, M. & Marín, A. (2007). A strengthened formulation for the simple plant location problem with order, *Oper. Res. Lett.*, pp.141-150.
Gorbachevskaya, L. E. & Kochetov, Y. A. (1992). A probabilistic heuristic for the two-level location problem, in *Proceedings of the 11-th Baikal International School-Seminar*, Irkutsk, pp. 249-252, (in Russian).
Hanjoul, P. & Peeters, D. (1987). A facility location problem with clients'preference orderings, *Regional Science and Urban Economics*, Vol. 17, pp. 451-473.
Hansen, P., Kochetov, Y. A. & Mladenovic, N. (2004). Lower Bounds for the Uncapacitated Facility Location Problem with User Preferences, *Les Cahiers du GERAD G-2004-24*.
Ishii, H., Lee, Y.L. & Yeh, K.Y. (2007). Fuzzy facility location problem with preference of candidate sites, *Fuzzy Sets and Systems*, Vol. 158, pp. 1922 – 1930.
Krarup, J. & Pruzan, P.M. (1983), The simple plant location problem: survey and synthesis. *European Journal of Operational Research*, Vol. 12, pp. 36-81.
Stackelberg, H.v. (1952), Marktform und Gleichgewicht, *Springer-Verlag, Berlin*, 1934. engl. transl: "The Theory of the Market Economy", *Oxford University Press*.
Vasil'ev, I.L. & Klimentova, K.B. (2010). The Branch and Cut Method for the Facility Location Problem with Client's Preferences, *Journal of Applied and Industrial Mathematics*, Vol. 4, No. 3, pp. 441–454.
Vasil'ev, I.L., Klimentova, K.B. & Kochetov, Y.A. (2009). New Lower Bounds for the Facility Location Problem with Clients' Preferences, *Computational Mathematics and Mathematical Physics*, Vol. 49, No. 6, pp. 1010–1020.