Solución heurística a un problema de diseño de territorios comerciales con restricciones de asignación conjunta mediante GRASP

Saúl I. Caballero-Hernández
Universidad Autónoma de Nuevo León
San Nicolás de los Garza, México
saul@yalma.fime.uanl.mx

Roger Z. Ríos-Mercado
Universidad Autónoma de Nuevo León
San Nicolás de los Garza, México
roger@yalma.fime.uanl.mx

Fabián López
Grupo ARCA
Monterrey, México
fabian.lopez@e-arca.com.mx

Resumen

En este artículo se presenta un enfoque de solución basado en GRASP para un problema de diseño de territorios de atención comercial cuya aplicación en el mundo real proviene de una empresa distribuidora de bebidas de la industria local. Algunos de los criterios a considerar en el diseño territorial son la minimización de una medida de dispersidad, balanceo con respecto a ciertas actividades nodales, contigüidad de los territorios y cumplimiento de restricciones de asignación conjunta de unidades básicas. Se presenta una descripción de los componentes del algoritmo y resultados que muestran la eficacia del método sobre un conjunto de instancias generadas a partir de datos reales.

1. Introducción

El diseño de territorios se puede ver como el problema de agrupar pequeñas unidades geográficas en grupos geográficos más grandes llamados territorios o distritos en una forma que estos son aceptables (u óptimos) de acuerdo a ciertos criterios de planeación relevantes [2].

El problema aquí tratado es motivado por una aplicación del mundo real encontrada en una empresa distribuidora de refrescos de la ciudad de Monterrey, México; y se puede considerar como un problema de diseño de territorios de atención comercial.

La compañía desea particionar el área de la ciudad en territorios disjuntos adecuados para sus propósitos comerciales. En particular, la compañía desea territorios balanceados con respecto a dos actividades (número de clientes y volumen de ventas). Adicionalmente, se desean territorios contiguos, de forma que se pueda llegar de una unidad básica a otra viajando dentro del territorio; compacidad de los territorios, de forma que los clientes esten relativamente cerca unos de otros; asignación conjunta de unidades básicas, de manera que dos unidades básicas marcadas de esta forma deben pertenecer al mismo territorio; el número de territorios es predeterminado.

Aunque han aparecido en la literatura enfoques de solución para problemas de diseño territorial, esta aplicación en particular presenta ciertas características que hacen al problema único. Vargas-Suárez, Ríos-Mercado y López [6] estudiaron un problema similar pero que no considera compacidad ni contigüidad y donde se busca minimizar el desbalanceo entre los territorios. Ríos-Mercado y Fernández [5] estudiaron el problema considerando compacidad y contigüidad pero sin restricciones de asignación conjunta.

En este trabajo se propone un algoritmo de solución para obtener soluciones iniciales de calidad basado en GRASP [1, 4]. Como función objetivo del modelo se ha tomado una medida de dispersidad de los territorios basada en distancia. Dado que puede no existir un diseño territorial en el que todos los territorios estén balanceados simultáneamente con respecto a las medidas de actividad, se han introducido tolerancias de desviación con respecto al valor

meta de balanceo de las actividades para modelar este requerimiento.

El resto del artículo está estructurado de la siguiente forma. En la Sección 2 se describe el problema y se presenta una formulación del mismo. En la Sección 3 se describen los componentes del algoritmo propuesto. La evaluación empírica se muestra en la Sección 4. Finalmente en la Sección 5 se presentan las conclusiones y trabajo a futuro.

2. Descripción del problema

El problema se modela como un grafo $G = (V, E)$ que representa el territorio que se desea particionar, una manzana o unidad básica i está asociado con un nodo, y un arco conectando los nodos i y j existe si las unidades básicas i y j son adyacentes. Cada nodo $i \in V$ tiene asociados varios parámetros como coordenadas geográficas (c_i^x, c_i^y), y dos medidas de actividad. Sea w_i^a el valor de la actividad a en el nodo i, donde $a = 1$ (número de clientes) y $a = 2$ (volumen de ventas). Un territorio es un subconjunto de nodos $V_k \subset V$. El número de territorios está dado por el parámetro p. Se requiere que cada nodo esté asignado a un solo territorio. Los territorios definen una partición de V. Una de las propiedades deseadas en una solución es que los territorios estén balanceados con respecto a cada medida de actividad. Definamos el tamaño del territorio V_k con respecto a la actividad a como: $w^a(V_k) = \sum_{i \in V_k} w_i^a$, $a = 1, 2$. Debido a la estructura discreta del problema y a la restricción de asignación única, es prácticamente imposible tener territorios perfectamente balanceados con respecto a cada medida de actividad. Para manejar esto, se mide el grado de balanceo calculando la desviación relativa de cada territorio con respecto al tamaño medio μ^a, siendo $\mu^a = w^a(V)/p$, $a = 1, 2$. Se cuenta con una colección de parejas de nodos H en la cual una pareja $[i, j]$ implica que los nodos i y j deben asignarse al mismo territorio. Otra característica importante es que todas las unidades básicas asignadas a cada territorio están conectados por una ruta que está contenida totalmente dentro del territorio. En otras palabras, cada territorio V_k debe inducir un subgrafo conexo de G. Adicionalmente, se requiere que en cada territorio, los clientes esten relativamente cerca entre ellos. Una forma de lograr esto para cada territorio es tener como referencia un nodo apropiado para ser su centro, y entonces definir una medida de distancia como

$$f(V_1, \ldots, V_p) = \max_{\substack{k=1,\ldots,p \\ j \in V_k}} \{d_{c(k)j}\},$$

donde $c(k)$ denota el índice del centro del territorio k y $d_{c(k)j}$ representa la distancia Euclidiana del nodo j al centro del territorio k. Se supone que se conocen todos los parámetros con certeza.

El problema se puede describir entonces como el de encontrar una p-partición de V satisfaciendo los criterios de balanceo, contigüidad, asignación conjunta y que minimiza la medida de dispersidad anterior.

Expresado formalmente, se desea encontrar una partición $\bar{V} = (V_1, \ldots, V_p)$ que minimize $f(\bar{V})$ sujeto a las restricciones de balanceo $(1 - \tau^a)\mu^a \leq w^a(V_k) \leq (1 + \tau^a)\mu^a \quad \forall k$, con $G(V_k, E(V_k))$ conexo y para cada $[i, j] \in H$, existe un territorio k tal que ambos i y$j \in V_k$.

Este problema también se puede formular en términos de programación entera como un problema de p-centro con restricciones adicionales de capacidad, contigüidad y de asignación conjunta. Dado que el problema de p-centro es NP-duro [3], se sigue que nuestro problema también es NP-duro.

3. Algoritmo propuesto

GRASP es una metaheurística que ha sido ampliamente usada para resolver con éxito muchos problemas de optimización combinatoria [4]. Es un método multiarranque que captura las buenas características de los algoritmos miopes puros y de los procedimientos de construcción aleatorizados. Cada iteración GRASP consiste en la construcción de una solución miope aleatorizada seguida de una etapa de postprocesamiento en la cual se intenta mejorar la solución encontrada en la etapa previa. Este procedimiento se repite varias veces y la

mejor solución encontrada sobre todas las iteraciones GRASP se devuelve como la solución aproximada.

El GRASP propuesto busca en la primera fase construir una solución que sea factible con respecto a las restricciones de asignación conjunta y de contigüidad. En esta etapa la función miope que guía la construcción es una función que pondera una medida de dispersidad y la violación relativa de las restricciones de balanceo con respecto a la cota superior de las actividades. En la etapa de posprocesamiento se lleva a cabo una búsqueda local en la que se busca mejorar el valor de la función objetivo y recuperar la factibilidad con respecto a las restricciones de balanceo.

3.1. Construcción

Durante esta fase se busca obtener una solución que cumpla con las restricciones de contigüidad y con las de asignación conjunta. En esta primera fase se pone un mayor esfuerzo en obtener soluciones de calidad respecto a la medida de dispersidad y se permite cierta violación de las restricciones de balanceo.

La idea en esta etapa es la de unir territorios iterativamente hasta alcanzar el número deseado (p). Para un par de territorios se evalua una función miope que pondera una medida de dispersidad y la violación relativa de las restricciones de balanceo para después decidir si se lleva a cabo la fusión de estos territorios.

El algoritmo arranca con n territorios, es decir que se asigna cada nodo a un territorio distinto. Después, a manera de preprocesamiento para asegurar que se cumplan las restricciones de asignación conjunta, se comienza por encontrar las rutas más cortas (P_{ij}) entre los nodos marcados con este tipo de restricciones. Dado un par de nodos i, j que se deben asignar conjuntamente, se procede a combinar los territorios que contienen a los nodos dentro de la ruta más corta de i a j para obtener de esta manera un nuevo territorio que no viole las restricciones de contigüidad.

En este punto, una vez asegurado el cumplimiento de las restricciones de asignación conjunta, se continúa combinando territorios de la siguiente forma hasta alcanzar los p territorios deseados. En una iteración dada, se analizan todos los arcos (i, j) que tienen extremos en territorios diferentes, de este modo se evalua la posibilidad de fusionar los territorios $V_{t(i)}$ y $V_{t(j)}$, donde $t(i)$ representa el índice del territorio al que pertenece el nodo i.

Denotemos como $f(V_k) = \max_{i,j \in V_k}\{d_{ij}\}$ la medida de dispersidad del territorio V_k, también definamos como $w^a(V_k) = \sum_{i \in V_k} w_i^a$ el tamaño del territorio V_k con respecto a la actividad a, $a \in A$. Sea (i, j) un arco del grafo con $i \in V_k$ y $j \in V_l$, definimos su función miope como

$$\phi(i, j) = \lambda F(i, j) + (1 - \lambda)G(i, j), \quad (1)$$

donde

$$F(i, j) = f(V_k \cup V_l),$$

define la medida de dispersidad para el territorio que surge de fusionar los territorios V_k y V_l, y

$$G(i, j) = \sum_{a \in A} g^a(i, j),$$

con $g^a(i, j) = (1/\mu^a)\,\text{máx}\{w^a(V_k \cup V_l) - (1 + \tau^a)\mu^a, 0\}$ que representa la suma de infactibilidades relativas con respecto a la cota superior de la restricción de balance de la actividad a. El parámetro λ se utiliza para ponderar los dos términos en la función miope. El Algoritmo 1 muestra el pseudocódigo del procedimiento constructivo.

3.2. Postprocesamiento

Después de la etapa de construcción se ejecuta una etapa de postprocesamiento en la que se lleva a cabo una búsqueda local. En esta etapa se busca mejorar el valor de la función objetivo y recuperar factibilidad respecto a las restricciones de balanceo que sean violadas. Durante la búsqueda local se utiliza una función de mérito que pondera la infactibilidad con respecto a las restricciones de balanceo y el valor de la función objetivo. Esta función es muy similar a la función miope utilizada en la fase constructiva con la excepción de que ahora la suma de infactibilidades considera también la violación con respecto a la cota inferior de las

Función Construccion(α)
Entrada: $\alpha :=$ Parámetro de calidad de la RCL de GRASP
Salida: Asignación factible S

```
0   Para cada i ∈ V : V_i ← {i}
1   Ē ← E;  S = {V_1, ..., V_n}
2   Para cada ([i, j] ∈ H):
3       Encontrar P_ij
4       Para cada l ∈ P_ij:  V_i ← V_i ∪ V_t(l),  S ← S \ V_t(l);
5   Mientras (|S| > p)
6       Calcular φ(i, j) de (1) para todo (i, j) ∈ Ē
7       Φ_mín ← mín_{i,j} {φ(i, j)};  Φ_máx ← máx_{i,j} {φ(i, j)}
8       RCL ← {(i, j) ∈ Ē : φ(i, j) ∈ [Φ_mín, Φ_mín + α(Φ_máx − Φ_mín)]};
9       Escoger (i, j) ∈ RCL aleatoriamente.
10      V_t(i) ← V_t(i) ∪ V_t(j) ;  S ← S \ V_t(j);  Ē ← Ē \ {(j_1, j_2) ∈ Ē : j_1, j_2 ∈ V_t(i)};
11  Termina Mientras
12  Regresar S
```

Algoritmo 1: Fase constructiva

restricciones de balanceo. Para una partición $S = (V_1, \ldots, V_p)$ su función de mérito es

$$\psi(S) = \lambda F(S) + (1 - \lambda)G(S)$$

donde

$$F(S) = \max_{k=1,\ldots,p} \left\{ \max_{j \in V_k} d_{c(k)j} \right\}$$

y

$$G(S) = \sum_{k=1}^{p} \sum_{a \in A} g^a(V_k),$$

con $g^a(V_k) = (1/\mu^a)\max\{w^a(V_k) - (1 + \tau^a)\mu^a, (1 - \tau^a)\mu^a - w^a(V_k), 0\}$ como la suma de las infactibilidades relativas de las restricciones de balance.

Se utiliza un vecindario $N(S)$ formado por aquellas soluciones que se pueden obtener a partir de S al mover una unidad básica i de su territorio $t(i)$ a un territorio vecino $t(j)$, donde j es la unidad básica en $t(j)$ adyacente a i. No se permiten movimientos en los que se generen territorios no contiguos o que violen las restricciones de asignación conjunta.

4. Experimentación

En esta sección se presentan los resultados obtenidos con una implementación en C++ del algoritmo presentado en este trabajo. Los experimentos se llevaron a cabo sobre un equipo SunFire V440 con sistema operativo Solaris 9.

Para las pruebas se generaron varios conjuntos de instancias a partir de datos reales proporcionados por nuestro socio industrial. Los tamaños de las instancias que se manejan en las pruebas son de 500 y 1000 nodos y el número de territorios buscados es 10 y 20, respectivamente. Con respecto a los niveles de tolerancia de desviación (τ^a) de los valores meta de las actividades, denotaremos como DS30, DS20, DS10 y DS05 a los conjuntos de instancias con valores τ^a de 0.3, 0.2, 0.1 y 0.05, respectivamente. Cada grupo de instancias consta de 10 instancias diferentes.

Como un primer experimento se realizaron pruebas para calibrar el parámetro de calidad de la lista restringida de candidatos de GRASP (α) con el fin de determinar el valor del parámetro para el cual el algoritmo presenta un mejor desempeño en cuanto al valor de la función objetivo.

Para llevar a cabo las pruebas en esta parte se tomaron dos conjuntos de instancias (uno de tamaño 500 y otro de 1000) considerados como los más dificiles, es decir, del tipo DS05. El algoritmo fue ejecutado realizando 500 iteraciones GRASP para valores α de 0.0, 0.1,

Tabla 1: Evaluación del parámetro de calidad (α)

	α= 0.0	α= 0.1	α= 0.2	α= 0.3	α= 0.4	α= 0.5
Instancias 500 nodos						
Desviación relativa del mejor	12.25	0.17	2.38	5.07	6.17	11.74
Veces mejor solución	0	9	4	0	0	0
Soluciones infactibles	0	0	0	0	0	0
Instancias 1000 nodos						
Desviación relativa del mejor	9.15	0.47	2.84	4.73	8.69	10.36
Veces mejor solución	1	8	3	1	0	0
Soluciones infactibles	2	0	0	1	2	2

Tabla 2: Instancias DS30 - Tamaño 500 nodos

Instancia	Iter.	Objetivo	dmaxc	Infeas.	MP BL (%)	M BL	SF F1
DU500-01.dat	173	0.054	122.378	0.00	80.09	22	23
DU500-02.dat	251	0.051	116.904	0.00	79.87	22	20
DU500-03.dat	459	0.053	120.827	0.00	84.25	25	10
DU500-04.dat	132	0.051	117.032	0.00	81.06	21	8
DU500-05.dat	32	0.053	120.274	0.00	79.94	21	16
DU500-06.dat	303	0.053	118.417	0.00	82.73	21	10
DU500-07.dat	38	0.052	117.776	0.00	81.87	22	10
DU500-08.dat	132	0.052	120.255	0.00	79.73	22	20
DU500-09.dat	184	0.053	119.369	0.00	79.85	20	19
DU500-10.dat	491	0.051	117.048	0.00	83.97	24	9

Tabla 3: Instancias DS20 - Tamaño 500 nodos

Instancia	Iter.	Objetivo	dmaxc	Infeas.	MP BL (%)	M BL	SF F1
DU500-01.dat	440	0.054	123.454	0.00	90.63	34	1
DU500-02.dat	186	0.051	118.039	0.00	90.42	34	1
DU500-03.dat	263	0.054	121.574	0.00	90.31	35	1
DU500-04.dat	412	0.053	117.998	0.00	89.82	33	0
DU500-05.dat	125	0.051	116.492	0.00	90.02	35	0
DU500-06.dat	231	0.052	114.213	0.00	89.96	33	0
DU500-07.dat	125	0.054	123.189	0.00	89.37	33	0
DU500-08.dat	139	0.055	122.293	0.00	89.87	34	1
DU500-09.dat	147	0.052	117.879	0.00	90.30	34	0
DU500-10.dat	339	0.051	116.700	0.00	89.63	34	3

0.2, 0.3, 0.4, 0.5 y el parámetro de ponderación de la función miope fijo ($\lambda = 0.7$). Durante la búsqueda local el algoritmo iteró hasta converguer a un óptimo local. Los resultados se pueden observar en la Tabla 1. Como puede apreciarse, los mejores resultados se observaron con un $\alpha = 0.1$.

Como una segunda etapa experimental, se

Tabla 4: Instancias DS10 - Tamaño 500 nodos

Instancia	Iter.	Objetivo	dmaxc	Infeas.	MP BL (%)	M BL	SF F1
DU500-01.dat	157	0.054	123.238	0.00	94.85	60	0
DU500-02.dat	209	0.054	123.314	0.00	94.76	60	0
DU500-03.dat	440	0.057	125.190	0.00	94.48	62	0
DU500-04.dat	454	0.055	122.370	0.00	94.52	58	0
DU500-05.dat	104	0.054	121.307	0.00	94.83	63	0
DU500-06.dat	22	0.055	123.278	0.00	94.51	59	0
DU500-07.dat	378	0.056	125.603	0.00	94.66	57	0
DU500-08.dat	155	0.056	124.490	0.00	94.88	59	0
DU500-09.dat	354	0.054	125.118	0.00	94.99	64	0
DU500-10.dat	13	0.054	123.715	0.00	94.87	60	0

Tabla 5: Instancias DS05 - Tamaño 500 nodos

Instancia	Iter.	Objetivo	dmaxc	Infeas.	MP BL (%)	M BL	SF F1
DU500-01.dat	40	0.053	119.364	0.00	96.18	77	0
DU500-02.dat	420	0.057	130.110	0.00	96.05	79	0
DU500-03.dat	285	0.054	120.516	0.00	96.11	79	0
DU500-04.dat	119	0.057	125.444	0.00	95.82	78	0
DU500-05.dat	202	0.056	128.156	0.00	96.28	83	0
DU500-06.dat	281	0.054	122.866	0.00	96.21	80	0
DU500-07.dat	163	0.057	124.938	0.00	95.91	79	0
DU500-08.dat	358	0.057	126.387	0.00	96.33	83	0
DU500-09.dat	58	0.056	125.138	0.00	96.17	85	0
DU500-10.dat	188	0.054	122.033	0.00	96.28	81	0

Tabla 6: Instancias DS20 - Tamaño 1000 nodos

Instancia	Iter.	Objetivo	dmaxc	Infeas.	MP BL (%)	M BL	SF F1
DU1000-01.dat	295	0.039	90.934	0.00	96.03	65	0
DU1000-02.dat	407	0.040	90.944	0.00	96.11	63	0
DU1000-03.dat	107	0.040	89.928	0.00	96.20	64	0
DU1000-04.dat	172	0.040	91.457	0.00	96.22	65	0
DU1000-05.dat	201	0.038	86.731	0.00	96.31	64	0
DU1000-06.dat	492	0.039	89.003	0.00	96.08	61	0
DU1000-07.dat	287	0.039	89.524	0.00	96.09	64	0
DU1000-08.dat	437	0.039	90.695	0.00	96.16	62	0
DU1000-09.dat	481	0.040	91.126	0.00	96.20	64	0
DU1000-10.dat	184	0.040	90.450	0.00	96.02	61	0

realizaron pruebas con el algoritmo con instancias de los cuatro tipos (DS30, DS20 DS10 y DS05) y para tamaños 500 y 1000 nodos con el fin de evaluar la calidad de las soluciones obtenidas y su comportamiento en las distintas fases del mismo. Estos experimentos se ejecu-

Tabla 7: Instancias DS10 - Tamaño 1000 nodos

Instancia	Iter.	Objetivo	dmaxc	Infeas.	MP BL (%)	M BL	SF F1
DU1000-01.dat	338	0.042	95.245	0.00	97.75	112	0
DU1000-02.dat	91	0.041	93.789	0.00	97.91	125	0
DU1000-03.dat	103	0.041	91.767	0.00	97.84	117	0
DU1000-04.dat	12	0.041	92.212	0.00	97.81	115	0
DU1000-05.dat	180	0.042	95.599	0.00	97.76	116	0
DU1000-06.dat	497	0.041	93.034	0.00	97.89	119	0
DU1000-07.dat	109	0.041	92.335	0.00	97.80	116	0
DU1000-08.dat	398	0.041	90.392	0.00	97.90	121	0
DU1000-09.dat	35	0.042	94.362	0.00	97.88	120	0
DU1000-10.dat	393	0.040	90.366	0.00	97.86	121	0

Tabla 8: Instancias DS05 - Tamaño 1000 nodos

Instancia	Iter.	Objetivo	dmaxc	Infeas.	MP BL (%)	M BL	SF F1
DU1000-01.dat	401	0.043	96.259	0.00	98.31	166	0
DU1000-02.dat	258	0.042	97.432	0.00	98.30	171	0
DU1000-03.dat	83	0.044	97.825	0.00	98.35	180	0
DU1000-04.dat	110	0.042	92.921	0.00	98.26	168	0
DU1000-05.dat	63	0.041	94.928	0.00	98.35	171	0
DU1000-06.dat	77	0.044	99.138	0.00	98.28	176	0
DU1000-07.dat	21	0.043	97.439	0.00	98.37	168	0
DU1000-08.dat	383	0.043	96.785	0.00	98.28	174	0
DU1000-09.dat	347	0.043	96.521	0.00	98.32	166	0
DU1000-10.dat	289	0.043	97.485	0.00	98.29	165	0

taron en las mismas condiciones que los primeros pero utilizando α fijo en 0.1.

En la Tabla 2 se presentan los resultados para 10 instancias de tamaño 500 con valores de tolerancia de 30 %. La primera columna muestra el nombre de la instancia, la segunda columna muestra el número de iteración en la que se encontró la mejor solución, la tercera columna muestra el valor de la función objetivo ponderada, la cuarta columna muestra el valor de la medida de dispersidad definida anteriormente, en la quinta columna se muestra la suma de infactibilidades relativas con respecto a las restricciones de balanceo, la columna MP BL muestra la mejora promedio obtenida con la búsqueda local, la columna M BL muestra el número promedio de movimientos realizados en la búsqueda local y la columna SF F1 muestra el número de soluciones factibles encontradas en la fase de construcción del algoritmo. Las Tablas 3 a 8 muestran los resultados para el resto de los grupos de instancias.

Como puede apreciarse el algoritmo fue capaz de encontrar soluciones factibles para todas las instancias. Esto se debió en gran medida al muy buen desempeño de la búsqueda local ya que en la fase constructiva un gran número de instancias son infactibles. La búsqueda local mejora a la fase constructiva entre un 79 y 96 % para las instancias de 500 nodos y entre un 92 y 98 % para las de 1000 nodos. También se puede ver que en las instancias en que la desviación permitida es más pequeña el número de movimientos promedio realizados en la búsqueda local es más grande. En

cuanto a los tiempos se observó que las instancias de 500 nodos toman en promedio 231 segundos. Para las instancias de 1000 nodos el tiempo de ejecución promedio fue de 758 segundos. En ambos casos se observó que aproximadamente el 80 % del tiempo lo empleó la fase constructiva.

5. Conclusiones

En este trabajo se ha presentado un GRASP para una variante del problema de diseño de territorios comerciales. Está variante contempla restricciones de asignación conjunta lo cual representa una contribución importante de nuestro trabajo ya que hasta donde se tiene conocimiento este caso no ha sido tratado anteriormente. Los resultados conseguidos por la heurística son bastante prometedores ya que se logra conseguir instancias factibles de buena calidad para problemas de 500 y 1000 unidades básicas. En particular se observó un excelente desempeño de la búsqueda local.

Como trabajo futuro se está trabajando además en la implementación de una vecindad basada en intercambio de unidades básicas entre territorios vecinos para tener una mayor flexibilidad en la fase de postprocesamiento. Otra de las áreas de oportunidad es evaluar la sensitividad del algoritmo con respecto al parámetro de ponderación de la función miope (1).

Agradecimientos

El presente trabajo es apoyado por el Consejo Nacional de Ciencia y Tecnología (apoyo número 83499-Y). Se agradece además a dos revisores anónimos cuya crítica y observaciones ayudó a mejorar la presentación del trabajo.

Referencias

[1] T. A. Feo y M. G. C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6(2):109–133, 1995.

[2] J. Kalcsics, S. Nickel y M. Schröder. Toward a unified territorial design approach: Applications, algorithms, and GIS integration. *Top*, 13(1):1–74, 2005.

[3] O. Kariv y S. L. Hakimi. An algorithmic approach to network location problems. I: the p-centers. *SIAM Journal on Applied Mathematics*, 37(3):513–538, 1979.

[4] M. G. C. Resende y J. L. González Velarde. Reactive GRASP: Procedimientos de búsqueda miope aleatorizados y adaptativos. *Inteligencia Artificial*, 19:61–76, 2003.

[5] R. Z. Ríos-Mercado y E. Fernández. A reactive GRASP for a sales territory design problem with multiple balancing requirements. Reporte técnico PISIS-2006-12, Programa de Posgrado en Ingeniería de Sistemas, Universidad Autónoma de Nuevo León, San Nicolás de los Garza, México, Septiembre 2006.

[6] L. Vargas-Suárez, R. Z. Ríos-Mercado y F. López. Usando GRASP para resolver un problema de definición de territorios de atención comercial. En M. G. Arenas, F. Herrera, M. Lozano, J. J. Merelo, G. Romero y A. M. Sánchez, editores, *Memorias del IV Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB)*, pp. 609–617, Granada, España, Septiembre 2005.