UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

FACULTAD DE INGENIERÍA

DISEÑO Y DESARROLLO DE UN MODELO PARA LA ZONIFICACIÓN ÓPTIMA

T E S I S

QUE PARA OBTENER EL GRADO DE:

DOCTORA EN INGENIERÍA

INVESTIGACIÓN DE OPERACIONES

P R E S E N T A

MARÍA BEATRÍZ BERNÁBE LORANCA

DIRECTOR DE TESIS:

MARÍA AUXILIO OSORIO LAMA

MÉXICO, D.F **ENERO 2010**

Agradecimientos

Quiero agradecer a mi Directora de tesis, la Dra. María Auxilio Osorio Lama y a mi Co-Director Ricardo Aceves García por todo el apoyo, orientación, consejos, recomendaciones, ayuda y confianza que me brindaron durante mi trabajo doctoral.

Agradezco a mi comité tutoral: Dr. Servio Tulio Guillen Burguete, Dr. Miguel Ángel Gutiérrez Andrade y al Dr. Javier Ramírez Rodríguez por sus valiosas observaciones en mis evaluaciones y en la revisión final de esta tesis.

Agradezco a la Benemérita Universidad Autónoma de Puebla por el apoyo de superación académica que recibí de PROMEP.

Mi primer contacto y apoyo con CONACYT, ha sido en este trabajo doctoral a través de la Universidad Nacional Autónoma de México, en la cual, aprendí más de lo que esperaba. Muchas gracias a estas dos instituciones.

Un agradecimiento muy especial al Dr. Javier Ramírez Rodríguez por creer en mí y por el trabajo conjunto que desarrollamos en publicaciones.

Agradezco al Dr. Juan Carlos Duque Cardona, por su importante contribución en el estado del arte de esta tesis.

Agradezco al Dr. Alberto Ochoa Rodríguez por todo lo que le he aprendido, y por su apoyo en la Habana, que permitió cumplir el trabajo que nos habíamos propuesto.

Agradezco al Dr. Carlos Guillen Galvan por su valiosa ayuda para comprender los aspectos matemáticos en multiobjetivo.

Por el lado académico, quiero finalizar agradeciendo al Dr. Carlos Coello, quien sin conocerme, me brindó la información necesaria para concluir la parte multiobjetivo de esta tesis. Sus sugerencias y ayuda fueron de vital importancia.

Agradezco a mis padres Isabel y Raúl por la formación y enseñanzas que recibí sobre los valores necesarios para cumplir, seguir y terminar.

Con una gratitud muy particular agradezco a cada una de las personas que estuvieron conmigo alentándome, extrañándome en las reuniones que no pude llegar, comprendiendo mis ausencias, retardos y momentos de abstracción.
Muchas gracias a mis amigos y amigas: Julieta, Amelia, Beatriz, Ana, María Elena, José Eduardo, Alejandro, Enrique, Aarón y Ramiro.

No puedo dejar de mencionar y agradecer a una gran persona, quien me compartió de sus ideas y de su tiempo en algunos momentos difíciles de esta tesis: Enrique Zamora Alcocer.

Gracias a Pablo Salazar por la nueva música contemporánea que trajo en el momento justo.

Dedico este trabajo a Melissa Mendoza Bernábe.

Gracias hija por estar en toda la lucha, por acompañarme en los momentos duros donde juntas hemos crecido, aprendido y planeado sonrisas.
Sé que un sueño no se obtiene si no se tiene la inspiración y la fuerza, aquí, Melissa es altamente responsable de ello.

Índice general

Introducción

Justificación

En general, en problemas de zonificación, el cual se entiende informalmente como un proceso de agrupación de zonas geográficas, se parte del supuesto de la existencia de una relación entre las características de los datos que componen la zona metropolitana, las variables de la población y las propiedades del problema.

En este tipo de problemas de zonificación, se desea saber la manera en que una variable censal/poblacional se encuentra distribuida o concentrada en determinados espacios territoriales.
Las aplicaciones de este tipo de problemas son diversas, principalmente centradas en resolver problemas de carácter poblacional.
Por ejemplo, si se desea atender a un sector de la población que no tenga servicios básicos de drenaje, entonces se puede abordar el problema agrupando unidades de áreas territoriales en grupos más grandes, tal que dichas áreas pertenecientes a los grupos, sean muy cercanas entre ellas (para mejor desplazamiento entre las zonas) y considerar en la agrupación las variables relacionadas con servicios de drenaje.

Esto significa la necesidad de crear y analizar grupos pequeños debido a que no es posible estudiar toda la extensión territorial como una sola unidad. En este punto y de acuerdo a la definición de diseño territorial DT que se define posteriormente, se ubica a zonificación óptima ZO como un caso derivado de diseño territorial.

Cuando se refiere a zonificación óptima, se asume que zonificación óptima es un problema de tipo diseño territorial.
El problema de zonificación óptima consiste en encontrar respuesta a diversos problemas poblacionales (distribución, concentración, densidad o centralización de la población de una zona metropolitana).
Se interpreta así que zonificación óptima, debe buscar agrupaciones de agebs tal que exista en cada grupo un equilibrio o balanceo sobre una variable poblacional específica (homogeneidad), y por otro lado, que cumpla con la propiedad de compacidad, exigida en diseño territorial. Por tanto, este problema debe resolverse con un método que pueda agrupar agebs a la vez que se minimizan las funciones objetivo de compacidad geométrica en el espacio geográfico y homogeneidad para alguna variable descriptiva de las agebs.

Es en este punto donde la propuesta para resolver el presente trabajo de tesis, se asiste de modelos de diseño territorial y algoritmos de particionamiento como soporte en su modelación.
Adicionalmente la solución para el conflicto de las dos funciones objetivo, se fundamenta de la teoría multiobjetivo para hallar un conjunto de soluciones no dominadas.
Esto indica que la solución a zonificación óptima reside en encontrar un conjunto de particiones compactas y homogéneas. Dicho conjunto de soluciones es una colección de pares de soluciones no dominadas integradas por (compacidad, homogeneidad).
El particionamiento implícito debe garantizar que cada partición sea compacta y que la suma de alguna variable poblacional cumpla la homogeneidad tanto como sea posible.

Diseño territorial

Definición. El diseño de territorio DT puede ser visto como un problema de agrupación de áreas geográficas pequeñas (áreas básicas) en conglomerados geográficos más grandes llamados territorios, del tal forma que la agrupación aceptable es aquella última que cumpla con criterios predeterminados del problema que ocupa. Dependiendo del contexto, estos criterios bien pueden ser de motivo económico (promedio de ventas potenciales, trabajo o número de vendedores) o tienen un fondo demográfico (número de habitantes, población votante) [Zoltners83].

Estos criterios en diseño territorial, en la formulación teórica, suelen expresarse como funciones objetivo y/o restricciones en un modelo de optimización. En este escenario, DT se ha demostrado

como un problema NP-duro [Altamn97].

Información básica a procesar en diseño territorial
Frecuentemente en un problema de regionalización o de diseño territorial, es común partir de información estadística a nivel regional y urbano además de la elección de las unidades geográficas que serán utilizadas.
En este sentido, se tienen al menos dos alternativas: la primera es utilizar "agrupaciones territoriales" oficialmente establecidas (unidades geográficas como municipios, provincias, comunidades autónomas, etc.) y la segunda opción es la creación de unidades territoriales que estén directamente relacionadas con el fenómeno que se estudia (unidades analíticas).
En esta tesis, se han estudiado los datos geográficos disponibles; de este modo, se escogieron a las áreas geoestadísticas básica agebs como datos a agrupar, las cuales son unidades geográficas oficialmente establecidas y de fácil acceso [Web3].

Aplicaciones en diseño territorial
Las aplicaciones sobre problemas de regionalización se presentan como problemas de asignación de puntos de venta, localización de servicios, problemas poblacionales y censales (como lo es zonificación óptima ZO), distritación electoral etc.
Estas aplicaciones son tratadas como problemas de optimización combinatoria con un enfoque de diseño territorial que tienen implícito un proceso de agrupación para las áreas geográficas disponibles.

Agrupación en diseño territorial
Hasta ahora, se sabe que la gran mayoría de los problemas derivados de diseño territorial, utilizan, entre otras herramientas, algún método de clasificación para crear grupos.
Entonces, cuando en un problema se busca crear agrupaciones con restricciones geográficas y los datos a agrupar son espaciales y/o unidades territoriales descritas por variables, se supone que este problema pertenece a DT.

La agrupación aquí debe asegurar que el número de grupos que se forman sea estrictamente menor a la cantidad total de los datos a agrupar, que los grupos sean no vacíos y que no exista intersección en los grupos.
Los datos a agrupar son unidades geográficas de naturaleza espacial, lo que implica que la clasificación debe formar grupos que cumplan con propiedades de compacidad geométrica, homogeneidad, contigüidad, conexidad u otras. Estas propiedades generalmente se plantean como restricciones o como medidas de disimilitud y/o similitud en la función objetivo.

Conglomerado geográfico
Los puntos anteriores hablan también del problema de conglomerado geográfico CG siempre y cuando la agrupación procese datos espaciales (agebs en este caso) y se optimice la función objetivo de compacidad geométrica.
El objetivo del conglomerado geográfico, al igual que el tradicional conglomerado, es encontrar conglomerados que corresponden a una partición del conjunto de objetos optimizando una función objetivo.
A estos métodos se les conoce como métodos de hard-clustering o clustering particional y se ha demostrado que estos métodos son Np-difícil o duro [Vicente05].
Se ubica así a conglomerado geográfico como un problema de clustering particional y además de tipo diseño territorial DT.
En este trabajo, el problema de conglomerado geográfico consiste en resolver la agrupación que optimice compacidad geométrica minimizando distancias entre agebs.

Una vez que esta agrupación es obtenida, se calcula homogeneidad a dicha partición compacta. El resultado es una partición compacta y homogénea, definida en este trabajo como zonificación óptima y se resuelve en el capítulo 5.

Estrategia de solución en zonificación óptima
En la zonificación óptima, la creación de grupos toma como datos a agregar a las agebs, descritas por un vector de variables poblacionales de carácter censal y por coordenadas geográficas.
En este proceso de agregación, de naturaleza NP-difícil, las clases de zonas que se forman están sujetas al cumplimiento de la minimización de dos funciones de costo: una medida de distancia en el espacio geográfico y otra de equilibrio u homogeneidad para variables poblacionales. Esto implica que la optimización de dichas funciones objetivo se aborde con algún método heurístico, mientras que la solución a la competencia entre estas dos funciones se resuelva con métodos multiobjetivo para obtener un conjunto de soluciones eficientes no dominadas.

La estrategia metodológica para resolver el problema de zonificación óptima ZO es la siguiente:

a) Formular un modelo combinatorio de particionamiento geográfico que considere las propiedades de diseño territorial (restricciones de las agebs: compacidad y homogeneidad).
   Se resuelve primero el particionamiento geográfico atendiendo solo una función objetivo: compacidad geométrica.
   Una vez obtenida la partición compacta, se calcula paralelamente la homogeneidad a dicha partición.
b) Los puntos anteriores exigen desarrollar un algoritmo que optimice tanto compacidad como homogeneidad con el uso de métodos heurísticos para garantizar la generación de soluciones de calidad.
c) Mientras se generan las soluciones con el método heurístico, es necesario emplear un método multiobjetivo para la construcción de un conjunto de soluciones eficientes no dominadas en una frontera de Pareto [Pareto96].

Objetivo
Siendo zonificación óptima un problema de tipo diseño territorial que debe dar respuesta a distintos problemas de distribución poblacional, el objetivo de este trabajo de tesis se sitúa en el diseño, desarrollo e implementación de un algoritmo de particionamiento multiobjetivo sobre agebs donde debe optimizarse una función de costo bi-objetivo compuesta por compacidad para ubicación geográfica de las agebs y homogeneidad para las características censales/poblacionales.

Para resolver ZO, se ha diseñado un modelo de optimización combinatoria bi-objetivo de agrupación sobre unidades geográficas agebs.
La heurística que optimiza esta función bi-objetivo es búsqueda por entorno variable y para obtener el frente de Pareto se hace uso de un método de la teoría del orden que encuentra un conjunto de soluciones no-dominadas a través de los puntos minimales definidos como minimales-ZO.
Dichos puntos, son pares de soluciones no comparables y además satisfacen la dominancia Pareto.

Contribuciones
Las contribuciones de este trabajo se pueden ver en 4 direcciones:

Diseño Territorial
En esta área, se presenta una nueva metodología de agrupación multiobjetivo para unidades geográficas donde se optimizan dos objetivos: compacidad y homogeneidad. Generalmente, cuando el problema en diseño territorial es multicriterio/multiobjetivo, se optimiza una función y el resto se establece como restricciones.

Agregación Territorial
A pesar que la agregación en diseño territorial es un problema NP-duro, es necesaria su solución en problemas de diseño territorial.
La agrupación para los objetos espaciales agebs de este problema de zonificación óptima, se ha resuelto con un método de particionamiento inspirado en nubes dinámicas, aspectos básicos de optimización de k-medias y particionamiento alrededor de los medoides con el uso de búsqueda

por entorno variable (esta heurística no había sido empleada para este tipo de problemas).
En este trabajo se han aprovechado las propiedades tanto de los métodos de particionamiento clásicos como de búsqueda por entorno variable para responder a la agregación en zonificación. Esto se traduce en una importante aportación de agregación para diseño territorial.
En la tabla 2.5.2 se presenta un interesante resumen de las metodologías en diseño territorial donde se distingue claramente tres aportaciones de esta tesis en esta área.

Optimización Heurística
Búsqueda por entorno variable es una heurística relativamente reciente, sin embargo, hasta donde se sabe, no ha sido muy aplicada a resolver problemas de diseño territorial. En este trabajo se ha demostrado estadísticamente que búsqueda por entorno variable es un método heurístico de eficiente aproximación en problemas de particionamiento compacto para datos espaciales agebs.

Sistemas de información geográfica SIG
Muchos han sido los esfuerzos utilizados para articular los resultados de agregación para datos geográficos con un sistema de información geográfico. En esta tesis se ha integrado las estructuras de los grupos obtenidos por el particionamiento para agebs con los esquemas del SIG Map-X, de tal modo que es posible observar en mapas los grupos resultantes. Esto implica una aportación en el ámbito de los sistemas de información geográfica con diseño territorial.

Particionamiento multiobjetivo
El particionamiento multiobjetivo ha sido un tema poco estudiado debido a la complejidad implícita. Sin embargo, en este trabajo se aporta en esta línea, un método multiobjetivo para crear un conjunto de soluciones eficientes de particiones compactas y homogéneas sobres agebs.
El método que construye este conjunto de soluciones, se apoyó de la teoría del orden, una rama de las matemáticas discretas.
Haciendo uso de la dominancia Pareto y dentro de las soluciones de particionamiento que se generan por búsqueda por entorno variable, se buscan soluciones no comparables y al mismo tiempo no dominadas interpretadas como minimales. De esta manera, este método es capaz de obtener un conjunto de soluciones eficientes no dominadas (particiones compactas y homogéneas).

Desarrollo de zonificación óptima
Cuando se han utilizado métodos alternativos para solucionar la clasificación geográfica obteniéndose óptimos locales como una respuesta no satisfactoria, se han buscado otros métodos de clasificación que respondan a una agrupación confiable.
Se ha comprobado que los algoritmos de particionamiento son un método adecuado para agrupar datos espaciales, por tanto, se ha puesto atención a un algoritmo exhaustivo de particionamiento conocido como pam (partitioning around mediods), que a pesar de no ser de propósito específico para agrupar unidades territoriales, si ha sido posible adaptarlo para agrupar agebs. Además, se implementó a pam en un esfuerzo por experimentar su alto costo computacional con las agebs y al mismo tiempo disponer de un valor “óptimo” para problemas pequeños.
Para reducir el alto costo de pam se ha hecho uso de recocido simulado, lo que ha permitido dotar a pam de herramientas adicionales para escapar con mayor eficiencia de soluciones sub-óptimas [Bernábe06b, 08].

Sin embargo, el propósito ha sido diseñar un algoritmo ideal para agrupar agebs. En este sentido y con el propósito de acercarse a la solución de ZO, se comenzó con la construcción de un algoritmo de particionamiento *propio* para agebs llamado conglomerado geográfico que utiliza recocido simulado donde se optimiza la función objetivo de compacidad [Bernábe09].
Por otro lado, al mismo algoritmo de particionamiento, se le insertó búsqueda por entorno variable.
Para evaluar la calidad de las soluciones generadas por las dos heurísticas, se ha aplicado un diseño de experimentos estadístico factorial box behnken [Bernábe09a]
Los resultados de esta evaluación proporcionaron evidencia suficiente para afirmar que búsqueda por entorno variable ofrece “mejores” soluciones que recocido simulado para el problema de

conglomerado geográfico.
Este resultado ha permitido elegir búsqueda por entorno variable como método heurístico de aproximación para optimizar los dos objetivos en zonificación óptima. Las soluciones generadas por la heurística también son evaluadas con el orden Pareto NoComparable y con la dominancia Pareto. El conjunto de soluciones resultantes proporciona un conjunto de soluciones no dominadas (minimales), que forman el frente de Pareto.

Para fines de organización, esta tesis se encuentra estructurada en 5 capítulos.
En el primer capítulo se presenta la descripción del problema de zonificación óptima y un resultado antecedente de alto costo computacional, que da lugar a proponer un algoritmo de particionamiento adecuado para la solución aproximada a ZO.
En el capítulo 2 se resume el estado del arte en diseño territorial, métodos de solución multiobjetivo y clustering multiobjetivo.
El capítulo 3 se ocupa de mostrar los temas teóricos de los cuales se ha apoyado esta tesis: particionamiento, optimización combinatoria, recocido simulado y búsqueda por entorno variable.
En el capítulo 4 se describe la modelación y los algoritmos que resuelven el problema de conglomerado geográfico (particionamiento que minimiza compacidad). Las dos propuestas heurísticas que se han insertado por separado al modelo de conglomerado son búsqueda por entorno variable y recocido simulado. Una evaluación de ambas heurísticas también se describe en este capítulo con el fin de elegir aquella que tenga un “mejor comportamiento” para usarla como método de aproximación en ZO.
Para la solución a zonificación óptima, que es el propósito principal de esta tesis, en el capítulo 5 se expone la modelación y metodología que resuelve el particionamiento con compacidad geométrica y homogeneidad para variables. Se describe y aplica un método que se apoya en teoría del orden para generar el frente de Pareto, el cual consiste en un conjunto de soluciones óptimas no dominadas: minimales-ZO.
Por último se discuten las conclusiones y el trabajo futuro; un anexo A sobre la integración de un sistema de información geográfica y análisis de datos espaciales, y un anexo B donde se propone un algoritmo de clasificación en paralelo sobre redes complejas aplicado a las agebs [Ochoa09] y finalmente las referencias.

Resumen

El problema de zonificación óptima es un problema de diseño territorial que debe resolver la agrupación compacta y homogénea de áreas geoestadísticas básicas agebs para dar respuesta a situaciones diversas de concentración y/o distribución de la población[1] y de espacios territoriales[2].

Dado que este problema es de elevado costo computacional, la solución a zonificación óptima se formula con un modelo de optimización combinatoria que plantea y resuelve la tarea específica de particionamiento geográfico bi-objetivo bajo el cumplimiento de restricciones espaciales para unidades territoriales de una zona metropolitana.

En este sentido, el presente trabajo de tesis se ocupa del diseño y desarrollo de un algoritmo de particionamiento bi-objetivo que optimice la función de costo integrada por compacidad geométrica en la ubicación espacial[2] y homogeneidad para variables censales[1] de áreas geoestadísticas básicas agebs.

Para aproximar estas dos funciones objetivo, se hace uso de búsqueda por entorno variable y al mismo tiempo, para resolver el conflicto entre compacidad y homogeneidad, se aplica un método multiobjetivo de eficiencia de Pareto, el cual se apoya en la teoría del orden con el fin de proporcionar un conjunto de soluciones óptimas no dominadas llamadas minimales-ZO.

Abstract

The problem of optimum zonification is a problem of territorial design that must solve the compact and homogeneous grouping of Basic Geo-Statistical Areas (AGEBs) to answer different situations of concentration and/or distribution of population[1] and territorial spaces[2].

Since this problem has a high computational cost, the solution to an optimum zonification is formulated as a combinatorial optimization model that states and solves the specific task of bi-objective geographical partitioning, fulfilling spatial restrictions for territorial units of a metropolitan zone.

In this sense, this thesis work deals with the design and development of a bi-objective partitioning algorithm that optimizes the cost function comprised by geometric compactness of the spatial location[2] and homogeneity for the census variables[1] of basic geo-statistic areas (AGEBs).

To approximate these two objective functions, variable neighborhood search is used and at the same time, to solve the conflict between homogeneity and compactness, Pareto's multi-objective efficiency method is applied, which is supported by order theory, with the goal of providing a set of non-dominated optimal solutions called ZO-minimals.

Capitulo1: Descripción del problema

Introducción

En este capítulo se describe el problema de particionamiento geográfico PG (llamado también conglomerado geográfico), y por otro lado el problema de zonificación óptima ZO, el cual es el propósito de esta tesis.
Se justifica a zonificación óptima y a PG como un sub-problema de diseño territorial. Se presentan los argumentos necesarios para incluir algoritmos de particionamiento y heurísticos en la resolución tanto de PG como de ZO.
Por otro lado, se menciona brevemente un algoritmo de particionamiento sobre “medoides” como un antecedente de alta complejidad computacional que dio lugar a proponer métodos alternativos de solución aproximados para la clasificación sobre agebs.

1.1 Justificación y supuestos en particionamiento geográfico y zonificación óptima

Para proporcionar argumentos suficientes sobre los supuestos básicos en zonificación óptima ZO y definir a ZO, es necesario citar algunos fragmentos importantes en relación a diseño territorial DT.
Las siguientes líneas proporcionan aspectos relevantes para comprobar la pertenencia de ZO y conglomerado geográfico en diseño territorial.

El estado actual de diseño territorial

Los problemas de diseño de territorio consisten en la agrupación de áreas o unidades geográficas pequeñas (áreas básicas) en conglomerados/grupos geográficos más grandes llamados territorios, de tal forma que la agrupación (agregación territorial) aceptable es aquella última que cumple con los criterios predeterminados del problema a tratar.
Estos criterios se distinguen como restricciones espaciales de continuidad y compacidad [Kalcsics05].

Distritación electoral es la aplicación más conocida de DT y consiste en particionar unidades de área (unidades generalmente administrativas), en un predeterminado número de zonas (los distritos) tal que las unidades en cada una de las zona sea contigua, cada zona debe ser geográficamente compacta y la suma de las poblaciones de las unidades de área en cualquier distrito es tan homogénea como sea posible o con cierta tolerancia dentro de un rango predeterminado [Mebhrotra98].
Esta descripción de distritación electoral junto con la definición de DT de [Kalcsics05], convienen al propósito de definición de agrupamiento en zonificación óptima y conglomerado geográfico que más adelante se definen.

Complejidad computacional en diseño territorial

Tan solo por que los problemas de diseño territorial incluyen procesos de particionamiento, el cual se ha demostrado como NP-duro, se asume que DT es un problema intratable. En la literatura se encuentra propuestas importantes que justifican la complejidad computacional de DT.

En [Anderberg73, Hartigan01] se encuentra una amplia serie de comentarios y referencias de algoritmos para el problema NP-difícil de agrupamiento.
En [Garey79] se muestra que la capacidad del problema de clustering es NP-completo, y que los métodos de programación de algoritmos exactos o de optimización entera para resolver este problema en instancias grandes son ineficientes.
El trabajo publicado en [Kalcsics05] presenta muchas y excelentes referencias sobre variedad de problemas de DT que son de naturaleza NP-difícil.
[Hess71] es una buena cita donde se distingue la complejidad de DT para el caso de distritación y de territorio de ventas que se intentó resolver a mediados de los años 60´s.
En términos de complejidad computacional, [Altamn97] muestra que el problema de diseño de zonas es NP-completo.

Los problemas de diseño territorial tienen aplicaciones muy diferentes que van desde distritación política (DP) hasta diseño territorial para escuelas, localización de áreas para asistir a servicios de emergencia (diseño de territorio de servicios).

Existen diferentes modelos de optimización que incluyen algoritmos heurísticos para dar respuesta al problema de diseño territorial. Trabajos que utilizan algoritmos para el diseño de zonas y que ya han sido usados en gran variedad de tareas son:
Distritación de escuelas [Ferland90], el diseño de zonas para análisis epidemiológicos y socioeconómicos [Haining94, Openshaw95a, Openshaw99], diseño de territorios de ventas [Fleischmann98] y diseño para procesos de censo [Martin97, 98], entre otros.

El supuesto inicial para ZO, dado un análisis exhaustivo de las referencias anteriormente citadas, es que zonificación óptima es una aplicación de diseño territorial debido a:
a) Los datos a clasificar son espaciales de naturaleza compleja (agebs).
b) La gran cantidad de agebs y el proceso de clasificación implícito se traduce en unidades de cómputo de alto costo, dando origen a un problema combinatorio.
c) La clasificación geográfica debe considerar las características del problema como restricciones que se interpretan y modelan a través de medidas de similitud y/o disimilitud (compacidad geométrica y homogeneidad).

1.2 Zonificación en diseño territorial y particionamiento
Claramente se ha identificado a ZO como un sub-problema de DT. Esta distinción es mayor por el proceso de agrupamiento exigido tanto en DT como en zonificación óptima.

El tema de clasificación se extiende en el capítulo 3, por ahora, se subraya que dentro de los métodos de clasificación existentes, se ha empleado en la agrupación para las agebs, las propiedades de los algoritmos de particionamiento, en particular el método de nubes dinámicas y particionamiento alrededor de los medoides.

De acuerdo a lo anterior, se proponen las siguientes definiciones:

Definición 1.
La *agregación* en zonificación óptima es un problema de particionamiento sobre áreas geoestadísticas básicas agebs en grupos geográficos más grandes llamados grupos o conglomerados, donde la partición final es aquella que cumple con los criterios geográficos de compacidad geométrica y homogeneidad para las variables censales de las agebs.

Definición 2.
El *problema de zonificación óptima* es un problema de diseño de territorio y consiste en la agrupación de agebs en grupos geográficos más grandes, de tal forma que la agrupación o partición aceptable es aquella última que optimice los criterios de compacidad y homogeneidad. Las agebs que pertenecen a cada grupo deben estar muy cercanas geográficamente de tal forma que cada grupo debe ser lo más compacto posible, y, la suma o promedio de un variable censal en cada grupo debe ser tan homogénea como sea posible con una tolerancia dentro de un rango aceptable en los valores de las variables de las agebs.

Del mismo modo, conglomerado geográfico es un problema derivado de DT dado que:
a) Los datos a clasificar tienen una clara componente espacial y una descripción de variables de tipo censal (agebs).
b) El proceso de particionamiento tiene que agrupar una importante cantidad de agebs lo que implica resolver un problema combinatorio de alto costo.
d) La clasificación geográfica debe considerar las características del problema como restricciones que se interpretan y modelan a través de medidas de disimilitud (compacidad geométrica).

De acuerdo al párrafo anterior, se propone la siguiente definición:

Definición 3.
Un problema de diseño de grupos geográficos, es llamado conglomerado geográfico cuando n unidades de áreas agebs son particionadas en k zonas tal que el valor de función de minimización de distancias entre agebs es optimizado y depende de las restricciones en la topología de la zona metropolitana ($k \ll n$).
Se admite que siendo DT un problema de complejidad computacional, el uso de métodos heurísticos en la agregación es la propuesta más utilizada en la mayoría de problemas DT y consecuentemente también para el conglomerado geográfico y zonificación óptima.

1.3 Estructura de desarrollo en zonificación óptima

Para resolver el problema optimización geográfica y zonificación óptima, en esta tesis se presenta una propuesta de programación matemática combinatoria que plantea y resuelve el particionamiento geográfico sobre agebs. Se optimiza la minimización de una función bi-objetivo integrada por compacidad y homogeneidad que actúa sobre unidades territoriales agebs de una zona metropolitana para dar respuesta a problemas poblacionales y/o de carácter censal.

El proceso de particionamiento para agebs y de optimización para la función bi-objetivo en zonificación óptima se ha desarrollado de la siguiente manera:

Primero: Se resuelve el problema de particionamiento optimizando compacidad geométrica con dos métodos de gran eficiencia en la resolución heurística de problemas difíciles de optimización combinatoria: búsqueda por entorno variable VNS y recocido simulado RS [Bernábe09, 09a].
Segundo: Se evalúa la calidad de las soluciones con un diseño de experimentos box-behnken y superficies de respuestas. Se concluye que para el problema que se trata en esta tesis, VNS es "mejor" que RS [Bernábe09, 09a, 09b].
Tercero: De acuerdo al punto anterior, se hace uso de VNS para asegurar la obtención de soluciones aproximadas en la función bi-objetivo de compacidad y homogeneidad.
Cuarto: Para dar respuesta a la mezcla de las 2 funciones en conflicto, se ha aplicado un método para obtener soluciones no dominadas, el cual encuentra puntos no comparables que forman el frente de Pareto [Pareto96].

Aportaciones particulares de zonificación óptima en diseño territorial
De la literatura existente, es posible afirmar que el problema de ZO que se resuelve en este trabajo, se ubica como una aportación en DT debido a los siguientes argumentos:
Las unidades geográficas a agrupar son agebs, en México, los problemas de esta categoría han utilizado manzanas para resolver la agrupación.
Ni recocido simulado ni VNS han sido utilizados como métodos de aproximación para este tipo de datos agebs.
Los métodos heurísticos aplicados no han sido evaluados sistemáticamente con box behnken y superficies de respuestas para determinar valores adecuados de los parámetros.
La mezcla de dos funciones objetivo en problemas de agregación territorial es un tema poco atendido. Una aportación importante aquí consiste en insertar al algoritmo de particionamiento para zonificación óptima, un método basado en teoría del orden para generar soluciones no dominadas [Günter01].
Hasta donde se sabe, la integración del sistema de información geográfica map-x a un método de regionalización no se ha reportado.

1.4 Aproximaciones no exactas en regionalización

Dada la revisión de la literatura en DT, se ha resumido y concluido que al menos existen dos aproximaciones metodológicas no exactas para la solución del problema de regionalización, las cuales pueden ser divididas en dos grandes bloques:

1) Aplicación de métodos convencionales de agregación (jerárquicos o de partición) en los cuales la restricción de continuidad geográfica se satisface indirectamente a través de la inclusión de las variables geográficas (por ejemplo distancias entre áreas) dentro del grupo de variables de clasificación. La inclusión de este tipo de variables favorecerá la creación de regiones

espacialmente compactas, lo cual se traduce en la satisfacción de la restricción de continuidad geográfica.
Esta aproximación suele ser más apropiada cuando se tiene un elevado número de áreas por agregar. También es útil cuando la compacidad es una característica deseable. Por otro lado, la utilización de este tipo de aproximaciones implica, en algunos casos, el otorgar gran importancia a las variables geográficas para garantizar la satisfacción de la restricción de continuidad geográfica. Esto puede hacer que el papel de las variables no geográficas (por ejemplo variables socioeconómicas) dentro el proceso de agregación pasaría a ser secundario.

2) La utilización de métodos de agregación que incluyen explícitamente la restricción de continuidad geográfica. Estos métodos requieren como información de entrada no sólo las variables de clasificación, si no también información sobre las relaciones de vecindad entre las áreas (generalmente una matriz binaria de continuidades geográficas).
Estos modelos suelen ser utilizados cuando no se desea imponer condiciones sobre la forma de las regiones, esto es, cuando la condición de compacidad no es requerida. (Más información sobre estos modelos puede ser encontrada en [Murtagh95, Gordon99, Duque06].
Se distinguen características de ZO en el primer bloque, el cual, ha merecido especial atención en este trabajo.

1.4.1 Antecedentes: una propuesta de agrupamiento exhaustiva a problemas pequeños de diseño territorial

Los inicios “prehistóricos” para resolver el problema de agrupar agebs consistió de la aplicación de técnicas multivariadas. Los grupos resultantes estaban formados por agebs “dispersas” (no compactos) y óptimos locales, lo cual viola los principios de agregación territorial. Estos resultados ha sido reportados en [Bernábe04, 05, 06c].

Se admite así que estas técnicas, no son las adecuadas para agrupar agebs, lo que implica reemplazar este estudio con otros métodos. En este sentido, se procede a la implementación de un algoritmo de clasificación que consiga una clasificación compacta con un bajo costo de computo.

Si bien k-medias y sus variantes son algoritmos de agrupamiento ampliamente conocidos y aplicados que pueden ser una buena opción para agrupar agebs, tambien es cierto que muchas de sus implementaciones generan soluciones locales o que bajo alteraciones convenientes pueden dar buenos resultados tanto en tiempo como en calidad.
En este sentido, se ha hecho un analisis de los metodos de clustering en “Clustering in an object oriented environment” [Anja96]. Este análisis aportó información importante para elegir a pam (particionamiento alrededor de los medoides) para agrupar agebs como primera opción para obtener una agrupación “exacta” en problemas pequeños.

Métodos de particionamiento de k medoides

El método de clustering de k-medoides encuentra objetos representativos llamados medoides en los conglomerados. Un medoid se puede definir como el objeto de un grupo, cuya disimilitud media a todos los objetos en el grupo sea mínima, es decir se trata de un punto más céntrico en el conjunto de datos dado.

El exponente de este tipo de particionamiento es pam (partitioning around medoids). Pam, propuesto por Kaufman y Rousseeuw [Kaufman87], comienza de un conjunto inicial de medoides e iterativamente reemplaza uno de los medoides por uno no medoide si este mejora la distancia total del conglomerado resultante.
Pam trabaja muy bien para conjuntos de datos pequeños pero no para gran tamaño de datos. El costo computacional es $O(k(n-k)^2)$ [Mark02].

Sin embargo, pam por la flexibilidad en el método puede aplicarse a problemas de carácter espacial dado que divide un número de observaciones en k conglomerados aún cuando los datos tienen atributos espaciales o atributos geográficos [Web1].

En la práctica, aplicando pam a las agebs, se pudo comprobar el costo computacional. Estos resultados ya han sido reportados [Bernábe06b].

1.4.2 Agrupación alrededor de los medoides para agebs (pam)

Pam es un método no jerárquico donde el conjunto de datos es particionado en un número previamente especificado de k conglomerados, y luego iterativamente se asignan las observaciones a los conglomerados hasta que se satisface algún criterio de parada (función a optimizar), por ejemplo, que la suma de cuadrados dentro de los conglomerados sea mínima [Kaufman87].

La función pam está basada en la búsqueda de k objetos representativos llamados medoids (medoides) entre los objetos del conjunto de datos. Estos medoids son calculados tal que el total de disimilitud de todos los objetos hacia su medoide más cercano es mínima: es decir, la meta es encontrar un subconjunto $\{m_1 \ldots m_k\} \subset \{1 \ldots n\}$ que minimiza la función objetivo:

$$\sum_{i=1}^{n} \min_{t=1,\ldots,k} d\,(i, m_t)$$

Cada objeto es asignado al conglomerado correspondiente al centroide más cercano. Esto es, el objeto i es colocado en el conglomerado v_i cuando el medoide m_{vi} está más cercano a i que cualquier otro medoide m_w, o:

$$d(i, m_{v_i}) \leq d(i, m_w) \quad \text{para todo} \quad w = 1, \ldots, k$$

Actualmente el algoritmo de pam consta de 2 pasos:

primero

construir los centroides iniciales:

m_1 es el objeto con el más pequeño $\sum_{i=1}^{n} d(i, m_1)$

m_2 disminuye el objetivo (1) tanto como sea posible

.

.

m_k disminuye el objetivo (1) tanto como sea posible

segundo

intercambio:

repetir hasta lograr convergencia

considerar todos los pares de objetos (i, j) con

$i \in \{m_1, \ldots, m_k\}$ y $j \notin \{m_1, \ldots, m_k\}$

y hacer el intercambio $i \leftrightarrow j$ para cualquiera que decrezca más el objetivo.

La función objetivo de pam depende de las disimilitudes entre los objetos por tanto esta función solamente necesita la matriz de distancias como entrada.

Cabe señalar que la variante interesante de pam con respecto a k-medias es que la meta de k-means es minimizar una suma de distancias euclidianas cuadradas, implícitamente asumiendo que cada conglomerado tiene una distribución normal esférica. La función pam es más robusta porque esta minimiza una suma de disimilitudes no cuadradas. Además pam no necesita suposiciones iniciales para los centroides de los conglomerados, contrariamente a k-means.

Aplicación y resultados de agrupación para agebs con particionamiento sobre medoides

Es importante mencionar que el proceso de preparación de agebs consiste en concentrarlos y adaptarlos a una matriz de distancias, así es posible contar con una matriz correcta en la entrada al agrupamiento pam (ver anexo A).

Una vez que se cuenta con la matriz de distancias y a pam como algoritmo de particionamiento, el resultado de la agrupación es una partición que contiene los códigos de agebs.

A pesar de que pam no es algoritmo de regionalización, aún con su complejidad computacional, ha sido posible emplearlo para agrupar agebs para instancias pequeñas, y solo en estas condiciones, se puede decir que pam pertenece a las herramientas de regionalización de alta complejidad.
Para reducir el costo de pam, se pueden tener aproximaciones no exactas si se hace uso de algún método heurístico. Se ha elegido a recocido simulado como método de aproximación en pam para escapar eficientemente de óptimos locales [Bernábe08].
Se afirma así que el problema de clasificación de agebs con pam y recocido simulado pertenece al problema de regionalización aproximada.

Por otro lado, olvidando a pam, se analizaron otros métodos alternativos de particionamiento para construir un algoritmo de particionamiento ***propio y adecuado*** a las agebs que optimice compacidad con el uso de recocido simulado y de búsqueda por entorno variable.
Para ello, se ha propuesto un modelo de particionamiento geográfico considerando como función objetivo la minimización de distancias para hacer cumplir la compacidad y las restricciones se plantean de acuerdo a las propiedades de particionamiento [Bernábe09, 09a]. Estos algoritmos se exponen en el capítulo 4.

El objetivo de obtener primero una solución de partición compacta, reside en que es posible calcular la homogeneidad a dicha solución y resolver así el problema de optimización para zonificación óptima (además de evaluar las soluciones bajo la dominancia Pareto NoComparable definida en el capítulo 5).

1.5 Formulación matemática de particionamiento multiobjetivo para zonificación óptima

El propósito de esta sección es presentar a zonificación óptima en un modelo matemático. Se precisa este modelo en el capítulo 5.

La función objetivo de compacidad se detalla en el capítulo 4, aquí se considera dicha función con la función de homogeneidad en un modelo matemático para zonificación óptima con fines introductorios.

Definición de zonificación óptima ZO
Sea una ageb un dato espacial definido por sus componentes en el espacio y una descripción de 144 variables censales dada por un vector. La siguiente nomenclatura básica para formular ZO es:

M= mapa territorial
T= territorio
DT= datos censales
MS= matriz de disimilitud
UGB= unidad geográfica básica (ageb)
GT= grupo territorial
m = número de GT
n= número de UGB tal que m<<n
i= índice de grupo territorial
j= índice de unidad geográfica básica
Ci= centroide del i-ésimo grupo territorial GTi

El modelo en cuestión es entero mixto y hace uso de las variables binarias para modelos de este tipo.

A es el conjunto de atributos cuantificables del cual será seleccionado un subconjunto de variables de acuerdo al problema.
VA_{kj} es el valor del k-ésimo atributo contenido en la j-ésima UGB_j
α_k , β_k parámetros de tolerancia para VA_k en cualquier UGB
$(VA_{ki}) = \sum_{j=1}^{n} VA_{kj} X_{ij}$ es el valor para el k-ésimo GT

(X_{ij} es la variable de decisión)

$\overline{VA_k} = i / m \sum_{j=1}^{n} VA_{kj}$ es el valor meta para el k-ésimo atributo en cualquier UGB

$d_{ij} = d(C_{i,} UGB_j)$ X_{ij} es la distancia de la j-ésima UGB en el i-ésimo GT respecto a su centroide.

Entonces en cálculo de distancias entre agebs puede expresarse como:

$$Dij = \sum_{j=1}^{n} d(C_i, UGB) X_{ij} \qquad a)$$

Y el valor de homogeneidad para variables se expresa como:

$$H_{\text{variable-ageb}} = \sum_{i=1}^{m} (\overline{VA_k} - VA_{ki}) \qquad b)$$

Entonces, si

f1: es el costo de minimizar la distancia entre agebs de acuerdo a la ecuación a), y
f2: es el costo de minimizar la homogeneidad de una variable censal de las agebs de acuerdo a la ecuación b).

El problema se plantea como:

Minimizar $y = f(x) = (f1, f2)$

sujeto a

$GT_i \neq \varnothing$ para i = 1,…, k (los grupos no son vacíos)

$GT_i \cap GT_j = \varnothing$ para $i \neq j$ (no existen agebs repetidos en distintos grupos)

$\bigcup_{i=1}^{k} GT_i = UGB$ (la unión de todos los grupos son todos los agebs)

$\alpha_k \leq VA \leq \beta_k$ (cotas para las variables)

$\sum_{i=1}^{m} X_{ij} = 1$ (es la asignación de agebs)

$X_{ij} = 1$ si $UGB \in GT_i$ o $X_{ij} = 0$ si $UGB \notin GT_i$ (son las variables de decisión)

$y = (y1, y2) \in Y \subset R^2$ (es el vector objetivo)

Capitulo 2: Estado del arte

Introducción

En este capítulo se presenta un resumen de las diversas alternativas bibliográficas conocidas hasta hoy, para solucionar problemas de diseño territorial. Se muestran además algunas propuestas importantes para resolver problemas multiobjetivo de las cuales se han aprovechado algunos aspectos para solucionar zonificación óptima.

2.1 Agregación y diseño territorial: revisión de la literatura

Los problemas de diseño territorial generalmente son planteados desde diferentes enfoques dado que existe una gran cantidad de aplicaciones.

Sin embargo, independientemente del problema, existen factores comunes en diseño territorial como son las definiciones, los modelos matemáticos que se utilizan de apoyo y los alcances.

Puede observarse en la literatura sobre regionalización o diseño territorial, que los autores proponen términos que pueden dar lugar a interpretaciones diversas: zonificación, distribución regional, categorización de zonas, agregación territorial, reordenamiento territorial o diseño de regiones. Sin embargo, la terminología no marca notables diferencias conceptuales entre unas y otras.

Por ejemplo, el término de alineación es utilizado con frecuencia en lugar de diseño [Kalcsics05]; agregación refiere a agrupación territorial sobre datos espaciales o unidades geográficas, clasificación y clustering generalmente se admite que son sinónimos, regionalización se identifica como diseño territorial etc.

Diseño de territorio

El diseño de territorio puede ser visto como un problema de agrupación de áreas geográficas pequeñas (áreas básicas) en conglomerados geográficos más grandes llamados territorios, de tal forma que la agrupación aceptable es aquella última que cumpla con criterios predeterminados del problema que ocupa. Dependiendo del contexto, estos criterios bien pueden ser de motivo económico (promedio de ventas potenciales, trabajo o número de vendedores) o tienen un fondo demográfico (número de habitantes, población votante). Es en este sentido donde empiezan a formalizarse conceptos de restricciones espaciales, continuidad y compacidad y que además son muy demandadas [Kalcsics05, Zoltners83].

El diseño de zona es un problema geográfico complejo presente en varias tareas justamente geográficas siendo distritación electoral el caso más conocido [Macmillan94].

El problema de diseño o planeación de zonas o diseño de zonas geográficas ocurre cuando n unidades de áreas son agregadas en k zonas tal que un valor de función es optimizado, dependiendo de las restricciones en la topología de las zonas (por ejemplo conectividad interior) [Bação05].

Distritación electoral consiste en particionar unidades de área (unidades generalmente administrativas), en un predeterminado número de zonas (los distritos) tal que las unidades en cada una de las zona sea contigua, cada zona debe ser geográficamente compacta y la suma de las poblaciones de las unidades de área en cualquier distrito es tan homogénea como sea posible o con cierta tolerancia dentro de un rango predeterminado [Mehrotra98].

2.2 Naturaleza NP-duro de diseño territorial

En el capítulo 1 se mencionó la situación que rodea a DT como un problema de categoría NP-difícil. Se repiten las citas para subrayar esta complejidad en el contexto de estado del arte en DT.

En [Anderberg73, Hartigan01] se hallan varios comentarios y referencias de algoritmos para el problema NP-difícil de agrupamiento.

La capacidad del problema de clustering como NP-completo y los métodos ineficientes para algoritmos exactos o de optimización entera para resolver este problema en instancias grandes, puede verse con detalle en [Garey79].

[Kalcsics05] ofrece muchas y excelentes referencias sobre variedad de problemas de DT que son de naturaleza NP-difícil.
En términos de complejidad computacional, [Altman97] muestra que el problema de diseño de zonas es NP-completo.

Aplicaciones
Los problemas de diseño territorial son motivados por aplicaciones muy diferentes que van desde distritación política hasta diseño territorial para escuelas, localización de áreas para asistir a servicios de emergencia (diseño de territorio de servicios), donde las principales aplicaciones son distritación política y diseño territorial para servicios.

Existen diferentes modelos de optimización que incluyen algoritmos heurísticos para dar respuesta al problema de diseño territorial. Trabajos que utilizan algoritmos para el diseño de zonas y que ya han sido usados en gran variedad de tareas son:
Distritación de escuelas [Ferland90], el diseño de zonas para análisis epidemiológicos y socioeconómicos [Haining94, Openshaw95, Openshaw99], diseño de territorios de ventas [Fleischmann88] y diseño para procesos de censo [Martin97, 98], entre otros.

La tarea de diseño de territorio para ventas es muy común en las compañías que requieren de operar su fuerza de ventas y necesitan dividir el área de mercado en regiones con responsabilidad. De igual manera está relacionado el problema de diseño de territorios de servicio para atender a vendedores o algún medio técnico [Kalcsics05].

2.3 Criterios y requerimientos generales: modelación y propuestas de solución
En un problema específico de planeación de distritación política, donde el área bajo estudio es un área gubernamental, tal como una ciudad o estado; esta área tiene que ser dividida en un número dado de territorios. Como cada territorio elige un solo miembro que lo represente, el principal criterio para la planeación es tener aproximadamente el mismo número de votantes en cada territorio, esto es, territorios de tamaño similar, con el propósito de respetar el principio de “un hombre-un voto” [Kalcsics05].
La solución a distritación política requiere un proceso de clasificación de zonas basado en restricciones, donde la agrupación fuerza generalmente a realizar muchas comparaciones, y el problema se convierte en NP-duro [Bação05]; por tanto se exige un modelo de optimización apoyado en una metaheurística para lograr una agrupación con buena solución.
El problema de diseño de zona no solo comprende un gran número de tareas geográficas, también este problema exige ser descrito y formalizado matemáticamente. Incluso, este problema puede verse como un caso especial de la mochila o de clustering [Bação05].

Tanto planeación de zonas como el proceso de administración y ordenamiento de zonas tienen como fin definir bien unidades de zonas que puedan ser “manejadas” basándose en características específicas y necesidades propias de un problema. Una vez establecida esta definición, la planeación y manejo de estas “unidades de zonas” constituyen la columna vertebral de la estrategia total [Bação05a].
El objetivo de un plan de división de zonas consiste en delinear áreas más pequeñas que pueden manejarse de una manera más flexible [Bação05a].

Durante las últimas décadas, ha habido varias propuestas para ello y algunas aún se encuentran en vías de desarrollo, pero sin perder el propósito de identificar estas unidades de área.
Dependiendo del problema, para la agregación y delineación de zonas, suelen ser considerados criterios físicos, límites políticos, límites administrativos, distancias arbitrarias o unidades medioambientales.

Para la mayoría de los proyectos que requieren de diseño, programación y administración de una zona costera, se usan los límites administrativos en lugar de adoptar las propiedades que reflejan el impacto del ecosistema que vienen fuera del área.

La tarea de diseñar una zona costera de unidades de área puede verse como un caso especial del problema más general de diseño de zona geográfico [Bação05a].

Los requerimientos típicos de planeación de áreas, consiste en el diseño de territorios que sean similares en tamaño, por ejemplo, en relación con el conflicto de ventas potenciales o cargas de trabajo o reducir dentro de los territorios, el tiempo de atención a clientes o servicios inesperados. Por otro lado, debido a las regulaciones legales, la introducción de nuevos productos en el mercado, cambios en el mercado etc., las decisiones para el diseño de territorio deben ser frecuentemente reevaluadas, especialmente cuando se tiene un gran número de áreas básicas y territorios. Esta situación se convierte en una tarea muy larga, y por consiguiente se aborda como proceso combinatorio, que debe ser tratado con algoritmos de optimización para acelerar el proceso y aproximarse a obtener una buena solución [Kalcsics05].

La tarea típica del diseño de la zona presenta generalmente restricciones adicionales complejas, como lo es contigüidad [Macmillan94].
Para ocuparse del problema del diseño de la zona, es necesario recurrir a analizar un número de diversos algoritmos que se tienen propuestos (para una revisión cuidadosa en el contexto de distritación electoral, ver a [Williams95]).

Hay dos problemas importantes en la aplicación de los algoritmos automáticos del diseño de la zona. Primero, la mayor parte de los algoritmos descritos en la literatura no están disponibles pues resultan de esfuerzos de la investigación y la mayor parte de ellos nunca fueron puestos en ejecución como paquetes de software. En segundo lugar, la mayoría de los algoritmos se basan en una perspectiva regional del problema del diseño de la zona, es decir utilizan áreas mientras que las unidades básicas para la zona que diseñan no están bien determinadas.
Sin embargo, actualmente se disponen de propuestas algorítmicas más o menos claras para el diseño de la zona y utilizan diversas estrategias de optimización: recocido simulado, búsqueda tabú y programación lineal asociada a branch-and bround [Mehrotra98, Openshaw95a].

Los algoritmos genéticos GA [Altman98] siguen siendo en gran parte inexplorados en este campo, una buena referencia es [Bação05] pero no proporciona ningunos detalles en cómo los algoritmos genéticos fueron aplicados a este problema particular. Sin embargo, estos algoritmos se han utilizado extensivamente como procedimientos de búsqueda en campos relacionados tales como el problema P-mediano [Correa 01] y análisis de conglomerados [Murthy96].

Los problemas de diseño territorial para distritación tienen ya varias décadas, y las propuestas de solución son semejantes a las actuales. Desde finales de los años 50s, el problema de modelar distritación comenzó cuando la legislación no tenía más recursos manuales para dar respuesta a esta situación o simplemente la corte permanecía indiferente.
En este contexto, existían ya varias irregularidades detectadas en la operatividad de distritación, por ejemplo en la ciudad de Connecticut E.U y voluntarios de dependencias gubernamentales desarrollaron una medida de compacidad y otra de localización heurística de hogares para dibujar de forma independiente los distritos políticos constitucionales y ayudar a modelar el problema. La propuesta estaba basada en desarrollar una heurística que mapeara de manera compacta y contigua distritos con población equilibrada. El criterio de minimización y medida de compacidad es el momento de inercia poblacional: la sumatoria de las distancias al cuadrado de cada persona a su centro de su distrito, de hecho, esa era su función objetivo [Hess65].
De esta forma, ingenieros y analistas desarrollaron un método apropiado basado en un criterio familiar que aún es indispensable: compacidad, contigüidad y equilibrio poblacional.

Históricamente, entre los diferentes tipos de problemas de partición de territorio, fue el llamado problema de distritación electoral lo que condujo al uso de metodologías científicas para buscar construir distritos políticos tan cercanos como fuera posible para el proceso de votación y generar confianza en la imparcialidad en el proceso de la partición. Esto último es particularmente crucial ya que es posible diseñar una partición que favorezca a cierto grupo político, social o étnico. Es muy conocido en la historia de E.E.U.U. que el gobernador del estado de Massachusetts, Albright Gerry

(1744-1814), en un esfuerzo por garantizar su re-elección, manipuló la división de su estado para concentrar a sus votantes adeptos para elegir a un representante y esparcir a un número muy grande de sus antagonistas en un número pequeño de distritos.

Uno de los primeros trabajos en este mismo tema aparecido en [Vickrey61], describió un proceso heurístico en que una zona se construye, a cada iteración.
Después, en 1965, el primer modelo de programación matemática fue propuesto por Hess formulando el problema como un modelo de localización [Hess65].

En [Shirabe05] se considera un problema de localización de unidades espaciales para usos particulares hacia regiones acordando criterios específicos. Se entiende que unidades espaciales son áreas territoriales donde el autor propone una nueva formulación exacta de contigüidad que puede ser incorporada en cualquier modelo de programación entero mixto para unidades espaciales de localización. Además, el modelo supuestamente garantiza la robustez de contigüidad indiferente de otros criterios tales como compacidad.

En los problemas de distritación, los modelos de criterio múltiples parecen ser una representación más adecuada de problemas de distritación para situaciones del mundo real que por su misma naturaleza son multidimensionales.
En [Tavares07] se trata de solucionar el problema de dividir un territorio en zonas "homogéneas". Cada zona está compuesta de un conjunto de unidades territoriales elementales. Un mapa de distrito se forma dividiendo el conjunto de unidades elementales en zonas conexas sin inclusiones. Cuando el criterio múltiple es considerado, el problema de enumerar todas las soluciones para tal modelo se conoce como NP-difícil.

Durante las últimas tres décadas, muchos investigadores, académicos y practicantes de diferentes campos han desarrollado modelos, construido algoritmos e implementado soluciones referentes al problema de distritación. Unidades elementales de un territorio son agrupadas en conglomerados más grandes o distritos, produciendo un mapa de distrito o una partición.
Son muchas preguntas prácticas relacionadas a este problema:
- Cómo definir los distritos electorales de un país.
- Cómo establecer las diferentes zonas de trabajo para un equipo de ventas de agentes viajeros.
- Cómo definir las áreas en las redes metropolitanas de internet.
- Cómo definir las áreas para bienes manufacturados y consumidores de los mismos.

El mismo tipo de preguntas se formulan en problemas diversos como: políticas sobre distritación, ubicaciones escolares, definición de zonas de fuentes de poder eléctricas, problemas de identificación sobre aglomeración poblacional etc.

La partición de un territorio en diferentes zonas "homogéneas" evaluadas sobre la base del criterio múltiple, consiste en agrupar las unidades elementales de un territorio para formar un conjunto de distritos o zonas. Cada zona es el resultado de un proceso de agrupación de unidades elementales. Este problema puede ser modelado usando teoría de grafos y programación matemática con modelación binaria-entera. Cada unidad elemental es asociada a un vértice del grafo, y un par de unidades elementales contiguas definen un borde del grafo. Algunos valores numéricos también se asocian a los bordes y/o vértices. Las zonas deben cumplir ciertas restricciones, más o menos, técnicas, éticas, ecológicas, sociales, y otras. Diferentes mapas de un territorio forman un conjunto de soluciones diferentes dónde cada uno se evalúa de acuerdo a un conjunto de criterios consistente. Así, la búsqueda de una solución óptima, en general, no tiene ningún sentido y la "mejor solución o una buena solución", es frecuentemente, un compromiso en que la mejora de un criterio dado conduce a una degradación de las evaluaciones en por lo menos uno de los criterios restantes. Esto lleva al concepto de solución no-dominada [Tavares07].

La literatura ha puesto especial énfasis en que los problemas de planeación territorial requieren de un modelo de agregación territorial, y que en la mayoría de los casos, debe ocuparse de construir grupos de zonas "pequeñas" para facilitar tanto el análisis como la solución del problema.

2.4 Procesos de regionalización y clustering (conglomerado)

Gran parte de las metodologías utilizadas en el campo de la regionalización utilizan técnicas enmarcadas dentro del análisis conglomerado. En este contexto, el problema de agregación de datos espaciales puede ser visto como un caso particular de conglomerado en el cual se debe asegurar la continuidad geográfica entre los elementos agrupados. Este caso especial de análisis de clustering es llamado generalmente análisis de conglomerado con restricción de continuidad espacial o regionalización[1].

Dentro de este tipo de análisis, destacan tres estrategias metodológicas: 1) la agrupación en dos etapas, 2) la inclusión de las coordenadas geográficas como variables de clasificación y 3) la utilización de instrumentos adicionales para controlar la restricción de continuidad geográfica.

La agrupación en dos etapas consiste en aplicar, en una primera etapa, un método clásico de conglomerado, jerárquico o de particionamiento, utilizando únicamente los datos no-geográficos[2].

En la segunda etapa los grupos son revisados en términos de su continuidad geográfica, así, si a un conglomerado pertenecen áreas que son geográficamente discontinuas, entonces dichas áreas serán asumidas como regiones diferentes [Ohsumi83].

Entre las ventajas asociadas a dicha metodología [Openshaw95] resalta el hecho de que en la primera etapa se garantiza la homogeneidad de las zonas creadas utilizando únicamente las variables de interés. Por otro lado, este método puede resultar útil como un medio para obtener evidencias de una dependencia espacial entre los elementos. Sin embargo, para los objetivos de la regionalización, puede convertirse en un inconveniente el hecho de que el número de grupos no dependa del investigador sino del grado de dependencia espacial[3].

La segunda estrategia consiste en aplicar métodos clásicos de clustering pero forzando la continuidad geográfica por medio de la inclusión de las coordenadas geográficas de las áreas a agrupar, además de las variables de interés, en el cálculo de las similaridades o disimilaridades [Perruchet83]. Los principales inconvenientes derivados de la inclusión de variables geográficas como variables de clasificación son: la necesidad de que el componente geográfico tenga el peso suficiente para asegurar la continuidad geográfica [Wise97], y, dado que los elementos a agrupar son de tipo área, la solución puede variar en función del método utilizado para localizar el punto que representa cada una de ellas, sobre todo en aquellos casos donde las áreas son considerablemente grandes [Horn95].

Por último, la más utilizada de las estrategias para la resolución de problemas de regionalización, consiste en controlar la restricción de continuidades geográficas utilizando instrumentos adicionales como la matriz de contactos o su correspondiente gráfico de continuidades. Dichos elementos son utilizados para realizar las modificaciones apropiadas a los algoritmos clásicos de conglomerado jerárquicos o de particionamiento, asegurando el cumplimiento de la restricción de continuidad.

Para el caso de los modelos de particionamiento, los más utilizados en el contexto de la regionalización, destacan dos metodologías: la programación matemática y los algoritmos iterativos.

---

[1]Un resumen de estas metodologías de agrupación se encuentra en Gordon (1999) y para los casos especiales de conglomerado restringido en Fisher (1980), Murtagh (1985) y Gordon (1996).

[2]Aquellos atributos propios de los elementos o de sus relaciones, por ejemplo, población, ingreso medio, número de hospitales, movilidad laboral entre municipios, etc.

[3] A mayor dependencia espacial habrá una tendencia hacia la creación de pocas regiones

[4] Openshaw planteó el problema de la unidad espacial modificable (PUEM), definiéndolo como una fuente potencial de error que puede afectar los resultados de aquellos estudios basados en información agregada geográfica, ya que estos pueden variar en función de la configuración de dicha agregación. Para más información ver Openshaw (1977), Openshaw

y Taylor (1981), y en un contexto econométrico ver Fotheringham y Wong (1991) y Amrhein y Flowerdew (1992).)

Con respecto a los métodos de programación matemática, [Macmillan94] define el problema de regionalización como un problema de optimización combinatoria con el cual se determinará una agregación territorial óptima dado un número predeterminado de grupos a conformar. La solución propuesta por estos autores para asegurar la continuidad geográfica consiste en exponenciar la matriz de contactos teniendo en cuenta que para la formación de un grupo con n elementos continuos es necesario que la (n-1)ésima potencia de dicha matriz no contenga elementos nulos. Esta solución implica que el espacio factible definido por las restricciones sea no convexo lo cual no asegura la obtención de un óptimo global. Dado que la solución de este tipo de problemas por métodos convencionales de optimización resulta extremadamente complejo, se han desarrollado otras metodologías en el campo de la regionalización que han resultado muy efectivas en aquellos casos donde el número de elementos a agrupar es muy grande.

Dentro de este tipo de soluciones destacan distintos algoritmos como el "automatic zoning problem" [Openshaw77], que tienen como objetivo dividir un territorio en distintas áreas buscando además que dichas unidades territoriales minimicen, en lo posible, los efectos asociados al "problema de la unidad espacial modificable" (PUEM)[4].

Los modelos expuestos hasta ahora, se caracterizan por ser métodos supervisados, lo cual implica que el investigador conoce a priori la estructura de los datos que se utilizan y que describen un fenómeno que se desea estudiar. Pero también existen métodos no controlados, útiles en casos donde el investigador desea analizar una gran cantidad de datos sobre los cuales no existe información de los factores que afectan el sistema. En estos casos surge la posibilidad de realizar un análisis no paramétrico que permita encontrar patrones y relaciones en un gran volumen de información. Una de las principales aplicaciones de estos métodos dentro del campo de la regionalización son los "self organization maps" propuesto por [Kohonen82]. Esta metodología, desarrollada en el campo de la inteligencia artificial ha generado opiniones divergentes entre los investigadores dada la falta de fundamentación teórica que dificulta la interpretación de los resultados [Openshaw92].

Modelación básica en diseño territorial
S.W Hess, J.B Weaver (1965); "Nonpartisan political redistriting by computer"
Las propuestas de modelos para diseño territorial han sido requeridas desde la década de los sesentas y las ideas centrales siguen estando presentes en los modelos recientes. Como caso particular en [Hess65].
Los criterios fueron definidos como:
- Población equilibrada -un hombre un voto-, fue una regla básica de distritación constitucional.
- Contigüidad requiere que cada distrito sea una sola tierra parcelada.
- Compacidad no estaba operacionalmente definida y se dejó solo como medida.
Compacto significa informalmente "condensado" en lugar de cubrimiento.
Medidas no geométricas de compacidad fueron medianamente aceptadas, entonces una medida numérica se necesitó para manipular los datos.

El problema de establecer los distritos en un estado, con una población de n individuos, se centra en asignar, cada de esos individuos a uno y solo uno de los k distritos del estado, donde k es mucho más pequeño que n. Es deseable también que el número de individuos asignados a los distritos este equilibrado y que los distritos sean en algún sentido compactos.
Esta idea da lugar a proponer a 2 nociones matemáticas que aún son de gran apoyo como lo es equilibrio poblacional: si n_j es la población del distrito j, $\sum n_j = n$ (1)
La suma de la población distrital es igual a n de la población del estado.
Cuando se tienen k distritos, divide (1) por n/k el promedio de la población distrital para obtener
$\sum r_j = k$ (2)
donde r es el radio poblacional del distrito j, un valor para el cual es idealmente igual 1 para todos los distritos. Así se consideran funciones a ser optimizadas de r_j bajo la condición (2).

F. Bação, S. Caeiro, M. Painho, P. Goovaerts, M. Costa; "Delineation of estuarine management units: evaluation of an automatic procedure"

Las características para el problema de diseño de zonas son similares a los de problemas de clustering.

Sea el conjunto inicial de unidades de área $X=\{x_1, x_2, \ldots, x_n\}$ donde x_i es la i-ésima unidad de área. El número de zonas es denotado por k y Z_i es el conjunto de todas las unidades de área que pertenecen a la zona Z_i. Entonces

$Z_i \neq \varnothing$ para i = 1, … , k

$Z_i \cap Z_j \neq \varnothing$ para $i \neq j$

$$\bigcup_{i=1}^{k} Z_i = X$$

Esto constituye un conjunto de restricciones básicas iniciales que pueden ser aplicadas igualmente al diseño de zonas como de conglomerado [Bação05a].

2.5 Resumen de métodos de regionalización y referencias

La siguiente tabla y figura resume los aspectos relevantes de los problemas de regionalización tanto de la literatura existente como de sus propuestas de solución [Duque07].

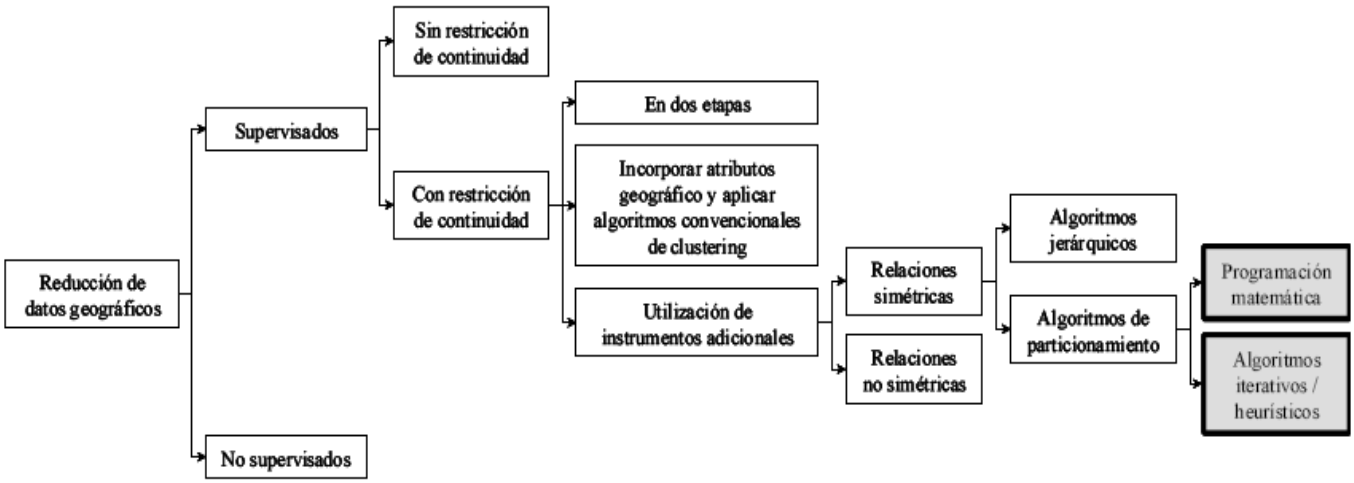


Figura 1. Resumen de las metodologías para la reducción de datos geográficos

2.5.1 Métodos de regionalización

Métodos de regionalización								
			Sin una explícita restricción de contigüidad espacial			Con una explícita restricción de contigüidad espacial		
características	Algoritmos de agrupamiento convencionales	Maximizan la región de compacidad	Modelos exactos	Algoritmos para agrupar mediante métodos jerárquicos adaptados	Regiones sin semillas	Modifica una región inicial factible	Modelos basados en teoría de graficas	Heurísticos mixtas
Se requiere el numero de regiones	✓	✓	✓	✓	✓	✓	✓	✓
Se debe considerar la información de todos los pares de relaciones	✓	✓	✓	✓	✓	✓	✓	✓

Las relaciones entre aéreas no tienen que depender geográficamente	✓	✗	✓	✓	✓	✓	✓	✓
Deben proporcionar vínculos entre vecinos	✗	✗	✓	✓	✓	✓	✓	✓
Se garantiza continuidad espacial	✗	✗	✓	✓	✓	✓	✓	✓
La forma de la región no es limitada	✓		✓	✓	✓	✓	✓	✓
Garantiza la solución optima	✓	✓	✗	✓	✓	✓	✓	✗
Puede usarse para resolver problemas grandes de regionalización	✓	✓	✗	✗	✓	✓	✓	✗
Puede fácilmente adaptarse un criterio de agregación	✗	✗	✗	✗	✓	✓	✓	✗

Tabla 1. Métodos de regionalización y sus principales características [Duque07]

2.5.2 Revisión bibliográfica de los métodos de regionalización

Método	Tópico tratado	Referencia
1.-Regionalización en base a algoritmos de agrupamiento convencionales	Regionalización en 2 estados. Sensibilidad de los resultados de regionalización a diferentes algoritmos de agrupamiento. Variante de recocido simulado del algoritmo k-means. **Variante de pam con recocido simulado** **Particionamiento compacto con el uso de búsqueda por entorno variable**	Openshaw (1973); Openshaw and Wymer (1995) **Bernabe (2007)** **Bernabe (2009)**
2.-Regionalización en base a la maximización de la región de compacidad	Consideración de fronteras naturales. Consideración de agregaciones pre existentes. Asignación fraccional. Formulación de programación entera. Medida cuantitativa de compacidad. Tipos de movimientos de áreas entre regiones. Uso de algoritmos genéticos Combinación de pesos del criterio de agregación múltiple.	Openshaw and Wymer (1995) Mills (1967); Segal and Weinberger (1977); George, Lamar, and Wallace (1997) George, Lamar, and Wallace (1997) Helbig, Orr, and Roediger (1972) Hess et al. (1965); Helbig, Orr, and Roediger (1972) Weaver and Hess (1963) Kaiser (1966) Bacao, Lobo, and Painho (2005) Kaiser (1966); Cliff et al. (1975); Zoltners (1979); Wise, Haining, and Ma (1997)

3.-Modelos de optimización exactos	Forma alternativa de definir adyacencia entre áreas. Forma alternativa de satisfacer restricciones de contigüidad Complexidad del problema de regionalización. Restricciones en el nivel de contigüidad basadas en el nivel de adyacencia. Restricciones en el nivel de contigüidad basadas en ruta de ruptura de restricciones. Restricciones en el nivel de contigüidad basadas en la potencia de la matriz de contigüidad. Soluciones obtenidas por la unión de las regiones factibles prediseñadas.	Zoltners and Sinha (1983) Duque, Church, and Middleton (2006) Cliff and Haggett (1970); Cliff et al. (1975); Keane (1975); Altman (1998) Zoltners and Sinha (1983) Duque (2004) Macmillan and Pierce (1994) Garfinkel and Nemhauser (1970); Mehrotra, Johnson, and Nemhauser (1998)
4.- Algoritmos de agrupamiento jerárquico adaptativo	Comparación de métodos jerárquicos. Métodos jerárquicos e identificación de patrones espaciales. Estrategias de agrupamiento jerárquico monofónico y no monofónico con restricciones de contigüidad. Reducción de la matriz de similaridad. Contigüidad espacial basada en umbral de contigüidad Algoritmos de jerarquía sub óptimos.	Spence (1968); Webster and Burrough (1972); Byfuglien and Nordgard (1973); Margules, Faith, and Belbin (1985 Byfuglien and Nordgard (1973) Ferligoj and Batagelj (1982) Openshaw (1973) Perruchet (1983) Bunge (1966)
5.- región por semillas	Algoritmos de múltiples inicios. Uso de rejillas regulares e irregulares. Formas de conservar la región. Formas de prevenir enclaves. Formas para regiones iníciales.	Thoreson and Liittschwager (1967) Thoreson and Liittschwager (1967) Vickrey (1961); Taylor (1973); Openshaw (1977a); Openshaw (1977b); Rossiter and Johnston (1981) Gearhart and Liittschwager (1969) Gearhart and Liittschwager (1969)
6.-Modificación de una solución factible inicial	Tipos de movimientos de áreas entre regiones. Uso de recocido simulado Uso de búsqueda tabú. Formas de verificar la contigüidad espacial.(intercambio de puntos).	Nagel (1965); Openshaw (1977a); Openshaw (1978); Sammons (1978); Ferligoj and Batagelj (1982); Browdy (1990); Horn (1995) Browdy (1990); Macmillan and Pierce (1994); Macmillan (2001); Openshaw and Rao (1995) Openshaw and Rao (1995); Duque and Church (2004) Macmillan and Pierce (1994); Macmillan (2001)
7.- Modelos basados en teoría de graficas	Uso de arboles desplegados	Maravalle and Simeone (1995)

8.-Modelos de heurísticas mixtas	Separación de los problemas de regionalización dentro de sub problemas. Uso de concentración de heurísticas.	Maravalle and Simeone (1995) Maravalle and Simeone (1995)
9.- Modelos multiobjetivo	**Uso de heurísticas y técnicas para obtención de maximales (soluciones no dominadas)**	**Bernabe (2009)**

En general, el problema de diseño y/o planeación de zonas es complejo y comprende muchos aspectos para su análisis así como un gran número de tareas por resolver (geográficas, de análisis, de modelación, simulación, etc.).

El conflicto de organización y modelación sobre entes geográficos, problemas poblacionales, reordenamiento territorial o división de zonas, debe enfrentarse primero que nada a investigar y recoger trabajos relacionados con el problema que ocupa y a partir de ello, apoyarse en las propuestas para ayudar a describir claramente la situación actual y objetivos que se persiguen.

Cuando la descripción del problema se va formalizando, es necesario incluir algunos conceptos, lemas y teoremas que bien pueden apoyarse en la literatura existente y proponer los propios, o bien elegir alguno que se adapte idealmente. Posteriormente estos se irán formalizando tanto en etapas tempranas del planteamiento del problema, como en el diseño (modelación).

De toda la revisión bibliográfica, se aseguran las contribuciones de este trabajo de zonificación óptima en DT, las cuales están señaladas en la sub-sección 2.5.2

2.6 Estado del arte en soluciones multiobjetivo: métodos para obtener soluciones no-dominadas

Una buena forma de concentrar la revisión de la literatura en métodos multiobjetivo se presenta a través de los resúmenes de los trabajos en esta línea.

1. **Encontrando la Maxima de un conjunto de vectores**

Sea U_1, U_2, .., U_d conjuntos totalmente ordenados y sea V un conjunto de vectores de n d-dimensional en $U_1 X U_2 \ldots X U_d$. Un ordenamiento parcial es definido sobre V de manera natural. El problema de encontrar todos los elementos maximales de V con respecto al orden parcial es considerado en la maxima [Kung75].

Este trabajo ha sido de gran ayuda para resolver ZO, incluso el método desarrollado en este trabajo para obtener las soluciones no dominadas, llamadas minimales-ZO, se apoya principalmente en el artículo “On finding the maxima of a set of vectors” (H. T. Kung, F. Luccio, F. P. Preparata) [Kung75]. De esta publicación, se encuentra un algoritmo implementado en código C (llamado nodom o ndominated) y está disponible en [Web2].

2. **Un método interactivo para elegir una solución sobre una amplia frontera eficiente**

Aquí se presenta un método interactivo para elegir una solución entre las que componen la frontera eficiente de un problema multiobjetivo. Este método está diseñado para una situación donde esta frontera ya ha sido obtenida usando un metaheurístico multiobjetivo (aunque también se podría haber conseguido con cualquier otro método) encontrándose formada por una gran cantidad de puntos, lo cual supone un problema para que un decisor real tome una decisión final. La principal característica de este método es que el esquema de interacción que se lleva a cabo es simple: tan solo se le pide al decisor que elija una solución entre un conjunto dado de soluciones representantes. Se aplica un método para ayudar a un decisor real a encontrar su solución más preferida entre una amplia frontera eficiente correspondiente a un problema real de optimización combinatoria multiobjetivo, previamente resuelto usando un metaheurístico multiobjetivo [Caballero06].

3. **Sobre soluciones óptimas en problemas de optimización multiobjetivo**

Se estudian los principales tipos de conceptos de óptimo considerados en problemas de

optimización multiobjetivo, cuando la ordenación de alternativas se regula mediante un cono K convexo: solucionesK-maximales, débilmenteK-maximales, fuertementeK-maximales, propiamenteK-maximales.
Se dan caracterizaciones en problemas generales de optimización vectorial y condiciones suficientes en problemas de maximización de funciones de valor vectoriales y escalares, particularizando después al caso de conos poliédricos. Se concluye con algunas aplicaciones y cuestiones prácticas [Rios08].

4. Bases de Gröbner parciales y optimización combinatoria multiobjetivo

La programación discreta multiobjetivo, es bien conocida como una rama de la familia en problemas de optimización con un largo espectro de aplicaciones. El caso lineal ha sido abordado por muchos autores durante los últimos años. Sin embargo, el caso polinomial, no ha sido profundamente estudiado debido a sus dificultades teóricas y computacionales. Este artículo presenta un enfoque algebraico para la solución de estos problemas. Se propone una metodología basada en la transformación del problema de optimización polinomial en un problema de solución de uno o más sistemas de ecuaciones polinomiales y se usa bases Gröbner para resolver estos sistemas. Diferentes transformaciones ofrecen diferentes metodologías, esas serán analizadas y comparadas desde un punto de vista teórico y por algunos experimentos computacionales a través de algoritmos que inducen (inductores) [Blanco07].

5. Soluciones no dominadas en problemas multiobjetivo

La teoría de estructuras de dominación, es un nuevo procedimiento de solución a problemas multiobjetivo, pero presenta bastantes lagunas, debidas, sin duda, a la novedad del tema. Se propone en este trabajo caracterizar completamente los puntos no dominados, por distintos procedimientos, así como seleccionar entre ellos un subconjunto más deseable ("soluciones propias"). Se abordan también condiciones para soluciones no dominadas en el espacio de decisiones [Coladas09].

6. Sobre óptimos locales en problemas multiobjetivo de optimización combinatoria

En este artículo, la optimalidad local en optimización combinatoria multiobjetivo se utiliza como una línea de base para el diseño y análisis de dos algoritmos de mejora iterativa. Ambos algoritmos buscan en un vecindario que es definido sobre una colección de conjuntos de soluciones factibles y su criterio de aceptación se basa sobre relaciones de rentablidad superior. Se dan pruebas de la solidez e integridad de estos algoritmos [Paquete07].

Por otro lado, a pesar de que el problema de particionamiento multiobjetivo ha recibido poca atención, se ha encontrado en la literatura aportaciones importantes.
Se expone en el capítulo 3 que el clustering puede ser abordado como un problema de optimización combinatoria cuando los conglomerados son una partición de un conjunto de objetos. Cuando se optimiza una función objetivo, basta incorporar al algoritmo clustering algún método heurístico y encontrar soluciones globales. En este punto, existen muchos trabajos que han resuelto este problema con muy buenos resultados [Bernábe09, Vicente05, Ochoa09, Bação05].

Además, dada la fuerte relación de los problemas de diseño territorial con particionamiento geográfico, se sabe que estos problemas exigen en su solución que algún método de clasificación obtenga grupos en el espacio geográfico donde las condiciones para la agrupación son propiedades espaciales de DT (continuidad, conexidad, compacidad, homogeneidad etc.). De la misma manera, la agrupación de los datos geográficos en DT ha dado buenos resultados si se optimiza una función objetivo dejando a otras propiedades como restricciones.
Sin embargo, el particionamiento multiobjetivo en DT supone una solución alternativa a este tipo de problemas ya que ofrece un conjunto de soluciones donde al decisor le favorece contar con más de una solución y así elegir la más adecuada a una aplicación en particular.
Pocos son los trabajos en particionamiento multiobjetivo debido a varias fuentes de complejidad: la modelación y caracterización del problema, la implementación articulada con algún método heurístico y la generación de soluciones no dominadas.

2.7 Literatura en clustering multiobjetivo

En un enfoque general de los trabajos relacionados en clustering multiobjetivo, se presentan 5 resúmenes:

1. Clustering multiobjetivo con tecnología de optimización metaheurística

Rafael Caballero, Manuel Laguna, Rafael Marti.

"Se desarrolla un procedimiento metaheurístico de agrupación a problemas multiobjetivo. Se trata de encontrar una buena aproximación de la frontera eficiente para esta clase de problemas y proporcionar un medio para mejorar la toma de decisiones en múltiples ámbitos de aplicación y, en particular los relacionados con la comercialización. El procedimiento se basa en la búsqueda tabú y dispersión de metodologías.

La agrupación ha sido un problema objeto de numerosos estudios, sin embargo, la mayor parte de los trabajos se ha centrado en un solo objetivo, esto es, la existencia de varios criterios de agrupamiento y/o múltiples fuentes de datos ha recibido escasa atención en la literatura. El procedimiento en este trabajo es general, y aborda varios problemas de tipo multiobjetivo y combinatoria en el análisis de datos llevando a la práctica una amplia experimentación con los datos reales y artificiales, en particular en un problema de segmentación de marketing para mostrar la eficacia del procedimiento propuesto" [Laguna 2006].

2. Clustering multiobjetivo alrededor de los medoides

Julia Handl, Joshua Knoeles.

"La gran mayoría de los actuales algoritmos de agrupamiento se centran en torno a la noción de una característica, es decir, los datos están representados por sus propiedades intrínsecas, vistas en característica de vectores. Sin embargo, algunas aplicaciones requieren que la agrupación de los datos sea exclusivamente extrínseca: sólo las relaciones entre los elementos de datos se conocen (es decir, sus similitudes o diferencias). En este sentido los autores desarrollan un sencillo y eficiente algoritmo de agrupamiento multiobjetivo para este escenario. El algoritmo resultante está demostrado en un rango de conjunto de datos, incluyendo una matriz de disimilaridad derivado de datos reales y no datos figurados" [Handl06].

3. Aplicación de optimización multiobjetivo para datos agrupados

Sriparrna Saha, Sanghamitra Bandyopadhyay.

"La agrupación es un problema central en reconocimiento de patrones y extracción de datos con innumerables aplicaciones que abarca muchos campos. El concepto de la principal dificultad de la agrupación eficaz es que una "buena" solución es un tanto mal definida para los datos no etiquetados. Por lo tanto, un número largo de medidas válidas de agrupación de calidad se han diseñado. La mayoría de algoritmos de agrupación optimizan solo un objetivo (a menudo implicitamente) y en consecuencia estan limitados en su ámbito de aplicación. En este articulo, investigamos un algoritmo de optimización multiobjetivo, la optimización de un número de medidas de calidad diferente simultáneamente y poder encontrar mejores soluciones del problema de agrupación. Los resultados experimentales muestran una clara ventaja del enfoque multiobjetivo los cuales muestran un nivel mucho más robusto que el clásico algoritmo k-means y algunas versiones de un solo objetivo para conjuntos de seis datos, algunas veces superándolos sustancialmente" [Saha06].

4. Gráfico de agrupación multiobjetivo con descenso de vecindad variable

Igor Navernіouk.

"Gráfico de agrupación es un problema combinatorio bien conocido que aparece en muchas diferentes áreas. La tarea es la partición del vértice del conjunto de un grafo a fin de minimizar una función de costo dada. La agrupación tiene aplicaciones en redes de interacción proteína-proteína, minería de datos y muchas otras áreas. En el contexto de la optimización multiobjetivo, se tiene más de una función de costo, y en lugar de encontrar una única solución óptima, es de interés encontrar un conjunto de soluciones de Pareto óptimas. Se presenta un algoritmo multiobjetivo con descenso de vecindad variable para éste problema y sus resultados en una colección de datos del mundo real y sintético. De los conjuntos de datos que tienen una agrupación "correcta" conocida, este algoritmo constantemente encuentra interesantes soluciones compatibles (restricción del

problema, que no se pueden encontrar por cualquier método lineal de un solo objetivo), demostrando una clara ventaja del enfoque multiobjetivo. Adicionalmente, la forma del frente de Pareto generada por el algoritmo puede dar pistas para las áreas del espacio de la función de costo que contiene soluciones no triviales. Comparando este método de un algoritmo de agrupación con un único objetivo y un algoritmo multiobjetivo, en todos los conjuntos de datos, este algoritmo requiere sustancialmente mayor tiempo computacional, pero produce resultados de mayor calidad" [Navernioukó3].

5. **Estrategía de evolución k-means basada en un algoritmo de clasificación acentuada de soluciones no dominadas**

Sameeh Ullah, Fakhri Karray, and Jin-Myung Won

"En este articulo, un nuevo método es propuesto basado en la estrategia de evolución k-means apoyada en un algoritmo de clasificación acentuada de soluciones no dominadas (NSES). En el enfoque de aprendizaje en una distancia métrica, los puntos de datos son transformados en un nuevo espacio donde las distancias euclidianas entre los puntos similares y los disimilares son su mínimo y su máximo, respectivamente. Sin embargo en NSES basado en el agrupamiento K-means, se optimizan los rendimientos globalmente en el componente gaussiano por un sistema de clasificación acentuada. Esta agrupación híbrida y el enfoque de clasificación mejoran el rendimiento del lenguaje natural en los sistemas de enrutamiento de llamadas. La clasificación acentuada realiza la tarea del modelo acústico de conmutación basada en la medida de confianza para la consulta de la persona que llama" [Sameeh08].

Capitulo 3: Marco teórico

Introducción

Este capítulo está dedicado a proporcionar algunos conceptos básicos teóricos que han sido importantes para establecer claramente el problema de zonificación óptima y particionamiento geográfico.
Los temas que se abordan son: clasificación, optimización, metaheurísticas, y metaheurísticas de búsqueda por entorno variable y recocido simulado.

3.1 Clasificación y particionamiento

Entre las técnicas más usadas en el análisis de datos, se encuentran las técnicas de clasificación [Sokal63]. Estas técnicas tienen como objetivo agrupar objetos en clases o clusters (conglomerados), internamente lo más homogéneos posible, de tal forma que los objetos pertenecientes a un conglomerado estén más próximos entre sí que los pertenecientes a grupos diferentes.
Generalmente para fines prácticos de esta metodología, es importante disponer de una medida D de desemejanza entre los objetos.
Muchos son los métodos de clasificación propuestos a lo largo de la historia, pero es posible distinguir dos categorías: métodos jerárquicos y métodos no jerárquicos o de enlace único.
En los primeros, el método genera, a distintos niveles de desemejanza ε, distintas soluciones de agrupamiento o particiones entre los objetos; por el contrario, para los segundos la solución se obtiene una vez fijado un nivel de desemejanza determinado [Bailey94].
Dentro de los métodos no jerárquicos se encuentran, entre otros, los métodos de partición iterativa o de optimización y las técnicas de recubrimiento.
La característica esencial que distingue a los métodos de optimización es que producen una única partición de los objetos en un número particular k, especificado apriori, de conglomerados no solapados, como resultado de la minimización o maximización de alguna función objetivo [Anderberg73].
Regularmente, estos métodos empiezan con una partición inicial del conjunto de objetos en k conglomerados, para cada uno de los cuales se define un centroide; se localiza, entonces, cada objeto en el conglomerado que tenga su centroide más cerca, para posteriormente, calcular los nuevos centroides y volver a relocalizar cada objeto. Así sucesivamente hasta que no se produzcan cambios en los conglomerados [MacQueen66].

Generalmente los objetos son representados por D atributos descriptores en forma de vectores en el espacio R^D, y con una medida de comparación de similaridad, como la distancia y se crean los conglomerados con objetos similares. En el proceso de la conformación de los grupos o conglomerado, no existe conocimiento previo acerca de cómo se debe conformar un conglomerado; por tal motivo, el proceso de clustering es también conocido como clasificación no supervisada.
En la agrupación se utiliza la información de una serie de variables para cada objeto y de acuerdo a estas variables, se mide la similitud entre estos objetos. Determinada la similitud, los objetos se agrupan en grupos homogéneos internamente y diferentes entre sí.
Las medidas de similaridad dependen de las asumsiones y el uso que se le da a los datos; diferentes resultados de los mismos, pueden resultar de las diferentes medidas de similaridad, donde cada una puede ser igualmente válida para un dominio de uso en particular.

3.1.1 Distancias y similitudes

Las similitudes y disimilitudes son los conceptos básicos que permitirán determinar si dos individuos u objetos son parecidos o diferentes. La similitud tiene el sentido de medir cuán similares son dos individuos, por lo tanto, entre mayor sea su valor, mayor será el parecido entre los individuos, y entre más cercano a cero menor será este parecido. La disimilitud, por el contrario, mide cuán diferentes son dos individuos, como es el caso de las distancias que conocemos; por lo tanto entre más cercana a cero sea la disimilitud menos diferentes serán los individuos (es decir, es más probable que pertenezcan a una misma clase) y entre mayor sea esta más diferentes serán.

En particionamiento, entendemos que individuos homogéneos que pertenecen a una misma clase, tienen las mismas o similares características en la tabla de datos que contiene las variables descriptivas, o los individuos son bastante cercanos en el caso de una tabla de proximidades. Esto es, dos objetos que pertenecen a una misma clase deben ser más parecidos entre ellos que con respecto a cualquier otro individuo de otra clase [Trejos09].

Distancias
La distancia expresa la proximidad o lejanía entre dos objetos. En matemática, la distancia entre dos puntos del espacio euclídeo equivale a la longitud del segmento de recta que los une, expresado numéricamente.

Regularmente en particionamiento se asume que son conceptos "similares" las distancias y las medidas de disimilitud. Sin embargo, cabe recordar la definición clásica:

Definición 1.
Desde el punto de vista formal, para un conjunto de elementos X se define distancia o métrica como cualquier función binaria $d(a,b)$ de $X \times X \in \Re$ que verifique las siguientes condiciones:

No negatividad $d(a,b) \geq 0 \quad \forall\ a,b \in X$

Simetría: $d(a,b) = d(b,a) \quad \forall\ a,b \in X$

Desigualdad triangular: $d(a,b) \leq d(a,c) + d(c,b) \quad \forall\ a,b,c \in X$

$\forall x \in X : d(x,x) = 0$

Si x,y ∈ X son tales que d(x,y)=0 entonces x=y

Si se deja de exigir que se cumpla esta última condición, al concepto resultante se le denomina pseudodistancia o pseudometrica.
La distancia es el concepto fundamental de la topología de espacios métricos. Un espacio métrico no es otra cosa que un par (X,d) donde X es un conjunto donde definimos una distancia d.
En el caso de que se tuviera un par (X,d) y d fuera una pseudodistancia sobre X entonces se dice que se tiene un espacio pseudometrico.
Si (X,d) es un espacio métrico y $E \subset X$ podemos restringir d a E de la siguiente forma: $d' : E \times E \to \Re$ de forma que si $x,y \in E$ entonces $d(x,y) = d(x,y)$ (es decir, $d' = d \mid_{E \times E}$). La aplicación d es también una distancia sobre d, y como comparte sobre $E \times E$ los mismos valores que d, se denota también de la misma manera, es decir, (E,d) es un subespacio métrico de (X,d) [Trejos09].

Similitudes
Definición 2.
Sea Ω el conjunto de individuos, u objetos a clasificar, que supone posee *n* elementos.
Una **similitud** es una función *s* : Ω × Ω → R^+ tal que:
1. para cada *i* E Ω , se tiene *s*(*i, i*) = max{*s*(*i, j*)/*j* E Ω };
2. para cada *i, j* E Ω , hay simetría: *s*(*i, j*) = *s*(*j, i*).
Con sólo estos dos requisitos se pueden construir funciones que den una idea de la similitud entre individuos. La definición de una similitud dependerá de cómo es la descripción de los individuos, es decir, qué tipo de variables son las que los describen [Pizza99, Trejos09].

Las medidas de similitud para variables cuantitativas podrían abordarse utilizando medidas para variables binarias, pero una transformación de variables cuantitativas a binarias implica pérdida de información. Lo más adecuado es considerar medidas de similaridad que puedan aplicarse directamente. Una de tales medidas es el coeficiente de correlación de Pearson.
Un coeficiente de similaridad sugerido por [Gower86] es particularmente interesante:

$$S_{ij} = \frac{\sum_{k=1}^{p} w_{ijk} S_{ijk}}{\sum_{k=1}^{p} w_{ijk}}$$

donde $S_{ij} = 1 - \frac{|x_{ik} - x_{jk}|}{R_k}$

Disimilitudes

La definición de disimilitud siguiente coincide con la de distancia formal antes descrita.

Una **disimilitud** es una función $d : \Omega \times \Omega \to \Re^+$ tal que:

1. Para cada $i \in \Omega$ se tiene $d(i,i) = 0$
2. Para cada $i, j \in \Omega$, hay simetría: $d(i, j) = d(j, i)$

   Si a la definición anterior uno le añade:
3. $d(i, j) = 0 \Leftrightarrow i = j$
4. La desigualdad triangular: $i, j, k \in \Omega \quad d(i, j) \le d(i, k) + d(k, j)$ para cada entonces la disimilitud es lo que llamamos una **distancia.**

Caso cuantitativo

La disimilitud más usada es la distancia euclídea clásica:

$$d(i, j) = \sqrt{\sum_{k=1}^{p} (X_i^k - X_j^k)^2}$$

Dicho de otra forma, esta distancia $D_{ij} = \left(\sum_{k=1}^{p} (x_{ik} - x_{jk})^2 \right)^{\frac{1}{2}}$ es un caso particular de una métrica más general llamada distancia de Minkowski $D_{ij} = \left(\sum_{k=1}^{p} (x_{ik} - x_{jk})^n \right)^{\frac{1}{n}}$,

donde D_{ij} es la distancia entre los individuos i y j , p es el número de variables y
n = 1, 2, ... ∞

Más aún, una distancia euclídea puede ser definida a partir de una métrica, esto es, de una matriz simétrica definida y positiva *M*. En tal caso, se podría poner

$$d^2(i, j) = \|x_i - x_j\| M = (x_i - x_j)^t M (x_i - x_j)$$

El uso de la distancia clásica tiene sentido cuando las variables observadas sobre los individuos son cuantitativas, pues en este caso tienen sentido las operaciones expresadas en la fórmula de la distancia. Hay que mencionar que esta distancia tiene un inconveniente si se usa sin precaución debido a que cada término de la sumatoria es elevado al cuadrado, la distancia euclídea tiene tendencia a magnificar las grandes diferencias entre las observaciones, por lo que si hay un dato aberrante este comportamiento atípico se traducirá en un valor muy grande de la distancia. Por ello, antes de cualquier análisis multivariado, siempre se recomienda hacer un estudio univariado de cada variable [Trejos09].

Algunos autores prefieren usar una como la siguiente, llamada "city block": $d(i, j) = \sum_{k=1}^{p} |x_i^k - x_j^k|$

Otra distancia usada en ocasiones, es la llamada distancia de Chebychev:

$$d(i, j) = \max\{|x_i^k - x_j^k| / k = 1, ..., p\}$$

Agregaciones

Los métodos de clasificación usan generalmente una noción de proximidad entre grupos de elementos, para medir la separación entre las clases que se buscan. Para ellos, se introduce el concepto de agregación, que no es más que una disimilitud entre grupos de individuos:

Sean $A, B \subset \Omega$, entonces la agregación entre A y B es: $\delta(A, B)$

Tal que δ es una disimilitud en el conjunto de partes $\mathrm{P}(\Omega)$:

i) $\delta(A, A) = 0$ para todo $A \in \mathrm{P}(\Omega)$

ii) $\delta(A, B) = \delta(B, A)$ para todo $A, B \in \mathrm{P}(\Omega)$

Usualmente, la medida de agregación está basada en la disimilitud d medida sobre Ω.

En efecto, denotando A y B dos subconjuntos de Ω, las agregaciones más usadas son:

1. Agregación del salto mínimo o del vecino más cercano:

$$\delta_{\min}(A, B) = \min\{d(a, b) \,|\, a \in A, b \in B\}$$

2. Agregación del salto máximo:

$$\delta_{\max}(A, B) = \max\{d(a, b) \,|\, a \in A, b \in B\}$$

3. Agregación del salto promedio:

$$\delta_{prom}(A, B) = \frac{1}{card(A) + card(B)} \sum_{\substack{a \in A \\ b \in B}}^{n} d(a, b)$$

El agrupamiento también es conocido como clustering y de ahí se distinguen distintos tipos de clustering que se describen más adelante. De este modo, clustering, conglomerado o grupo se usa indistintivamente.

Algoritmos de agrupamiento jerárquicos J y no jerárquicos NJ

En esta sección se mencionan algunas propiedades de los métodos de agrupamiento J y NJ. Se ha puesto especial atención a los algoritmos no jerárquicos debido a que de estos algoritmos, se han identificado algunos aspectos esenciales para la construcción del algoritmo de particionamiento para unidades geográficas desarrollado en este trabajo.

A pesar de que no existe una definición general de un agrupamiento, se han desarrollado algoritmos que encuentran diferentes clases de conglomerados: esféricos, lineales, irregulares, entre otros. Motivados por su amplia gama de aplicaciones existen muchos trabajos donde han desarrollado técnicas para agrupar datos de diferentes tipos: binarios, nominales y otras clases de variables discretas, variables continuas, similaridades y disimilaridades.

Divisiones de algoritmos de agrupamiento

Existen al menos 2 divisiones principales de algoritmos de agrupamiento: el agrupamiento particional y agrupamiento jerárquico. El agrupamiento particional, desarrolla una partición de los datos tal que los objetos en un grupo son más similares a algunos que ellos a los objetos en otros grupos. Este método construye k particiones de los datos, donde cada partición representa un grupo o conglomerado. Cada grupo tiene al menos un elemento y cada elemento pertenece a un solo grupo.

También, crea una partición inicial e iteran hasta un criterio de paro. Los más populares utilizan k-medias y k-medoides (por ejemplo, pam, clara y clarans) [Kaufman87].

El agrupamiento jerárquico, combina grupos pequeños en grupos grandes, o particiona los grupos grandes. En el agrupamiento basado en localidades, los objetos del grupo se basan en las relaciones locales, y por consiguiente la base de datos puede examinarse en un paso.

Clasificación de los algoritmos de agrupamiento

Se habla de algoritmos directos/constructivos cuando no optimizan ninguna función criterio. Si usan una función criterio a optimizar se habla de algoritmos indirectos o por optimización [Ester96].

Jerárquicos: estructura progresiva de árbol
Los métodos jerárquicos actúan sobre la matriz de desemejanzas (o de semejanzas) para construir un árbol que describirá, mediante uniones o divisiones sucesivas, las relaciones entre los objetos. Los algoritmos que han seguido la vía de la jerarquización se distinguen entre sí en función del criterio adoptado para la unión o división de los grupos.
La principal ventaja de los métodos jerárquicos es la facilidad de interpretación del árbol resultante. Entre sus desventajas se suelen citar las siguientes:
El algoritmo de clasificación sólo pasa una vez por los datos, por lo que una partición inicial desacertada no puede ser subsanada; en el caso del método del enlace simple, aparece el efecto del encadenamiento; pueden generar soluciones diferentes simplemente reordenando los datos en la matriz de desemejanzas o semejanzas, lo que causa problemas de no estabilidad de las soluciones [Aldenderfer84].
Por otro lado, dentro del agrupamiento jerárquico se distinguen a los algoritmos aglomerativos o incrementales que parten de patrones aislados y tienden a unir agrupamientos de acuerdo a algún umbral fijado (por ejemplo, agnes). Los algoritmos divisivos o decrementales se generan a partir de agrupamientos ya establecidos y tienden a crear nuevos agrupamientos más homogéneos.

No jerárquicos
En la literatura sobre clasificación, los métodos no jerárquicos NJ, han recibido menor atención que los métodos jerárquicos. Esta circunstancia es aún más acentuada en lo que se refiere a las técnicas de recubrimiento, los cuales conducen a conglomerados no disjuntos, que permiten que un objeto cualquiera pueda aparecer en más de una clase final a la vez. Su objetivo básico es recubrir el conjunto de objetos o con una familia de conglomerados maximales [Everittt93]. Estos métodos están muy condicionados por los objetivos del estudio, las características intrínsecas de las variables observadas para los objetos, así como por la medida de desemejanza elegida para relacionar los objetos.

En NJ el conjunto de datos es particionado en un número pre-especificado de conglomerados k, y luego iterativamente se va reasignando las observaciones a los conglomerados hasta que algún criterio de parada (función a optimizar) se satisface. De esta forma, los métodos de particionamiento tienen la ventaja de que satisfacen un criterio de optimalidad aunque sea aproximadamente.

A los métodos no jerárquicos con frecuencia se les conoce como “agrupación k-medias” y pueden clasificarse en umbral secuencial, umbral paralelo y la división para la optimización.

En el método de umbral secuencial, se selecciona un centro de grupo y se agrupan todos los objetos dentro de un valor de umbral que se especifica previamente a partir del centro. Después, se selecciona un nuevo centro o semilla de grupo y el proceso se repite para los puntos si agrupar. Una vez que un objeto se agrupa con una semilla, ya no se considera conglomerado con semillas subsecuentes. El método de umbral paralelo funciona de manera similar excepto que se seleccionan simultáneamente varios centros de grupo y se agrupan los objetos de nivel umbral dentro del centro más próximo. El método de división para la optimización difiere de los otros dos procedimientos de umbral en que los objetos pueden reasignarse posteriormente a otros grupos, a fin de optimizar un criterio general, como la distancia promedio dentro de los grupos para un número determinado de conglomerados.

K-medias es el método no jerárquico más ampliamente difundido, sin embargo, otros métodos de particionamiento han surgido con la idea de superar algunos de los problemas que tiene k-medias, (como lo es la obtención de óptimos locales). Algunos de estos algoritmos son daisy, pam, clara, fanny, agnes, diana y mona, con un enfoque tanto estadístico como de optimización y difuso [Anja96].
En general, los métodos de clasificación por particiones buscan una sola partición de mediante la optimización de algún criterio. Existen básicamente dos tipos de métodos:
• los que fijan a priori el número de clases
• los que no fijan este número.

Los primeros tienen la ventaja de la sencillez y rapidez, mientras que los segundos tienen la ventaja obvia de buscar el número de clases. Sin embargo, estos últimos tienen la gran desventaja de depender de un gran número de parámetros que deben ser estimados por el usuario y cuya manipulación no es fácil sin una adecuada experimentación y práctica.

Un esquema llamado de nubes dinámicas es un clásico ejemplo de algoritmos de particionamiento que fijan a priori el número de grupos. Estos métodos están basados en el principio que una clase puede ser representada por algún objeto, sea éste un punto promedio, un individuo o grupo de individuos de la clase, un conjunto de parámetros, etc. A este representante se le llama comúnmente núcleo. El primer algoritmo de este tipo fue propuesto por Forgy [Forgy65], y luego fueron propuestos otros similares.
La idea subyacente es:
• asignar los individuos al núcleo más cercano
• calcular los núcleos con las clases formadas en el paso anterior
• iterar los pasos anteriores hasta obtener estabilidad.
Se parte de una configuración inicial de núcleos, y se puede probar que el método converge a una partición que no mejora el criterio. Dependiendo del contexto y del tipo de núcleo, se define un criterio a ser mejorado.
Los métodos de clustering existentes difieren uno del otro en la forma de estructurar los conglomerados. Aquellos que encuentran conglomerados que corresponden a una partición del conjunto de objetos se les conoce como métodos de hard-clustering o clustering particional, siendo el más conocido el algoritmo k-means [Vicente05].

A los métodos que asignan a cada objeto un valor de pertenencia con respecto a cada conglomerado se les conoce como métodos de soft-clustering, y entre los más representativos de este tipo de clustering se encuentran los algoritmos fuzzy k-means y expectación maximización. Estos métodos no han sido analizados ni aplicados a los datos geográficos que nos ocupan, pero bien pueden ser considerados como trabajo futuro.

El método propuesto en este trabajo se considera dentro de las técnicas de hard-clustering o clustering particional [Vicente05]. Para ello se han estudiado los métodos descritos en la siguiente sección con el fin de considerar y respetar algunas propiedades básicas para construir un algoritmo de particionamiento para zonificación óptima que se define en el capítulo 5.

3.2 Métodos clásicos de particionamiento
Se ha insistido que en los métodos de particionamiento, se busca una única partición de los objetos en estudio en k clases disjuntas. La teoría tradicional de los métodos de particionamiento son fundamentalmente: k-means de Forgy, nubes dinámicas de Diday, algoritmo de transferencias de Régnier o k-means de McQueen, particiones principales, entre otros [Trejos09].

En la clasificación por particionamiento se tiene que siendo $\{x_1, \ldots, x_n\}$ el conjunto finito de n objetos a clasificar y $k<n$ el número de clases en los cuales que se desea clasificar a los objetos. Una partición $P = (C_1, \ldots, C_k)$ de Ω en k clases $C_1, \ldots, C_k$, está caracterizada por las siguientes 2 condiciones: $1) \Omega = \bigcup_{i=1}^{k} C_i \qquad 2) C_i \cap C_j = \varnothing$ para $i \neq j$

Es posible permitir eventualmente que algunas de las clases C_i sea vacía, de manera que en realidad las particiones $P = (C_1,\ldots, C_k)$ que se consideran son particiones Ω de en k *o menos* clases. Sin embargo, las particiones óptimas de acuerdo al criterio de inercia contienen exactamente k clases no vacías [Pizza99, Trejos09].

Criterio de inercia
Como se ha mencionado, se quiere obtener clases lo más homogéneas posibles y tal que estén suficientemente separadas. Este objetivo se puede concretar numéricamente a partir de la siguiente propiedad:

Supóngase que se está en presencia de una partición $P = (C_1, C_2, ..., C_k)$ de Ω donde $g_1, g_2, ..., g_k$ son los centros de gravedad de las clases donde:

$$g_k = \frac{1}{n} \sum_{i \in C_k} X_i ,$$

g es el centro de gravedad total: $g = \frac{1}{n} \sum_{i=1}^{n} X_i$.

Si se denota $I = \frac{1}{n} \sum_{i=1}^{n} \left\| x_i - g \right\|^2$ la inercia total de la nube de puntos,

$$B(P) = \sum_{k=1}^{K} \frac{|C_k|}{n} \left\| g_k - g \right\|^2$$

la inercia *inter-clases,* es decir los centros de gravedad respecto al centro de gravedad total, y

$$W(P) = \sum_{k=1}^{K} I(C_k) = \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in Ck} \left\| x_i - g_k \right\|^2$$

la inercia *intra-clases*, es decir la inercia al interior de cada clase, entonces se tiene la igualdad de Fischer: $I = B + W$.

Obsérvese que B mide precisamente la "separación" de la nube de puntos, al medir la inercia entre los centros de gravedad; si esta inercia es grande se deduce que los centros de gravedad están bastante separados (son dispersos). Por su parte, W mide la homogeneidad de las clases; en efecto si W es pequeño entonces cada $I(C_k)$ es pequeño y así la dispersión al interior de cada clase es pequeña.

Como la inercia I es fija, dada la nube de puntos, entonces al minimizar B se maximiza automáticamente W . Por lo tanto, los dos objetivos (homogeneidad al interior de las clases y separación entre las clases) se alcanzan al mismo tiempo al querer minimizar W . Así, el objetivo en el método de nubes dinámicas es encontrar una partición P de Ω y representantes de las clases, tales que $W(P)$ sea la mínima. [Trejos09].

Método de k-medias

Existe un poco de confusión en la literatura acerca del método de las k-medias, ya que hay dos métodos distintos que son llamados con el mismo nombre. Originalmente [Forgy65] propuso un primer método de reasignación-recentraje que consiste básicamente en la iteración sucesiva, hasta obtener convergencia, de las dos operaciones siguientes:

- Representar una clase por su centro de gravedad, esto es, por su vector de promedios.
- Asignar los objetos a la clase del centro de gravedad más cercano.

Poco después, [MacQueen66] propone un método muy similar, donde también se representan las clases por su centro de gravedad, y se examina cada individuo para asignarlo a la clase más cercana. La diferencia con el método de Forgy es que inmediatamente después de asignar un individuo a una clase, el centro de ésta es recalculado, mientras que Forgy primero hacia todas las asignaciones y luego recalculaba los centros. Es claro que el método de McQueen depende del orden en que se presentan los datos. Este método de McQueen ya había sido propuesto en Francia por S. Régnier en 1965 [Trejos09].

Sin embargo, en el contexto de la búsqueda de una partición de consenso, llamada partición central, variantes del método de Forgy son propuestas en Francia como método de nubes dinámicas por E. Diday a partir de 1967 [Trejos09].

Es McQueen, [McQueen66] quien propone el nombre k-means, que se usa hasta la fecha, aun si estos métodos también reciben nombres como nubes dinámicas, centros móviles, o reasignacion-recentraje.

Método de Forgy

Denotaremos Ω el conjunto de *n* individuos que queremos clasificar y supondremos que están descritos por *p* variables cuantitativas $x^1, x^2, ..., x^p$.

En el caso en que se está en presencia de variables cuantitativas, tiene sentido el cálculo de promedios y de distancias euclídeas, por lo tanto, también tiene sentido que cada clase esté representada por su centro de gravedad, esto es, por un individuo ficticio cuyas coordenadas son los valores promedio de las variables para los individuos pertenecientes a la clase. Este es el caso más simple y usado más corrientemente. Generalmente, se usa la distancia euclídea en este contexto.

Como se mencionó anteriormente, el método de las k-medias consiste en:

1. Escoger una partición inicial, al azar o con base en algún otro criterio.
2. Calcular los centros de gravedad de la partición.
3. Asignar cada objeto a la clase del centro de gravedad más cercano.
4. Repetir los pasos 2 y 3 mientras las clases en el paso 3 se modifiquen, esto hasta que se obtiene la estabilidad en la partición.

Se prueba que efectivamente el método alcanza la estabilidad después de unas pocas iteraciones.
Conviene hacer notar que, en una implementación computacional, la escogencia al azar es más bien de una muestra de K objetos iniciales que servirán de núcleos iniciales, y luego se asignan todos los demás objetos a la clase del núcleo más cercano, formándose entonces la partición inicial.

Método de transferencias

Un segundo tipo de métodos de particionamiento son los algoritmos del tipo de *transferencias*, originalmente propuestos por Régnier y por McQueen. Consisten en hacer la transferencia entre una clase y otra, de un único elemento de a la vez, haciendo mejorar algún criterio en cada iteración.

El algoritmo general es como sigue (aquí *W* es un criterio general de clasificación, no necesariamente la inercia intra-clases):

1. Se da una partición inicial $P = (C_1, C_2, \ldots, C_k)$ de Ω.
2. Se toma un elemento x ϵ Ω arbitrario, con $x \in C_k$.

Llamamos $C_{k'}^{k}(x)$ la partición de P_k consistente en transferir x de C_k hacia $C_{k'}$ en la partición *P* y dejar las demás clases iguales.

3. Sea *P** tal que $W(P^*) = \min\{ W(C_{k'}^{k}(x))\}$: k´=1, …, k

Entonces ponemos *P* := *P**

4. Se repiten los pasos 2 y 3 para todos los elementos x ϵ Ω .
5. Se detiene cuando al aplicar 4 no ocurre ninguna nueva transferencia.

En el caso euclídeo, se tiene *n* individuos descritos por *p* variables cuantitativos y R^p está provisto de una distancia euclídea. Se busca la partición $P = (C_1, \ldots, C_k)$ que minimice la inercia inter-clases *W*. Por tanto, al pasar x de C_k a $C_{k'}$ se debe minimizar

$$W(C_{k'}^{k}(x)) = \sum_{h \notin \{k,k'\}} I(C_h) + I(C_k \setminus \{x\}) + I(C_{k'} \cup \{x\})$$

En el caso general, Ω es arbitrario, con *d* un índice de disimilitud sobre Ω. El criterio *W* que se define sobre la partición *P* toma en cuenta la relación de equivalencia *R* asociada a *P*: W(P) = sup{d(*i,j*) : *iRj; i,j* ∈ Ω}

Si se tiene
$P = (C_1, \ldots, C_k)$, con $x \in C_k$, para transferir x a $C_{k'}$ es necesario que:
$\sup\{ d(x,y): \in C_{k'} \} < \sup\{ d(x,y): \in C_k \}$

Debe observarse que, al igual que en el método de k-medias, aquí también la partición final P* depende de la partición inicial. Así mismo, el número K de clases es dado a priori.

Sin embargo las clases también se pueden vaciar en el transcurso del algoritmo. Igualmente, ese número k puede no ser un número "natural" de clases para. Para dar las K clases iniciales en el caso euclídeo, también se puede usar el método de polos de atracción [Trejos09].

Método de nubes dinámicas

Se quiere obtener una partición de Ω en K clases bien agregadas, bien separadas y de intersección vacía. El número K de clases es dado a priori y los datos pueden ser de cualquier naturaleza.
Este método fue introducido por [Diday72] generalizando el método de k-medias de Forgy. Se basa en que cada clase debe tener una representación (llamada núcleo), y luego se hace una búsqueda iterada de núcleos y de particiones, hasta optimizar un cierto criterio.
En el método general de nubes dinámicas, cada clase estará representada por un núcleo, que será un elemento representativo de los integrantes de la clase. El algoritmo general de nubes dinámicas es el siguiente:

1. Se da una partición inicial de Ω
.2. Se calculan los núcleos, mediante una función de representación.
3. Se forma una partición, asignando cada elemento al núcleo más próximo, mediante una función de asignación.
4. Se repiten los pasos 2 y 3 hasta que las clases se estabilicen.

La escogencia de los núcleos iniciales, se hace generalmente de manera aleatoria. En el caso general, se escoge K veces m elementos entre los individuos. Se usa un criterio aditivo del tipo

$$W(P)= \sum_{k=1}^{k} \sum_{x_i \in C_k} D(x_i, N_k) ,$$

donde N_k es el núcleo de C_k (formado por m objetos) y D es una medida de disimilitud (por ejemplo, una agregación) entre los objetos x_i y los núcleos N_k (que son conjuntos de objetos). El núcleo N_k se define como el subconjunto de C_k con m elementos que minimice $\sum_{i \in C_k} D(x_i, N_k)$.

Se puede probar que en cada iteración se mejora W y además se converge a una clase estable [Trejos09].

Es claro que el método de k-medias corresponde al método de nubes dinámicas cuando los núcleos son centros de gravedad.

Variantes del método de nubes dinámicas

Existe una serie de variantes al método de nubes dinámicas. Básicamente, para cada una de ellas se debe definir el criterio a optimizar, los núcleos (función de representación), y la forma de asignar elementos a las clases (función de asignación).

Métricas adaptativas. El método de k-medias tiene la tendencia de formar clases esféricas con misma cardinalidad. Por ello, no es útil cuando se trata de identificar clases que tengan una misma forma de dispersión, quizá no necesariamente esférica, pero con una o varias direcciones de prolongamiento (sobre un eje discriminante, por ejemplo). Por tanto, en este caso se quita la restricción de que la medida de distancia sea la misma durante todo el algoritmo. Más bien se trata de buscar iterativamente la distancia que mejor se adapte a los datos.
En presencia de objetos descritos por variables cuantitativas, el criterio es para el caso de una sola métrica M:

$$W(P)= \sum_{k=1}^{k} \sum_{i \in C_k} \left\| x_i - g_k \right\|^2 M \quad \text{o bien} \quad W(P)= \sum_{k=1}^{k} \sum_{i \in C_k} \left\| x_i - g_k \right\|^2 M_k$$

para el caso que se tiene una métrica M_k asociada a cada clase C_k

En cada iteración del algoritmo, se calcula no solo los centros de gravedad gk, sino también las métricas. En el primer caso $M=\det(V)^{1/p}V^{-1}$, donde V es la matriz de varianzas intra-clases, mientras

que en el segundo caso $M_k=\det(V_{k)}^{1/p}V_k^{-1}$ donde V_k es la matriz de varianzas intra-clases de clase C_k.

Descripción del problema de conglomerados como uno de optimización

Dado un conjunto de n objetos denotado por $X = \{x_1, x_2, ... , x_n\}$, en que $x_i \in R^D$, sea K un número entero positivo conocido apriori, el problema del clustering consiste en encontrar una partición:
$P = \{C_1,C_2, ..., C_k\}$ de X, siendo C_j un conglomerado conformado por objetos similares, satisfaciendo una función objetivo $f : R^D \rightarrow R$, y las condiciones:

$C_i \cap C_j = \emptyset$ para $i \neq j$, y $C_j \cup C_i = X$

Para medir la similaridad entre dos objetos x_a y x_b se usa una función de distancia denotada por $d(x_a, x_b)$, siendo la distancia euclidiana la más usada para medir la similaridad. Así la distancia entre dos diferentes elementos

$x_i = (x_{i1}, ..., x_{iD})$ y $x_j = (x_{j1}, ..., x_{jD})$

es $d(x_i, x_j) = \sqrt{\sum_{l-1}^{D} (X_{il} - X_{jl})^2}$

Los objetos de un conglomerado son similares cuando las distancias entre ellos es mínima; esto permite formular la función objetivo f, como:

$$\sum_{j=1}^{k} \sum_{x_i \in C_j} d(x_i, \overline{x_j})^2 ; \qquad (1)$$

es decir, se desea minimizar (1); donde x_j, conocido como elemento representativo del conglomerado, es la media de los elementos del conglomerado Cj,

$$x_j = \frac{1}{|C_j|} \sum X_i \in C_j \qquad (2)$$

y corresponde al centro del conglomerado. Bajo esas características, el clustering es un problema de optimización combinatoria, y ha sido demostrado que es un NP-difícil [Vicente05].

3.2.1 Naturaleza combinatoria de los algoritmos de particionamiento

Dado que el clustering puede ser abordado como un problema de optimización combinatoria de categoría NP-dificil, es similar decir que el agrupamiento no jerárquico es un problema combinatorio.

Es necesario hacer notar que, cuando se quiere obtener una partición en K clases de un conjunto con n individuos, no tiene sentido examinar todas las posibles particiones del conjunto de individuos en K clases. En efecto, se está en presencia de un problema combinatorio muy complejo. Solo para efectos de ilustración, mencionemos que el número de particiones de un conjunto con 60 elementos en 2 clases es aproximadamente 1018, y para 100 elementos en 5 clases anda por 1068. De hecho, se puede probar que el número S(n,K) de particiones diferentes de un conjunto de n individuos en K clases, cumple la ecuación de recurrencia

$S(n,K) = S(n - 1,K - 1) + kS(n - 1,K)$

Esto lleva a que

$$S(n,k) = \frac{1}{K!} \sum_{i-1}^{K} (-1)^{K-i} \binom{K}{i} i^n$$

De lo anterior se deduce la necesidad de contar con métodos y algoritmos que den una solución satisfactoria del problema propuesto, aunque evidentemente puede que no se obtenga la mejor solución en todos los casos [Trejos09].

Algoritmo k-medias en optimización

El algoritmo k-medias es una de las heurísticas de búsqueda local comúnmente utilizadas para resolver el problema de clustering. Como ya se ha mencionado, la idea básica es obtener los k

centros iniciales y formar conglomerados asociando todos los objetos de X a los centros más cercanos, después se recalculan los centros. Si esos centros no difieren de los centros anteriores, entonces el algoritmo termina; en caso contrario, se repite el proceso de asociación con los nuevos centros hasta que no haya variación en los centros o se cumpla algún otro criterio de parada como poco numero de reasignaciones de los objetos [Vicente05].

Los k diferentes centros iniciales $\{\overline{x}\}_{j=1,..,k}$ se seleccionan aleatoriamente de X.

La asociación del objeto $x_i \in X$ con el centro más cercano $\overline{x_j}$ del conglomerado C_j es dada si $d(x_i,\overline{x_j}) < d(x_i,\overline{x_p})$ para todo j, p=1, …, k y j≠p. Los centros son recalculados usando la expresión (2).

El pseudocódigp de k medias se presenta así:

Algoritmo k-means (X = {x1, ..., xn}, K)

1. Seleccionar aleatoriamente de X centros iniciales $\{\overline{x_i}\}_{i=1,...,k}$
2. Para cada $x_i \in X$
   2.1 Asociar x_i con el centro más cercano:
   $C_j = C_j \cup \{x_i\}$ si $d(x_i, \overline{x_j}) < d(x_i, \overline{x_p})\ \forall\ j, p = 1,\ldots, k \wedge j \neq p$
3. Fin para
4. Calcular los centros $\overline{x_i^*} = 1/|C_i| \sum(|C_i| / j=1)\ xj, 1$ para $x_j \in C_i$
5. Si no hay más reasignaciones: $\overline{x_i^*} = \overline{x_i}\ \forall$ i, parar

   Caso contrario considerar $\overline{x_i^*}$ como nuevo centro $\overline{x_i}$ e ir al paso 2

Los principales inconvenientes de k medias son su sensibilidad a la inicialización (debido a esto el resultado final depende del estado inicial). K medias es un algoritmo de búsqueda local que minimiza la función (1) pero no garantiza una configuración óptima de los conglomerados y se debe tener conocimiento previo del valor de k [Vicente05].

Para la inicialización de k-medias, se han discutido en la literatura 3 métodos descritos en este capítulo: Forgy y McQueen (aleatorios) [Forgy65, MacQueen66].

Por otra parte, el algoritmo de Kaufman and Rousseeuw [Kaufman87], el cual es un algoritmo con una “intención heurística” (se le pude insertar un método de aproximación para reducir el alto costo en unidades de cómputo). Este algoritmo identifica los objetos más representativos que prometen tener en su alrededor gran cantidad de objetos. Se sabe que la inicialización aleatoria junto con la propuesta de Kaufman y Rousseeuw ofrece una mejor inicialización para k-medias.

Esta ha sido una buena idea para agrupar las agebs con pam, sin embargo, también se necesito un estudio detallado de k-medias donde se consideraron algunos de sus aspectos básicos en el diseño del particionamiento para las agebs.

3.3 El problema de optimización

La optimización es el proceso de tratar de encontrar la mejor solución posible para un determinado problema. En un problema de optimización existen diferentes soluciones, un criterio para discriminar entre ellas y el objetivo es encontrar la mejor. De forma más precisa, estos problemas se pueden expresar como encontrar el valor de unas *variables de decisión* para los que una determinada *función objetivo* alcanza su valor máximo o mínimo. El valor de las variables en ocasiones está sujeto a unas *restricciones*.

Dicho de otra forma, el término optimización se usa para describir el proceso de hallar la "mejor" solución de entre un conjunto de opciones llamadas "espacio de búsqueda". La mejor solución es dicha en el sentido de óptimo global referente a una cierta función objetivo.

Definición 0. Dado un problema de minimización de la función

$$f : A \subseteq S = \Re^n \to \Re, \quad A \neq \varnothing$$

El valor $-\infty < f^* := f\left(\vec{x}^*\right)$ es llamado el óptimo global (o mínimo) si y solo si para todos los valores $\vec{x} \in A$ se cumple que $f\left(\vec{x}^*\right) \leq f(x)$

A la función f se le conoce como función objetivo.

Algunas clases de problemas de optimización son relativamente fáciles de resolver. Este es el caso, por ejemplo, de los *problemas lineales*, en los que tanto la función objetivo como las restricciones son expresiones lineales. Estos problemas pueden ser resueltos con el conocido método simplex; sin embargo, muchos otros tipos de problemas de optimización son muy difíciles de resolver. De hecho, la mayor parte de los que podemos encontrar en la práctica entran dentro de esta categoría.
La idea intuitiva de problema "difícil de resolver" queda reflejada en el término científico NP-hard utilizado en el contexto de la complejidad algorítmica. En términos coloquiales se puede decir que un problema de optimización difícil es aquel para el que no podemos garantizar el encontrar la mejor solución posible en un tiempo razonable [Lara03].

3.4 Complejidad computacional
Muchos problemas requieren para su solución una serie de pasos, los cuales al agruparse, forman lo que se conoce como algoritmo. Los algoritmos se describen de manera conceptual y posteriormente se pueden llevar a su implementación en una computadora; normalmente los algoritmos se estudian para mejorar su eficiencia.
Se entiende informalmente por eficiencia de un algoritmo si éste es más rápido y/o que ahorre memoria de manera considerable.
Una manera de estandarizar el análisis de la eficiencia de un algoritmo es comparando el número de ciclos de ejecución que éste representa; así los resultados dependerán del algoritmo de manera conceptual y no de la computadora en la que se implemente.

Un algoritmo de tiempo polinomial es aquél cuya complejidad temporal es de orden O(p(n)), donde p(n) es un polinomio. Este tipo de algoritmos describen problemas a los cuales se conoce como problemas tratables.
Un algoritmo de tiempo exponencial es aquel que no puede ser acotado por una función polinomial. Este tipo de algoritmos describen problemas a los cuales se conoce como problemas intratables.
Otra clasificación de problemas, dada por Alan Turing es que un problema pueda o no ser resuelto por una computadora (como conocemos a la computadora actual, es decir una máquina determinista). A aquellos para los cuales la afirmación es cierta se les conoce como problemas decidibles, a los que no pueden ser resueltos por una computadora se les llama problemas indecidibles.

Un algoritmo se dice polinomialmente acotado si la complejidad de su peor caso está acotada por una función polinomial del tamaño de la entrada, es decir, si existe un polinomio p tal que para cada entrada de tamaño n el algoritmo termina a lo sumo en p(n) pasos. (un problema se dice polinomialmente acotado si existe un algoritmo polinomialmente acotado que lo resuelva).
A la clase que contiene a todos los problemas que son resueltos en un tiempo polinomial se les conoce como de clase P.
Un algoritmo no determinista es aquel en el que cada estado puede transitar a varios estados diferentes de manera simultánea y que no se puede predecir en un principio.

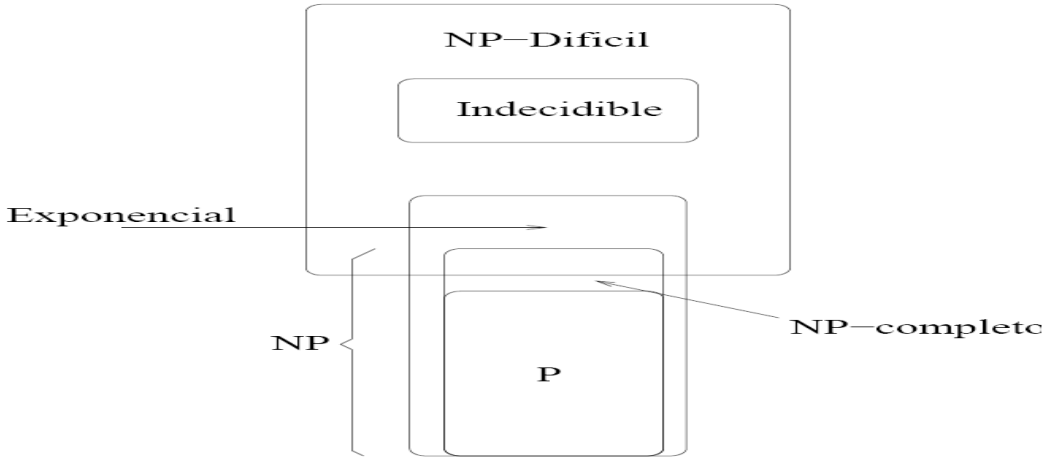
Una computadora no determinística es, de manera informal, aquella que en cualquier paso de sus cálculos puede encontrarse con dos o más cursos alternativos de acción y puede producir copias de sí misma, incluyendo el contenido de su memoria, y continuar con su trabajo de forma independiente para cada alternativa. Considerando otro punto de vista, se puede pensar en un algoritmo no determinístico como aquel que escoge de forma arbitraria uno de los posibles cursos de acción cada vez que se enfrenta con varias posibilidades. La elección puede pensarse como una pregunta que realiza el algoritmo para saber qué alternativa lo conduce a la solución. Empleando esta interpretación de cálculos no determinísticos, se dice que un algoritmo no determinístico resuelve un problema si existe alguna secuencia de preguntas que pueden realizarse para alcanzar una solución.

Problema NP

Un problema se encuentra dentro del conjunto NP si, y sólo si, puede resolverse mediante un algoritmo no determinístico en tiempo polinomial. El nombre NP proviene de “nondeterministic polynomial-bounded” (polinomialmente acotado no determinístico).
Se les llama NP a la clase que contiene a los problemas tales que se resuelven a través de un algoritmo no determinista en un tiempo polinomial.

Un problema se dice polinomialmente transformable en otro si existe una función f tal que en tiempo polinomial transforma los casos de un problema en el otro incluyendo las soluciones. Un problema se dice ser NP-completo si pertenece a la clase NP y todos los problemas de la clase NP son polinomialmente transformables a él. Un problema se dice que es NP-difícil si cumple con que todos los problemas de la clase NP son polinomialmente transformables a él pero él no necesariamente pertenece a la clase NP.

La clase de complejidad NP-completo es el subconjunto de los problemas de decisión en NP tal que todo problema en NP se puede reducir en cada uno de los problemas de NP-completo. Se puede decir que los problemas de NP-completo son los problemas más difíciles de NP y muy probablemente no formen parte de la clase de complejidad P. La razón es que de tenerse una solución polinómica para un problema de NP-completo, todos los problemas de NP tendrían también una solución en tiempo polinómico [Lara03].



Jerarquía de la complejidad

3.5 Heurísticas y optimización combinatoria

La existencia de una gran cantidad y variedad de problemas difíciles, que aparecen en la práctica y que necesitan ser resueltos de forma eficiente, impulsó el desarrollo de procedimientos eficientes para encontrar buenas soluciones aunque no fueran óptimas. Estos métodos, en los que la rapidez del proceso es tan importante como la calidad de la solución obtenida, se denominan heurísticos o aproximados.

“Un método heurístico es un procedimiento para resolver un problema de optimización bien definido mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución” [Lara03].

Los problemas de optimización se dividen de manera natural en dos categorías: problemas de optimización con variables continuas y problemas de optimización con variables discretas. A estos últimos se les llaman problemas de optimización combinatoria.
En problemas de tipo continuo se busca la solución sobre un conjunto de números reales con ciertas propiedades de continuidad y generalmente de convexidad; en los problemas de optimización combinatoria se busca la solución sobre un conjunto finito o infinito numerable de objetos para los que se pueda definir una función que evalúe la “calidad” de cada objeto.

En práctica de los problemas de optimización combinatoria, se han desarrollado una amplia variedad de algoritmos para intentar resolverlos. Estos algoritmos pueden clasificarse como exactos o aproximados. Mientras los primeros garantizan la solución óptima de instancias pequeñas (para instancias grandes se pueden tardar una vida y más), los segundos sacrifican optimalidad, poniendo énfasis en la obtención de soluciones satisfactorias en tiempo reducido.

En contraposición a los *métodos exactos* que proporcionan una solución óptima del problema, los *métodos heurísticos* se limitan a proporcionar una buena solución del problema no necesariamente óptima. Lógicamente, el tiempo invertido por un método exacto para encontrar la solución óptima de un problema difícil, si es que existe tal método, es de un orden de magnitud muy superior al del heurístico (pudiendo llegar a ser tan grande en muchos casos, que sea inaplicable).

Una característica recurrente en los problemas de optimización combinatoria es el hecho que son muy “fáciles”' de entender y enunciar, pero generalmente son “difíciles” de modelar y resolver.

Podría pensarse que la solución de un problema de optimización combinatoria se restringe únicamente a buscar de manera exhaustiva el valor máximo o mínimo en un conjunto finito de posibilidades y que usando una computadora veloz, el problema carecería de interés matemático, sin pensar por un momento, en el tamaño de este conjunto, y, en la mayoría de los casos, este conjunto de posibilidades crece de manera alarmante conforme crece el tamaño de la instancia del problema.

Es frecuente, que los problemas de este tipo tengan n! (n factorial) o más soluciones factibles, (donde n es el tamaño de la instancia) y si una computadora pudiera ser programada para examinar soluciones a razón de un billón de soluciones por segundo; la computadora terminaría su tarea, para n=23 en alrededor de 820 años, para n=24 en alrededor de 19,674 años y así sucesivamente. Por lo que no tiene sentido resolver de esa forma un problema si al interesado no le alcanza su vida para ver la respuesta [DelosCobos96].

En vista de que un gran número de problemas combinatorios resultan NP-completos, y bajo las consideraciones anteriores, el interés de la optimización combinatoria pasa a ser el de desarrollar algoritmos para los cuales el número de etapas computacionales elementales sea aceptablemente pequeño. Equivalentemente, que encuentren la solución en un tiempo razonable o en tiempo polinomial.

Para los problemas de optimización en la clase NP-dura, a la fecha, no se conocen algoritmos que los resuelvan en tiempo polinomial. Es por esto, que en las últimas décadas se han desarrollado algoritmos de tipo heurístico para resolver instancias grandes de problemas que pertenecen a esta clase. Estos algoritmos, no necesariamente encuentran la mejor solución al problema, pero en general se pueden obtener “buenas” soluciones cuando se aplican, aquí se entiende “buena” solución en términos de cercanía al valor óptimo. La ventaja principal de este tipo de algoritmos es que son rápidos y corren en un tiempo polinomial.
En términos generales, se dice que los algoritmos aproximados aportan soluciones cercanas a la óptima en problemas complejos (NP-duros) en un tiempo razonable.

Los factores que pueden hacer interesante su uso son: a) Cuando no hay un método exacto de resolución, o b) éste requiere mucho tiempo de cálculo y memoria (ineficiente) o c) Cuando no se necesita la solución óptima, basta con una de buena calidad en un tiempo aceptable.

Se concluye entonces que los problemas de optimización combinatoria tienen como objetivo encontrar el máximo (o el mínimo) de una determinada función sobre un conjunto finito de soluciones y no se exige ninguna condición o propiedad sobre la función objetivo o la definición del conjunto de soluciones.
Dada la finitud sobre el conjunto de soluciones, las variables han de ser discretas, restringiendo su dominio a una serie finita de valores. Habitualmente, el número de elementos del conjunto de soluciones es muy elevado, y es nada práctico hacer la evaluación de todas sus soluciones para determinar el óptimo.

Atendiendo esta situación, el uso de métodos heurísticos es adecuado cuando:
• El problema es de una naturaleza tal que no se conoce ningún método exacto para su resolución.
• Aunque existe un método exacto para resolver el problema, su uso es computacionalmente muy costoso.
• El método heurístico es más flexible que un método exacto, permitiendo, por ejemplo, la incorporación de condiciones de difícil modelización.
• El método heurístico se utiliza como parte de un procedimiento global que garantiza el óptimo de un problema. Existen dos posibilidades:
- El método heurístico proporciona una buena solución inicial de partida.
- El método heurístico participa en un paso intermedio del procedimiento,

La aplicación de los puntos anteriores al problema NP-difícil de zonificación óptima, implica la necesidad de utilizar métodos heurísticos.
De este modo, en el capítulo siguiente, se presenta el uso del método heurístico de recocido simulado en la solución al problema de particionamiento para agebs, entendido como un problema de optimización combinatoria. La estrategia consiste en insertar recocido simulado en un método de particionamiento *propio y apropiado para agebs* y generar buenas soluciones. Por otro lado, a dicho particionamiento, se le incorporó búsqueda por entorno variable. Los detalles se muestran en el capítulo 4.

3.5.1 Metaheurísticas: una visión global

En los últimos años han aparecido una serie de métodos bajo el nombre de metaheurísticos con el propósito de obtener mejores resultados que los alcanzados por los heurísticos tradicionales. El término metaheurístico fue introducido por Fred Glover en 1986 [Glover86]. En este trabajo se utiliza la acepción de heurísticos para referir a los métodos clásicos en contraposición a la de metaheurísticos los más recientes y complejos.

Los procedimientos meta-heurísticos se sitúan conceptualmente “por encima” de los heurísticos en el sentido que guían el diseño de éstos. Así, al enfrentarnos a un problema de optimización, podemos escoger cualquiera de estos métodos para diseñar un algoritmo específico que lo resuelva aproximadamente.

Otra definición importante es “una metaheurística es un proceso maestro iterativo que guía y modifica las operaciones de heurísticas subordinadas para producir, de forma eficiente, soluciones de alta calidad. En cada iteración, puede manipular una solución (completa o incompleta) o un conjunto de soluciones. Las heurísticas subordinadas pueden ser procedimientos de alto o bajo nivel, o simplemente una búsqueda local o método constructivo” [Pelta00].

Dada la amplia variedad de metaheurísticas existentes, se han intentado realizar varias clasificaciones de las mismas. Por ejemplo, en función del origen del método, se habla de algoritmos “bioinspirados” (algoritmos genéticos, de colonia de hormigas, etc) vs. “no bioinspirados”. Otra posibilidad es categorizarlos en función de la utilización o no de memoria, o en relación del uso de una función objetivo estática o dinámica.

Una clasificación interesante es la que surge al diferenciar los métodos que mantienen una única solución, frente a los que mantienen un conjunto o población de soluciones [Pelta00].

3.6 Metaheurística de recocido simulado

El método de recocido simulado RS o "simulated annealing" SA, está inspirado en el proceso físico de enfriamiento utilizado en el tratamiento de metales y pertenece a una clase más amplia de algoritmos conocidos como algoritmos de umbral.

RS tiene la propiedad de que, en el límite converge al conjunto de soluciones óptimas del problema de optimización combinatoria, y en un tiempo polinomial produce buenas soluciones.
La técnica de recocido simulado se usa para resolver problemas de optimización combinatoria. Fue propuesta por [Kirkpatrick83] y originalmente se basó en una analogía entre la simulación de recocido de sólidos y el reto de resolver problemas de optimización combinatoria de gran escala.

La característica sobresaliente de la técnica es su aplicación general y la habilidad para obtener soluciones arbitrariamente cercanas a la óptima. Sin embargo, el obtener soluciones de alta calidad puede requerir de mucho esfuerzo computacional.

3.6.1 Conceptos preliminares

Para enunciar el algoritmo de recocido simulado se requieren de algunos conceptos preliminares. La búsqueda local constituye una clase interesante de los algoritmos heurísticos y está basado en el mejoramiento paso a paso de la función de costo al explorar las vecindades de soluciones cercanas. Este algoritmo tiene una fuerte relación con el algoritmo de recocido simulado. El uso del algoritmo de búsqueda local presupone la definición de una solución, una función de costo y una estructura de vecindades. A continuación se dan algunas definiciones y ejemplos de las mismas.

Definición 1. Una *instancia* de un problema de optimización combinatoria puede formalizarse como una pareja (S, f) donde S denota el conjunto finito de todas las soluciones posibles y f la *función de costo*, mapeo definido por:

$$f : S \rightarrow R \qquad (1)$$

Para el caso de minimización, el problema es encontrar $i_{opt} \in S$ que satisfaga

$$f(i_{opt}) \leq f(i) \text{ para toda } i \in S \qquad (2)$$

en el caso de maximización, la $i_{opt} \in S$ que satisfaga

$$f(i_{opt}) \leq f(i) \text{ para toda } i \in S \qquad (3)$$

A la solución i_{opt} se le llama *solución globalmente óptima* y $f_{opt} = f(i_{opt})$ denota el costo óptimo mientras que S_{opt} denota el conjunto de soluciones óptimas. Un problema de *optimización combinatoria* es un conjunto I de instancias.

En la definición 1 se ha distinguido entre un *problema* y una *instancia* del problema. De manera informal, una instancia está dada por los "datos de entrada" y la información suficiente para obtener una solución mientras que un problema es una colección de instancias del mismo tipo.
En general un *algoritmo exacto* es un procedimiento paso a paso que resuelve un problema. Se dice que un algoritmo *resuelve* un problema, si puede aplicarse a cualquier instancia y siempre garantiza una solución. Para este caso un algoritmo exacto es aquel que al aplicarse a una instancia (S, f), obtiene $i_{opt} \in S_{opt}$.

Definición 2. Sea (S, f) una instancia de un problema de optimización combinatoria. Entonces una estructura de *vecindades variables* es un mapeo

$$N: S \rightarrow 2^S \qquad (4)$$

Que define para cada solución i ϵ S un conjunto S_i sub. S de soluciones que son "cercanas" a i en algún sentido. El conjunto S_i se llama *vecindad* de la solución i y cada j ϵ S, se llama un *vecino* de i. Además se supone que j ϵ S_i ⇔ i ϵ S_j

Definición 3. Sea (S, f) una instancia de un problema de optimización combinatoria, *N* una estructura de vecindades. Entonces un *mecanismo de generación* es un medio para seleccionar una solución j de la vecindad S_i de la solución i.

Dada una instancia de un problema de optimización combinatoria y una estructura de vecindades. El algoritmo de búsqueda local es un algoritmo que itera sobre un número de soluciones, comienza con una solución inicial, que a menudo se genera aleatoriamente, después se aplica un mecanismo de generación que continuamente trata de encontrar una mejor solución en la vecindad de la solución actual , esto es, una solución con menor costo. Si se encuentra una solución con éstas características, la solución actual se reemplaza por esta solución. De otra manera el algoritmo continúa con la solución actual. El algoritmo termina cuando el mecanismo de generación no puede encontrar una mejor solución a partir de la solución actual. El algoritmo de búsqueda local en pseudo-código se muestra a continuación [DelosCobos96].

Comienza
INICIALIZA ($i_{inicial}$)
i: $i_{inicial}$
Repite
GENERA (j ϵ S_i)
Si f (j) ≤ f (i) entonces i :=j
Hasta f (j) ≥ f (i) , para toda j ϵ S_i
Fin
Algoritmo de búsqueda local en pseudo-código.

Un concepto importante en el análisis de algoritmos de búsqueda local es el de optimalidad local.

Definición 4. Sea (S, f) una instancia de un problema de optimización y *N* una estructura de vecindades, entonces $\bar{i}$ se llama una *solución óptima local* o simplemente un *óptimo local* con respecto a *N* si $\bar{i}$ es mejor que o igual a, todas sus soluciones vecinas con respecto al costo. Específicamente, en el caso de minimización, $\bar{i}$ se llama *solución mínima local* o simplemente un *mínimo local* si:

$$f(\bar{i}) \leq f(j) \text{ para toda } j \in S_i \qquad (5)$$

y en el caso de maximización, $\bar{i}$ se llama una *solución máxima local* o simplemente un *máximo local* si

$$f(\bar{i}) \geq f(j) \text{ para toda } j \in \qquad (6)$$

Definición 5. Sea (S, j) una instancia de un problema de optimización combinatoria y sea *N* una estructura de vecindades. Entonces *N* se llama **exacta** si para cada $\bar{i}$ ϵ S que es localmente óptimo con respecto a *N*, $\bar{i}$ también es globalmente óptimo.
De aquí se observa que, los algoritmos de búsqueda local terminan con un óptimo local a menos que la estructura de vecindades sea exacta. Generalmente es difícil definir estructuras de vecindades exactas, ya que la definición de una estructura de vecindades exacta frecuentemente involucra hacer una búsqueda exhaustiva en todo el conjunto S.

La calidad del óptimo local obtenido por el algoritmo de búsqueda local, depende fuertemente de la solución inicial y para muchos problemas de optimización combinatoria no se tiene a la mano una

solución inicial apropiada. Además la complejidad del problema, en el peor de los casos, no se conoce para muchos problemas.
Entre las ventajas de usar el algoritmo de búsqueda local es que se puede aplicar de manera general, a diferentes problemas específicos de optimización combinatoria, sin alterar su estructura, ya que únicamente se requiere de especificar la función de costo y la estructura de vecindades.

Algunas formas de relajar las desventajas del algoritmo de búsqueda local son las siguientes:

- Ejecutar el algoritmo de búsqueda local para un número grande de diferentes soluciones iniciales. Es claro que de manera asintótica (garantizando que todas las soluciones han sido usadas como inicial) el algoritmo encontrará el óptimo global con probabilidad uno.
- Introducir estructuras de vecindades complejas, de tal forma que se explore una parte grande del conjunto de soluciones. A menudo encontrar una estructura de este tipo requiere de un conocimiento experto del problema de optimización combinatoria tratado.
- Aceptar con ciertos límites transiciones correspondientes a un incremento del valor de la función de costo; note que en el algoritmo de búsqueda local solamente se aceptan transiciones que correspondan a soluciones que decrementen el costo.

Esta última alternativa es la que se explora en el resto del trabajo ya que corresponde a la filosofía del algoritmo de recocido simulado.

El proceso de recocido de un sólido
Recocido denota un proceso de calentamiento de un sólido a una temperatura en la que sus granos deformados recristalizan para producir nuevos granos. La temperatura de recocido o de recristalización, depende del tipo de material, del grado de deformación del mismo, además de su uso futuro. Seguida a la fase de calentamiento, viene un proceso de enfriamiento en donde la temperatura se baja poco a poco.

De esta manera, cada vez que se baja la temperatura, las partículas se reacomodan en estados de más baja energía hasta que se obtiene un sólido con sus partículas acomodadas conforme a una estructura de cristal. Si se comienza con un valor máximo de la temperatura, en la fase de enfriamiento del proceso de recocido, para cada valor de la temperatura *T* debe permitirse que se alcance su equilibrio térmico. Sin embargo, si el proceso de enfriamiento es demasiado rápido y no se alcance en cada etapa el equilibrio térmico, el sólido congelará en un estado cuya estructura será amorfa en lugar de la estructura cristalina de más baja energía. La estructura amorfa está caracterizada por una imperfecta cristalización del sólido.

El equilibrio térmico está caracterizado por la distribución de Boltzmann [Toda83]. De acuerdo a esta distribución, la probabilidad que el sólido esté en un estado *i* con energía E_i a la temperatura *T*, viene dada por

$$P_T\{X = i\} = \frac{1}{Z(T)} \exp\left(\frac{-E_i}{k_B T}\right), \tag{7}$$

donde *X* es una variable aleatoria que denota el estado actual del sólido. *Z*(*T*) es una constante de normalización llamada la *función partición*, que está definida como

$$Z(T) = \sum_j \exp\left(\frac{-E_j}{k_B T}\right), \tag{8}$$

donde la sumatoria se extiende sobre todos los posibles estados y k_B es una constante física conocida como la constante de Boltzmann. El factor $\left(\frac{-E_i}{k_B T}\right)$ se conoce como el factor de Boltzmann.

Obviamente (7) es una función de densidad de probabilidad ya que siempre es mayor o igual a cero y la suma sobre todos los valores es igual a la unidad. Se puede observar en (7) que cuando

el valor de *T* disminuye, la distribución de Boltzmann se concentra en los estados de menor energía mientras que si la temperatura se aproxima a cero, únicamente los estados con mínima energía tienen una probabilidad de ocurrencia diferente de cero.
Por lo dicho anteriormente, el proceso de recocido consta de dos pasos fundamentales que son:

- Incrementar la temperatura del baño térmico a un valor máximo.
- Decrementar *cuidadosamente* la temperatura del baño térmico hasta que las partículas se reacomoden por sí mismas en un estado de mínima energía, denominado el estado fundamental del sólido.

En el proceso de elevar la temperatura del sólido, todas las partículas se reacomodan aleatoriamente. En el estado fundamental, las partículas se acomodan en una retícula altamente estructurada y la energía del sistema es mínima. El estado fundamental del sólido se obtiene solamente si la temperatura máxima es suficientemente elevada y el proceso de enfriamiento es suficientemente bajo. De otra manera se obtendría un estado amorfo denominado meta-estado.

El proceso físico de recocido puede modelarse exitosamente usando métodos de simulación el algoritmo introducido para tal propósito se basa en técnicas Monte Carlo y genera una sucesión de estados del sólido de la siguiente manera:

Dado un estado *i* del sólido con energía E_i, se genera el estado siguiente *j* aplicando un mecanismo de perturbación que transforma el estado actual en el siguiente estado por medio de una pequeña distorsión, por ejemplo, por el desplazamiento de una partícula. La energía del siguiente estado es E_j. Si la *diferencia de energía*, $E_j - E_i$, es menor o igual a cero, el estado *j* se acepta como el estado actual. Si la diferencia de energía es mayor que cero, el estado *j* se acepta con una probabilidad que está dada por

$$\exp\left(\frac{E_i - E_j}{k_B T}\right), \qquad (9)$$

donde *T* denote la temperatura del baño térmico y k_B es la constante de Boltzmann.
La regla de decisión descrita arriba se conoce como el *criterio de metropolis* y al algoritmo se le conoce como *algoritmo de metropolis.*
Si la temperatura se baja poco a poco, el sólido puede alcanzar su equilibrio térmico en cada temperatura. En el algoritmo de metrópolis esto se lleva a cabo generando un número grande de transiciones para un valor dado de la temperatura [Gutierrez91].

Equivalencias de recocido simulado en optimización combinatoria
La simulación del proceso de recocido puede usarse para describir un proceso de generación de sucesiones de soluciones de un problema de optimización combinatoria en donde se vaya obteniendo, conforme el proceso avanza, mejores soluciones al mismo. Para este propósito, se puede observar una analogía entre el sistema físico y un problema de optimización combinatoria en donde cada solución del problema puede verse como un estado del sólido y los valores de la función objetivo para cada solución como los niveles de energía del sólido. En resumen, se puede pensar en las siguientes equivalencias.

- Las soluciones de un problema de optimización combinatoria son equivalentes a los estados de un sistema físico.
- El costo de una solución es equivalente a la energía de un estado.

Además, se introduce un parámetro que juega el papel equivalente de la temperatura. Este parámetro se llama el parámetro de control. El algoritmo de recocido simulado puede verse como una iteración del algoritmo de metropolis, evaluado en valores decrecientes del parámetro de control [Gutierrez91].

Definición 6. Sea (S, F) denote una instancia de un problema de optimización combinatoria y denote por i y j dos soluciones con costo f (i) y f (j), respectivamente.

Entonces el criterio de aceptación determina si j se acepta a partir de i al aplicar la siguiente probabilidad de aceptación:

P c {aceptar j} = 1 si f (j) ≤ f (i)

exp((f (i)- f (j)/c) si f (j) > f (i) (10)

donde c ϵ R^+ denota el parámetro de control.

Claramente, el mecanismo de generación corresponde al mecanismo de perturbación en el algoritmo de metropolis, mientras que el criterio de aceptación corresponde al criterio de metropolis.

Definición 7. Una transición es una acción combinada que transforma la solución actual en una subsecuente. Esta acción consiste de los siguientes dos pasos:

(i) aplicación del mecanismo de generación,

(ii) aplicación del criterio de aceptación.

Sea C_K denote el valor del parámetro de control y L_K el número de transiciones generadas en la k-ésima iteración del algoritmo de metropolis.

Entonces el algoritmo de recocido simulado (algoritmo RS) puede describirse en pseudo-código como se muestra abajo.

Comienza

INICIALIZA $((i_{inicial}, c_0, L_0)$

$k := 0$

$i := i_{inicial}$

Repite

Para $l :=$ a L_k haz

Comienza

GENERA $(j$ de $S_i)$

Si $f(j) \leq f(i)$ entonces $i := j$

Si no

si $\exp\left(\frac{f(i)-f(j)}{c_k}\right) >$ número aleatorio en $[0,1]$ entonces $i := j$

fin para

$k := k+1$

calcula - longitud (L_k)

calcula - control (c_k)

hasta criterio de paro

fin

Un rasgo característico del algoritmo de recocido simulado es que, además de aceptar mejoramientos en el costo, también acepta soluciones más malas en costo. Inicialmente, para valores grandes de c, puede aceptar grandes soluciones deterioradas; cuando c decrece únicamente pequeñas desviaciones serán aceptadas y finalmente, cuando el valor de c se aproxima a cero, no se aceptarán desviaciones. Este hecho significa que el algoritmo de recocido simulado tiene la capacidad de escapar de mínimos locales.

La probabilidad de aceptar desviaciones está implementada al comparar el valor de

exp((f (i)- f (j))/c,

con un número aleatorio generado de una distribución uniforme en el intervalo [0,1). Además, debe ser obvio que la velocidad de convergencia del algoritmo está determinada al escoger los parámetros L_k y c_k , k = 0,1,.... Si los valores k c_k decrecen rápidamente o los valores de L_k no son

grandes, se tendrá una convergencia más rápida que cuando los valores de c_k decrecen lentamente o los valores de L_k son grandes.

Comparando el algoritmo de recocido simulado con el algoritmo de búsqueda local, resulta evidente que recocido simulado puede verse como una generalización de búsqueda local y viene a ser idéntico a búsqueda local, en el caso que el parámetro de control c se tome igual a cero. En lo que respecta a la ejecución de ambos algoritmos, generalmente el algoritmo de recocido simulado obtiene soluciones de mejor calidad que el algoritmo de búsqueda local [Gutierrez91].

Aspectos generales del algoritmo
En las aplicaciones del algoritmo de recocido simulado, comúnmente se desea implementar éste, de manera que la sucesión de soluciones estén generadas a partir de valores decrecientes del parámetro de control. Las soluciones se generan continuamente tratando de transformar la solución actual en una subsecuente por medio de la aplicación del mecanismo de generación y el criterio de aceptación. Las aplicaciones del algoritmo de recocido simulado requieren de la especificación de los siguientes puntos:

a) Una descripción concisa de la representación del problema consiste de una representación del espacio de soluciones y una expresión de la función de costo. La función de costo debe escogerse de manera que represente la efectividad de las soluciones con respecto al objetivo de optimización.

b) La generación de ensayos para transformar la solución actual en una subsecuente consiste de tres pasos. Primero, se debe generar una nueva solución aplicando un mecanismo de generación. Enseguida se debe calcular la diferencia de costo de las dos soluciones, por último, se hace una decisión de aceptar o no, la nueva solución. La evaluación de la nueva solución es lo que más tiempo consume en el algoritmo de recocido simulado y por lo tanto debe hacerse lo más eficientemente posible. El mecanismo de generación usualmente se escoge de tal manera que se obtengan modificaciones simples para que puedan ser ejecutadas rápidamente, por ejemplo permutaciones, cambios o inversiones. El cálculo de la diferencia de costo se hace tomando en cuenta las diferencias entre ambas soluciones. Para muchas aplicaciones éste es el camino más rápido para calcular las diferencias en el costo. La decisión de aceptar la nueva solución se basa en un criterio de aceptación (definición 6).

c) Ejecutar el proceso de recocido, requiere de la especificación de los parámetros que determinan el programa de enfriamiento. Estos parámetros son el valor inicial del parámetro de control, una función que especifique el decremento del parámetro de control, la longitud de cada bloque donde permanece constante el parámetro de control y el criterio de paro.

En [Aarts89] se muestra que si se selecciona el mecanismo de generación, el criterio de aceptación como en la definición 6, una estructura de vecindades y un programa de enfriamiento adecuado; el algoritmo de recocido simulado converge con probabilidad 1 al conjunto de soluciones óptimas.

3.7 Metaheurísticas de búsqueda
Las *metaheurísticas de búsqueda* aportan estrategias para afrontar la resolución de un problema realizando una búsqueda sobre un espacio cuyos elementos representan las soluciones candidatas alternativas. La representación de las soluciones se realiza a través de una codificación que incluya toda la información necesaria para su identificación y evaluación. Una búsqueda sobre un espacio consiste en generar una sucesión de puntos del espacio pasando de uno a otro por medio de una serie de transformaciones o movimientos. Un procedimiento de búsqueda para resolver un problema de optimización realiza recorridos sobre el espacio de las soluciones alternativas y selecciona la mejor solución encontrada en el recorrido. Las metaheurísticas de búsqueda proporcionan pautas para obtener recorridos que, con alto rendimiento, proporcionen soluciones de alta calidad.

La descripción general del proceso de resolución de un problema es, partiendo de una situación inicial, aplicar iterativamente una operación para modificar la situación actual, hasta que se alcance la situación buscada. Un proceso de búsqueda basado en transformaciones o movimientos sobre un espacio de soluciones posibles consiste en la selección iterativa de movimientos para transformar una solución hasta que se cumpla cierto criterio de parada.
El criterio de parada determina cuando se considera resuelto el problema sin que sea necesario disponer, en una situación intermedia, de información de lo cerca que se está de solucionarlo. Sin embargo, las búsquedas inteligentes deben utilizar este y otro tipo de información tanto en el criterio de parada, como en la selección de los movimientos.

En los problemas de optimización, la selección de movimientos y el criterio de parada se diseñan teniendo en cuenta, al menos, un indicador de la calidad de las soluciones encontradas en el recorrido. La evaluación de la calidad de las soluciones se realiza a través de una o varias funciones objetivo, considerando las restricciones del problema. La estrategia de búsqueda establece los criterios y mecanismos que guiaran el recorrido. La estrategia de búsqueda puede incorporar técnicas de una o varias metaheurísticas, junto a heurísticas específicas para el problema. Por su generalidad, la descripción y análisis de las metaheurísticas de búsqueda se realizan sobre problemas de optimización [Hansen96, 03].

3.7.1 Estructura de entornos

Los procedimientos de *búsqueda por entornos* recorren el espacio de soluciones *U* mediante un conjunto de transformaciones o movimientos. Las soluciones que se obtienen de otra mediante uno de los movimientos posibles se denominan vecinas de esta y constituyen su *entorno*. El conjunto de movimientos posibles da lugar a una relación de vecindad y una *estructura de entornos* en el espacio de soluciones, su elección es trascendental e intervienen en ella, la implementación y evaluación eficiente de los movimientos y las propiedades de la estructura de entorno resultante.
El esquema general de un procedimiento de búsqueda por entornos consiste en generar una solución inicial y, hasta que se cumpla el criterio de parada, seleccionar iterativamente un movimiento para modificar la solución. Las soluciones son evaluadas mientras se recorren y se propone la mejor solución del problema encontrada.

El entorno de una solución está constituido por las soluciones a las que se puede acceder desde ella por uno de los movimientos posibles.
Formalmente, una *estructura de entornos* sobre un espacio o universo de búsqueda *U* es una función E: $U \rightarrow 2^S$ que asocia a cada solución $x \in U$ un entorno $E(x) \subseteq U$ de soluciones vecinas a x.
Gran cantidad de métodos heurísticos propuestos en la literatura pertenece a la clase de procedimientos de búsqueda por entornos [Pelta00].
La descripción en pseudocódigo de la búsqueda por entornos se muestra a continuación:

```
1.  Procedimiento búsqueda por entornos
2.  {
3.  x ← genera Solución (U)
4.  x* ← x
5.  Do {
6.        x ← selecciona solución (E(x));
7.          Si (objeto (x) mejora objeto (x*))
8.                x* ← x;
9.          }while (no criterio de parada)
10. }
```

La elección de la estructura de entornos es fundamental en el éxito de los procesos de búsqueda ya que determina la calidad del conjunto de movimientos aplicados. Aparte de la factibilidad y el grado de mejora de los movimientos aplicados es importante la versatilidad de los mismos.
Los movimientos combinados aparecen al ejecutar sucesivamente varios movimientos sobre una solución. Una adecuada combinación de movimientos enriquece los entornos, con lo que se pueden realizar pasos más amplios en el acercamiento al optimo, aunque se corre el riesgo de

perjudicar la eficiencia del algoritmo al tener que contemplar un número mayor de movimientos posibles en el proceso de selección.

Otra característica importante de los movimientos es la factibilidad de las soluciones aportadas. Los movimientos factibles son aquellos que siempre proporcionan una solución factible. Esto puede estar ligado o no al hecho de que se aplique solo a soluciones factibles. En muchos casos, aplicar movimientos más simples, pero no necesariamente factibles, y descartar las soluciones producidas que no sean factibles, es menos eficiente que adaptar el diseño de los movimientos para que sean factibles, sobre todo cuando dicha comprobación es costosa o cuando la probabilidad de que resulte factible sea baja.

Formalmente, los procedimientos que solo consideran movimientos factibles están asociados al concepto, algo más restrictivo, de estructura de entornos como una función E: $S \rightarrow 2^S$ que asocia a cada solución factible $x \in S$ un entorno $E(x) \subset S$ de soluciones factibles vecinas a x.

Las principales metaheurísticas de búsqueda por entornos se centran solo en el procedimiento de selección del movimiento.
Sin embargo, además de la selección de la propia estructura de entornos sobre la que articular la búsqueda, existen otras cuestiones relevantes en el éxito del procedimiento de búsqueda por entornos, como son: la evaluación de la función objetivo, el procedimiento de generación de la solución inicial y el criterio de parada.

La posibilidad de realizar una evaluación eficiente de la solución obtenida tras el movimiento es especialmente importante en aquellos problemas en los que la evaluación de la función objetivo es costosa. Son aplicables las pautas de las metaheurísticas de relajación para evitar cómputos excesivos en la obtención de valoraciones exactas que no son imprescindibles en la conducción de la búsqueda.
Además, se puede contar con procedimientos que evalúan la calidad de los movimientos sin tener que realizar una evaluación completa de la nueva solución desde cero. Para ello se utilizan procedimientos que actualizan inmediatamente el valor de la función objetivo tras el movimiento, utilizando el valor anterior y los cambios producidos por el movimiento.

Las pautas de las metaheurísticas constructivas se utilizan para el diseño de los procedimientos de generación de la solución inicial. En este sentido, las características fundamentales son la calidad y dispersión de la solución de arranque desde la que se inician las búsquedas por ejemplo *grasp,* la cual propone un procedimiento para conseguir diferentes soluciones de alta calidad.
Por último, en relación a los criterios de parada, los parámetros más comunes se refieren a un límite al número de iteraciones, movimientos, operaciones elementales o tiempo de cómputo total o sin que se produzca alguna mejora.

Otras dos características fundamentales en el procedimiento de búsqueda por entorno resultante de aplicar metaheurísticas son las capacidades de exploración y de explotación. La *exploración* se refiere a la capacidad del método para explorar las diferentes regiones del espacio de búsqueda para alcanzar la zona en la que se encuentra la solución del problema. La *explotación* de la búsqueda se releja en el esfuerzo y capacidad por mejorar las soluciones con las que trabaja el procedimiento. Existe un amplio consenso en que estas dos características deben modularse adecuadamente para conseguir el éxito práctico de las aplicaciones de las metaheurísticas [Hansen03].

Búsquedas locales
El término *local* se emplea con bastante frecuencia en los estudios teóricos y prácticos del campo de las metaheurísticas de búsqueda. Las estructuras de entorno suelen reflejar algún concepto de proximidad o vecindad entre las soluciones alternativas del problema. Por tanto, el análisis del entorno de la solución actual en el recorrido de búsqueda para decidir como continuarla, representa un estudio local del espacio de búsqueda.

Las *metaheurísticas de búsqueda local* establecen pautas de selección de la solución del entorno a la solución actual, dando lugar a búsquedas locales heurísticas con alto rendimiento. Las búsquedas locales no informadas solo tienen en cuenta la estructura de entornos para guiar la búsqueda. Las búsquedas monótonas utilizan la evaluación de la función objetivo para admitir solo cambios en la solución actual que supongan una mejora.
Por tanto, las búsquedas locales monótonas quedan atrapadas al llegar a una solución que no admite mejora dentro de su entorno. Las búsquedas globales emplean diversos métodos para escapar de esta situación.

3.7.2 Búsqueda por entorno variable

La búsqueda por entorno variable (*variable neighborhood search*, VNS) es una metaheurística reciente que consiste en cambiar de forma sistemática la estructura de entorno [Hansen03]. La idea original fue considerar distintas estructuras de entornos y cambiarlas sistemáticamente para escapar de los mínimos locales. El VNS básico obtiene una solución del entorno de la solución actual, ejecuta una búsqueda monótona local desde ella hasta alcanzar un óptimo local, que reemplaza a la solución actual si ha habido una mejora y modifica la estructura de entorno en caso contrario.

Una variante de esta estrategia básica es la búsqueda descendente por entorno variable VND, la cual, aplica una búsqueda monótona por entornos cambiando de forma sistemática la estructura de entornos cada vez que se alcanza un mínimo local.
Otra es utilizar una selección de los entornos, por medio de la elección de un conjunto de sus atributos (variables), de tal manera que se produzca una descomposición, y por tanto se definen determinados subproblemas, este modelo es denominado búsqueda por descomposición de entornos variables VNDS.

En metaheurísticas “combinadas”, es difícil discernir si el mérito lo tiene una de las componentes o se está obteniendo beneficio de la interacción.
Tres ideas para comprender estas cuestiones para cualquier metaheurística de búsqueda, que han sido utilizadas con la VNS, son el estudio de la topografía de los óptimos locales de las trayectorias seguidas por los procesos de búsqueda, y de las fases de mejora y deterioro de las soluciones.

El estudio de la topografía de los mínimos locales da lugar a una descripción de los *perfiles de montañas y valles* encontrados durante la búsqueda. Cuando se aplica la VNS, el descenso desde una solución x', seleccionada al azar, puede volver al óptimo local x alrededor del que están centrados los entornos actuales, o a otro optimo local x'' cuyo valor puede ser mejor o no que el de x. Se ha estudiado la probabilidad de estos tres sucesos como una función de la distancia de x a x' al aplicarla al *problema de la máxima satisfactibilidad ponderada* [Hansen03a].

Conviene también observar si x'' está más cerca de x que x' (lo que puede ser interpretado como que x'' pertenece al mismo valle que x, con rugosidades locales en relieve) o no (lo que puede interpretarse como un indicio de que se ha encontrado un nuevo gran valle).
El relieve suministra esta información en base a una determinada cantidad de descensos desde puntos en entornos sucesivamente más amplios. Los perfiles varían considerablemente con la calidad de óptimo local x. Cuando x es una mala solución es suficiente alejarse un poco para obtener, con alta probabilidad, un óptimo local mejor. Cuando el óptimo local x es bueno, o muy bueno, debe alejarse bastante más para encontrar un nuevo gran valle y, además, la probabilidad de encontrar una solución mejor que la actual es entonces baja. Esto ilustra una debilidad del esquema básico de la VNS que tiende a degenerar en un *arranque múltiple* cuando la distancia de x a x' se hace grande. El remedio es el proporcionado por el esquema de la VNS sesgada.
Para algunos problemas se dispone de instancias para las que se conoce el óptimo global [Hansen03a].

No es “elegante” analizar las búsquedas heurísticas determinando lo frecuente que se alcanza el optimo global o el promedio de la diferencia relativa entre el valor optimo del objetivo y el alcanzado por la búsqueda.
Sin embargo, se obtiene mucha más información si se consideran las propias soluciones y no sólo su valoración, por ejemplo, cuando es posible obtener la solución óptima y la solución actual en tiempo real para compararlas y contar con la diferencia simétrica entre estas dos soluciones. Esto indica cuanta mejora queda por hacer y donde. Esto aún es mejor si una rutina permite la representación de la *diferencia* entre la solución actual en una iteración y en la siguiente [Hansen03, 03a].

Finalmente, como las representaciones de las soluciones para problemas grandes pueden ser difíciles de leer, y muchos problemas, en particular los problemas euclídeos, permiten una descomposición natural, una rutina de *enfoque* permite la representación de la información mencionada anteriormente para subproblemas seleccionados; es decir, en alguna región del espacio en la que se traza la ruta.

Es muy provechoso el uso de herramientas computacionales de visualización para guiar la búsqueda, así, se puede evaluar el trabajo realizado paso a paso por una VNS u otra metaheurística. Se pueden estudiar variantes en cada uno de esos pasos y recopilar información detallada para sacar conclusiones que pueden no ser evidentes en un rendimiento global de la heurística. Esto puede dar lugar al descubrimiento de fenómenos inesperados y, la profundización proporcionada por su explicación, puede a su vez dar lugar a principios para mejorar las heurísticas.

Se ha observado que seleccionando la primera mejora (es decir, el primer movimiento que reduce el valor de la función objetivo) se obtienen mejores resultados que seleccionando la mejor mejora, (es decir, el movimiento que más reduce el valor de la función objetivo) empezando desde una solución elegida al azar [Melián03, Hansen03a].

VNS se ajusta a las propiedades de las metaheurísticas como se lista a continuación:
Simplicidad: la metaheurística debe estar basada en un principio simple y claro.
Precisión: los pasos de la metaheurística deben formularse en términos precisos.
Coherencia: las operaciones del algoritmo heurístico deben derivarse naturalmente de los principios de la metaheurística.
Efectividad o *eficacia*: los procedimientos deben proporcionar soluciones óptimas o cercanas a la óptima para todos o al menos para la mayoría de los casos realistas.
Eficiencia: los algoritmos deben emplear un tiempo computacional moderado para proporcionar soluciones óptimas o cercanas a la óptima.
Robustez: el rendimiento de las heurísticas debe ser consistente sobre una variedad de casos
Amigable: las heurísticas deben expresarse claramente, ser fáciles de entender y usar.
Innovación: los principios de la metaheurística, y/o la eficiencia y efectividad de las heurísticas derivadas de ella, deben dar lugar a nuevos tipos de aplicación [Hansen03a].

La VNS está basada en un principio simple y relativamente no explorado: el cambio sistemático de la estructura de entornos durante la búsqueda. La *simplicidad* de la metaheurística contribuye a dotarla de la amplia aplicabilidad que se refleja en la variedad de las aplicaciones ya comentadas. Las reglas *precisas* en la descripción de la forma de efectuar tales cambios son cruciales. Todos los pasos de los esquemas básicos y extendidos de la VNS se traducen *coherentemente* de los principios en que se inspira. Para validar la *efectividad* de una metaheurística, estas deberían encontrar soluciones óptimas para la mayoría de los problemas de un banco de casos para el que se conozcan las soluciones, cuando se disponga de ellos. La VNS tiene probada *eficacia* en la resolución de problemas de varios bancos de prueba con resultados óptimos o muy cercanos a los óptimos, y con tiempo computacional moderado (o al menos razonable).

La metaheurística VNS se ha mostrado muy *eficiente* en muchos experimentos, ya que obtiene tan buenos o mejores resultados que la mayoría de las metaheurísticas en muchos problemas y de

una forma mucho más rápida. Además, la VNS se muestra *robusta* ya que ha probado su rendimiento en multitud de problemas, y no sólo con un ajuste fino de parámetros a algún conjunto de entrenamiento. La *amigabilidad* de los sistemas basados en la VNS cuentan con la ventaja de que los principios básicos en que se basa son fáciles de aplicar, y muy fáciles de usar. Esta cualidad se traduce en que en las aplicaciones realizadas, el número de parámetros se mantienen a un mínimo o incluso están ausentes.

La búsqueda por entorno variable es, por tanto, una metaheurística reciente con el merito de ajustarse a las cualidades de la lista anterior. Esto, junto con la frecuencia con la que las metaheurísticas incorporan las ideas en las que se basa la VNS para beneficiarse de sus efectos positivos, justifica un papel destacado de la VNS en el campo de las metaheurísticas.

3.7.3 Esquemas fundamentales

Un problema de optimización consiste en encontrar, dentro de un conjunto X de soluciones factibles, la que optimiza una función $f(x)$. Si el problema es de minimización se formula como sigue:

$$\text{Min } \{ f(x) \mid x \in X \} \qquad (1)$$

Donde x representa una *solución* alternativa, f es la *función objetivo* y X es el *espacio de soluciones* factibles del problema.

Una *solución optima* x^* (o mínimo global) del problema es una solución factible donde se alcanza el mínimo de (1).

Una *estructura de entornos* en el espacio de soluciones X es una aplicación $N : X \rightarrow 2^X$ que asocia a cada solución $x \in X$ un entorno de soluciones $N(x) \subset X$, que se dicen *vecinas* de x.

Las metaheurísticas de búsqueda local aplican una transformación o movimiento a la solución de búsqueda y por tanto utilizan, explícita o implícitamente, una estructura de entornos. Denotemos por N_k, $k = 1; \ldots ; k_{max}$, a un conjunto finito de estructuras de entornos en el espacio X. Los entornos N_k pueden ser inducidos por una o más métricas introducidas en el espacio de soluciones X. La mayoría de las heurísticas de búsqueda local usan sólo una estructura de entornos.

Una solución $x^* \in X$ es un *mínimo global* del problema (1) si no existe una solución $x \in X$ tal que $f(x) < f(x^*)$. Decimos que $x' \in X$ es un *mínimo local* con respecto N_k si no existe una solución $x \in N_k(x') \subseteq X$ tal que $f(x) < f(x')$.

Una búsqueda local descendente cambia la solución actual por otra solución mejor de su entorno, por tanto tienen el peligro de quedarse atascada en un mínimo local. Las metaheurísticas basadas en procedimientos de búsqueda local aplican distintas formas de continuar la búsqueda después de encontrar el primer óptimo local.

La VNS está basada en tres hechos simples:

1. Un mínimo local con una estructura de entornos no lo es necesariamente con otra.
2. Un mínimo global es mínimo local con todas las posibles estructuras de entornos.
3. Para muchos problemas, los mínimos locales con la misma o distinta estructura de entornos están relativamente cerca.

Esta última observación, que es empírica, implica que los óptimos locales proporcionan información acerca del óptimo global. Puede ser, por ejemplo, que ambas soluciones tengan características comunes. Sin embargo, generalmente no se conoce cuáles son esas características. Es procedente, por tanto, realizar un estudio organizado en las proximidades de este óptimo local, hasta que se encuentre uno mejor.

Los hechos 1 a 3 sugieren el empleo de varias estructuras de entornos en las búsquedas locales para abordar un problema de optimización.

El cambio de estructura de entornos se puede realizar de forma determinística, estocástica, o determinística y estocástica a la vez.

Están claros algunos aspectos del trabajo que aún queda por hacer para VNS. Aunque la VNS y otras metaheurísticas se han mostrado claramente útiles para la resolución aproximada de muchos problemas difíciles, les cuesta resolver problemas muy grandes. El tamaño de los problemas abordados se limita en la práctica por las herramientas disponibles para resolverlos más que por la necesidad de los potenciales usuarios de estas herramientas. Cuando las metaheurísticas se aplican a instancias realmente grandes sus fortalezas y debilidades aparecen más claramente. Las mejoras en este sentido se presentan altamente deseables, tanto para las metaheurísticas en general, como para la VNS en particular [Hansen03, 03a].

Capitulo 4: Recocido simulado y búsqueda por entorno variable en particionamiento geográfico

Introducción

El problema de particionamiento geográfico PG, (conocido también como conglomerado geográfico CG), agrupa agebs. La función objetivo implícita, calcula el costo mínimo de distancias entre estas. Este problema se distingue como una variante del problema de regionalización.

En este capítulo se resuelve este problema, el cual es discreto y entero mixto; su solución se formula con un modelo de optimización combinatoria bajo el criterio de compacidad como función objetivo.
El algoritmo de particionamiento que se expone en este capítulo, se apoya de los métodos clásicos de partición que ya se han descrito en el capítulo 3.

El carácter combinatorio de particionamiento para agebs exige para su solución el uso de métodos aproximados. Para remediar esta situación, se dispone de dos métodos heurísticos de eficiencia comprobada en la solución de problemas de optimización combinatoria: búsqueda por entorno variable VNS y recocido simulado RS.
Por último, se evalúa la calidad de las soluciones con un método estadístico factorial conocido como box behnken y superficie de respuestas.

4.1 Particionamiento geográfico

En el algoritmo de particionamiento geográfico PG, las condiciones espaciales de las variables geográficas dentro del grupo de variables de clasificación favorecen la creación de regiones espacialmente compactas, lo cual se traduce en la satisfacción de la restricción de continuidad geográfica.
Estas propiedades son ajustadas al algoritmo de particionamiento para agebs dado el elevado número de áreas por agrupar y por la exigencia de cumplir la compacidad en la agregación.

Por otro lado, la utilización de este tipo de aproximaciones implica, en algunos casos, el otorgar gran importancia a las variables geográficas para garantizar la satisfacción de la restricción de continuidad geográfica. Esto asume que el papel de las variables no geográficas (por ejemplo variables socioeconómicas) dentro del proceso de agregación pasaría a ser secundario. Sin embargo, este tipo de variables son muy importantes cuando la agregación resuelve problemas como equilibrio entre variables, homogeneidad o balanceo. Esta condición de homogeneidad se resuelve en el capítulo 5.

El problema de agrupamiento de datos espaciales es visto como un caso particular de conglomerado cuando se cubre la continuidad geográfica entre los datos. En esta dirección, para la agrupación de agebs bajo el criterio de compacidad, se construyó un modelo de particionamiento geográfico con las propiedades del particionamiento clásico.
La optimización para este caso se soluciona cuando se toma como función objetivo la minimización de distancias entre los objetos a agrupar con el fin de lograr la compacidad geográfica (tan buscada en problemas de diseño geográfico).
Se resuelve la agrupación de agebs de tal forma que las agebs que componen los grupos estén entre ellas muy cercanas geográficamente donde se requiere el uso de una función de costo que minimice distancias entre estas.
El método consiste en elegir aleatoriamente agebs como centroides que identifican el número de grupos. Los agebs no centroides que tengan la distancia más corta hacia un determinado centroide-ageb, son los integrantes de un grupo. Esta idea informal es la que se entiende como compacidad geométrica.

Definir formalmente compacidad especialmente para problemas de diseño territorial, no es tan simple, sin embargo, en la definición 1 se plantea la compacidad para unidades geográficas.
Se consideran los siguientes conceptos que son muy utilizados para resolver compacidad y conexidad para problemas de agrupación geográfica.

Por otro lado, se han estudiado otras medidas de distancia, como la geodésica, propuesta en [Sanchez06], para eventualmente hacer pruebas al agrupamiento para agebs [Gower86].

4.1.1 Restricciones en particionamiento geográfico

La condición de compacidad es la que se satisface implícitamente dentro del algoritmo que se expone en este capítulo.

Conexidad

La conexidad de una zona puede definirse como la factibilidad de viajar entre las unidades geográficas que la forman sin salirse de la zona, o bien como la posibilidad de conectar cualesquiera dos unidades geográficas de una zona mediante unidades geográficas que también formen parte de dicha zona. Para hacerlo, deben determinarse las unidades geográficas que serán consideradas como contiguas o vecinas ya que en base a esta contigüidad se podrá evaluar si una zona es conexa o no.

Compacidad geométrica

El concepto de compacidad geométrica surge como una restricción que busca evitar la creación de zonas con formas irregulares. Una forma de comprender el concepto de compacidad geométrica es pensar que la forma de cada zona generada debe ser tan "parecida" como sea posible a un cuadrado o a un círculo. Otra manera de expresar esta idea consiste en considerar que una zona es geométricamente compacta si es de forma "homogénea localizada dentro de un espacio definido y limitado que no presenta irregularidades o figuras confusas sobre un área amplia". Aunque estas descripciones reflejan la idea intuitiva de compacidad geométrica, no proporcionan una definición rigurosa o exacta que pueda ser utilizada para determinar si un plan de zonificación es o no compacto. Como consecuencia ninguna de las medidas propuestas hasta el momento ha resultado totalmente satisfactoria. Por lo cual deberá ponerse especial atención en la selección o propuesta de una medida adecuada para este problema.

Algunos ejemplos de las medidas de compacidad propuestas son los siguientes:

$\sum_{j}\sum_{i\in Z_j} d_{ij}$ donde d_{ij} representa la distancia euclidiana entre la i-esima unidad geográfica y el centro de la zona que la contiene.

$\sum_{i}\frac{pr_i^2}{a_i}$ donde pr_i es el perímetro y a_i el área de la i-esima zona.

$\left(\sum_{j\in J} R_j(x) - R\right) \Big/ 2R$ donde $R_j(x)$ es el perímetro de la zona j en la solución x y R es el perímetro del territorio total.

$\sum_{j\in J}\left(1 - \frac{2\pi\sqrt{A_j(x)/\pi}}{R_j(x)}\right) \Big/ m$ donde $R_j(x)$ es el perímetro de la zona j en la solución x y $A_j(x)$ es un círculo que tiene la misma área que la zona j.

Ciertas características que debe tener una buena medida de compacidad son:

1. La medida debe ser aplicable a todas las formas geométricas, regulares e irregulares.
2. La medida debe ser independiente de escala y orientación.
3. La medida debe ser carente de dimensiones, y estar definida preferiblemente en una escala de 0 a 1, con 1 describiendo una región altamente compacta.
4. La medida no debe ser muy afectada por uno o dos puntos extremos.
5. La medida debe corresponder con la "intuición".

La siguiente definición de compacidad es la utilizada en el algoritmo para agrupar agebs:

Definición 1. Compacidad
Si denotamos por Z={1, 2, …, n} al conjunto de *n* objetos a clasificar, se trata de dividir Z en *k* grupos $G_1, G_2, \ldots, G_k$ con *k*<*n*, de tal forma que:

- $\bigcup_{i=1}^{k} G_i = Z$
- $G_i \bigcap G_j = \phi \qquad i \neq j$
- $|G_i| \geq 1 \qquad i = 1, 2, \ldots, k$

Un grupo G_m con $|G_m| > 1$ es compacto si para cada objeto $t \in G_m$ cumple:

$$\min_{i \in G_m,\ i \neq t} d(t,i) < \min_{j \in Z - G_m} d(t,j) \qquad \text{(CV 1)}$$

Un grupo G_m con $|G_m| = 1$ es compacto si su objeto *t* cumple:

$$\min_{i \in Z - \{t\}} d(t,i) > \max_{j,l \in G_f \ \forall f \neq m} d(j,l)$$

El criterio de vecindad entre objetos para lograr la compacidad esta dado por los pares de distancias descritos en (CV 1).

En la figura 1 se muestra un ejemplo de cómo dos agebs cumplen las restricciones anteriores. El grupo azul que solamente consta de un ageb está bien clasificado porque la distancia entre él y el ageb más cercano es mayor que la distancia mayor entre dos agebs de un mismo grupo. Así mismo para el grupo rojo que tiene más de un ageb se muestra en la figura como el ageb *t* de este grupo está bien clasificado porque la menor distancia que hay entre él y el ageb más cercano de su mismo grupo es menor que la distancia entre este y el ageb *j* más cercano de diferente grupo.

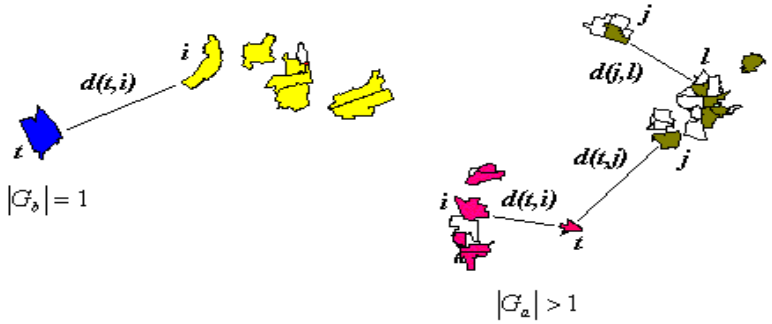


Figura. 1 Cuatro grupos compactos

Se usa la definición de compacidad para la modelación del problema de conglomerado geográfico.

4.2 Modelo para particionamiento geográfico
La siguiente modelación de particionamiento se ha diseñado para adaptarla a cualquier heurística. En este caso VNS y RS son las heurísticas a incluir en dicho modelo.

Modelación PG
Datos:
UG= total de agebs

Sea el conjunto inicial de n unidades geográficas UG={$x_1, x_2, \ldots, x_n$}, donde
x_i es la i-ésima unidad geográfica, (i=índice de UG)
k es el número de zonas (grupos)
Dado que se desean forman grupos y para referirnos a estos, definimos:
Z_i como el conjunto de las unidades geográficas que pertenecen a la zona i
n es el número de unidades geográficas
C_t es el centroide
d(i,j) es la distancia euclidiana del nodo i al nodo j (de un ageb a otro)

Entonces se tienen como restricciones:
$Z_i \neq \varnothing$ para i = 1,…, k (los grupos no son vacíos)
$Z_i \cap Z_j = \varnothing$ para $i \neq j$ (no existen agebs repetidos en distintos grupos)
$\bigcup_{i=1}^{k} Z_i = UG$ (la unión de todos los grupos son todos los agebs)

Una vez que se ha decidido el número k de centroides c_t, t = 1,...k, a utilizar hay que seleccionarlos en forma aleatoria y enseguida asignar los agebs a los centroides de la siguiente manera:

Para cada ageb i

$$\underset{t=1,\ldots,k}{Min} \{d(i, c_t)\}$$

cada ageb es asignado al centroide más cercano c_t.

Para cada valor de k se calcula la suma de las distancias de los agebs asignados a cada centroide y se escoge el mínimo y nit es el número de iteraciones. Esto puede expresarse como:

$$\underset{k=1,..,n_{it}}{Min} \left\{ \sum_{t=1}^{k} \sum_{i \in c_t} d(i, c_t) \right\} \quad (1)$$

Algoritmo de recocido simulado
Se sabe que RS es un método de búsqueda por entornos caracterizado por un criterio de aceptación de soluciones vecinas que se adapta a lo largo de su ejecución. Hace uso de las variables ya conocidas y que denotamos así:
Temperatura inicial T_i, Temperatura final Tf, alfa α y L(t).

Pseudocódigo de RS

```
INPUT (To, α, L(t), Tf)
T← To                                  /* Valor inicial del parámetro de control
Sact ← Genera solución inicial
WHILE T >= Tf DO                       /* Condición de parada
   BEGIN
   FOR cont ← 1 TO L(T) DO             /*Velocidad de Enfriamiento (T)
      BEGIN
      Scand ← Selecciona solución N(Sact)   /*Generación de una nueva solución
      ∂← costo(Scand)-costo(Sact)      /*Cálculo de la diferencia de costos
      IF  U(0,1) < e^(-∂/T)  OR        /*Aplicación del criterio de aceptación
      END
   T←∝(T)                              /*Mecanismo de enfriamiento
   END
{Escribe como solución la mejor de las Sact visitadas}
```

Finalmente este pseudocódigo con la inclusión del modelo PG, queda integrado de la siguiente manera:

4.3 Algoritmo de recocido simulado y particionamiento para el problema de conglomerado geográfico

Sea n el número de objetos a clasificar
UG_{ij} denota que el objeto i está asignado al centroide j
i=1, ... , n; j=1,…,k
Sea M={M_1,M_2,,M_k} Una solución de K centroides
T_0 temperatura inicial
T_f temperatura final
L(t) número de iteraciones que se van a realizar con la misma temperatura

1. Inicio

/ Obtiene Solución inicial*
Generar aleatoriamente centroides iniciales M = {M_1, M_2, …, M_k}
/ 'Cualquier ageb puede ser centroide obtenido de forma aleatoria*
costo_act ← Costo (M)*
/ 'Esta asignación representa ya una solución inicial, es una solución propuesta generada por el paso anterior. En los siguientes pasos se genera otra solución (solución vecina) para determinar que tan buena es con respecto a la actual y decidir si se cambia o no la solución actual.*

Mientras T ≥ Tf
/ 'mientras el sistema no sea frio*
Para cont=1 hasta L(t) hacer
/ 'número de ciclos a realizar con la misma temperatura (parámetro de RS)*
C← Genera una solución aleatoria
/ 'se genera la solución que se compara con **
costo_cand ←Costo (C)
/ 'se obtiene el costo de solución candidata que se ha generado*
∂ ← costo_cand – costo_act
/ 'diferencia de costos para obtener el valor de probabilidad de aceptación de la solución candidata*
Si U(0,1) < $e^{-\partial/T}$ o ∂ < 0 hacer
/ 'si la probabilidad de aceptación aún es alta*
M←C
/ 'si se acepta la solución candidata*
costo_act ← costo_cand
Fin Si
fin para
t ← α t
/ 'se está enfriando el sistema*
fin Mientras

2. Función costo (Sol)

/ 'determina que tan buena es la solución SOL, es decir, que tanto minimiza el objetivo*

i ← 1
/ 'inicializa primer objeto*
cost ← 0
Mientras i ≤ n
/ 'para cada objeto en Ug hacer*
si Ug_i no es centroide entonces
dmin ← dist(Sol_1 , Ug_i)
/ 'representa la distancia del objeto$_i$ hacia Sol_1 (primer centroide donde Sol representa al conjunto de todos los centroides. Se calcula la distancia cada objeto a su centroide más cercano, (distancia de un objeto $_i$ que no es centroide hacia Sol_1 que es el centroide 1)*
j ← 2
/ 'paso al segundo centroide*

Mientras j ≤ k

Si dist (Sol_j , Ug_i) < dmin

/* *'se calcula la distancia del objeto i hacia Sol_j (otro centroide)*

dmin ← dist (Sol_j , Ug_i)

Fin si

j ← j + 1

/* *'paso al siguiente centroide*

Fin Mientras

cost ← cost + dmin

fin si

i ←i + 1

Fin Mientras

Costo (Sol) ← cost

Una vez implementado el algoritmo anterior, se efectúo un conjunto de pruebas para comprobar la funcionalidad general: el tiempo de respuesta debe reducir considerablemente el costo computacional, la calidad de las soluciones debe estar muy cercana al óptimo, el número de conglomerados y parámetros almacenados en el archivo de salida tienen que corresponder a la entrada del algoritmo y cada iteración debe estar asociada con el valor de la solución sub-óptima.

La siguiente figura 2 muestra una diagrama de flujo de datos del particionamiento con recocido simulado.

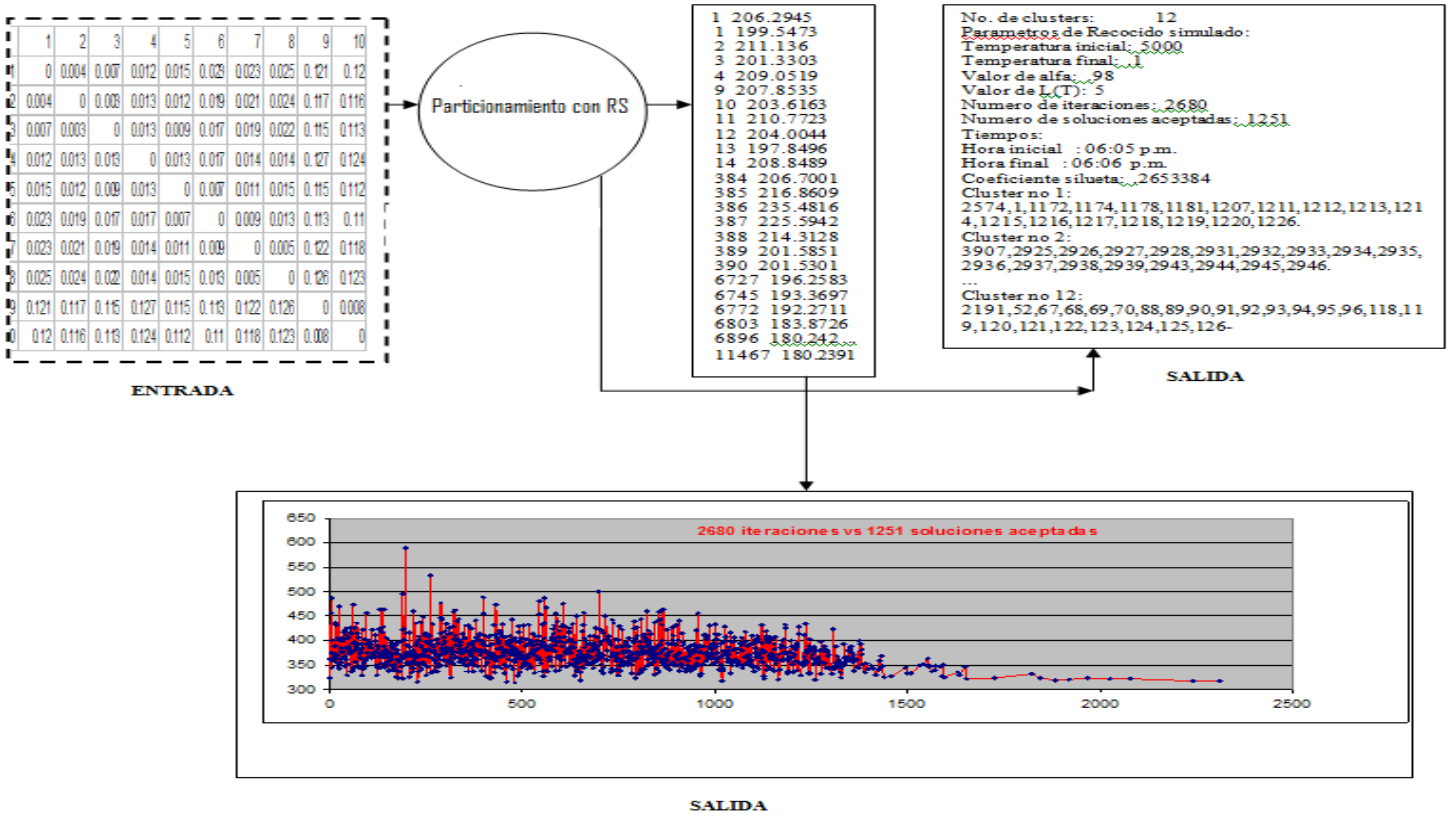


Figura 2. Diagrama de flujo de datos en PG con RS

Primera fase de pruebas para recocido simulado

La primera parte de pruebas ha sido experimental y aleatoria. La siguiente tabla 1 muestra algunas pruebas para diferentes números de grupos.

iteración	FC	Ti	Tf	alfa(α)	l(t)	G
2670	26.06359	5000	0.1	0.98	5	5
2658	9,014,701	5000	.1	.98	5	40
4812	4,728,044	5000	.1	.98	5	5
2659	250,508	5000	.1	.98	5	5
5042	4,009,179	5000	.1	.98	5	8
1749	5,178,715	5000	.1	.98	5	4
14234	160,896	10000	.0001	.99	1	12
12292	170,194	9000	.0001	.99	1	12
10725	1,707,811	9000	.001	.99	1	12
9388	1,709,711	8000	.01	.99	1	12
7888	1,619,069	5000	.1	.98	5	12
2653	11.2403	5500	0.055	0.985	4	24
3173	11.5392	5250	0.01	0.985	4	24
2863	12.02921	5250	0.1	0.985	4	24
2909	11.8841	5000	0.055	0.985	4	24
3466	11.6377	5250	0.055	0.985	5	24
2131	12.33139	5250	0.055	0.985	3	24
2200	12.21459	5250	0.055	0.98	4	24
3714380	11.02201	50000	0.01	0.98	5000	24
663177	11.0599	9900	0.1	0.98	1000	24
332692	11.463	9000	0.1	0.98	500	24

Tabla 1. Pruebas experimentales para PG con RS

Analizando los resultados de la tabla anterior, se observa que temperatura inicial (Ti) debe ser un valor alto (mayor o igual a 5000) para que la exploración en el espacio de soluciones sea aceptable. En cada iteración se multiplica la temperatura actual por alfa. Con este mecanismo de enfriamiento, al principio (cuando la temperatura es muy alta) el sistema se enfría rápidamente (cuando hay mayor probabilidad de aceptar una solución peor que la actual), cuando la temperatura es baja, el valor de la temperatura actual decrece cada vez menos. Si el valor de alfa es muy pequeño, el sistema se enfriará rápidamente; entre más cercano este a cero, el numero de exploraciones será más pequeño aunque la temperatura inicial sea muy alta. Por ejemplo, con alfa=0.1 solo se realizarán alrededor de 4 iteraciones. Entre más cercano este el valor de alfa a uno, el numero de iteraciones será mayor y por tanto se explorará más espacio de soluciones. Entonces se ha elegido a alfa (α) con un valor de 0.98 o .985.

La temperatura final (Tf) se eligió entre 0.055 y .1. En “teoría”, el algoritmo debe finalizar cuanto la temperatura actual sea igual a cero, pero al ir enfriando el sistema multiplicando el valor de la temperatura actual con alfa, el valor de la temperatura actual se irá haciendo cada vez más pequeño sin llegar a ser cero, por lo que un valor muy pequeño es adecuado como criterio de finalización.

Sin embargo, se observó que aumentando la temperatura inicial y L(t) mejora el valor de la función objetivo pero no como se esperaba, lo que implica que las pruebas deben estar controladas de alguna manera sistemática, lo que significa que las pruebas pueden modelarse para evaluar la calidad de las soluciones generadas.

Con la aplicación de un diseño de experimentos estadístico factorial es posible encontrar un balance y adecuación de los valores de los parámetros de recocido simulado en el control de la obtención de buenas soluciones.

4.3.1 Diseño de un experimento box behnken para particionamiento geográfico con recocido simulado

El diseño estadístico de experimentos es el proceso de planear un experimento para obtener datos apropiados que puedan ser analizados mediante métodos estadísticos con el objetivo de producir conclusiones válidas y objetivas.

La metodología estadística es un enfoque objetivo para analizar un problema que involucre datos sujetos a errores experimentales.
En el diseño de un experimento es importante "caracterizarlo", es decir se determinan los factores controlables e incontrolables que influyen en la ocurrencia de defectos. En este sentido, es posible diseñar un experimento que estime la magnitud y dirección de los efectos del factor. Esto representa que en la medida que cambia la variable de respuesta, se identifica como se modifica cada factor. Sin embargo, cuando cambian los factores simultáneamente se producen resultados distintos de los que se obtienen con ajustes de factores individuales.
En [Montgomery91] se encuentran los detalles sobre diseño de experimentos.

Para este caso, donde se busca evaluar la calidad de las heurísticas para el problema de conglomerado geográfico, una de las pruebas que es importante realizar sobre los resultados obtenidos es evaluar la calidad de estos usando una metodología que permita identificar el efecto de los parámetros de control sobre la función de costo, modelar la dependencia de esta función respecto a los parámetros y finalmente poder hacer un estudio sobre la influencia de los parámetros en la búsqueda por encontrar mínimos ya sea locales o generales de la función.
Para ello, se ha considerado un diseño experimental box behnken y de superficies de respuestas que ha permitido observar los efectos descritos en el párrafo anterior.
Este tipo de experimento es una prueba o serie de pruebas en las cuales se inducen cambios deliberados en algunas variables de entrada del sistema mientras otras se mantienen fijas, de tal forma que es posible identificar las fuentes de los cambios en las variables de salida [Montgomery91].

Hasta donde se ha estudiado, no se disponen de metodologías claras para establecer cómo encontrar parámetros correctos de una heurística determinando a su vez la calidad de las soluciones generadas. Sin embargo en [Barr95] se presenta un buen método para diseñar experimentos en heurísticas.

En este capítulo se muestran los resultados que aseguran encontrar un balance entre los parámetros que estén cercanos al óptimo.
La metodología para este propósito se ha publicado en [Bernábe09], por tanto, aquí solo se dan a conocer los principales resultados.

Análisis de los resultados experimentales para PG con RS

En la siguiente tabla se indican los niveles de los parámetros para RS que el diseño box behnken BB arrojó para que se ejecutaran las pruebas correspondientes.

parámetro	nivel alto	nivel central	nivel bajo
T_I	5500	5250	5000
T_F	0.1	0.055	0.01
A	0.99	0.985	0.98
L(t)	5	4	3
Grupos	24	18	12

Tabla 2. Niveles utilizados en el experimento box behnken para RS

Con estos niveles de diseño box behnken se corrieron 46 pruebas experimentales que se muestran en la tabla 3. La nomenclatura utilizada en la tabla es: C (corrida), Ti (temperatura Inicial), Tf (temperatura final), α (alfa), Lt (L(t)), G (grupos), FC (función objetivo).

C	Ti	Tf	α	Lt	G	**FC**
1	5000	0.01	0.985	4	18	**13.5279**
2	5500	0.01	0.985	4	18	**13.5878**
3	5000	0.1	0.985	4	18	**14.0342**
4	5500	0.1	0.985	4	18	**14.1222**
5	5250	0.055	0.98	3	18	**13.9166**
6	5250	0.055	0.99	3	18	**14.1286**
7	5250	0.055	0.98	5	18	**13.2346**
8	5250	0.055	0.99	5	18	**13.8935**
9	5250	0.01	0.985	4	12	**16.2161**
10	5250	0.1	0.985	4	12	**16.553**
11	5250	0.01	0.985	4	24	**11.5392**
12	5250	0.1	0.985	4	24	**12.0292**
13	5000	0.055	0.98	4	18	**16.3016**
14	5500	0.055	0.98	4	18	**14.1095**
15	5000	0.055	0.99	4	18	**13.9159**
16	5500	0.055	0.99	4	18	**13.9545**
17	5250	0.055	0.985	3	12	**15.6353**
18	5250	0.055	0.985	5	12	**16.0845**
19	5250	0.055	0.985	3	24	**12.3314**
20	5250	0.055	0.985	5	24	**11.6377**
21	5250	0.01	0.98	4	18	**13.5198**
22	5250	0.1	0.98	4	18	**14.304**
23	5250	0.01	0.99	4	18	**13.3445**
24	5250	0.1	0.99	4	18	**13.7725**
25	5000	0.055	0.985	3	18	**13.6595**
26	5500	0.055	0.985	3	18	**13.5348**
27	5000	0.055	0.985	5	18	**14.0258**
28	5500	0.055	0.985	5	18	**13.0667**
29	5250	0.055	0.98	4	12	**16.8496**
30	5250	0.055	0.99	4	12	**17.1076**
31	5250	0.055	0.98	4	24	**12.2146**
32	5250	0.055	0.99	4	24	**11.7276**
33	5000	0.055	0.985	4	12	**16.6959**
34	5500	0.055	0.985	4	12	**16.7826**
35	5000	0.055	0.985	4	24	**11.8841**
36	5500	0.055	0.985	4	24	**11.2403**
37	5250	0.01	0.985	3	18	**13.5575**
38	5250	0.1	0.985	3	18	**13.2107**
39	5250	0.01	0.985	5	18	**13.6996**
40	5250	0.1	0.985	5	18	**14.7604**
41	5250	0.055	0.985	4	18	**13.9274**
42	5250	0.055	0.985	4	18	**13.8217**
43	5250	0.055	0.985	4	18	**13.5833**
44	5250	0.055	0.985	4	18	**13.9886**
45	5250	0.055	0.985	4	18	**13.6392**
46	5250	0.055	0.985	4	18	**12.9008**

Tabla 3. Corridas determinadas por el experimento box behnken para RS

El experimento ha proporcionado información relevante para ajustar valores para lograr un “óptimo” próximo al real. Se ha considerado solo el caso para 24 grupos, donde el óptimo fue obtenido con pam con un valor de 9.279. El valor más cercano es el que corresponde a la corrida 36 (11.2403). Sin embargo, de los resultados obtenidos a lo largo del experimento estadístico, puede verse en la siguiente figura 3 que existe una zona de experimentación para lograr un sub-óptimo de 10.3597. Estos datos han sido analizados en [Bernábe09].

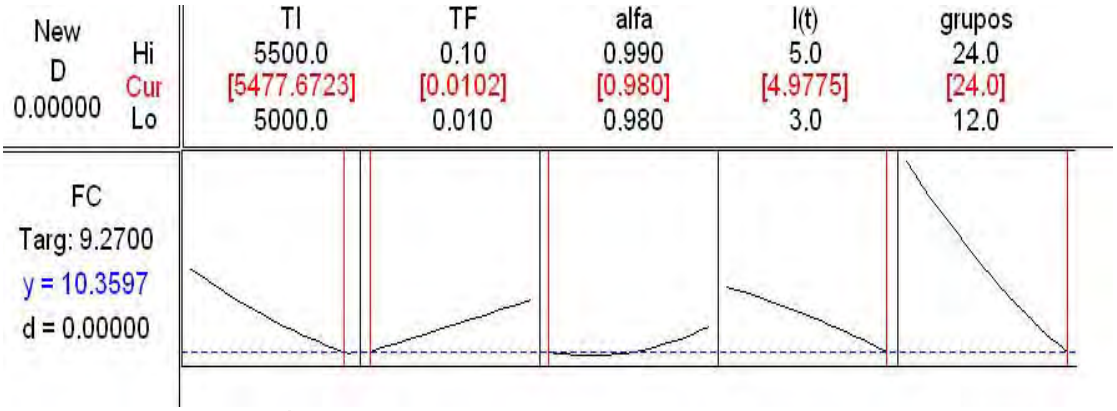


Figura 3. Representación del modelo de segundo orden para 24 grupos

Se concluye que los parámetros de la heurística de recocido simulado implementada para el problema de PG, son sensibles para las siguientes condiciones:

a) En términos generales entre mayor sea el número de grupos se alcanza un mejor óptimo.
b) La temperatura inicial entre más cerca este de 5000 unidades independientemente del número de grupos, el costo de la función objetivo converge al óptimo.
c) Al fijar la temperatura final y alfa, la variación de los otros parámetros restantes debe estar bajo control. Si se estaciona .01 para temperatura final y .98 en alfa es posible lograr un buen mínimo en la función de costo.
d) Cuando se han considerado la variación de todos los parámetros, un valor de alfa de .980 debe ser exigido mientras que la temperatura final debe ser pequeña con un valor de .01.

El experimento se inició a partir de los resultados analizados en corridas empíricas donde se determinó que 24 es un buen número de grupos para distintos análisis [Bernábe04].

4.4 Solución para particionamiento geográfico con búsqueda por entorno variable

Retomando la modelación para particionamiento geográfico PG, en esta subsección se muestra la solución a PG con búsqueda por entorno variable VNS. Se consideran las propiedades de VNS vistas en el marco teórico de esta tesis.

Recuérdese que VNS está basada en la observación de que los mínimos locales tienden a agruparse en una o varias zonas del espacio de búsqueda. Por lo tanto, una vez que se consigue un óptimo local, resulta beneficioso aprovechar la información que este contiene. Por ejemplo, que distintas variables puedan tener valores iguales o cercanos al óptimo global.
VNS propone explorar vecindarios cercanos, y progresivamente más lejanos de la solución actual en la búsqueda de mejores soluciones.
Asociada con cada iteración de VNS, existe una solución actual *sa* y un vecindario de orden *k*. En cada iteración se ejecutan dos pasos: primero, se genera una solución vecina de *sa*, $sp \in Nk(sa)$; segundo, se aplica un procedimiento de búsqueda local sobre *sp*, dando lugar a una nueva solución *sol*.
Si *sol* mejora la solución actual *sa*, entonces la búsqueda se reinicia desde *sol* utilizando $k = 1$. En caso contrario, se hace $k = k + 1$ y se repite el proceso desde *sa*. El algoritmo se detiene después de cierto número de veces que se haya realizado la exploración de la secuencia completa *N*1*, N*2*, …, Nkmax* [Pelta00].

Este esquema básico descrito en el párrafo anterior, se propone como el algoritmo siguiente:

Procedimiento de búsqueda por entorno variable:
Begin
/* Nk; k = 1; … ; kmax, estructuras de vecindarios
/* sa: solución actual
/* sp: solución vecina de sa

```
/* sol: solucion localmente optima
Repeat Until ( finalizacion ) Do
k=1;
    Repeat Until ( k = kmax ) Do
    /* generar vecino sp del k-esimo vecindario de sa (sp ∈ Nk(sa))
    sp = ObtenerVecino(sa,Nk);
    sol = BusquedaLocal(sp);
        If (sol es mejor que sa) Then
       sa = sol;
        Else
        k=k+1;
        endIf
    endDo
endDo
End.
```

El siguiente pseudocódigo incluye al procedimiento anterior que se inserta a su vez en la modelación de PG.
Se comentan líneas importantes para resaltar el funcionamiento tanto de los ciclos como de la obtención de la función de costo.

4.4.1 Algoritmo de particionamiento geográfico con búsqueda por entorno variable
Sea n el número de objetos a clasificar
UGij denota que el objeto i está asignado al centroide j
i=1,..,n; j=1,…,k
Sea M={M_1,M_2,,M_k} una solución de K centroides
MaxVNS ´máximo *número de iteraciones que recorre toda la estructura de vecindades*
MaxBL *número de iteraciones de BL para cada vecindad*

1. Inicio
'Obtiene Solución inicial
Generar aleatoriamente centroides iniciales M = {M_1, M2, … , Mk}
'Cualquier ageb puede ser centroide obtenido de forma aleatoria

coste_act ← Coste (M)*
'Esta asignación representa ya una solución inicial, es una solución propuesta generada por el paso anterior. En los siguientes pasos se genera otra solución para determinar que tan buena es con respecto a la actual y decidir si se cambia o no la solución actual de acuerdo a VNS

```
cont=1
Mientras  cont < MaxVNS hacer
        KVecindad=1
        `Variable de control para saber en cual vecindad se está buscando una solución
        Mientras KVecindad <> n
                C← Genera una solución aleatoria de  Kvecindad
                'Obtiene un vecino de Kvecindad
                Sol_vecindad← Busqueda Local (C,)
                If Coste (Sol_vecindad)  < coste_act
                        M← Sol_vecindad
                Else KVecindad=Kvecindad+1
```
Pasa a la siguiente vecindad, solo cambia de vecindad si la sol actual (M)l no se mejora, esto es, si es mejor que M se queda en la vecindad (teóricamente existe en esa vecindad una buena solución).
```
        End
        Cont=cont+1
```

End
Regresa M ´*solución de costo menor encontrada*

2. Función coste (Sol)
'determina que tan buena es la solución SOL, es decir, que tanto minimiza el objetivo

i ← 1
'inicializa primer objeto
cost ← 0
Mientras i ≤ n
'para cada objeto en Ug hacer
si Ug_i no es centroide entonces
dmin ← dist(Sol_1 , Ug_i)
'representa la distancia del objeto$_i$ hacia Sol_1 (primer centroide donde Sol representa al conjunto de todos los centroides. Se calcula la distancia cada objeto a su centroide más cercano, (distancia de un objeto $_i$ que no es centroide hacia Sol_1 que es el centroide 1)
j ← 2
'paso al segundo centroide
Mientras j ≤ k
Si dist (Sol_j , Ug_i) < dmin
'se calcula la distancia del objeto i hacia Sol_j (otro centroide)
dmin ← dist (Sol_j , Ug_i)
Fin si
j ← j + 1
'paso al siguiente centroide
Fin Mientras
cost ← cost + dmin
fin si
i ←i + 1
Fin Mientras
Coste(Sol) ← cost

En búsqueda local se trata de mejorar la solución "actual" buscando soluciones alrededor de esta, pero si encuentra una mejor antes de que se cumpla el número máximo de iteraciones, se regresa esa solución mejorada, de otra forma es necesario hacer cumplir el número máximo de iteraciones ya que si no se mejora la solución y no existe un parámetro de control, búsqueda local se podría ciclar.

La siguiente figura muestra el resultado de un ejemplo de la implementación del algoritmo de PG con VNS. Se ha incluido en el diagrama de flujo de datos el mapa que refleja la agrupación.

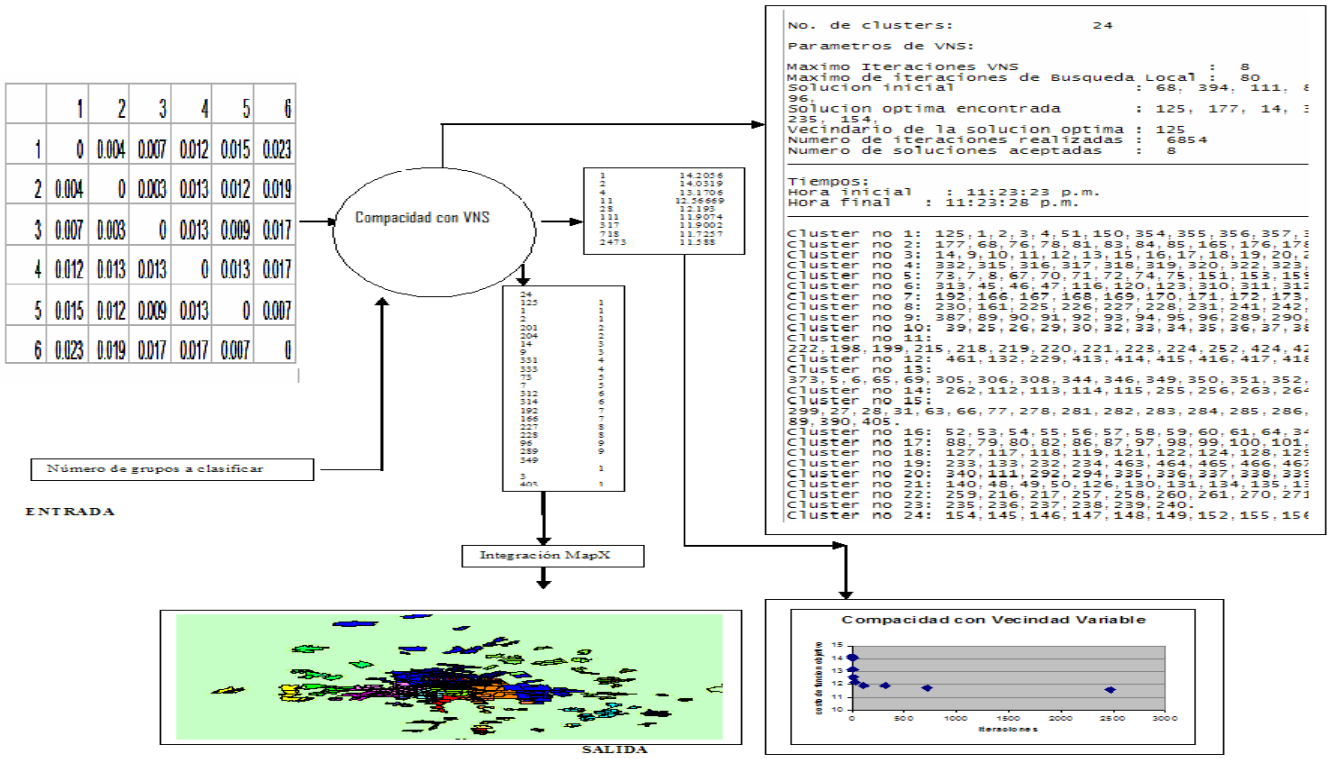


Figura 4. Diagrama de flujo de datos en PG con VNS

La siguiente tabla 4 muestra un conjunto pequeño de pruebas que demuestran la funcionalidad de la implementación y por otro lado se observa que para este problema de PG, VNS aproxima muy bien la función de costo [Bernábe09a], sin embargo este supuesto se comprueba más adelante.

La notación para las pruebas es: fc (función de costo), bl (búsqueda local), ev (estructura de vecindad) y g (grupos).

Iteración	fc	bl	ev	g
12402	9.798796	20	3	44
342	8.284898	15	2	42
6098	7.453398	70	32	30
118663	10.6685	537	641	24
296043	10.8771	537	641	24
400725	10.7212	916	1092	24
9486	10.9851	537	641	24
274396	10.8937	1072.99	641	24
591340	10.85999	537	1278.81	24
179993	11.21711	1.01	641	24
23805	10.7856	537	641	24
55618	11.0746	916	190	24

Tabla 4. Pruebas experimentales para PG con VNS

4.4.2 Diseño de un experimento box behnken para particionamiento geográfico con búsqueda por entorno variable

En esta sección se muestra una serie de resultados dados por un experimento estadístico aplicado en las soluciones generadas por VNS y RS para el problema de particionamiento geográfico.

Pruebas experimentales aleatorias para VNS

Algunos de los resultados obtenidos por VNS se han usado para elegir los valores de los niveles de los parámetros y definir así la región de experimentación de box behnken BB.
Estos niveles pueden verse en la siguiente tabla 5. La nomenclatura utilizada es:
ev estructura de vecindad, bl búsqueda local, fc función de costo y g grupos.

parámetros	nivel alto	nivel central	nivel bajo
bl	848	530	212
ev	640	400	160
g	24	18	12

Tabla 5. Niveles usados durante la experimentación

El resultado de BB implica desarrollar una serie de 15 pruebas aleatorias tal como se muestra en la tabla 6.

corridas	g	bl	ev	fc
15	18	530	400	12.6586
2	24	212	400	10.9011
4	24	848	400	10.8866
1	12	212	400	15.4535
5	12	530	160	15.3206
7	12	530	640	15.3221
14	18	530	400	12.5667
6	24	530	160	11.0177
13	18	530	400	12.5597
10	18	848	160	12.4411
11	18	212	640	12.6957
8	24	530	640	10.8371
3	12	848	400	15.1598
12	18	848	640	12.4726
9	18	212	160	12.8541

Tabla 6. Corridas experimentales para BB en el problema de PG con VNS

En la corrida 8, se ha observado que para 24 grupos con bl=530 y ev=640 se obtiene una función de costo de 10.8371. Entre las pruebas realizadas, esta corrida 8 es la que más se aproxima al óptimo de 9.27 y requiere de tiempo de cómputo de al menos 13 minutos.

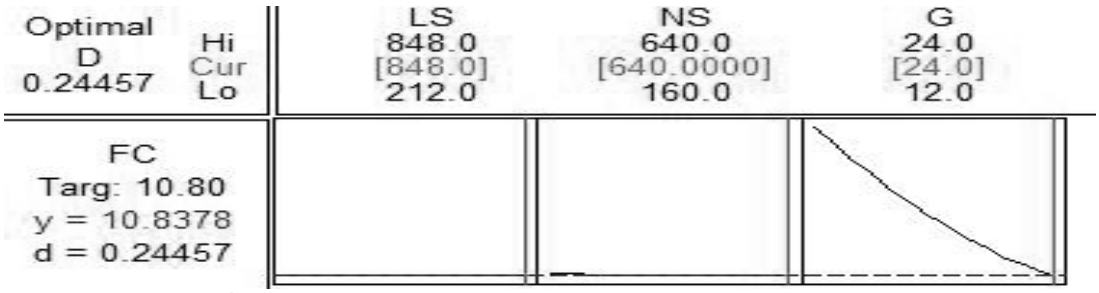


Figura 5. Regresión para VNS con un modelo de segundo orden (LS=bl y NS=ev)

La reflexión de este experimento deja algunos puntos que deben atenderse.

A través de todo el desarrollo experimental para VNS se puede afirmar que el mejor valor de la función de costo se logra para ev=640 y bl debe estar entre 848 y 530. Por otra parte, al comparar empíricamente este resultado con el obtenido en el experimento para RS, VNS logra mejor valor de la función de costo fc con respecto a la calidad de las soluciones pero no en tiempo.

Además, cuando se utilizó la ecuación de predicción se pudo estimar que es posible lograr una función de costo de 10.3597 para RS si se fijan los parámetros de Ti=5477.6723, Tf=0.102, alfa=.980 y l(t)= 4.9775, pero en la práctica esto no se ha comprobado.

En terceras pruebas no sistematizadas, el desempeño de VNS fue mejor en cuanto a calidad, lo que provoca que este análisis se extienda para comparar las soluciones con las 2 heurísticas [Bernábe09c].
En la siguiente sección se ha buscado un método para comparar la calidad de las soluciones generadas por las dos heurísticas.

4.5 Modelo experimental para comparar la calidad de soluciones generadas por búsqueda por entorno variable y recocido simulado

De los resultados obtenidos en las secciones anteriores, donde el experimento se centro en evaluar la calidad para VNS y RS considerando solo como factor común a 24 grupos, se obtuvieron resultados que proporcionan información para ampliar el estudio.
En consecuencia, se asume hipotéticamente que hasta el momento, VNS obtiene mejores soluciones que RS en cuanto a calidad.

Prueba T-student para búsqueda por entorno variable VNS y recocido simulado RS

Para comprobar que VNS proporciona mejor calidad en las soluciones que RS, el primer paso consiste en plantear una prueba de hipótesis T-student de dos colas o simplemente T. De este modo, dados los resultados anteriores, para esta prueba T, se han considerado las pruebas 8 de VNS y 36 de RS de la secciones antes vistas.
Para obtener una muestra representativa en esta prueba T, se replicaron 10 corridas aleatorias tanto para VNS como para RS.
En este escenario, se propone como hipótesis nula Ho que la calidad de las soluciones para VNS y RS es la misma. Considerando 13 grados de libertad con un error marginal de media μ y varianza σ^2. La prueba de dos colas indicó que el valor del estadístico de prueba es 37.7. Este resultado implica que Ho debe rechazarse.
Se concluye entonces que se acepta la hipótesis alternativa H_1 (la calidad de las soluciones de VNS y RS son diferentes). De este modo, la prueba se reemplaza por una de cola derecha con una hipótesis nula Ho: la calidad de las soluciones para RS es mejor que VNS. Se rechaza nuevamente Ho dado el estadístico de prueba de 31.5.
Por último, se asegura que existe evidencia significativa para afirmar que la calidad de las soluciones generadas por VNS es mejor que las obtenidas por RS. Estos resultados se han reportado en [Bernábe09c].

Es importante subrayar que hasta este momento, el factor grupos g es el único que es común en los experimentos aplicados a las heurísticas.
La manera en que ambas heurísticas puedan participar en un experimento imparcial y equitativo es considerar al tiempo como factor común.

Modelación de los tiempos de respuesta para recocido simulado y búsqueda por entorno variable

Hasta ahora, se han descrito los experimentos para evaluar a VNS y SA por separado. Pero el objetivo es comparar la calidad de las funciones de costo de ambas heurísticas “juntas” y objetivamente. Para poder hacer esta comparación, es necesario realizar dos experimentos apareados que permitan hacer una **modelación de los tiempos de cálculo** de cada una de las heurísticas.

Una vez realizadas diversas y aleatorias pruebas preliminares para examinar cómo se podría proponer un nuevo modelo experimental donde participen las 2 heurísticas justamente considerando al tiempo como factor comparable, se han identificado los niveles de los parámetros para ambas.

La idea básica para incluir al tiempo en el modelo del diseño experimental es realizar una serie de pruebas sistemáticas que proporcionen información de los parámetros para que tanto VNS como RS se ejecuten en el mismo intervalo de tiempo. La función de costo para este propósito ahora es el tiempo T (t).

Diseños experimentales compuesto central para VNS y box behnken para RS
Las siguientes tablas muestran los niveles usados para el diseño tanto para VNS y RS con las respectivas pruebas correspondientes al diseño.

Cabe señalar que se asocian 2 parámetros en el nuevo diseño experimental para VNS, por tanto, un box behnken no se ha podido adaptar y se ha elegido un diseño compuesto central [Montgomery91].

parámetros	nivel alto	nivel bajo
LS	916	158
NS	1092	190

Tabla 7. Niveles en el diseño compuesto central

Los niveles de la tabla anterior indican que se deben realizar para VNS un conjunto de 13 pruebas, las cuales se muestran en la siguiente tabla 8. Se considera como objetivo o función de costo al parámetro tiempo T, t.

corridas	Ev	bl	t
13	641	537	314
12	641	537	314
4	1092	916	509
9	641	537	312
8	641	1073	313
6	1279	537	624
7	641	1.01	130
10	641	537	318
3	190	916	90
11	641	537	287
1	190	158	90
2	1092	158	1050
5	3.19	537	3

Tabla 8. Pruebas VNS de acuerdo al diseño de experimentos

El conjunto de pruebas que se desarrollaron para VNS debido al diseño de la tabla 7 está organizado en la tabla 8.

Para el diseño asociado a RS se definió un box behnken como puede verse a continuación:

parámetros	nivel alto	nivel bajo
T_i	50000	500
T_f	0.1	0.01
A	0.99	0.98
L(t)	1000	5

Tabla 9. Niveles en el diseño box behnken para RS

corridas	TI	TF	α	L	T
5	25250	0.055	0.98	5	2
2	50000	0.01	0.985	502.5	174
10	50000	0.055	0.98	502.5	119
14	25250	0.1	0.985	5	3
13	25250	0.01	0.985	5	1
16	25250	0.1	0.985	1000	285
17	500	0.055	0.985	5	2
27	25250	0.055	0.985	502.5	154
22	25250	0.1	0.98	502.5	129
6	25250	0.055	0.99	5	2
7	25250	0.055	0.98	1000	219
4	50000	0.1	0.985	502.5	150
12	50000	0.055	0.99	502.5	239
19	500	0.055	0.985	1000	209
23	25250	0.01	0.99	502.5	262
15	25250	0.01	0.985	1000	354
21	25250	0.01	0.98	502.5	126
3	500	0.1	0.985	502.5	96
25	25250	0.055	0.985	502.5	156
8	25250	0.055	0.99	1000	444
1	500	0.01	0.985	502.5	125
11	500	0.055	0.99	502.5	158
18	50000	0.055	0.985	5	2
24	25250	0.1	0.99	502.5	217
20	50000	0.055	0.985	1000	313
9	500	0.055	0.98	502.5	76
26	25250	0.055	0.985	502.5	152

Tabla 10. Pruebas para SA de acuerdo al diseño de experimentos

Las tablas anteriores muestran una serie de pruebas que responden a los diseños construidos para cada una de las heurísticas. Con estos experimentos desarrollados se lograron los siguientes modelos de regresión para estimar los tiempos de cálculo de cada una. Primero se muestra el modelo para RS.

Modelo de regresión para SA

Termino	Coeficiente
Constante	154
TI	27.583
TF	-13.5
ALPHA	54.25
L	151
TI*TI	-23.583
TF*TF	8.792
ALPHA*ALPHA	17.667
L*L	-1.958
TI*TF	1.25
TI*ALPHA	9.5
TI*L	26
TF*ALPHA	-12
TF*L	-17.75
ALPHA*L	56.25

La figura 6 muestra un ejemplo que bajo el modelo descrito, fue posible obtener los valores de los parámetros para RS con 24 grupos. De este modo, se consiguió el costo de la función para un tiempo de 100 segundos.

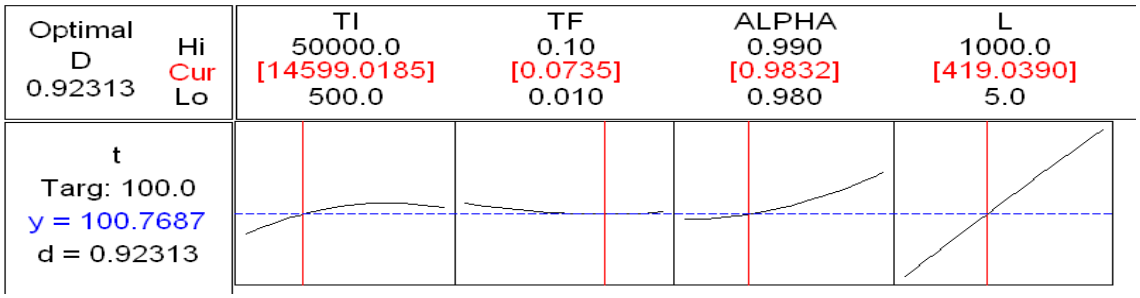


Figura 6. Ejemplo para que RS logre un valor de costo en 100 segundos

Por otro lado, la gráfica de contornos que se muestra en la siguiente figura indica las zonas de experimentación alrededor de t= 120, 140, 160, y 180 segundos. Por ejemplo, cuando alfa se mantiene en .985 y L(t) en 502.5, para lograr que SA corra en 160.044 segundos, se debe fijar a Ti en 39377.3 y Tf en .06309.

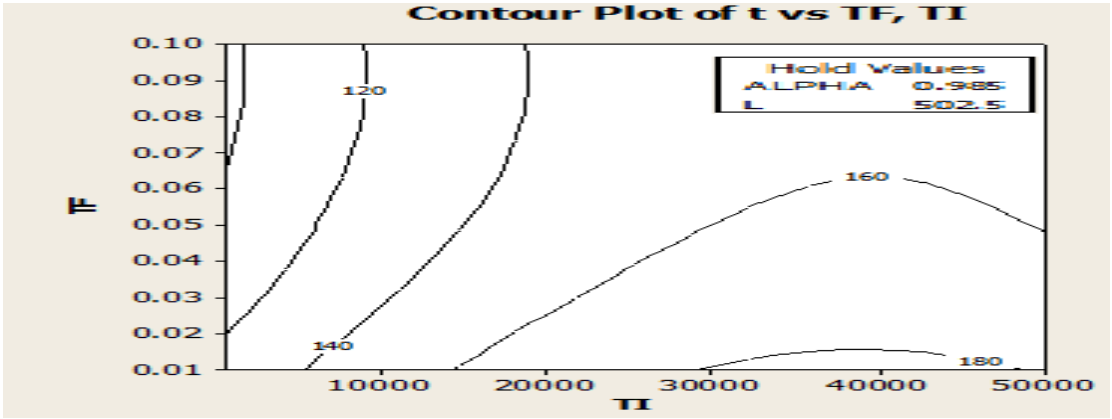


Figura 7. Ejemplo de un contorno para SA

De la misma manera que se realizó el experimento para RS y reduciendo algunos resultados para VNS, se ha obtenido el modelo siguiente:

Modelo de regresión para VNS

Termino	Coef
Constant	309
EV	282.153
BL	-35.275
EV*EV	44.063
BL*BL	-1.938
EV*BL	-135.25

En la figura 8 se observa que aplicando el modelo anterior para VNS se obtienen los valores de los parámetros con 24 grupos obteniendo así, un valor para la función de costo tiempo en 100 segundos.

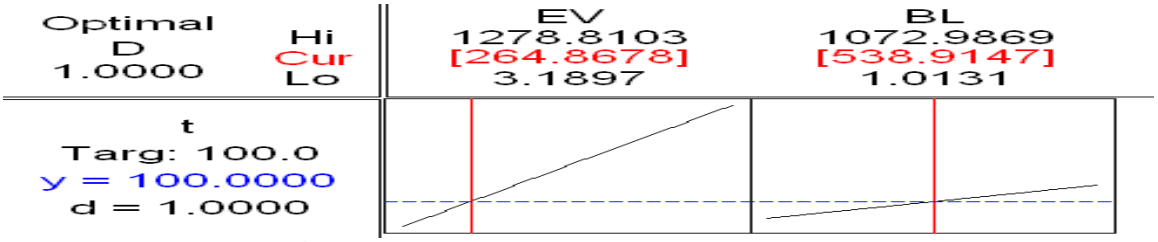


Figura 8. Valores de parámetros en VNS para t=100

El gráfico de contornos de la figura 9 revela las zonas de experimentación para tiempos alrededor de 200, 400, 600 y 800. Si se fija ev=1159.3 y bl=1051.71, VNS debe obtener una solución en 400.187 segundos

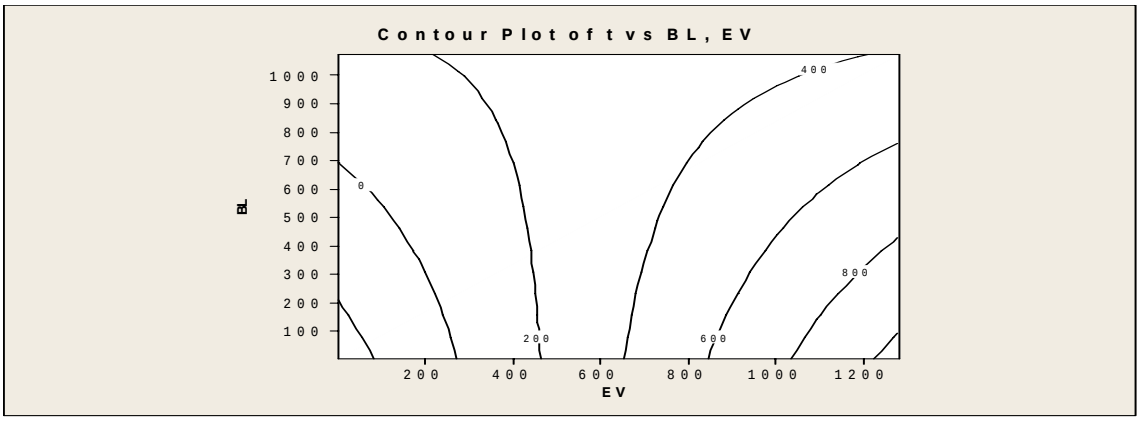


Figura 9. Ejemplo de Contorno para VNS

Con los modelos y figuras anteriores, fue posible desarrollar las pruebas de comparación para los siguientes tiempos de cálculo en ambas heurísticas. 100, 150, 200, 250, 300, 350 y 400 segundos.

Finalmente, se han calculado los niveles de los parámetros que permiten llevar a cabo esta prueba. La siguiente tabla resume los resultados de las funciones de costo T para cada uno de los tiempos de cálculo señalados.

t (s)	FC VNS	FC SA
100	11.14	11.4678
150	10.78029	11.4848
200	10.55189	11.0438
250	10.5982	11.18271
300	10.8914	11.14
350	10.5982	11.0181
400	10.99091	10.92021

Tabla 11. RS y VNS: tiempo y función de costo

De esta tabla se concluye que la función de costo para la heurística VNS es de mejor calidad que RS puesto que en la gran mayoría de los casos tiene un valor menor.

En el gráfico siguiente se puede observar el resultado obtenido. Se distingue claramente como la calidad de las soluciones de VNS supera a RS (SA) en todas las pruebas excepto en la última de 400 segundos.

Grafico 1. Comparación de la calidad de soluciones entre VNS y RS

4.6 Conclusión de resultados

A lo largo de este capítulo se ha podido comprobar que VNS para el problema de particionamiento geográfico para pruebas hechas a 24 grupos logra mejores resultados que RS.

La metodología utilizada permite comprobar el supuesto inicial sobre la calidad de las heurísticas, cuando en un experimento por separado para VNS y RS, se identificó que VNS daba mejores resultados que SA.
Con este dato, fue posible realizar un juego de hipótesis estadístico que resaltaba una vez más el resultado y finalmente desarrollar un modelo estadístico para estimar el tiempo de cálculo entre VNS y RS. De este modo, ha sido posible predecir el tiempo en que ambas heurísticas actuaran en un mismo intervalo, de forma equitativa observando así el valor de su función de costo.

Por último, fijándose en las cualidades de VNS, se elige a VNS como método heurístico de aproximación para el problema de agregación en zonificación óptima ZO, donde se optimizan los objetivos de compacidad geométrica y homogeneidad para variables.

Las figuras siguientes reflejan los mejores resultados en VNS y SA

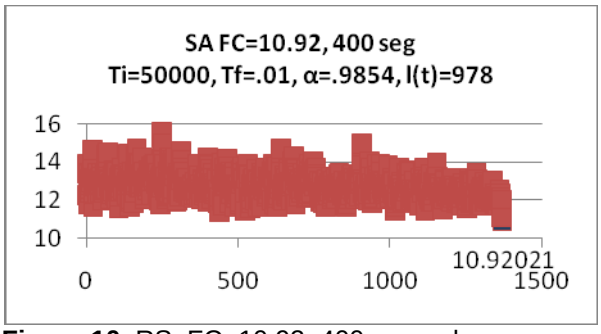


Figura 10. RS: FC=10.92, 400 segundos y **Ti=50000, Tf=.01, α=.9854, l(t)=978**

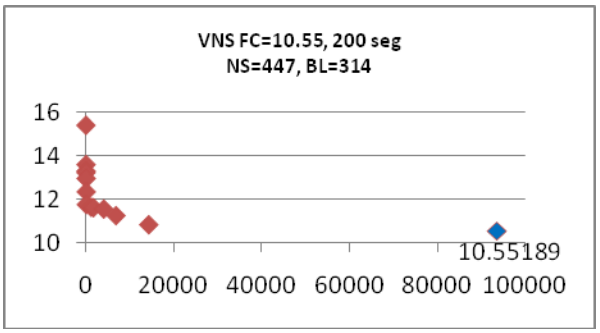


Figura 11. VNS: FC=10.55, 200 segundos y **NS=447, BL=314**

Capítulo 5: Zonificación óptima multiobjetivo

Introducción

En este capítulo se presenta una propuesta multiobjetivo para resolver el problema de zonificación óptima ZO tal y como se ha definido en la introducción y capítulo 1 de esta tesis.

El método para resolver zonificación óptima consiste en la agrupación de agebs que minimiza simultáneamente las funciones de compacidad para la ubicación geográfica y homogeneidad en las 144 variables censales disponibles [Web3].
La optimización se realiza con búsqueda por entorno variable VNS mientras va obteniendo para cada solución generada, un par de valores "mínimo" (compacidad, homogeneidad) denotado como (c, h).
Dichas soluciones generadas por la heurística, son evaluadas paralelamente bajo el orden Pareto NoComparable y dominancia Pareto DP, lo cual, genera un conjunto de pares (c^*, h^*) no comparables y no dominados que proporcionan un conjunto llamado soluciones minimales-ZO que forman el frente de Pareto.

Esto se puede expresar informalmente como sigue:
Sea A= minimales {(f 1, f 2)| f 1 es el valor de compacidad y f 2 es el valor para homogeneidad}.
El conjunto A esta sujeto a las restricciones de particionamiento y homogeneidad que se detallan en secciones más adelante.
Entonces si (A, orden Pareto NoComparable) es un conjunto parcialmente ordenado, entonces (f 1, f 2)=conjunto de minimales de A.

El orden Pareto NoComparable es la negación de Dominancia de Pareto DP:

Dominancia de Pareto DP. Se dice que un vector $\vec{u} = (u_1,\ldots, u_k)$ domina a $\vec{v} = (v_{1,\ldots,}v_k)$ (denotado mediante $\vec{u} \preccurlyeq \vec{v}$), si y solo si, es $\vec{u}$ es parcialmente menor que $\vec{v}$. Es decir, si

$$u_i \leq v_i \; \forall \; i \in \{1,\ldots,k\}$$

y

$$\exists \; i \in \{1,\ldots,k\}: u_i < v_i$$

Las secciones posteriores están encargadas de detallar los aspectos tanto teóricos como de modelación alrededor del problema multiobjetivo que nos ocupa.

5.1 Problemas multiobjetivo: teoría básica

En muchas disciplinas, resolver un problema o tomar una decisión significa hallar la mejor solución común a un conjunto de relaciones.

No es fácil resolver un problema multiobjetivo, sin embargo, estos problemas pueden ser más claros identificando las relaciones entre las características del problema, sus restricciones y los objetivos principales que se desean mejorar en conjunto. Esto es, para este tipo de problemas, es posible tener una expresión como una función matemática.
Al referirse a la mejora en conjunto, se dice que se debe *optimizar* a todas las funciones de manera simultánea, definiéndose entonces un problema del tipo descrito a continuación [Lara03]:

Definición 1. *Un problema multiobjetivo (MOP) puede definirse en el caso de minimización (y análogamente para el caso de maximización) como:*

Minimizar $f(x)$

Dado que $f : F \subseteq R^n \longrightarrow R^q$, $q \geq 2$

y se evalúa en

A= {a $\in F: g_i(a) <= 0, i \in \{1, \ldots ,\}\} \neq \emptyset$

El conjunto A es llamado la región factible y se dice que el problema se encuentra sujeto a las restricciones g_i donde $g_{i:}\ R^n \rightarrow R$ son funciones cualesquiera.
Otra notación equivalente puede verse como:

Notación 2 de MOP
La optimización simultánea de más de un objetivo, significa optimizar una función de la forma
$f : S \rightarrow T$, donde $S \subseteq R^n$ y $T \subseteq \mathbb{R}^k$

Es claro ahora que no existe un elemento de *S* que produzca un óptimo de forma simultánea para cada uno de los *k* objetivos que componen f.
El problema de optimización multiobjetivo, puede establecerse de la siguiente forma:
Encontrar un vector $x^* = [X_1^*, X_2^*, \ldots, X_n^*]^n$ que satisfaga las *m* restricciones:
$g_i \geq 0$ donde i = 1,2, …, m (1)
y las *p* restricciones:
$h_i(x) = 0$ donde i = 1,2, …, p (2)

y optimice la función vectorial
$f(x) = [f_1(x), f_2(x), \ldots, f_k(x)]^n$

donde $x=[x_1, x_2, \ldots, x_n]^T$ es el vector de variables de decisión
En otras palabras, se desea determinar la solución particular $X_1^*, X_2^*, \ldots, X_n^*$, del conjunto *S* formado por todos los valores que satisfacen (1) y (2), que dé lugar a los valores óptimos para todas las funciones objetivo [Lara03, Cruz04].

Algunas veces las funciones objetivo del problema deben escalarse o cambiar de signo, para presentar el problema de la forma anterior, ya que originalmente los problemas multiobjetivo vienen dados de tres formas: en el que todas las funciones se maximizan, en donde todas las funciones se minimizan y en donde algunas funciones se maximizan mientras que otras se minimizan.

A partir de la definición 1, el sentido común nos lleva a pensar en la optimización multiobjetivo como la búsqueda de un vector que representa el conjunto de variables de decisión y el cual optimiza (maximiza o minimiza) en conjunto a las funciones objetivo. Sin embargo en este caso es de principal importancia notar que dichas funciones pueden estar en conflicto unas con otras.

Una manera de ilustrar esta situación es posible con el ejemplo que se muestran a continuación:

Ejemplo 1.
Supóngase que se tienen que optimizar las funciones $f1(x) = x+2$; $f2(x) = x^2$ y $f3(x) = 4-x$ con $0 < x < 100$. Mientras que $f1(x) = x + 2$ y $f2(x) = x^2$ crecen se tiene que $f3(x) = 4-x$ decrece, por lo que, la mejor solución será aquella que nos lleve a un mejor compromiso entre las tres funciones.

Además de la posibilidad de estar en conflicto unas con otras, las funciones en un problema multiobjetivo pueden representar cantidades que no son equivalentes entre sí. Por ejemplo, si el problema consiste en optimizar $f = (f1; f2, \ldots, fn)$ que son las ganancias totales de una empresa, pero *f*1 representa una ganancia actual en pesos, *f*2 representa una ganancia en recursos humanos, *f*3 representa un ahorro de energía, etc.

Debido a las situaciones anteriores, en optimización multiobjetivo no se utiliza el término óptimo para referirse a la mejor solución del problema, sino el término “conjunto de soluciones eficientes”; no necesariamente se debe encontrar una única solución mejoradora de todas las demás. Puesto que se está buscando un vector compromiso, la solución de un problema multiobjetivo consiste en dar un conjunto de vectores eficientes con los cuales se pueda tomar una decisión [Lara03].

Dicho de otra forma técnicamente: sea P un problema de optimización múlti-objetivo. Se dice entonces que una solución S1 es Pareto-óptima (definida más adelante), cuando no existe otra solución S2 tal que mejore en un objetivo sin empeorar en otro.

La siguiente figura muestra un ejemplo de 3 funciones en conflicto.

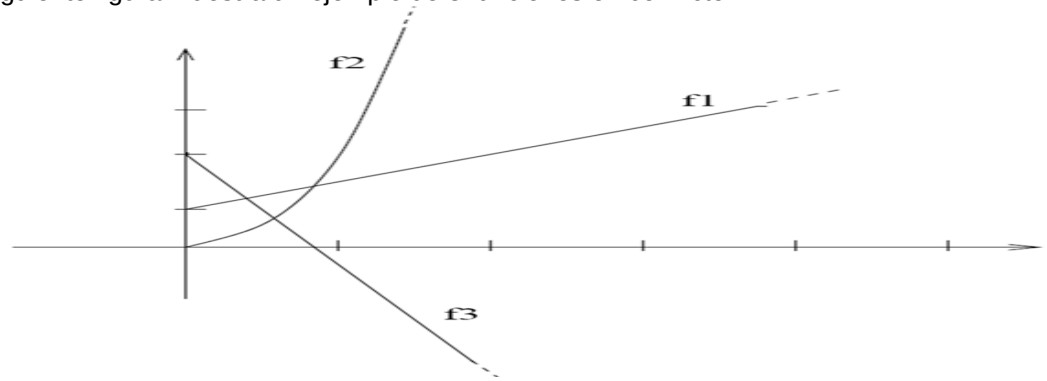


Figura 1. Funciones en conflicto, *f*3 decrece mientras *f*1 y *f*2 crecen.

Es por eso por lo que interesa disponer, no de una solución, sino de varias, para que en el momento de tomar decisiones se consideren todas las soluciones conocidas como "óptimas posibles". [Lara03].

Para ello se dispone actualmente de algunas metodologías: 1) métodos basados en el concepto de eficiencia de Pareto, 2) métodos basados en la combinación de objetivos y 3) métodos basados en la asignación de prioridades. El método 1) es la metodología utilizada para desarrollar esta tesis.

5.1.1 Conjuntos parcialmente ordenados

Resolver un problema multiobjetivo es contar con un conjunto de vectores no dominados. Esto conduce al concepto de conjunto parcialmente ordenado, el cual es estudiado por las matemáticas dentro de las áreas de álgebra y combinatoria [Gierz03].

Para comprender mejor este concepto se enuncian las siguientes definiciones y sus principales propiedades.

Definición 2. *Sea A un conjunto. Al subconjunto* $R \subseteq A \times A$ *se le llama* ***relación binaria****. A los elementos* $(x; y) \in R$ *se les denota como xRy y se dice que x está relacionado con y. La relación R se llamará:*

a) reflexiva *si ocurre que* xRx *para toda* $x \in A$
c) simétrica *si* xRy *entonces* yRx *para todos* $x, y \in A$
d) antisimétrica *si* xRy & yRx *entonces* $x = y$ *con* $x, y \in A$
e) asimétrica *si* xRy *entonces* yRx *para todos* $x, y \in A$
f) transitiva *si* xRy & yRz *entonces* xRz *para todos* $x, y, z \in A$

Con este concepto tenemos la siguiente definición.

Definición 3. *Sea A un conjunto dado no vacío y R una relación binaria definida en A, se dice que R es una* ***relación de orden*** *si cumple*

1. *Reflexividad: Todo elemento de A esta relacionado consigo mismo, es decir* $\forall x \in A$, xRx
2. *Antisimetría: Si dos elementos de A se relacionan entre si, entonces ellos son iguales, es decir,* $\forall x,y \in A$, xRy, $yRx \Rightarrow x=y$
3. *Transitividad:* $\forall x,y,z \in A$, xRy, $yRz \Rightarrow$, xRz

Una relación de orden R sobre un conjunto A puede denotarse con el par ordenado $(A , \leq)$.

Es frecuente encontrar en la literatura la siguiente definición:

Definición 4. *Una relación binaria que cumple con ser reflexiva, antisimétrica y transitiva es llamada una* ***relación de orden parcial*** *y se representa comúnmente mediante el símbolo (≼).*

Definición 5. Un **orden parcial** es una relación binaria R sobre un conjunto A que es reflexiva, antisimétrica, y transitiva, es decir, para cualesquiera a, b, y z en A se tiene que:
aRa (reflexividad).
Si aRb y bRa, entonces a = b (antisimetría).
Si aRb y bRz, entonces aRz (transitividad).

Un conjunto con un orden parcial se denomina **conjunto parcialmente ordenado o poset**.

A veces se usa la expresión conjunto ordenado para uno parcialmente ordenado, siempre que quede claro que no se hará referencia a otras clases de orden. En particular, a un conjunto totalmente ordenado también se lo llama ordenado, en especial en campos donde éstos son más comunes que los parcialmente ordenados. Por ejemplo, en R^n puede definirse el orden parcial ≤ de manera natural como $x \leq y \; x,y \in R^n \Leftrightarrow x_i \leq y_i$ con i=1, …, n

Concluyendo lo anterior, se dice entonces que un **conjunto parcialmente ordenado** CPO es un conjunto equipado con una relación binaria de orden parcial.

Esta definición formaliza el concepto intuitivo de orden, secuencia, o arreglo de los elementos del conjunto. Un orden tal no necesariamente debe ser total, es decir, no se necesita que se puedan comparar unos con otros todos los elementos del conjunto; esto sin embargo puede ocurrir en algunos casos (en otras palabras, el orden total es un caso particular del orden parcial). La propiedad de totalidad de esta relación se puede también describir como que todo par de elementos son comparables bajo la relación.

Formalmente, un **orden total, orden lineal, orden simple, o simplemente orden** en un conjunto A es una relación binaria sobre A que es antisimétrica, transitiva, y total; esto es, si se denota una tal relación por ≤, lo siguiente vale para cualesquiera a, b, y c en A:

Si a ≤ b y b ≤ a, entonces a = b (antisimetría).
Si a ≤ b y b ≤ c, entonces a ≤ c (transitividad).
a ≤ b o b ≤ a (totalidad o completitud).

Un conjunto dotado de un orden total se denomina **conjunto totalmente ordenado**, **linealmente ordenado, simplemente ordenado, o cadena.** Nótese que la condición de totalidad implica reflexividad, esto es, a ≤ a para todo a ∈ X; por lo tanto, un orden total es también un orden parcial, esto es, una relación binaria reflexiva, antisimétrica, y transitiva. Un orden total, entonces, puede también definirse como un orden parcial que sea "total", i.e. que cumpla con la condición de totalidad.

Los conjuntos totalmente ordenados forman una subcategoría completa de la categoría de conjuntos parcialmente ordenados, siendo los morfismos funciones que respetan el orden, es decir, funciones f tales que si a ≤ b entonces f (a) ≤ f (b). Entonces una función biyectiva entre dos conjuntos totalmente ordenados que respete los dos órdenes es un isomorfismo en esta categoría.

El orden parcial anteriormente definido se denomina no estricto o reflexivo; de tal forma que un **orden parcial estricto o irreflexivo** es una relación binaria que es irreflexiva y transitiva, y por lo tanto asimétrica. De forma equivalente, asimétrica (y por lo tanto irreflexiva) y transitiva. Es decir, para cualesquiera a, b, y c en X se tiene que:

¬(aRa) (irreflexividad).
Si aRb, entonces ¬(bRa) (asimetría).

Si aRb y bRc, entonces aRc (transitividad).

Si R es un orden parcial no estricto, entonces S = R − {(a, a) | a ∈ X} es el orden parcial estricto correspondiente. Análogamente, todo orden parcial estricto S tiene uno no estricto correspondiente, a saber, S ∪ {(a, a) | a ∈ P}, o la "clausura reflexiva" de R.

En R^n existen vectores que no pueden compararse bajo el orden parcial ≤, por ejemplo (3.5, 7.8, 4, 6) y (4, 7, 4, 7). Este ejemplo ilustra que no cualquier conjunto cumple con la propiedad de que todos sus elementos puedan compararse (como en R^n).
Los enunciados anteriores se formalizan también para describir más adelante el concepto de Pareto.

Definición 6. Dados A un conjunto y ($\preccurlyeq$) una relación de orden parcial sobre él, llamamos a la pareja (A, $\preccurlyeq$) un **conjunto parcialmente ordenado** también referido como **Poset.**

Definición 7. Dado (A, $\preccurlyeq$) un Poset, el subconjunto X⊆A se dice ser un **orden total** o cadena con respecto a ($\preccurlyeq$), si y solo si, se cumple que $x \prec y$ ó $y \prec x$ para todos x,y ∈ X. En este caso se dice que (X, $\preccurlyeq$) es un conjunto totalmente ordenado.

Si se tiene que $x \not\preccurlyeq y$ para todos (x,y) ∈ X, se dice que X es una anticadena. Las cadenas pueden ser finitas o infinitas y también pueden estacionarse en uno de sus extremos como:

$$x_0 \prec x_1 \prec x_2 \prec \ldots \prec x_{i-1} \prec x_i \prec x_{i+1} \prec \ldots$$

0

$$\ldots \prec x_{i+1} \prec x_i \prec x_{i-1} \prec \ldots \prec x_2 \prec x_1 \prec x_0$$

El siguiente enunciado es de vital importancia para generar soluciones no dominadas:

A partir de un orden parcial puede definirse la relación de dominación ($\prec$) de la manera siguiente:
$x \prec y \iff x \preccurlyeq y \wedge x \neq y$

Cuando ocurre que $x \not\preccurlyeq y \wedge y \not\preccurlyeq x$ se dice que son no comparables, lo cual es denotado por $x \parallel y$.

Por otro lado es importante también citar el lema de Zorn, también llamado de Kuratowski-Zorn [Campbell78], es una proposición de la teoría de conjuntos que afirma lo siguiente:

Todo conjunto parcialmente ordenado no vacío en el que toda cadena (subconjunto totalmente ordenado) tiene una cota superior, contiene al menos un elemento maximal.

Un **elemento maximal** de un conjunto parcialmente ordenado P, es un elemento de P que no es menor que ningún otro. El término elemento minimal se define de manera dual.

Definición 8. Sea (P, ≤) un conjunto parcialmente ordenado; m ∈ P es un elemento **maximal** de P si el único x ∈ P tal que m ≤ x es x = m. La definición de elemento minimal se obtiene reemplazando ≤ por ≥.

A primera vista parecería que m debería ser un elemento máximo, lo que no es siempre cierto ya que la definición de elemento maximal es algo más débil. De hecho, pueden existir elementos maximales sin que haya un máximo. La razón es que, en general, ≤ es sólo un orden parcial en P; si m es un maximal y p ∈ P, cabe la posibilidad de que ni p ≤ m ni m ≤ p, con lo que m no sería máximo. Esto permite, además, que haya más de un elemento maximal en un conjunto.

Sin embargo, si m ∈ P es maximal y P tiene un máximo, se cumplirá que máx(P) ≤ m; por definición de máximo se debe tener m ≤ máx(P) y por lo tanto m = máx(P); en otras palabras, un máximo, si existe, es también el único maximal.

No es difícil ver que si ≤ es un orden total en P, las nociones de máximo y maximal coinciden. Sean $m \in P$ un elemento maximal, y $p \in P$ arbitrario; por la condición de orden total, o bien $p \leq m$ o bien $m \leq p$; en el segundo caso se tendría $p = m$ por definición de maximal, con lo cual $p \leq m$, y por consiguiente, m = máx(P).

No siempre existen los elementos maximales, ni siquiera en el caso en que P esté totalmente ordenado [Gierz03].

Formalizaciones de minimal y poset
Lo descrito anteriormente se resume y formaliza a continuación [Lara03]:

Definición 9. Un elemento $x^* \in A$ es llamado elemento **minimal del poset** $(A, \preccurlyeq)$ si no existe un elemento $x \in A$ tal que $x \prec x^*$. El conjunto de todos los elementos minimales se denota $M(A, \preccurlyeq)$.

Definición 10: El conjunto $M(A, \preccurlyeq)$ de todos los elementos minimales de $(A, \preccurlyeq)$ se dice completo si para cada $x \in A$ existe al menos un elemento $x^* \in M(A, \preccurlyeq)$ tal que $x^* \preccurlyeq x$.

Lema 1. Dado un poset $(A, \preccurlyeq)$ tal que A es finito, se cumple que su conjunto de elementos minimales $M(A, \preccurlyeq)$ es completo.

Demostración.
Sea $x_0 \in A$. Si $x_0 \in M(A, \preccurlyeq)$, el lema es cierto por la propiedad de reflexividad $\preccurlyeq$. Supongamos entonces que $x_0 \notin M(A, \preccurlyeq)$. Esto implica que existe un elemento $x_1 \in A$ tal que $x_1 \preccurlyeq x$. Dado que A es finito, existe un elemento x_n para alguna cadena $x_n \preccurlyeq \ldots \preccurlyeq x_1 \preccurlyeq x_0$ con $n \leq |A|$ en el cual la cadena transitiva se detiene, esto quiere decir que ningún elemento de A domina a x_n y por lo tanto se sigue directamente que $x_n \in M(A, \preccurlyeq)$.

Lema 2. Sea $M(A, \preccurlyeq) \neq \emptyset$ el conjunto de elemento minimales de algún conjunto parcialmente ordenado $(A, \preccurlyeq)$. Sea también $x \in A$ el conjunto $G(x) := \{y \in A: y \preccurlyeq x\}$:

Entonces cumple las siguientes implicaciones:
a) $x \in M(A, \preccurlyeq) \Leftrightarrow G(x) \setminus \{x\} = \emptyset$
b) Si $M(A, \preccurlyeq)$ es completo y $x \notin M(A, \preccurlyeq)$ entonces $(G(x) \setminus \{x\}) \cap M(A, \preccurlyeq) \neq \emptyset$

Demostración.
a) Por definición $x \in M(A, \preccurlyeq) \Leftrightarrow (\nexists\, y \in A; y \prec x) \Leftrightarrow (\nexists\, y \in A; y \preccurlyeq x \wedge y \neq x)$
Son implicaciones ciertas. La última expresión puede escribirse como $(\nexists\, y \in G(x): y \neq x$. Esto entonces equivale a decir que $G(x) \setminus \{x\} = \emptyset$.

b) Sea $\in A$ tal que $x \notin \in M(A, \preccurlyeq)$. Considerando la parte a) es un hecho que $G(x) \setminus \{x\} = \emptyset$. Dado que $M(A, \preccurlyeq)$ es completo podemos decir que existe un elemento $x^* \in M(A, \preccurlyeq)$ tal que $x^* \neq x \wedge x^* \prec x$. Entonces empleando la definición G(.) concluimos que necesariamente $x^* \in G(x) \setminus \{x\}$, por tanto $(G(x) \setminus \{x\}) \cap M(A, \preccurlyeq) \neq \emptyset$.

Para concluir esta sub-sección, se cita a minimales y maximales como componentes de un diagrama de Hasse.

Elementos maximales y minimales en un diagrama de Hasse
Sea (A≤) un CPO. Sean $m,n \in A$. Entonces
1) n es un elemento maximal en A si y solo si $(\forall x)(n \leq x \Longrightarrow n=x)$
2) m es un elemento minimal en A si y solo si $(\forall x)(x \leq m \Longrightarrow x=m)$

Intuitivamente, un elemento n de un CPO (A≤) es maximal de A si no hay un elemento en A que sea mayor estrictamente que n.

Análogamente, un elemento m de A es un minimal de A si no hay un elemento de A que sea menor estrictamente que m [Gonzalez].

Búsqueda por entorno variable en el particionamiento multiobjetivo "identifica implícitamente ramas" del diagrama de Hasse donde cada punto no dominado de cada rama es el conjunto maxima (que contiene a todos los maximales) y es el frente de Pareto y es el conjunto de puntos no comparables minimales-ZO.

5.1.3 Esquema de dominación

En el ámbito de la optimización multiobjetivo se tiene que decidir un cierto esquema de mejoría de una solución sobre otra, es decir cuáles soluciones se elegirán para ser más aptas; a esta relación de mejoría de un individuo sobre otro es a lo que llamaremos *Esquema de Dominación.*

La definición de un Esquema de Dominación se basa principalmente en que, la solución de un problema multiobjetivo no es única y por lo tanto, el tomador de decisiones debe elegir de entre una gama de posibles soluciones que no se pueden mejorar entre sí, es decir que no se dominan, y elegir la que mejor convenga a sus necesidades [Lara03].

Para el problema de ZO, si el algoritmo VNS basa su mecanismo de selección en dicho esquema de dominación entonces dará como resultado un conjunto de soluciones razonablemente buenas para elegir de entre ellas.

5.1.3.1 Óptimo y frente de Pareto

Una opción común a usarse como relación de dominación es la conocida dominación de Pareto definida a continuación:

Definición 11. Dado el problema multiobjetivo

Minimizar: $f(x)$

donde

$f : F \subseteq R^n \rightarrow R^q \; q \geq 2$

con A ⊆ F la región factible. Decimos que un vector $x^* \in A$ es no dominado o un óptimo Pareto si no existe un vector $x \in A$ tal que

$f(x) < f(x^*)$

Así, la respuesta al problema de hallar las mejores soluciones (las soluciones no dominadas, como quiera que se defina la dominación dentro de la técnica) en un problema multiobjetivo, es a lo que llamaremos el *conjunto solución* del problema, y al grupo de valores de la función objetivo con dominio restringido a los vectores del conjunto solución (es decir, los vectores no dominados), es lo que conoceremos como frente de Pareto.

El concepto de frente de Pareto fue introducido en el siglo XIX por el economista italiano Vilfredo Pareto y se ha utilizado en el estudio de problemas multiobjetivo con técnicas clásicas de Investigación de Operaciones. En general y sobre todo para problemas de la vida real no es sencillo hallar el frente de Pareto de manera analítica (y en la mayoría de los casos es imposible).

Un concepto íntimamente relacionado con el Frente de Pareto es el de óptimo de Pareto. Tanto el óptimo de Pareto como el Frente de Pareto son el marco sobre el que se trabaja dentro de la toma de decisiones multicriterio.

Definición 12. El conjunto $E(A; f)$ de soluciones de Pareto eficientes (también conocido como **conjunto de óptimos de Pareto**) se define de la manera siguiente:

$E(A, f) := \{ a \in A: \nexists\, b \in A \text{ que cumpla } f(b) < f(a)\}$

Es decir, el conjunto de todos los vectores no dominados bajo el esquema de Pareto.
En resumen, el conjunto de óptimos de Pareto es el espacio solución del problema y el Frente de Pareto es su imagen con respecto a la función $f : F \subseteq R^n \rightarrow R^q$ $q \geq 2$ a optimizar.

El óptimo Pareto para un problema multiobjetivo dado es un conjunto parcialmente ordenado, visto formalmente. En los problemas multiobjetivo se busca los elementos minimales del espacio de solución R^n visto como un poset, en la relación ≤ definida de manera natural.

Definición 13. Dado un problema MOP, llamaremos *frente de Pareto verdadero* y denotaremos como F^* al frente de Pareto teórico que cada problema posee en particular.

Llamaremos *frente de Pareto conocido* y denotaremos como F_k al frente de Pareto que se desprende del algoritmo VNS al término de la última solución en el último vecindario *k.*

(En la mayoría de los problemas el frente de Pareto teórico no se conoce. En algunos puntos de la literatura éste se denota como *Ftrue*.

Hasta este momento es posible resumir que:

La condición DP indica un conjunto parcialmente ordenado bajo el orden Pareto, donde el "ultimo" elemento de este orden es un maximal o minimal (según se maximice o minimice la función multiobjetivo). Se asume entonces que, este orden no es más que una rama de un diagrama de Hasse y de esta rama destaca solo un punto no dominado.

Intuitivamente, para el caso de dos objetivos, y dadas las condiciones del concepto de orden Pareto, es posible crear un método para hallar el conjunto de pares (c_i, h_i) no dominados, siempre que sea posible "atinar" a todas las ramas de Hasse de un mismo conjunto de pares de soluciones, para una corrida.

En la siguiente sección se describe el método para resolver zonificación óptima ZO que considera a la dominancia Pareto DP, el cual, permitió proponer la primera solución a ZO. Se explica además, las consecuencias de este método que dan lugar a proponer variantes en DP para obtener el conjunto no dominado que se busca.

5.2 Enfoque multiobjetivo para el problema de zonificación óptima
El problema de zonificación óptima requiere para su solución de la optimización de dos funciones objetivo: compacidad y homogeneidad. En este escenario, su proceso se inició con la agrupación compacta bajo el nombre de conglomerado geográfico, considerando en la modelación e implementación las propiedades de particionamiento clásico y de diseño territorial.

Para obtener "simultáneamente" los dos valores de las funciones compacidad-homogeneidad, la idea principal es contar primero con un conjunto de soluciones compactas (la óptima y locales), y ejecutar el proceso de homogeneidad a dichas soluciones justo en el momento que se van generando. Esto se logra por las características de VNS.

Considerando que en problemas multiobjetivo las funciones objetivo tienen el mismo dominio [Rios08], para obtener el valor de homogeneidad asociado, y evaluar simultáneamente las dos funciones bajo dominancia Pareto DP, es necesario revisar *todo* el conjunto de soluciones generadas con VNS independientemente de que sean óptimos locales o la solución final "óptima". En este punto, el cálculo de homogeneidad se realiza a *todo* el conjunto de soluciones compactas generadas mientras VNS no termine.
Finalmente, como la heurística lo permita y para cada iteración, esto proporciona tantos pares de soluciones integrados por valores de compacidad y homogeneidad (denotado como (c_i, h_i)).

Esto significa que a la vez que se van generando soluciones (c_i, h_i) con VNS y usando el orden Pareto DP, se identifican las soluciones que cumplan este orden, hallándose así, un subconjunto de soluciones (c_k, h_k) $k \leq i$, con un par "especial" ($c_{mínimo,}$ $h_{mínimo}$) no dominado.
Este par se dice que es un elemento distinguido, llamado minimal en la frontera de Pareto.

Hasta este punto, se ha descrito la manera para encontrar soluciones (c_i, h_i) para el problema de ZO usando el orden Pareto implícito en DP. Sin embargo, claramente se observó que ($c_{mínimo,}$ $h_{mínimo}$) es el único punto no dominado en la frontera de Pareto, lo que implica atender este problema dotando a DP de extensiones que aseguren proporcionar todas las soluciones no dominadas en el frente de Pareto.

En la literatura multiobjetivo, particularmente en la página de Carlos Coello [Web2], se encuentra disponible un algoritmo que, aplicado adecuadamente, permite obtener de un conjunto de soluciones, un subconjunto de vectores no dominados llamados elementos maximal o minimal. (En este caso, dicho algoritmo identifica los pares (c^*, h^*) no dominados).

Se usó la implementación de este algoritmo conocido como nodom [Web2], a todas las soluciones generadas por el método que considera dominancia Pareto para ZO descrito anteriormente.
El resultado para diversas pruebas, ha sido un conjunto de pares (c^*, h^*) no dominado. Dentro de este conjunto, existe un par idéntico al orden Pareto: ($c_{mínimo,}$ $h_{mínimo}$) que encontró el método para ZO.

El algoritmo disponible en [Web2], es conocido como "Algorithms to identified nonodom solutions in a multi-dimensional set". El documento original es propuesto por Kung [Kung75].
Después de un análisis completo a dicho artículo, se concluyó que es posible obtener el conjunto de soluciones no dominadas si se obtiene la maxima, Este conjunto maxima no es más que el conjunto de puntos minimales o maximales de un diagrama de Hasse.

La siguiente afirmación es cierta para ZO:
Bajo el "orden Pareto NoComparable" se puede obtener un conjunto de soluciones no dominadas para ZO si se obtienen soluciones no comparables y que además satisfagan la dominancia Pareto. De esta manera, dichas soluciones son también aquellas que se observan como "puntos" minimales de un diagrama de Hasse. El orden no comparable consiste en la negación proposicional lógica de DP.

Para comprender el enunciado anterior, se introdujo un primer enfoque para encontrar soluciones no dominadas adicionales al orden Pareto. Este enfoque se basa en la inclusión de cotas a las soluciones dadas por el orden Pareto. La "inclusión de cotas", no es más que la suma de un número real a cada uno de los pares de valores (c, h), en la relación de orden implícita en la dominancia Pareto DP, lo que denominamos informalmente como "holgura en el orden Pareto".

Si se dispone de una "holgura" en el orden Pareto, es decir, si es posible relajar las soluciones con una agregación de un valor adecuado a los pares (c, h) identificados por este orden, se asume que es probable desplazarse a otras ramas del diagrama de Hasse obteniendo así, dependiendo de la aleatoriedad de la heurística, el conjunto máxima total ZO o una parte de este conjunto y soluciones adicionales dominadas–relajadas denominadas "pre-minimales ZO".

El segundo enfoque encuentra el conjunto de pares de soluciones "no comparables" en ZO. Para obtener los pares no comparables se ha negado lógicamente el orden Pareto.
Esta propuesta con la incorporación adecuada de la dominancia Pareto, ha permitido encontrar todas las soluciones minimales en ZO (del conjunto de soluciones generadas para una corrida).
Llamamos para efectos prácticos, al orden Pareto NoComparable, como aquel orden que encuentra soluciones que sean no comparables y obedezcan también la dominancia Pareto DP.

Para mostrar que se están encontrando con este segundo enfoque todos los minimales para una determinada prueba, se ha utilizado nodom.

Para ZO si se denota a f1 como la función que obtiene el mínimo valor de compacidad y f2 es la función que obtiene el mínimo para homogeneidad, para obtener un conjunto de soluciones óptimas, la respuesta al problema de ZO, en términos generales, se reduce a:
Minimizar F=(f 1, f 2)
Sujeto a
los grupos no son vacíos,
la intersección entre grupos es vacía,
la unión de todos los elementos de los grupos son todos los agebs,
y las variables de la población pueden o no estar acotadas por valores permitidos en las agebs.

Minimizar (f 1, f 2) consiste en obtener un conjunto de pares (c_i, h_i) no dominados y no comparables entre todos los pares (c_i, h_i). Sea A= minimales {(f 1, f 2)}, el conjunto A esta sujeto a las restricciones de particionamiento y homogeneidad.

Si (A, orden Pareto NoComparable) es un conjunto parcialmente ordenado, entonces el minimal (f 1, f 2) es igual al conjunto de minimales de A.

En las siguientes secciones se detallan los aspectos de modelación.

5.2.1 Descripción del problema de zonificación óptima
De acuerdo al propósito de ZO, la meta es obtener una partición de datos espaciales agebs cuya composición está dada por 2 componentes: coordenadas geográficas en el plano R^2 y un vector de 144 características descriptivas de carácter censal.

La primera componente permite que se obtenga una matriz de distancias para el proceso de cálculo de la compacidad geométrica, una de las funciones de objetivo a minimizar. La segunda componente, el vector de descripción, se utiliza para optimizar la segunda función objetivo y consiste en minimizar alguna de las variables censales con el fin de calcular homogeneidad o equilibrio para dicha variable (previamente se hace una selección de las variables de interés) [Bernábe06].

En este sentido, la partición que se busca para ZO es un conjunto de clases donde sus elementos estén muy cercanos geográficamente y balanceados para alguna variable censal, de tal forma que se está frente a un problema bi-objetivo de particionamiento geográfico, donde se exige tanto el uso de métodos heurísticos para lograr soluciones aproximadas como de herramientas multiobjetivo, para encontrar un conjunto de pares de soluciones no dominadas que forman el frente de Pareto.

5.2.2 Discretización y representación
Se tiene un espacio físico de búsqueda para la agrupación geográfica. Las unidades geográficas son finitas y se denominan agebs (áreas geoestadisticas básicas); cada elemento es representado por su ubicación espacial y por un arreglo de variables descriptivas. El problema es discreto, combinatorio, entero-mixto y la agregación se realiza bajo las propiedades de particionamiento.

Se resuelve la agrupación de agebs de tal forma que, las agebs que componen los grupos estén entre ellas muy cercanas geográficamente, donde se requiere el uso de una función de costo que minimice distancias entre estas (ver capítulo 4). Por el lado de la homogeneidad, ésta se optimiza buscando un equilibrio entre una variable censal de interés.

La estrategia de agrupamiento consiste en elegir aleatoriamente agebs como centroides que identifican el número de grupos. Los agebs no centroides que tengan la distancia más corta hacia un determinado centroide-ageb, son los que conforman un grupo. Esta idea es la que se entiende como compacidad geométrica y se ha descrito en el capítulo 4.

Una vez formados los grupos bajo la minimización de distancias, se calcula la homogeneidad a la agrupación creada debido a que el dominio en este problema es una partición (en problemas multiobjetivo la función a optimizar tiene el mismo dominio para todos los objetivos [Rios08]). De este modo sobre una misma partición se optimiza la compacidad y homogeneidad.
Para aclarar esta situación, basta observar que en los problemas de optimización multiobjetivo se elige la mejor alternativa x de un conjunto X respecto a un objetivo amplio y poco preciso. Entonces se introduce una jerarquía de objetivos siendo los m de más bajo nivel lo suficientemente precisos para permitir medir los resultados de cada alternativa. Se tiene así, que elegir la mejor alternativa respecto a los m objetivos. Matemáticamente, tenemos un conjunto X subconjunto R^n espacio de alternativas y m aplicaciones f_i: X →R, i=1, … , m que representan los m objetivos [Rios08].

Antes de formalizar matemáticamente las funciones objetivo y su dominio, se indican los datos, las restricciones, los objetivos y la notación básica para la modelación de ZO:

Datos:
Los elementos que forman un grupo territorial GT son áreas geoestadísticas básicas agebs. Dos componentes describen a cada ageb: a) ubicación geográfica y b) vector de parámetros de características asociadas a la ageb (variables censales cuantificadas). Se asume que k es el número de grupos territoriales y n el número total de agebs $k \ll n$.

Restricciones:
1. Cada ageb debe pertenecer a un único grupo.
2. En un grupo el valor de cada parámetro en el valor de la variable censal.
3. Los grupos son disjuntos.
4. No existen grupos vacíos.
5. Las variables poblaciones pueden o no estar acotadas.
6. Pueden estar en la agrupación todas las variables o un subconjunto de ellas.
7. Los agebs asignados a cada grupo deben formar un grupo compacto.
8. Lo grupos deben estar balanceados con respecto a una meta de equilibrio para alguna característica medible.

Objetivos:
Las restricciones 7 y 8 se traducen en los objetivos:
Que los grupos sean lo más compactos posible.
Que los grupos sean homogéneos para una variable censal.

Primer objetivo: minimización de distancias
Este objetivo se ha resuelto y expuesto en el capítulo 4 con un algoritmo de agrupamiento.
Se elige un número k de grupos como centroides aleatorios c_t, t = 1,...k.
Enseguida se asignan los agebs a los centroides de la siguiente manera:
Para cada ageb i $\underset{t=1,\ldots,k}{\text{Min}} \{d(i, c_t)\}$ (cada ageb es asignado al centroide más cercano c_t).
Para cada valor de k se calcula la suma de las distancias de los agebs asignados a cada centroide y se escoge el mínimo y nit es el número de iteraciones. Esto puede expresarse como (1).

$$\underset{k=1,\ldots,nit}{\text{Min}} \left\{ \sum_{t=1}^{k} \sum_{i \in c_t} d(i, c_t) \right\} \qquad (1)$$

Sujeto a:

$Z_i \neq \varnothing$ para i = 1,…, k, $Z_i \cap Z_j = \varnothing$ para $i \neq j$

$\bigcup_{i=1}^{k} Z_i = UG$

UG = n (n es el número de objetos).

Segundo objetivo: Homogeneidad para variables censales
El segundo objetivo consiste en encontrar un equilibrio para una variable de interés donde participan algunas variables adecuadas con cotas o todas las variables con o sin cotas.

Para ilustrar esta situación, decimos que cuando se desea agrupar agebs de una zona metropolitana, donde estas se seleccionen previamente bajo condiciones en los valores, buscando un equilibrio para alguna variable censal, hablamos informalmente del problema de particionamiento para agebs, bajo criterios de homogeneidad.

Una forma de plantear como elegir variables de población, para que puedan ser usadas en un proceso de agrupación posterior y se mantenga un equilibro en una variable, puede verse como en la siguiente definición:

Definición 14. Selección para variables censales
Sea *P* un conjunto de m agrupaciones de agebs sobre una variable de la población:
(1) *P*={AG1, AG2, … , AGm} j=1, ... ,m, tales que, AGi ∩ AGj=Ø

Sea VC un conjunto de variables censales que describen a las agebs:
(2) VC={X1, X2, … , Xn} i=1, ... ,n. Entonces existe una función Xi: $P \rightarrow R^{+}$ i=1, … ,n

Una pregunta en AGj sobre Xi se propone como:
Xi(AGj)= Query de Xi en AGj, y se entiende como un Query para selección de variables.

Ejemplo, Se tienen 3 grupos compactos donde se requiere conocer el valor promedio de la variable censal Z167 (hogares con jefatura femenina). Sea

X1(AG1)=hogares con jefatura femenina en el grupo 1
X1(AG2)=hogares con jefatura femenina en el grupo 2
X1(AG3)=hogares con jefatura femenina en el grupo 3

Así si P={AG1, AG2, AG3}, P es subconjunto de *P,* y sea VC = X1=Z167.

Entonces en general, el problema asociado con variables censales/poblaciones, resuelve el valor de la variable Xi en el AGj, y se propone como una terna (*P,* VC, Query).

Y Query es una pregunta sobre las variables que describen a las agebs:
(3) Query(X1,X2, …,Xn).

Una vez contando con las variables que participan en la agrupación, para homogeneizar los grupos se propone informalmente calcular lo siguiente:

Obtener un promedio ideal para la variable de interés, esto sucede cuando todos los grupos tengan el mismo valor. Lo que no es común en la práctica, por lo que se propone calcular el promedio real a cada grupo y restar este valor al promedio ideal.
Al minimizar esta diferencia se puede obtener el costo de la función objetivo para homogeneidad.

En la siguiente sección se incorporan los dos objetivos que describen la modelación final de zonificación óptima.

5.2.3 Formulación multiobjetivo para zonificación óptima
Sea una ageb un dato espacial definido por sus componentes en el espacio y una descripción de 144 variables censales dada por un vector.

Ahora, con el fin de integrar los dos objetivos en un modelo entero-mixto, se tiene una variante en la notación. La nomenclatura básica para formular ZO es:

M= mapa territorial
T= territorio
DT= datos censales
MS= matriz de disimilitud
UGB= unidad geográfica básica (ageb)
GT= grupo territorial
m = número de GT
n= número de UGB tal que m<<n
i= índice de grupo territorial
j= índice de unidad geográfica básica
Ci= centroide del i-ésimo grupo territorial GTi

El modelo en cuestión es entero mixto y hace uso de las variables binarias para modelos de este tipo.

De acuerdo a la definición 14, se propone la siguiente formulación para la selección de las variables censales y sus cotas:

Sea A es el conjunto de atributos cuantificables, del cual será seleccionado un subconjunto de acuerdo al problema.

VA_{kj} es el valor del k-ésimo atributo contenido en la j-ésima UGB_j .

α_k , β_k parámetros de tolerancia para VA_k en cualquier UGB.
con $\alpha_k \leq VA \leq \beta_k$ (cotas para las variables).

$(VA_{ki}) = \sum_{j=1}^{n} VA_{kj} X_{ij}$ es el valor para el k-ésimo GT. Con X_{ij} como variable de decisión, dado que este modelo es entero mixto.

$\overline{VA_k} = i/m \sum_{j=1}^{n} VA_{kj}$ el valor meta para el k-ésimo atributo en cualquier UGB.

Por el lado de la minimización de distancias, se ha ajustado el objetivo propuesto en (1) y una sencilla variante en la notación:

$d_{ij} = d(C_{i,} UGB_j) X_{ij}$ la distancia de la j-ésima UGB en el i-ésimo GT, respecto a su centroide.

Entonces el cálculo de distancias entre agebs puede expresarse como:

$$Dij = \sum_{j=1}^{n} d(C_i, UGB) X_{ij} \qquad a)$$

Y el valor de homogeneidad para variables se expresa como:

$$H_{\text{variable-ageb}} = \sum_{i=1}^{m} \overline{(VA_k} - VA_{ki}) \qquad b)$$

Considerando lo anterior, una propuesta para el modelo para zonificación óptima es:
Minimizar y= f (x)= (f 1, f 2) c)

donde

f 1: es el costo de minimizar la distancia entre agebs de acuerdo a la ecuación a)

f 2: es el costo de minimizar la homogeneizar de una variable censal de las agebs.

Las funciones f1 y f2 como en c) están sujetas a:

$GT_i \neq \varnothing$ para i = 1,…, k (los grupos no son vacíos), con $GT_i \cap GT_j = \varnothing$ para $i \neq j$ (No existen agebs repetidos en distintos grupos).

$\bigcup_{i=1}^{k} GT_i = UGB$ (la unión de todos los grupos son todos los agebs),

$\sum_{i=1}^{m} X_{ij} = 1$ es la asignación de agebs,

$X_{ij} = 1$ si $UGB \in GT_i$ o

$X_{ij} = 0$, si $UGB \notin GT_i$ son las variables de decisión

$y = (y1, y2) \in Y \subset R^2$ es el vector objetivo

5.3 Estrategia para encontrar soluciones no-dominadas en zonificación óptima

Con el fin de encontrar soluciones óptimas Pareto en ZO, inicialmente se hizo uso del orden Pareto para las soluciones generadas por VNS, obteniéndose así un conjunto parcialmente ordenado de acuerdo a la dominancia de Pareto DP [Pareto96].
Sin embargo, lo que se pide es demostrar que los pares (c_i, h_i) contenidos en el conjunto de soluciones finales para una corrida, formaba un conjunto de soluciones no dominadas. Con ese fin y para efectos de "filtración", a dichas soluciones ZO, se ha aplicado nodom.

5.3.1 Primera etapa. Método para encontrar la maxima

Ndominated o nodom es una aplicación que desprende del artículo "On finding the maxima of a set of vectors" [Kung75].
Nodom es la implementación del algoritmo "Algorithms to identified nonodom solutions in a multi-dimensional set". El programa esta implementado en código C e identifica soluciones no dominadas de un conjunto de datos. El código y detalles para su uso están disponible en [Web2].

Se han implementado dos versiones para este algoritmo, la primera por Luis Vicente Santana Quintero and Antonio López Jaimes [Web2]. Este algoritmo tiene una complejidad O(n2) y actualmente es el más utilizado. La segunda versión se desarrollo por Kung, Luccio and Preparata y encuentra de un conjunto de soluciones, vectores no dominados (también llamados elementos maxima). Este algoritmo requiere O(n log n) comparaciones para k=2, y O(n ($\log^{k-2} n$)) comparaciones para k >= 3, donde n es el tamaño del conjunto de soluciones en la entrada y k es la dimensión del vector.

Se han usado las aplicaciones nodom, con el fin de verificar si las soluciones generadas por el método implementado en esta primera etapa para ZO es un conjunto no dominado.

Bajo el orden de DP no se genera todo el conjunto no dominado (ver sección 5.2). Esto se probó comparando el conjunto de soluciones generadas por ZO (donde se usó estrictamente DP), contra el vector resultante de nodom. La comparación proporcionó una clara intersección en una solución para ambas implementaciones.

Esto significa que nodom identifica del vector de soluciones que procesa, el conjunto maxima, donde sus elementos son los maximales y el método para ZO, en esta "primera versión", genera un conjunto parcialmente ordenado de orden Pareto DP que contiene solo un par (c_x, h_x) no dominado en dicho conjunto, el cual es un minimal.

Se concluye entonces que el conjunto parcialmente ordenado que genera el método que resuelve ZO bajo la dominancia Pareto DP, es solo una rama del diagrama de Hasse.
Lo anterior implica que en esta primera "versión" que considera el orden Pareto, las soluciones que se generan para ZO, solo satisfacen "parcialmente" las restricciones de este problema multiobjetivo.

Para ilustrar esta situación véase un sencillo ejemplo:

Ejemplo 2:
Se desea agrupar las agebs de la zona metropolitana del valle de Toluca en 5 particiones compactas y homogéneas tal que solo participen en la agrupación las siguientes variables cuyo valor este por encima del promedio:
Población masculina menor a 6 años (X001).
Población masculina entre 6 y 11 años (X003).
Población masculina entre 15 y 17 años (X007).

La homogeneidad se hará sobre X003. (Para comprender este proceso de restricciones para las variables véase el anexo a).

Las soluciones obtenidas por ZO y nodom son las siguientes:

Soluciones ZO		Soluciones nodom	
COM	HOM	COMP	HOM
61860	5308.8		
39683	**4251.6**	66069	1452.4
85076	3306.4	**19562**	**4251.6**
66069	1452.4	46862	3559.6
81851	4284.4		
71503	5509.6		
19562	**4251.6**		
58567	6533.6		
63424	5509.6		
72080	3227.6		
40281	4251.6		
55195	6612.8		
67249	2614.4		
65912	5351.6		
46862	3559.6		
83779	5736.8		

La primer columna es el costo de compacidad y la segunda el costo de homogeneidad.

Los pares de puntos señalados con rojo son un minimal para las dos aplicaciones (ZO y nodominated).
Se observa que para nodom los 3 pares de soluciones son el conjunto maxima. Para ZO los pares señalados en verde y rojo son nodos de una rama del diagrama de Hasse. Las soluciones en color negro se generaron por la heurística pero no fueron aceptadas por el orden Pareto.

En este escenario, resolver ZO consiste en encontrar un método que obtenga no solo todos los minimales que nodom identifica, se trata entonces de lograr aquellos pares (c, h) que pueden ser interesantes (muy cerca de los minimales) y que no se consideran por la relación de orden Pareto.

5.3.2 Segunda etapa. Soluciones relajadas para zonificación óptima

Obsérvese en el ejemplo anterior que en ZO, el primer punto (61860, 5308.8) es la solución inicial y la siguiente solución aceptada es el par (39683,4251.6) dado que es la primera solución después de la inicial que cumple la condición de dominancia DP.
Por otro lado, la solución (66069, 1452.4), puede ser de interés ya que mejora notablemente la solución para homogeneidad y empeora poco la compacidad.

La idea entonces es tener una “holgura en el orden Pareto”, es decir, “relajar” la condición de orden Pareto.
Para relajar el orden en DP, se ha sumado a las soluciones que se generan un épsilon ε a compacidad y otro ε a homogeneidad, lo suficientemente adecuado para que ZO acepte soluciones que a pesar de que un objetivo empeora el otro mejora o incluso que ambos empeoren pero no radicalmente.

Se ha puesto énfasis en el orden Pareto, es conveniente subrayar ahora que este orden se encuentra implícito en la siguiente definición que ya se ha citado:

Dominancia de Pareto DP. Se dice que un vector $\vec{u} = (u_1, \ldots, u_k)$ domina a $\vec{v} = (v_{1,\ldots,}v_k)$ (denotado mediante $\vec{u} \preccurlyeq \vec{v}$) si y solo si es $\vec{u}$ es parcialmente menor que $\vec{v}$. Es decir, si

$$u_i \leq v_i \; \forall \; i \in \{1,\ldots,k\}$$

y

$$\exists \; i \in \{1,\ldots,k\}: u_i < v_i$$

La variante para cumplir la relajación de soluciones consiste en la suma de un valor ε a las soluciones que cumplen el orden Pareto: ($u_i+\varepsilon \leq v_i+\varepsilon$).

Una vez que implementada esta variante en el orden Pareto, se obtuvieron los siguientes resultados para el mismo ejemplo anterior con ε=15000 para compacidad y ε=1000 para homogeneidad.

56341	2010.4
77551	2249.2
48778	3044.4
49403	3044.4
39570	2397.6

Los resultados de nodom son:

56341	2010.4
39570	2397.6

Para este ejemplo, no solo se han hallado todos los maximales, también soluciones cercanas a ellos que pueden ser importantes. Sin embargo, se realizaron más pruebas con la “relajación” en DP, de las cuales, se eligió la siguiente:

Ejemplo 3. En la prueba que se presenta a continuación, con las mismas características del ejemplo anterior y con la “holgura en el orden Pareto”, se observa que con un épsilon de 15000 para compacidad y otro de 1000 para homogeneidad, la aplicación de ZO no genera todos los maximales que nodom identificó.

Esto se debe a dos cosas: aleatoriedad de las soluciones o es necesario un épsilon mayor que debe calcularse de manera más precisa con otros métodos. Esta situación implica modificar el algoritmo para que independientemente de la aleatoriedad siempre se obtengan todos los minimales (ver figura 2).

93088	4791.6				
75392	3159.6	Aceptada Pareto		57105	2010.4
57105	2010.4	Aceptada Epsilon		24002	4251.6
86229	5920.8			41336	2170.4
67246	4187.6			20534	5476.8
75392	3159.6			47218	2094.4
50355	2094.4	Aceptada Epsilon		ndominated	
81668	2162.4				
66297	4342.4				
61579	4583.6				
42273	6558.8				
69485	4440.8				
48786	5476.8				
44808	2397.6	Aceptada Epsilon			
41336	2170.4	Aceptada Pareto			
77764	4564.4				
20534	5476.8				
24002	4251.6				
47218	2094.4	Aceptada Epsilon			
58903	5405.6				
ZO					

Figura 2. Prueba con soluciones relajadas en el orden Pareto

5.3.3 Tercera etapa. Soluciones no comparables para ZO: Pareto NoComparable

Entendido que los maximales son “puntos no dominados” pero además no comparables (para el problema que se resuelve en esta tesis), es necesario proponer un método para descubrir todos los pares de soluciones no comparables revisando de igual forma que estos sean no dominados con el orden Pareto.

Atendiendo esta situación, se ha analizado nuevamente el orden Pareto DP, lo que significa que la relación de orden implícita en Pareto debe ser reemplazada por una “relación de orden no comparable”.

Recordemos que la primera implementación, básicamente ha considerado con el orden Pareto que:

Dada una solución (a, b) la siguiente solución (a´, b´) es aceptada si: (*1)
a´ > a ∧ b´ = b ∨
b´ > b ∧ a´=a ∨
a´ > a ∧ b´ >b mejoran simultáneamente (mejor de los casos) ∨
a=a´ ∧ b=b´ (peor de los casos).

Esta última observación, da lugar a repasar las pruebas arrojadas por esta implementación no satisfactoria. Esto constituye la aportación final de esta tesis: Para encontrar soluciones no dominadas, intuitivamente se ha “negado” (1*) y se examina al mismo tiempo que respeten el orden Pareto en la dominancia Pareto DP, así, se encuentra el conjunto de soluciones minimales-ZO.

El enunciado “negar el orden Pareto”, induce a proponer una relación de orden dotando de propiedades adicionales a (*1). Esto permite formular una relación o conjunto parcialmente ordenado que identifica pares de soluciones no comparables y por otro lado estas soluciones deben satisfacer la condición de ser no dominadas bajo el orden Pareto. Finalmente, este conjunto de soluciones son minimales.

Consideremos el par (a, b) < (a´, b´).
Entonces (a, b) < (a´, b´) sii (a ≤ a´) ∧ (b≤ b´)
¬((a, b) < (a´, b´)) ≡
¬[(a < a´ ∨ a= a´) ∧ (b < b´ ∨ b = b´)] ≡
¬(a < a´ ∨ a= a´) ∨ ¬(b < b´ ∨ b = b´) ≡
a ≥ a´∧ a≠ a´ ∨ b ≥ b´∧ b≠ b´ ≡
(a > a´ ∨ b > b´)

De la misma manera para ¬ ((a´, b´) < (a, b)) ≡ (a´> a ∨ b´ > b)

Por tanto, se concluye que (a, b) , (a´, b´) son no comparables sii
(a > a´ ∨ b > b´) ∧ (a´> a ∨ b´ > b) (*2)

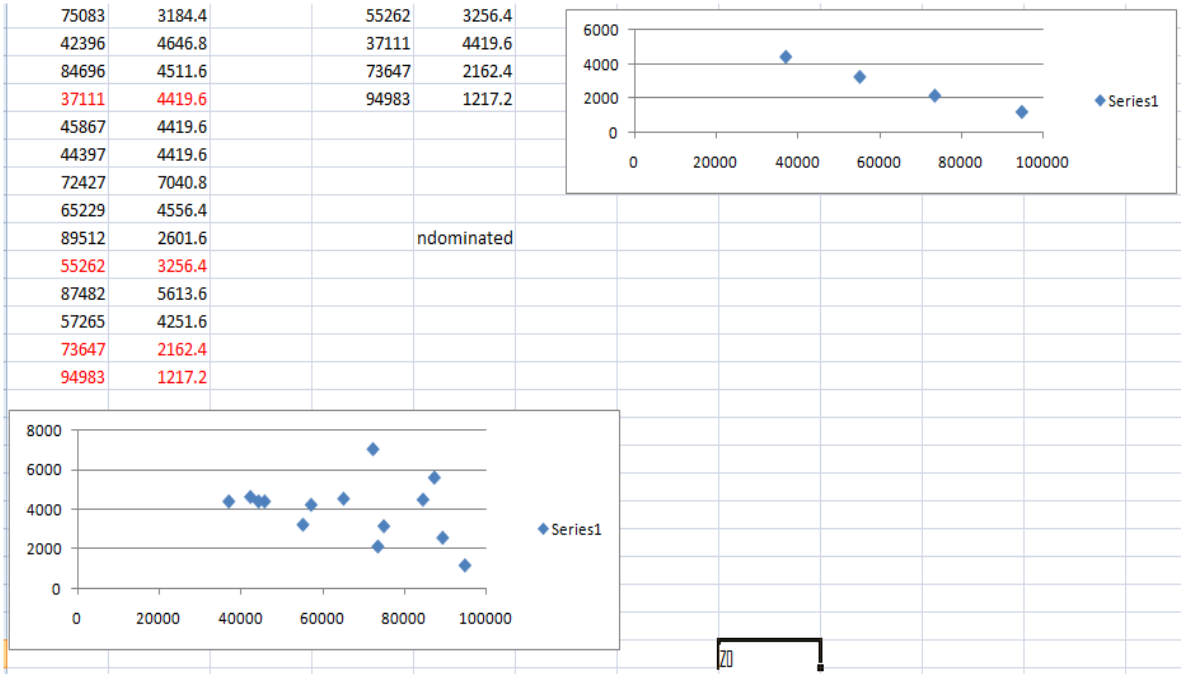
Bajo este orden llamado “Pareto NoComparable” (2*) y, combinadas adecuadamente con la dominancia Pareto, se obtienen todos los pares de soluciones minimales, lo que se confirma cuando nodom se aplica a todas las soluciones generadas por ZO.

Por último, se señala que:
Los minimales implican soluciones no comparables.
Soluciones no comparables no implica que estas sean minimales, sin embargo, si las soluciones son no comparables y además no dominadas entonces dichas soluciones son minimales.
Este conjunto de soluciones son definidas en esta tesis como minimales-ZO y son precisamente las soluciones que se han encontrado para formar el frente de Pareto para el problema de zonificación óptima.

De esta forma se ha logrado un frente de Pareto bien definido formado por soluciones no dominadas con soluciones minimales-ZO (ver ejemplos 4 y 5).

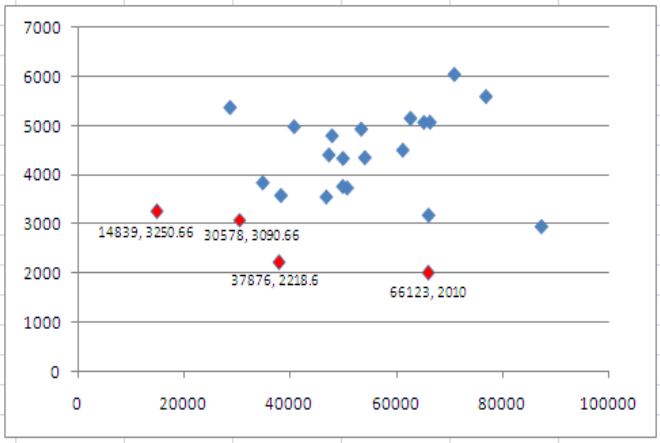
Ejemplo 4. En este ejemplo para 5 grupos, se puede ver que con el método de ZO se obtienen los pares de soluciones minimales al igual que nodom.

Sin embargo en este último método, que es la aportación final de este trabajo, se “liberan” algunas soluciones adicionales lejos de los maximales, los cuales deben ser filtrados. Esto se convierte en un trabajo futuro.



Ejemplo 5.

53450	4926		66123	2010
76989	5586.66667	no comparables	30578	3090.66667
66123	2010	no comparables	14839	3250.66667
50792	3736	no comparables	37876	2218.66667
65280	5064	*		
30578	3090.66667	no comparables		
54157	4348.66667	*		
66429	5068	*		
61297	4500.66667	*		
38310	3582.66667	*		
40802	4976	*		
34875	3840	no comparables		
62752	5144	*		
47349	4404.66667	no comparables		
46868	3552	no comparables		
49989	3764	*		
71020	6033.33333	*		
47952	4792	*		
14839	3250.66667	no comparables		
28715	5364.66667	no comparables		
37876	2218.66667	no comparables		
50007	4332.66667	*		
66145	3182.66667	*		
87440	2954	*		



Una contribución interesante de ZO es que nodom solo acepta en la entrada alrededor de 300 mil vectores para encontrarles sus puntos no dominados. ZO va generando soluciones en tiempo de ejecución y puede generar tantas soluciones no dominadas como los parámetros de la heurística lo permitan.

A continuación se presenta el pseudocódigo para resolver el problema de ZO:

5.3.4 Algoritmo para zonificación óptima: compacidad y homogeneidad

Algoritmo.

Sea:

n	Número de objetos a clasificar.
k	Número de grupos
Val_i	Valor que tiene el ageb i para la variable que se va a mantener homogeneidad
Ug	
MaxVNS	Número de veces que se va a recorrer la estructura de vecindades.
MaxBL	Número máximo de iteraciones para búsqueda local

kVecindad ← Generar un número aleatorio entre 1 y n

SolActual ← Genera una solución aleatoria que se encuentre en la vecindad *kVecindad*
costeActual ← getCosteComp(SolActual) , getCosteHom(SolActual)
// Este par de soluciones deben cumplir el "orden Pareto NoComparable"
cont ← 1
Mientras cont < MaxVNS hacer
    kVecindad ← 1
    Mientras kVecindad ≤ n
        SolCand← Genera una solución aleatoria que se encuentre en la vecindad *kvecindad*

```
            SolCand ← BusquedaLocal (SolCand)
            costeCand ← getCosteComp(SolCand), getCosteHom(SolCand)
            Si costeCand  < costeActual entonces
                  SolActual← SolCand
                  costeActual← costeCand
            Si no
                  kVecindad ← kvecindad + 1
            fin si
      fin Mientras
      Cont ← cont+1
fin Mientras
Regresa SolActual
```

Función getCosteComp (Sol)

*//Regresa un entero indicando que tan buena es la solución **Sol** en cuanto a compacidad (entre mas pequeño sea, la solución es mejor)*

```
i ← 1
cost ← 0
Mientras i ≤ n
    Si Ug_i no es centroide entonces
          dmin ← dist(Sol_1 , Ug_i )  //Distancia entre el objeto Sol_1 y el objeto i
          j ← 2
          Mientras j ≤ k
                Si dist (Sol_j , Ug_i) < dmin
                      dmin ← dist (Sol_j , Ug_i)
                Fin si
                j ← j + 1
          Fin Mientras
          cost ← cost + dmin
    fin si
    i ← i + 1
Fin Mientras

getCosteComp(Sol) ← cost
```

Función getCosteHom (Sol)

*//Regresa un entero indicando que tan buena es la solución **Sol** en cuanto a homogeneidad (entre mas pequeño sea, la solución es mejor)*

```
total ← 0
coste ← 0
Para i ← 1 hasta n hacer
    ng ← Obtener el número de grupo al cual pertenece el ageb i
    total ← total + Val_i
    totalGrupo_ng ← Val_i
fin Para
promedioIdeal ← total / k
Para j ← 1 hasta k hacer
    Coste ← coste + | totalGrupo_j - promedioIdeal |
fin Para
```

getCosteHom(Sol) ← coste

Función BusquedaLocal (Sol)

```
NumItera ← 0
SolMejorada ← Sol
costeSolMejorada ← getCosteComp(SolMejorada) , getCosteHom(SolMejorada)
Mientras NumItera ≤ MaxBL
    SolCand ← Generar solución aleatoria vecina de SolMejorada
    costeSolCand ← getCosteComp(SolCand) , getCosteHom(SolCand)
    Si costeSolCand < costeSolMejorada entonces
            SolMejorada ← SolCand
            NumItera ← MaxBL + 1
    Si no
            NumItera ← NumItera + 1
    fin Si
fin Mientras
BusquedaLocal(Sol) ← SolMejorada
```

6 Conclusiones y trabajo futuro

El objetivo de este trabajo consistió en encontrar un método que resolviera para zonificación óptima, la agrupación compacta y homogénea sobre agebs para dar respuesta a diversos problemas poblacionales de carácter censal.

Para resolver zonificación óptima, antes de proponer un método multiobjetivo para su solución, fue necesaria una revisión exhaustiva de la literatura en diseño territorial y de métodos de particionamiento. El propósito aquí fue resolver primero el particionamiento compacto (bajo el nombre de conglomerado geográfico), con dos métodos de eficiencia comprobada en problemas difíciles de optimización combinatoria: búsqueda por entorno variable VNS y recocido simulado RS.

Para elegir el método heurístico que mejor optimizara el particionamiento para agebs, se diseño un experimento estadístico factorial. La aplicación de esta metodología estadística proporcionó resultados para asegurar que búsqueda por entorno variable es la heurística que mejores resultados arroja para aproximar soluciones en particionamiento geográfico para agebs.

Se diseñó un modelo de particionamiento bi-objetivo como uno de “particionamiento en optimización combinatoria multiobjetivo” considerando en la función de costo, las dos funciones a optimizar: compacidad y homogeneidad. De esta manera, se desarrollo un algoritmo de particionamiento multiobjetivo y, para aproximar compacidad y homogeneidad simultáneamente se uso VNS logrando buenas soluciones. Para identificar de las soluciones generadas por VNS aquellas soluciones que sean no dominadas, se empleó una relación de orden Pareto NoComparable definida en este trabajo y comprobada su eficiencia en pruebas para ZO.

Los apoyos teóricos para definir el método de obtención de soluciones no comparables que forman el frente de Pareto para ZO, constituyen un buen estado del arte para problemas multiobjetivo. Dentro de la literatura estudiada en este trabajo, ha destacado el trabajo de Kung [Kung75], donde desprende una aplicación que identifica pares de soluciones no dominadas. Esta heramienta conocida como “nodom” permitió comprobar que las soluciones que se han generado en diversas pruebas para este trabajo son correctas.

Contribuciones

Una aportación interesante aquí es que nodom acepta alrededor de 300 mil soluciones en la entrada para procesarlas y el método para obtener soluciones no comparables en ZO genera tantas soluciones como la heuristica lo permita.

Una aportación adicional en este trabajo reside en que, el método para resolver la agrupación multiobjetivo para agebs puede ser empleado para otro tipo de datos, con algunas variantes en la implementación de acuerdo al tipo de datos a agrupar.

Para el caso de particionamiento multiobjetivo, el problema ha sido poco atendido. Las fuentes de complejidad son altas, y esto merece estudiarse. Sin embargo, en esta tesis, se ha resuelto un problema interesante como lo es ZO y además combina tres áreas: particionamiento, diseño territorial y optimización multiobjetivo.

Extensiones

El método que se ha propuesto, resuelve el problema de zonificación óptima con resultados satisfactorios en tiempo y calidad, sin embargo, esta afirmación requiere comprobación, con una serie de pruebas en un diseño de experimentos estadístico. En este trabajo no se realizo un experimento sistemático de pruebas para ZO. Este punto exige atenderse como trabajo posterior.

En este sentido, también se deben realizar pruebas adicionales con el algoritmo de intermediación de aristas que se ha desarrollado como una ópción de zonificación, esto permitirá comparar la eficiencia y calidad de los resultados generados por el método de ZO y el publicado en [Ochoa09].

Dado que el orden Pareto NoComparable, identifica de entre las soluciones minimales generadas, pares adicionales “irrelevantes”, se persigue encontrar una extensión a este orden que refleje un mejor comportamiento en el frente de Pareto.

La integración del sistema de información geográfica map-x a un problema de diseño territorial o particionamiento geográfico constituye una buena aportación. En la literatura vista no se encontró algún proceso de integración con map-x.
Sin embargo, el procedimiento de particionamiento con el sistema de información geográfica ha dado excelentes resultados cuando solo se resuelve la compacidad para agebs. Cuando se integran los modulos de particionamiento compacto-homogeneo, aún existe el inconveniente de editar manualmente algunos valores de los identificadores de agebs. Este error debe rastrearse en el código, lo que implica que debe ser solucionado, aunque se supone un trabajo arduo, se tienen varias opciones prácticas para resolver este problema.

Considerando la literatura revisada en el área multiobjetivo, se estima estructurar los tópicos, métodos, aplicaciones, autores etc. para categorizar y ubicar los problemas que se abordan y su solución (tal y como se presentó en el capitulo 2 para diseño territorial).

Finalmente no se puede negar que el campo de desarrollo en problemas multiobjetivo sigue siendo un reto, pero también es cierto que cada problema puede ser resuelto considerando los aspectos adecuados bajo un exhaustivo trabajo.

.

Anexo A. Integración de un sistema de información geográfica

En este anexo se presenta la estrategia de implementación para integrar el procedimiento de particionamiento geográfico con el sistema de información geográfica map-x (PG-sig).
Se describe también, el análisis de datos y el proceso de selección de variables censales con restricciones para crear la matriz de homogeneidad-distancias (la entrada a multiobjetivo).

Análisis de datos en particionamiento y regionalización

Muchas preguntas de interés en las ciencias sociales, las ambientales y las geo-ciencias, involucran objetos con una clara componente geográfica, es decir, que su localización en el espacio en relación con otros objetos es importante para su estudio (datos espaciales). Algunos ejemplos de este tipo de problemas son la dispersión de contaminantes, la interpretación de imágenes de satélite, la distribución de una enfermedad, la posible relación entre nivel de ingreso y preferencias electorales o la mejor localización para un cierto negocio, etc.

El principal propósito del análisis de datos espaciales es detectar y modelar posibles patrones que formen los datos estudiados en el espacio. Existe gran cantidad de métodos estadísticos que tienen como propósito describir y modelar datos. Sin embargo, una hipótesis fundamental de la estadística clásica es que las observaciones son independientes entre sí mientras que, los eventos espaciales, por su misma naturaleza, están relacionados unos con otros en el espacio como sucede, por ejemplo, con ciertos fenómenos ambientales y sociales. Entre los problemas relacionados con el análisis de datos espaciales se pueden mencionar la autocorrelación espacial, la identificación de “outliers” espaciales y el problema de las unidades de área modificables.

El problema de analizar fenómenos con una componente espacial se asemeja al de analizar datos que se relacionan entre sí en el tiempo y para el cual se han desarrollado técnicas estadísticas específicas, las cuales conforman el análisis de series de tiempo. Del mismo modo, dentro del análisis espacial se han desarrollado diversos métodos geoestadísticos para analizar los datos espaciales que toman en cuenta explícitamente su ubicación. Así, por ejemplo, el concepto estadístico de correlación entre objetos tiene su equivalente en la geoestadística: la autocorrelación espacial; en ella, la posición relativa de unas observaciones con otras puede ser importante y debe ser incluida explícitamente en el análisis.
Por otro lado, el análisis de datos espaciales presenta varios problemas prácticos causados por la escala, la falta de “orden” en el espacio (en contraste con, por ejemplo, las series de tiempo), la falta de una medida única de vecindad y la agregación de datos en áreas.

Muchas aplicaciones requieren trabajar con datos espaciales agrupados por unidades de área; esto es lo que ocurre al utilizar unidades básicas censales. En casos como este surgen problemas adicionales, debidos a la misma agregación (clasificación de unidades territoriales), que se conocen en su conjunto como el problema de las unidades modificables de área. Aparte de esto, hay muchas ocasiones en las que el problema requiere datos que han sido recolectados en unidades de área diferentes e independientes unas de otras como, por ejemplo, si se busca calcular la población que vive en una zona de riesgo o en un distrito electoral.

El problema de las unidades de área modificables

Hay muchos problemas geográficos en los que los datos se encuentran agregados en áreas, ya sea por que resulte natural hacerlo o porque sea necesario, como sucede con las unidades básicas censales para datos socioeconómicos siendo las zonas irregulares geométricamente hablando (como sucede con las agebs).
En particular, en las ciencias sociales suele ser común que, por razones de confidencialidad, los datos se publiquen agrupados por zonas aunque hayan sido recolectados a nivel individual. En general, el análisis de datos agregados presenta diversas limitaciones y dificultades, algunas de las cuales se discuten a continuación.

Áreas son las unidades administrativas o legales (municipios, estados, distritos electorales, etc.) así como áreas naturales tales como zonas clasificadas por tipo de vegetación o de suelo.

Los censos de población constituyen un ejemplo de datos agregados por área ya que la información se recopila por hogar pero se hace pública agrupada en unidades de área; en México, los datos se agregan por ageb (área geoestadística básica), las cuales son una partición de los municipios.

Aquí es importante enfatizar que, en general, la forma en que se diseñan las zonas en las que se agrupan los datos es independiente de la información misma, utilizándose criterios administrativos o que faciliten su recolección; es por ello que dichas áreas son consideradas arbitrarias, es decir, son tan sólo una regionalización de entre muchas posibles. Por ejemplo, los criterios utilizados para la creación de las agebs son independientes de aquellos usados para las secciones electorales [Fotheringham91].

También es necesario subrayar que los criterios de regionalización tienden a basarse en una sola variable (por ejemplo, número de habitantes) lo cual hace que no sean realmente adecuadas para analizar otros factores (como ingreso o nivel educativo). Así, aún cuando una regionalización se realice siguiendo un criterio de homogeneidad, es prácticamente imposible que el mismo sea válido para todas las variables.

Los resultados del estudio de datos agregados arbitrariamente por área están en función de la escala, la orientación y el origen de la regionalización empleada así como del número de zonas. A este fenómeno se le conoce como el "problema de las unidades de área modificables" [Openshaw84] y se puede enunciar como sigue: los resultados obtenidos del análisis de datos agregados dependen de la forma en que hayan sido agrupados dichos datos, es decir, depende tanto del número de zonas como de la forma de éstas. Así, el problema de las unidades de área modificables tiene dos partes, el llamado "efecto de escala" y el "efecto de la división en zonas". El efecto de escala provoca que, al calcular una estadística (como varianza o correlación) a diferentes escalas, se obtengan resultados distintos. Por su parte, el efecto de la división en zonas provoca que, al reagrupar los datos en sistemas de zonas diferentes aunque a la misma escala, se obtengan también distintos valores para una misma estadística [Fotheringham91]. Además, el problema de las unidades de área modificables surge tanto con la agregación de datos a nivel individual como cuando se agrupan áreas pequeñas para formar otras más grandes.
Esta situación ha sido muy discutida por diversos autores y en el estado del arte se amplían tanto los conceptos como los trabajos desarrollados, sus alcances y las limitaciones actuales.

Datos espaciales
Teniendo claro el papel de los datos espaciales y su asociado problema de unidades de áreas modificables, se entiende que analizar la cantidad, calidad y naturaleza de los datos espaciales se convierte en la tarea inicial para dar lugar a un proceso donde participan estos datos.

Dentro del conjunto de datos espaciales se distinguen unidades geográficas o territoriales o cualquier otro dato que cumpla con la definición siguiente:

Definición de dato espacial
Un dato espacial es aquel que tiene una clara componente geográfica, una ubicación en el espació bien definida y que su descripción está dada por vector de variables.
"Los datos geográficos son entidades espacio-temporales que cuantifican la distribución, el estado y los vínculos de los distintos fenómenos u objetos naturales o sociales".
Los datos espaciales refieren a entidades o fenómenos que cumplen los siguientes principios básicos:
Tienen posición absoluta: sobre un sistema de coordenadas (x,y,z).
Tienen una posición relativa: frente a otros elementos del paisaje (topología: incluido, adyacente, cruzado, etc.).
Tienen una figura geométrica que las representan (punto, línea, polígono, área).
Tienen atributos que lo describen (características del elemento o fenómeno).

En este sentido, las áreas geoestadísticas básicas agebs pertenecen a este conjunto de datos.

La ageb es el área geográfica que corresponde a la subdivisión de las AGEM. Constituye la unidad básica del marco geoestadístico nacional y dependiendo de las características que presentan las ageb, se clasifican en dos tipos:
Área geoestadística básica urbana
Área geoestadística básica rural

La ageb urbana es el área geográfica ocupada por un conjunto de manzanas que generalmente son de 1 a 50 habitantes, perfectamente delimitadas por calles, avenidas, etc.; este tipo de ageb se asigna en áreas geográficas de localidades que tengan una población igual o mayor a 2,500 habitantes. La segunda (rural), es una extensión territorial que puede llegar a tener hasta 10,000 hectáreas y contener un conjunto de localidades con menos de 2,500 habitantes cada una, asentadas en terreno de uso generalmente agropecuario o forestal. Para ponerlo en otras palabras, las agebs urbanas subdividen a las áreas del país que cuentan con 2,500 o más habitantes, o que son cabeceras municipales, éstas son denominadas localidades urbanas de acuerdo con la normatividad del marco geoestadístico nacional MGN las agebs rurales.

Normatividad del MGN de las agebs rurales
Subdividen al resto del país, el cual contiene a las denominadas localidades rurales, es decir, aquellas que tienen menos de 2,500 habitantes y que no son cabeceras municipales. Por lo anterior las agebs urbanas están contenidas en localidades urbanas y las agebs rurales contienen localidades rurales.

La clave de un ageb se compone por código de ageb, municipio, entidad y localidad.

	CAMPO	DESCRIPCIÓN
1	ENT	Clave de la entidad federativa (2 dígitos)
2	MUN	Clave del municipio (3 dígitos)
3	LOC	Clave de la localidad (4 dígitos)
4	ageb	Clave de la ageb (4 dígitos)

NOTA: La clave de la ageb se forma con la unión de los campos ENT, MUN, LOC y ageb.

La ageb es un área de extensión territorial bien definida, su ubicación en el espacio es determinada por latitud, longitud y altitud siendo así una unidad territorial o área geográfica.

La extracción de los datos que se utilizan en este trabajo se ha tomado del último censo de población y vivienda 2000 publicada por el INEGI siendo los datos de carácter censal descritos por 167 variables y su código de ageb para todos los estados de la república mexicana.

Dado que el proceso de agregación requiere las coordenadas geográficas que la base de datos original no contiene, se ha utilizado un sistema de información geográfica SIG y mapas físicos para este propósito.

La ventaja de utilizar las agebs obedece a que la clasificación a nivel ageb permite apreciar las diferencias con mucha mayor claridad debido a que los grandes promedios que están detrás de los indicadores estatales y municipales, disimulan o suavizan algunas situaciones.

Selección de variables y datos
Cuando se procesan datos espaciales y censales, se requiere de un análisis exploratorio para su adecuación; de este modo, es posible otorgar calidad a los datos finales además de ser accesibles a diferentes tratamientos posteriores.

En particular, en el proceso de agrupación para el problema de zonificación óptima los datos censales originales se han extraído de una base de datos que ofrece el INEGI. Estos datos están definidos de manera descriptiva por un vector de variables y se conocen como agebs (áreas geoestadísticas básicas).
Bajo ciertos procedimientos (citados a continuación), estos datos se han depurado y transformado para que se proceda al cálculo de distancias entre agebs.

El resultado que se busca es una matriz básica de distancias (matriz de disimilitud) y otra de variables (matriz de homogeneidad). Estas estructuras de distancias y variables se emplean como entrada a las diferentes propuestas de agrupamiento geográfico que se han desarrollado a lo largo de esta tesis y al final al procedimiento de agrupación bi-objetivo de compacidad y homogeneidad.

Proceso para preparación de datos

1. Limpieza de datos: Resuelve tanto redundancias y consecuencia de la integración como problemas de ruido o valores perdidos eliminando inconsistencias/conflictos entre datos.
2. Integración de datos: Obtiene los datos de diferentes fuentes de información solucionando problemas de representación y codificación. El propósito aquí es integrar los datos desde diferentes tablas o bases de datos.
3. Transformación de datos: Los datos son transformados o consolidados de forma apropiada para la extracción de información.
4. Reducción de datos: Selecciona/extrae datos relevantes para la tarea de clasificación.
5. Paralelamente a la selección de datos, es necesaria la selección de características. Este proceso consiste en la búsqueda del subconjunto de características que sean representativas para el problema. Una reducción de la dimensionalidad para variables utilizando componentes principales produjo las variables representativas.
6. Por último, en un proceso de particionamiento, generalmente la matriz de distancias es la que se lee y debe ser obtenida con métodos alternativos.

Cabe señalar que previo a la reducción de la dimensión de las variables con técnicas multivariadas, se ha puesto énfasis en el formato técnico de la información censal, la cual corresponde al XII censo general de población y vivienda del INEGI. Ésta información contiene los datos más confiables y ampliamente utilizados además de fácil obtención en nuestro país. Por esta razón la ageb (área geoestadística básica) es la unidad básica de información que tiene más ventajas sobre otras.
Para las agebs el concepto de dato refiere a la situación geográfica y características a las variables.

La evidencia empírica que se aporta para la agrupación se basa en los 4952 agebs (áreas geoestadísticas básicas) de la zona metropolitana del valle de México (ZMVM) y los 429 agebs de la zona metropolitana del valle de Toluca. Las descripción para cada ageb esta conformadas por 147 variables socioeconómicas disponibles para dichas áreas (reducidas a 76 por técnicas multivariadas) [Web3].

Por otra parte el INEGI proporciona en dicha base de datos, información adicional para referenciar los lugares geográficos correspondientes que se obtienen con la ayuda de un sistema de información geográfico SIG.

Las bases de datos obtenidas del INEGI están en formato de dbase, y se han convertido a bases de datos de access 97 debido a:

1. La disponibilidad para múltiples usuarios además de que access es un manejador de uso común y de fácil acceso. En algunos casos es necesario modificar los datos del censo, por ejemplo, para estimar el valor de algunas variables poblacionales para los años intermedios entre censo y censo.
2. El tamaño de las bases de datos se reduce considerablemente. Al migrar de dbase a access 97 el tamaño es hasta 80% menos que la base original.
3. Hasta el momento no se requiere el poder o seguridad de algún manejador de bases de datos más robusto.
4. Los datos agebs se transformaron a formato de texto y compatibles con excel para ser accesibles a pam y a los otros algoritmos desarrollados en este trabajo.
5. Finalmente estos datos deben adecuarse al proceso que se someten: reducción de la dimensión, obtener matriz de distancias, de variables y de correlaciones.

Hasta ahora, con los procedimientos descritos en resumen, es posible asegurar que las agebs están en condiciones de ser clasificadas.

A.1 Sistema de particionamiento con map-x

Una vez que se ha obtenido una partición sobre compacidad u homogeneidad-compacidad para las agebs, es importante dotar a los resultados de información adicional para que la agrupación se observe gráficamente en mapas. En este punto se ha desarrollado un sistema llamado particionamiento geográfico-sig (PG-sig), donde los diferentes algoritmos se han adaptado a subprogramas con una estructura modular bien definida con el fin de aprovechar las capacidades de los diferentes algoritmos y de las características de sistemas y modularidad.

Estructura general del sistema

La aplicación final de particionamiento geográfico-sig, es un sistema que tiene la siguiente funcionalidad a nivel usuario:

a) Elegir la entidad federativa sobre la cual se procesarán las agebs, cada entidad está dividida en un cierto número de zonas geográficas agebs.
b) Seleccionar las variables censales adecuadas para un problema particular.
c) Hacer uso del procedimiento de particionamiento con VNS o con RS o multiobjetivo VNS (cada uno en subprogramas separados).
d) Crear mapa del resultado de la partición.

Recordar que en el particionamiento compacto para agebs, dado que solo requiere la matriz de disimilitud como entrada, no “existen” las variables censales en dicho particionamiento.
Cuando a la partición sobre compacidad se le calcula la homogeneidad sobre variables, entonces si es necesaria la matriz de variables y de distancias.
El proceso de selección de variables (acotadas o no) para cualquiera que sea el agrupamiento, solo es necesario cuando no se requiere la “participación” de todas las variables en la partición (la mayoría de los problemas censales utilizan solo variables “adecuadas”). Una aplicación de este proceso puede verse en [Bernábe06].

A pesar de que la homogeneidad si exige un procedimiento de variables, también la compacidad, (aunque no es muy común estos casos), puede ser resuelta partir de una matriz reducida de distancia que obedece a restricciones de variables.

La manera de “combinar” las dos matrices de variables y distancias para que funcionen como entrada en el particionamiento multiobjetivo, se apoya de dos pasos en un solo procedimiento:

1. Un proceso de selección de variables adecuadas de acuerdo a un problema particular. El resultado es un subconjunto de variables que pueden estar restringidas en sus valores censales (este modulo se la ha llamado “modulo de consultas).
2. Del grupo de variables resultante del paso anterior, se obtiene una matriz de disimilitud ajustada para las variables seleccionadas.

Esto significa que se cuenta con una matriz de distancias sobre variables específicas.

En las siguientes líneas se describe la forma en que se desarrollo este procedimiento y también la integración del particionamiento con map-x.

Los algoritmos de particionamiento compacto y de homogeneidad-compacidad, se implementaron con las propiedades de la programación modular, lo que constituye un acoplamiento en el sistema PG-sig o pueden funcionar de forma independiente.
De esta manera las estructuras de datos de los resultados del agrupamiento se hacen compatibles con los requerimientos de las estructuras de datos del sistema de información geográfica map-x. El propósito aquí es ver gráficamente los resultados del particionamiento.

La estructura modular final del sistema PG-sig se muestra en la siguiente figura. Cada uno de los módulos de la figura a, se han implementado por separado, de tal modo que también pueden ser

trasladados y usados por otras aplicaciones. Sin embargo, recuérdese el algoritmo que resuelve la compacidad-homogeneidad visto con detalle en el capítulo 5.
Por tanto no significa que se efectué el cálculo por separado, el propósito de las figuras es mostrar una estructura que permite comprender visualmente la integración de los módulos.

Es importante señalar que se puede obtener también una agrupación solo para homogeneidad y esa es otra razón por la que todas las figuras tienen señalada la clasificación como compacta y homogénea en módulos separados. Esta aclaración vale para todas las demás figuras.

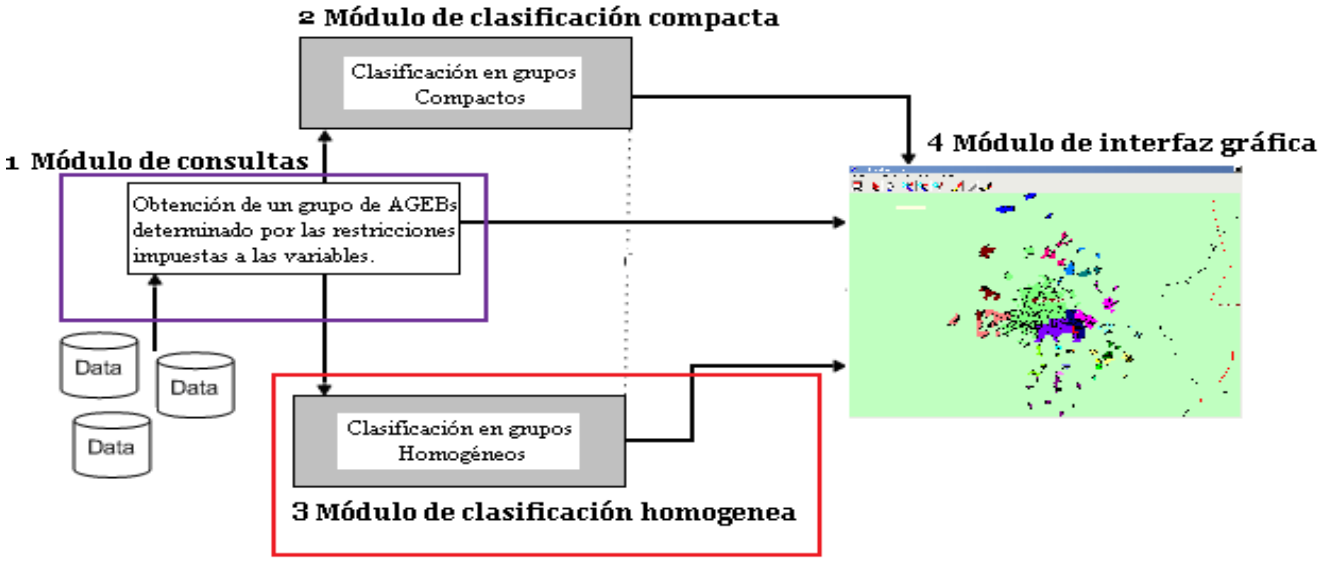


Fig. a. Diagrama modular de particionamiento y SIG.

A.2 Módulo de consultas (selección de variables)
Los algoritmos de clasificación tradicionales no condicionan las variables y para diversos problemas es útil restringir tanto las variables a agrupar como los valores de estas.
Por ejemplo, si se desea obtener una partición de mujeres económicamente activas, se necesitan solo aquellas variables de mujeres con algún porcentaje de estas y homogeneizar a “mujeres económicamente activas”.
En este sentido, se ha implementado un proceso de categorización que inicia con la extracción de variables poblacionales con un procedimiento de búsqueda en la base de datos de variables, proporcionando así, un conjunto de agebs cumpliendo con las características especificadas [Bernábe06].

A.2.1 Organización de la información en bases de datos
En la práctica e independientemente del problema poblacional a resolver, la información censal se depura o analiza previamente en un proceso de bases de datos, por ejemplo, lo que consume mucho tiempo.
Cuando la depuración de esta información es la selección de variables con valores definidos, suelen acudirse a paquetes de software que no son especializados para la resolución de problemas de este tipo.
El proceso de mover o transformar la información para seleccionarla con especificación de valores en las variables implica, generalmente, emigrar los datos de un paquete de software, lo que se traduce en una actividad costosa. Se busca así una aplicación que incluya procedimientos que resuelvan los puntos anteriores. Se atiende esta situación con el software PG-sig que se ha desarrollado y se divide en tres fases (subprogramas) que se describen a continuación:

1) Integración de los datos del XII censo del inegi así como los catálogos de codificación al software [Web3]. En esta parte, se elige para un estado, las variables poblacionales y las

características de estas (restringiendo las variables a determinados valores), generando una lista de agebs, es decir, una categoría determinada por las características de interés del usuario.

2) Implementación de un algoritmo que categorice la lista de agebs anteriormente generada en un número de grupos. El procedimiento de particionamiento al grupo caracterizado consigue grupos bajo los siguientes criterios: a) Que se mantenga homogeneidad en el número de habitantes o en cualquier otra variable poblacional disponible y b) que sean compactos.

3) Creación de una interfaz grafica para mostrar los resultados de la clasificación en un mapa.

A.2.2 Determinación de restricciones

Para resolver la selección de variables con restricciones en los valores, se ha diseñado un procedimiento donde se genera una categoría o lista de agebs con las cotas o restricciones impuestas a las variables de interés. En particular, para particionamiento multiobjetivo, se necesita un proceso especial para la integración de variables y distancias.

La organización de las agebs en la base de datos consiste de un proceso que utiliza en total 65 bases de datos: 32 para la información censal de cada estado de la república mexicana, 32 para la información geoespacial de los agebs para poder referenciarlos en los mapas y una base de datos con la información básica y la información común para todos los estados.
La base de datos principal contiene la información que es común para todos los estados e información para la configuración del sistema. Esta base de datos (catalogos97.mdb) debe estar en memoria secundaria para que la aplicación pueda ejecutarse. Su tamaño es de aproximadamente 19 Mb y a pesar de que tiene información que por el momento no se utiliza (como los catálogos de colonias para cada localidad), se ha dejado almacenada esa información para futuras ampliaciones.

La base de datos está compuesta de 6 entidades que se describen en la tabla 1 y la descripción puede verse en la figura 1.

Tabla 1 Descripción de las tablas de la base de datos principal.

Tabla.	Descripción.
Colonia	Contiene un catalogo de colonias para todos los estados, así se puede saber que colonias pertenecen a una determinada localidad de un determinado Municipio de un determinado estado.
Configuración	Contiene información para que la aplicación configure ciertos parámetros.
Entidad	Contiene un catalogo de los estados de la republica mexicana. También contiene el nombre del mapa a ser usado cuando se despliegue la información gráfica de un estado y un valor para cada estado que indica si su información censal esta presente.
Municipio	Contiene un catalogo de los Municipios de todos los estados.
Localidad	Contiene un catalogo de las localidades de cada Municipio.
Variables	Contiene una lista de variables disponibles en el censo así como su descripción.

Tabla 1 Descripción de las tablas de la base de datos principal.

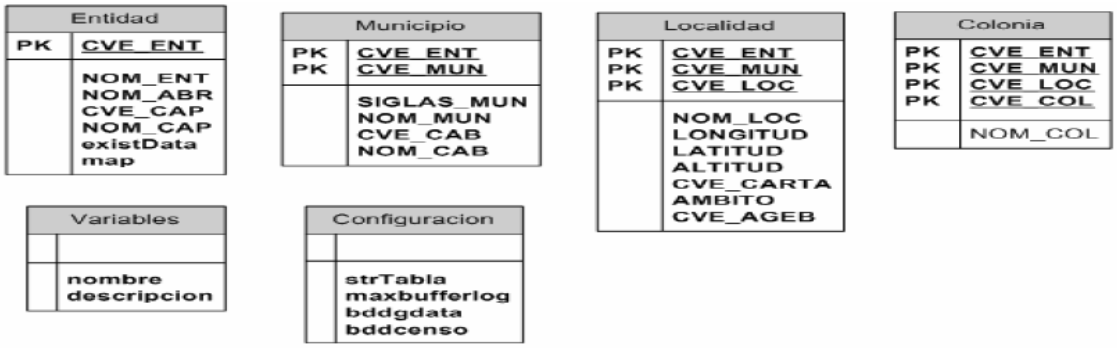


Fig. 1 Tablas de la base de datos Catalogos97.

A.2.2. 1 Bases de datos con la información geoespacial

Las 32 bases de datos contienen también la información geoespacial necesaria para referenciar correctamente los agebs en el mapa. Además, esta base de datos permite guardar la distancia entre cada par de Agebs del estado correspondiente.

El nombre de cada una de estas bases de datos se forma por "gdata" más la clave del estado al que corresponde la información (figura 2). Las claves de los estados están descritas en la base de datos catalogos97 en la tabla Entidad.

Fig. 2 Tablas de la base de datos con información geoespacial para cada estado.

A.2.2.2 Actualización de la información

El diseño de las bases de datos asegura que actualizar los datos censales o incluso cambiar de censo (siempre y cuando la información del censo se proporcione por agebs) es posible, incluso sin modificar ni una sola línea de código de la aplicación. Todos estos cambios se hacen sobre las bases de datos.

Difícilmente van a cambiar los nombres de los municipios y localidades así como la ubicación y delimitaciones de los Agebs.

Sin embargo, si se diera un cambio, solo se debe modificar la información de las tablas de la base de datos catalogos97 para la actualización de nombres de municipios o localidades. En el caso de algún cambio en las delimitaciones de los agebs lo que se altera es la capa (mapa) correspondiente al estado donde cambio. Y en el caso de un cambio en las claves de agebs también se deben modificar estas claves y actualizar las coordenadas geográficas en la tabla gdata de la base de datos con la información geoespacial.

Los reemplazos que van a darse con mayor frecuencia, son las actualizaciones de los datos del censo o el cambio a otro censo. Para cualquiera de estos casos simplemente se remplazan las bases de datos con la información censal por las "nuevas". Se deja el nombre del campo con la clave de ageb como "id"; si alguna variable fue removida del censo o variable nuevas fueron incluidas, hay que darlas de baja o de alta respectivamente en una tabla "variables" de la base de datos base.

En tiempo real, cuando se ejecuta la aplicación de selección de variables, se asume que las variables descritas en la tabla variables de la base de datos base están presentes para todos los estados, si algún estado no cuenta con información censal para alguna variable z y esta es elegida para análisis, el programa genera un error en tiempo de ejecución.

A.3 Categorización con restricciones

Generalmente no se trabaja sobre todas las zonas ni todas las variables, así que se extrae solo la información con características importantes para el problema censal a resolver. En este punto se ha construido un sub-programa de generación de consultas.

A.3.1 Generación y creación de consultas

De acuerdo a lo descrito hasta este momento, la aplicación se basa en trabajar sobre consultas previamente establecidas (selección de variables con valores).

Una consulta precisamente es un subconjunto de agebs (del total de agebs de un estado) determinados por el valor de algunas de sus variables. Una consulta también puede ser todos los agebs de un estado. De este modo, se elige un estado y se puede crear una nueva consulta o abrir una consulta anteriormente generada.

Para la creación de una nueva consulta, se selecciona una o más variables y se “impone” una restricción a cada una de las variables elegidas. Por tanto, la consulta generada estará formada de aquellos agebs en que cada variable elegida cumpla con la restricción impuesta. La figura 3 muestra la primera ventana que se presenta cuando se quiere generar una nueva consulta, en esta se eligen las variables de interés.

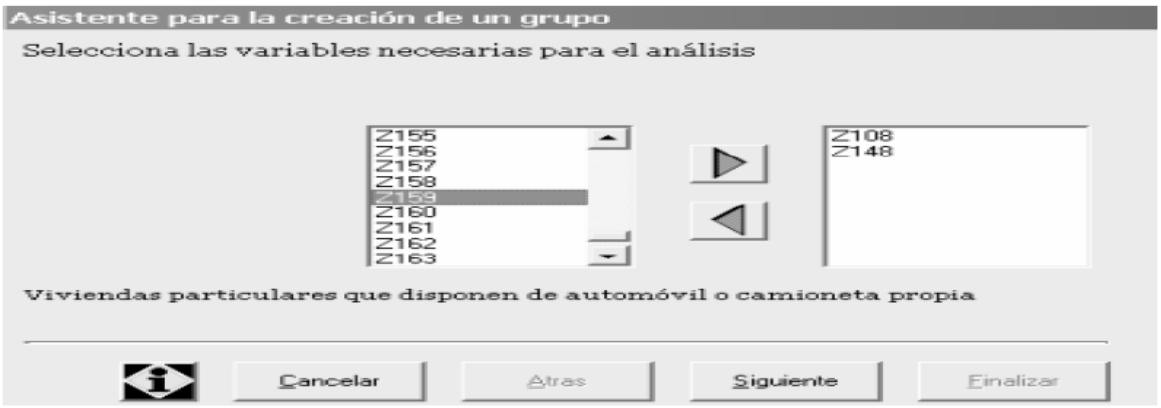


Fig. 3 Selección de variables. En la izquierda se encuentra una lista de todas las variables disponibles, en la derecha se van colocando las variables elegidas.

A.3.2 Restricciones a los valores de las variables en las consultas

La aplicación incluye las restricciones más utilizadas por los investigadores en problemas censales y otras muy generales para poder producir prácticamente cualquier restricción a una variable.

Los tipos de restricciones que el usuario puede escoger para cada variable seleccionada son:
RANGO. Que la variable este dentro de un rango de valores.
MAYOR. Que la variable sea mayor o igual a un valor.
MENOR. Que la variable sea menor o igual a un valor.

Para cada una de estas restricciones el valor introducido puede ser interpretado de tres formas distintas pudiendo realizar diferentes combinaciones para cada variable.

1. Como valor ordinario si Tipo = Valor. El valor dado se toma como simple valor de la variable. En la figura se muestra un ejemplo, a la variable Z148 se le está asignando la restricción de que sea mayor o igual a 200, es decir, la consulta contendrá a aquellos agebs cuyo valor de la variable Z148 sea mayor a 200 y que además se cumplan las restricciones de las otras 2 variables que se muestran (Z108 y Z159) (Figura 4).

Fig. 4 Imposición de una restricción: La variable Z148 debe ser mayor al valor 200.

2. Como porcentaje si Tipo = Porcentaje. El valor introducido es justamente el porcentaje de la variable.
El porcentaje se calcula como sigue: Sea x la variable seleccionada, n el valor introducido y m el valor máximo de la variable x en todas las agebs del estado (este valor se tomara como el 100%)
Porcentaje $_x$= n*m/100

En la figura 5 se muestra un ejemplo. A la variable Z159 se le asigna la restricción de que este entre el 5% y 40%, es decir, la consulta contendrá las agebs cuyo valor de la variable Z159 este entre el 5 y 40% y que además se cumplan las restricciones de las otras 2 variables que se muestran (Z108 y Z148).

Fig. 5 Imposición de una restricción: El valor de la variable Z159 debe estar entre el 5 y 40%, tomando como el 100% al valor más alto de la variable Z159.

En el capítulo 5 se tiene una expresión para las consultas “Query” y por otro lado se han publicado estos resultados en [Bernábe06].

A continuación se muestra la implementación de la función que regresa una cadena de texto con la subconsulta SQL para obtener el porcentaje de la variable varName donde value es el valor del porcentaje deseado:

```
Public Function percent(varName As String, value AsDouble) As String
percent = "(SELECT MAX(" & varName & ") FROM " & tabla & ") * " & Str(value) & " / 100"
End Function
```

3. Como el promedio de la variable. Si en lugar de un valor se introduce la palabra “promedio”, “average”, “avg” o “prom”.

El promedio de una variable x se calcula como sigue: $1/n \sum_{i=1}^{n} x_i$ donde n es el número de agebs del estado correspondiente y Xi es el valor de la variable x en el i-ésimo ageb.

En la figura 6 se muestra un ejemplo: a la variable Z108 se le asigna la restricción de que su valor este por encima del promedio, es decir, la consulta contendrá a aquellos agebs cuyo valor de la variable Z108 sea por encima del promedio y que además se cumplan las restricciones de las otras 2 variables que se muestran (Z159 y Z148).

Fig. 6 Imposición de una restricción: El valor de la variable Z108 debe estar por encima del promedio.

La implementación de la función que regresa una subconsulta que obtiene el promedio de la variable varName es:

```
Public Function average (varName As String) As String
average = "(SELECT AVG(" & varName & ") FROM " &tabla & ")"
End Function
```

Así, al crear una consulta con las variables y las restricciones vistas en los 3 ejemplos anteriores para el estado de Toluca, se genera una lista de agebs en las que las personas que trabajan como obrero en esas zonas está por encima del promedio (variable Z108), además, en estas zonas hay más de 200 viviendas que son propias y solo entre el 5% y 40% de las viviendas en cada una de estas zonas tienen automóvil propio.

El resultado de estas consultas dice que solo el 34.1% de los agebs de Toluca tienen esas características. En la figura 7 se ve la consulta en forma gráfica, donde se aprecia que los agebs con estas características están repartidos por todo el estado.

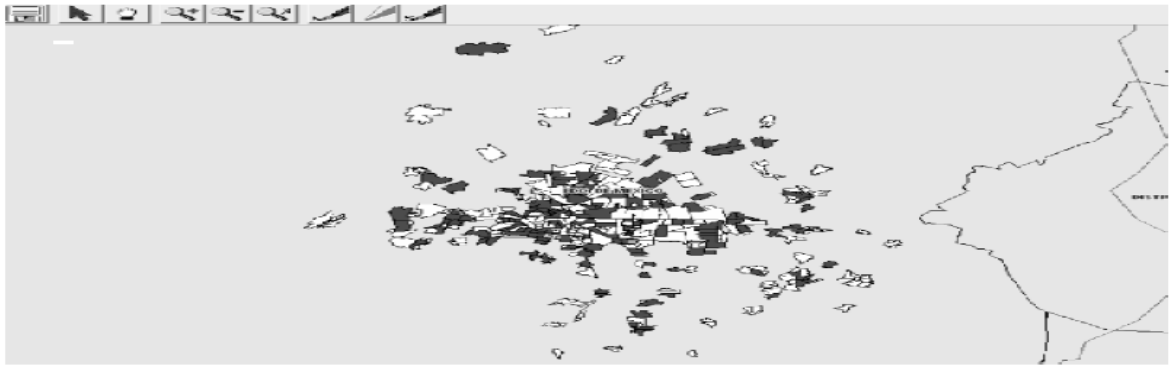


Fig. 7 Vista grafica de la consulta generada.

A.3.3 Guardando consultas

Es común que se trabaje con un grupo de agebs durante un tiempo, el crear la misma consulta cada vez que se ejecuta la aplicación es un proceso inútil. Por esto la aplicación tiene la opción de guardar la consulta generada y poder abrirla en cualquier momento (ver tabla2).

Número de línea.	Contenido
1	El numero 270378 que lo identifica como un archivo .qry válido.
2	Una cadena con la descripción de la consulta
3	Una cadena con la consulta SQL
4	Clave de la entidad federativa sobre la cual va a actuar la consulta.

Tabla 2 Estructura de un archivo que contiene una consulta.

Se deduce al ver la tabla anterior, que al guardar una consulta se guarda la entidad que se utilizó, de tal modo que al abrir la consulta guardada se cargara la entidad federativa con la cual se guardo la consulta. Si se quiere realizar la misma consulta pero sobre otro estado, modificar la línea 4 con un editor de texto y colocar la clave de la entidad con la que se quiere trabajar.

El guardar en el archivo la consulta SQL en lugar de la lista de agebs que cumplen con las restricciones impuestas, tiene 2 grandes ventajas: Primero, el archivo que contiene a la consulta es demasiado pequeño (1 KB por lo regular) y también nos aseguramos de que si cambiaron los datos censales, al abrir la consulta estará actualizada puesto que se ejecutara la sentencia SQL sobre los nuevos datos. Mientras que el tiempo que se pierde en obtener de la base de datos la lista de agebs correspondiente a la consulta SQL es despreciable.

A.3.4 Algoritmo para la creación de consultas

Se ha implementado la generación de consultas utilizando el motor de SQL Así, se formo una consulta SQL para cada variable seleccionada y al final se unieron estas consultas mediante un "AND" para formar una consulta SQL con las restricciones de todas las variables seleccionadas. El algoritmo para obtener la consulta SQL es el siguiente:

```
q ← "SELECT id FROM " + tabla + " WHERE"
para i ← 1 hasta n hacer
        s1 ← Valor1i
    s2 ← Valor2i
    si Valor1i  se va a tomar como el promedio de la variable
Vi entonces
        s1 ← "(SELECT  AVG(" + Vi + " FROM " + tabla + ")"
    fin si
    si Valor2i  se va a tomar como el promedio de la variable
Vi entonces
        s2 ← "(SELECT  AVG(" + Vi + " FROM " + tabla + ")"
    fin si
    si Valor1i se va a tomar como porcentaje entonces
        s1 ← "(SELECT MAX(" + Vi + ") FROM " + tabla + ") *
" +             Valor1i + "/100"
    fin si
    si Valor2i se va a tomar como porcentaje entonces
        s2 ← "(SELECT MAX(" + Vi + ") FROM " + tabla + ") *
" +             Valor2i + "/100"
    fin si
    si el tipo de restricción es RANGO entonces
        si s1=s2 entonces
            q ← q + Vi + " = " + s1
        sino
            q ← q + Vi + " BETWEEN " + s1 + " AND " + s2
        fin si
    si el tipo de restricción es MAYOR entonces
        q ← q + Vi + ">=" s1
    fin si
    si el tipo de restricción es MENOR entonces
        q ← q + Vi + "<=" s1
    fin si
    si i ≠ n entonces
        q ← q +  " AND "
    fin si
fin para
```

donde :
V_i : Nombre de la variable i
tabla : Nombre de la tabla en la base de datos que contiene la información censal.
n : Numero de variables seleccionadas
$Valor1_i$: Valor introducido por el usuario para indicar el inicio de rango, o el valor de las otras restricciones.
$Valor2_i$: Valor introducido por el usuario para indicar el final del rango.
Al final la cadena q es la que contiene la sentencia SQL completa.

El algoritmo anterior fue implementado en Visual Basic mediante dos funciones:

subQuery que regresa en una cadena de texto la consulta correspondiente a la restricción numero index impuesta a la variable contenida en varRes y getQuery que regresa una cadena de texto con la consulta SQL correspondiente a todas las restricciones. Dicho de otra forma, getQuery une con un AND las subconsultas generadas con subQuery y la regresa. Este modulo de restricciones puede verse en [Bernábe06].

A 3.5 Módulo de interfaz gráfica

Este modulo muestra los resultados gráficos en mapas sobre el particionamiento independientemente de que se seleccionen o no las variables que intervienen en el particionamiento.

Al igual que el modulo de consultas, este subprograma de interfaz gráfica se diseño de forma general, de tal modo que se puede adaptar fácilmente a otras aplicaciones realizadas en Visual Basic.

La información de los mapas proporcionados por el INEGI de los diferentes estados se hicieron con el software GIS MapInfo, por tanto, para poder manipularlos y obtener información de estos mapas, es necesario encontrar una herramienta adicional, o bien, convertir los mapas a otros formatos.

MapInfo, el sistema de información geográfica que se ha integrado, proporciona un control para manipular las capas (layers) creadas con su software. Este control llamado MapX (map-x) es el que se utilizo para este propósito.

MapX es un componente ActiveX que permite integrar la funcionalidad de MapInfo en una amplia variedad de aplicaciones nuevas y pre-existentes, para resolver necesidades específicas. Se integra usando lenguajes de programación estándar, tales como Visual Basic, Visual C++, Delphi, PowerBuilder y Oracle Express Objects. Esto fue la razón de utilizar MapX.

Los componentes de este modulo son dos formas: *frmMapXInterfaz* y *frmClusterProperties*, un modulo de clase Color y dos archivos de modulo: *modFuncMap* y *modMap*. Color y *modFuncMap* son independientes del modulo, al igual que los dos módulos anteriores se crea una instancia de la clase Color, la cual si es dependiente del módulo [Zamora06].

La figura 8 muestra la interacción entre los componentes de este módulo.

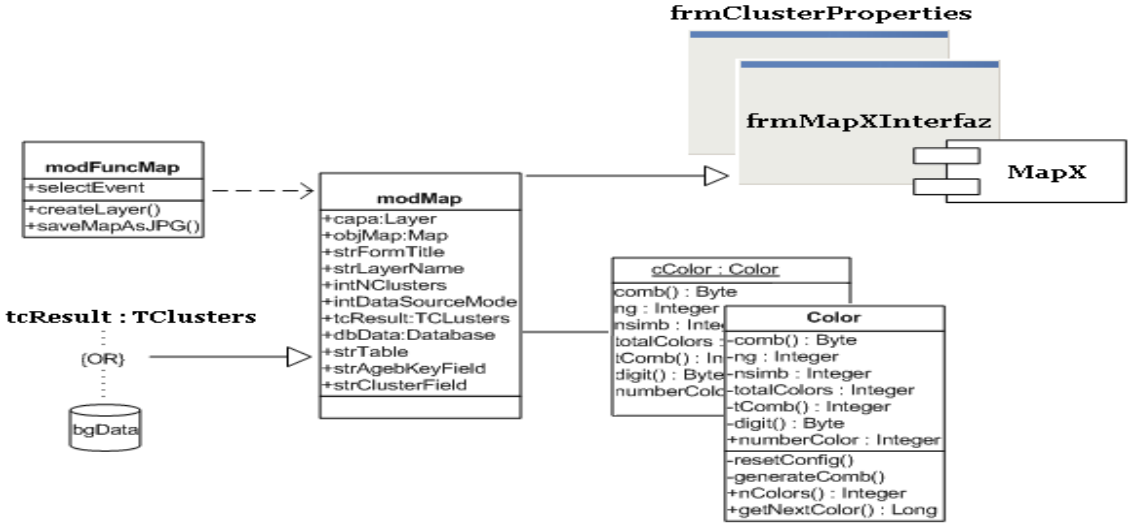


Fig. 8. Módulo de interfaz gráfica.

El punto de entrada a este modulo es un llamado a la forma *frmMapXInterfaz*, la cual, de acuerdo a los valores de las variables pasadas como argumentos, colorea determinadas zonas de un estado y muestra el mapa correspondiente en una ventana. La forma *frmconglomeradoProperties* tiene diversas opciones que le permiten al usuario cambiar la manera de cómo se muestra la información grafica en la ventana, se puede cambiar el color de los grupos, así como mostrar u ocultar algunos de estos.

El propósito de *modFuncMap* es concentrar todas las funciones generales a un control MapX, mientras que *modMap* tiene propiedades y métodos que interactúan sobre la información gráfica a ser mostrada en *frmMapXInterfaz*.

A continuación se muestra la declaración de los parámetros de entrada y respectiva descripción.

```
Public Const DATABASE_MODE As Integer = 1
Public Const TCLUSTER_MODE As Integer = 2

Public formTitle As String
Public layerName As String
Public nClusters As Integer
Public dataSourceMode As Integer
Public tcResult As TClusters
Public dbData As DAO.Database
Public strTable As String
Public strAgebKeyField As String
Public strClusterField As String
```

formTitle. Texto que va a contener la ventana.
layerName. Nombre de la capa (layer) de MapInfo a ser mostrada.
nClusters. Numero de grupos de zonas a ser mostrado, por lo que se generaran nCluster tonos diferentes para cada grupo utilizando la clase Color.
dataSourceMode. Especifica el modo en que se van a introducir los argumentos, tiene dos modos y dependiendo del que se escoja se tendrán que especificar los siguientes parámetros:

1. si *datasourceMode* = TCLUSTER_MODE

    tcResult. Objeto que contiene los agebs y el grupo al que pertenece cada uno.

2. si *datasourceMode* = DATABASE_MODE

    dbData. Referencia a la base de datos que contiene la información
    strTable. 'Nombre de la tabla en donde se encuentran los datos
    strAgebKeyField. Nombre del campo que contiene a las claves de los agebs
    strClusterField. Nombre del campo que contiene la pertenencia de grupo de cada ageb

La descripción del tipo de dato *TClusters* es una opción para transferir información de la clasificación.

Por cada grupo (*nCluster*) se llama al método *createLayer* definido en *modFuncMap*, el cual crea una capa (*Layer*) con el color pasado como argumento (el cual es generado con la clase Color para no repetir tonos entre los grupos) y la superpone en la capa actual (*layerName*). Esta forma de mostrar los grupos en el mapa facilita el mostrar y ocultar los grupos rápidamente.

Las operaciones como el zoom, el colocar texto en el mapa, figuras, cambiar el estilo del texto, copiar el mapa al portapapeles y demás funcionalidad que permite frmMapXInterfaz se facilita mediante el uso del modelo de objetos de MapX.

A 3.6 Integración de los módulos

Cuando se cuenta con los módulos terminados (de clasificación, consultas etc.), es posible integrarlos para que el sistema se ejecute en un solo componente.
Todos los algoritmos implementados funcionan por separado, la comunicación se hace a través de los parámetros, muy conveniente para los algoritmos implementados pero no le resta funcionalidad, es igual de fácil uso y acceso que si estuvieran integrados en un solo componente.

Sin embargo, para que todo lo desarrollado se ejecute como un sistema integral, se ha propuesto un sistema de creación de ventanas (*forms*) con dos archivos *modProject* y *modDatos*, con los cuales es posible integrar módulos en una sola aplicación.

Los componentes modProject y modDatos se encargan de definir variables con información de lo que el usuario va realizando y métodos que acceden a las bases de datos por lo que este se envía la información adecuada a los módulos.
En la figura siguiente figura se puede apreciar los diferentes accesos entre módulos y los llamados a procedimientos a través de los parámetros exactos. La palabra "cat" se refiere a la tabla catálogos de la base de datos principal.

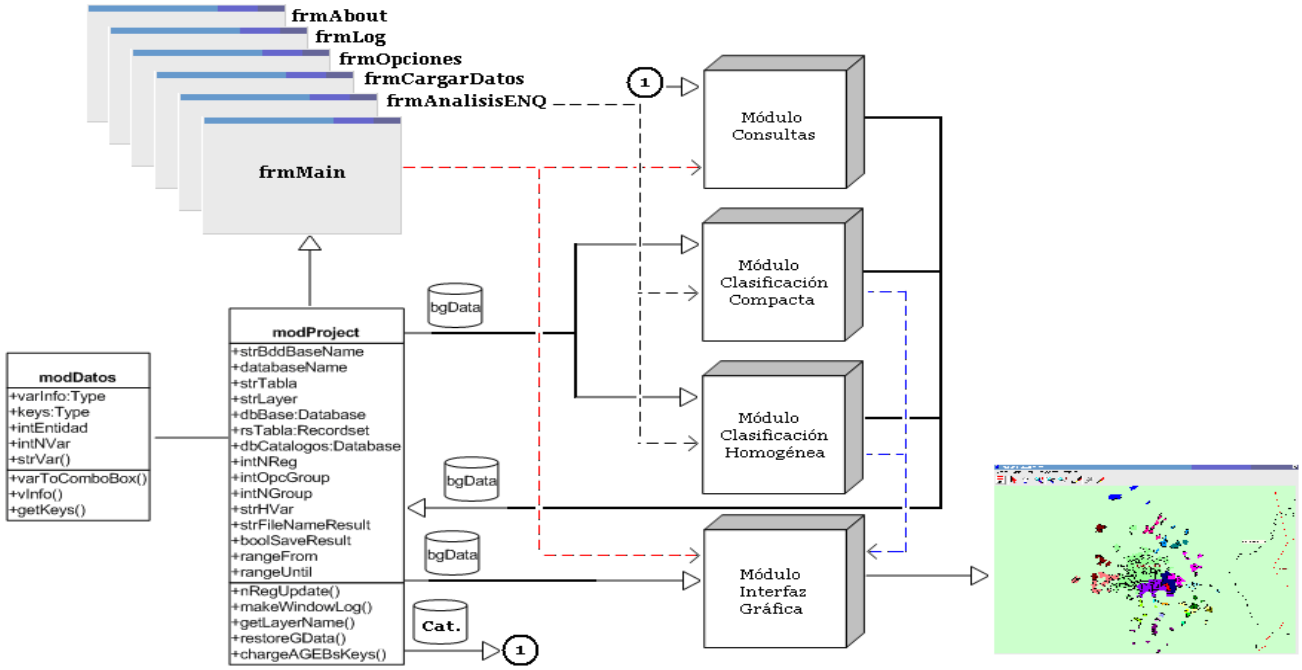


Fig. 9. Integración de todos los módulos en la aplicación final. La línea sólida especifica entrada y salida, mientras que la línea punteada define las llamadas entre componentes.

El proceso de integración de ZO con el map-x, se ha apoyado del trabajo [Zamora06].

Anexo B. Propuesta de un algoritmo de clasificación en paralelo sobre redes complejas aplicado a las agebs

Este anexo presenta el documento inédito de la publicación realizada en [Ochoa09]. Este trabajo se ha desarrolado en conjunto con dos investigadores del Instituto de Cibernética, Matemática y Física ICIMAF de la universidad de la Havana Cuba.

El propósito de este trabajo ha sido contar con una alternativa de agrupación sobre datos espaciales. La alternativa es aplicar la herramienta de intermediación de aristas, la cual es eficiente y conocida en el área de redes complejas.
Sin embargo, el costo computacional es alto y para manejar este aspecto, se ha implementado el algoritmo en paralelo.
Finalmente, se tiene planeado como trabajo futuro, desarollar un conjunto de pruebas a diferentes casos de distribución poblacional de las agebs. Por otro lado, con el fin de comparar resultados, se realizarán las mismas pruebas aplicando el método que resuelve ZO.

Towards a Parallel System for Demographic Zonification Based on Complex Networks

Alberto Ochoa[1] & Beatríz Bernábe[2] & Omar Ochoa[3]

ABSTRACT

This paper presents a novel method for the zone design problem that is based on a popular technique in the field of complex networks research: betweenness centrality. We use a parallel version of a community detection algorithm that outputs the graph partitioning that maximizes the so-called modularity metric. The method is put at the centre of an effort to build an open source interactive high performance computing platform to assist researchers working with population data.

RESUMEN

Este artículo presenta un método novedoso para el problema de diseño de zonas, que se basa en una técnica muy popular en el campo de las investigaciones de redes complejas: la intermediación de aristas. Se utiliza una versión paralela de un algoritmo de detección de comunidades que retorna el particionamiento del grafo que maximiza la tal llamada métrica de modularidad. El método se pone en el centro de un esfuerzo en pos de la construcción de un sistema de código abierto para computación interactiva de alto rendimiento que asista a los investigadores que trabajan con datos demográficos.
KEYWORDS: parallel algorithms, zone design problem, betweenness centrality, clustering, interactive high performance computing

1. INTRODUCCTION

The ultimate goal of the present work is the creation of a system to assist researchers that investigate population data. The social and economic impact of dealing with this kind of information can be assessed from the following sources [1, 2, 3].
Specifically, we address the problem of demographic zonification, which is understood here as the problem of clustering geographical units (GU) according to some measure of demographic similarity. Each GU is a small geographical area for which the following is known: latitude, longitude, radius and a vector of demographic features (statistics) computed over the population living in the area. A cluster in the intended solution is expected to be compact, which means that each GU is geographically close to a large fraction of the GUs in the same cluster. Besides, the cluster should be homogeneous as to the similarity measure being used, meaning each GU has rather high similarity values with many members of its cluster. In other words, the clustering can be seen as a two-objective optimization problem or as a single objective one with the restriction of getting compact solutions. In this paper we take the later approach.

[1] Institute of Cybernetics, Mathematics and Physics. ochoa@icmf.inf.cu
[2] UNAM, Systems Department & BUAP School of Computer Science. bety@cs.buap.mx
[3] Institute of Cybernetics, Mathematics and Physics. omar@icmf.inf.cu

Theoretically speaking, solving the above defined clustering problem is challenging. In terms of computational complexity the zone design problem has been shown to be NP-Complete [4]. One of the reasons why this problem is especially difficult is the size of the solution space. The dimension of most real world problems makes finding an exact solution unfeasible. Thus, heuristic techniques seem to be the best way available to produce solutions in a reasonable computational time. Unfortunately, the heuristics that have been used so far have also a high computational cost [5, 6] and often do not provide good solutions. It is worth noting that approaches based on classical statistics alone cannot solve the problem either [7]. We believe it is necessary to apply a combination of heuristics and statistical methods to obtain good results. In this paper we explore new ideas and develop better algorithms, including parallelism as an option.
To make things even more challenging we would like to create an open source tool for assisting *in situ* people that may have no or limited knowledge of parallelism and lack the necessary software and equipment. These people work interactively with population data, which means that they may change the set of features that describe the GUs, the geographical regions of interest or the metrics. Besides they need statistical tools for analysing their results. The system must fulfil all these requirements. This paper is a first step towards the construction of such a tool.

The method we have developed in the paper is motivated by results in complex network research [8, 9, 10, 11, 12]. Basically, what we do is to map the problem to an unweighted graph and then apply a community detection algorithm -based on edge betweenness centrality- that produces the desired partitioning of the GUs. However, the mapping from the distance and similarity matrices to the unweighted graph may not be so straightforward as far as a lot of weight information has to be codified in the graph topology.
The outline of the paper is as follows. Section 2 is devoted to a detailed exposition of the proposed method. The method is applied to the clustering of geographic units according to given demographic information, but with the restriction of geographic closeness. In Sect. 3 a hint is given about how to apply the method to the clustering of the demographic variables. Sect. 4 presents a short overview of the first steps given in the design and building of an open source interactive high performance computing platform to assist researchers working with population data. Sect. 5 shows some numerical results that illustrates the method in action. Finally the conclusions are given.

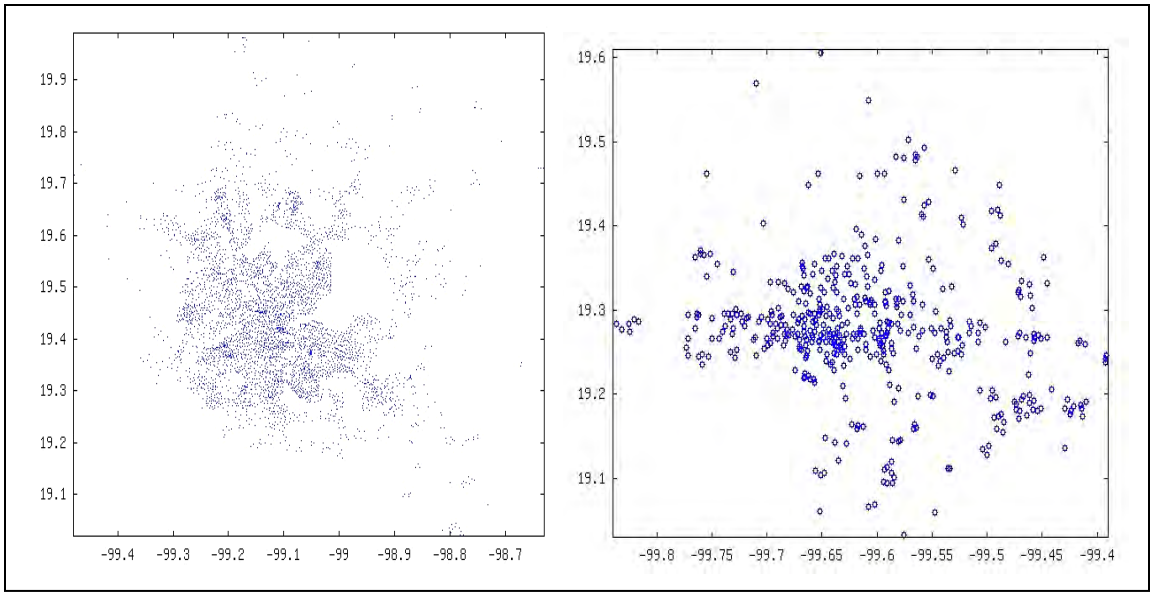


Figure 1: Geographical distribution (latitude vs. longitude) of 4292 and 466 GUs of the metropolitan areas of the valleys of Mexico (left) and Toluca respectively. A smaller marker size has been used in the first case.

2. SPATIALLY CONSTRAINED DEMOGRAPHIC CLUSTERING

Throughout the paper we shall use a simplified model of the zone design problem. Our goal is just to show that a method based on network partitioning is suitable for the problem. According to the model the geographic units are assumed to be points.

As it was said, the problem of demographic zonification is understood in this paper as the problem of clustering geographical units according to demographic similarities. Figure 1 shows the distribution of a set of such units for two areas of Mexico: the metropolitan areas of the valleys of Toluca and Mexico. We shall use both areas in our experiments, but the smaller region (Toluca) has been chosen as the main case study throughout the paper.

The main ideas of our algorithm, described next, are based on observations and conclusions drawn from research in complex networks. Indeed, we transform the problem of clustering the GUs according to two measures: demographic similarity and Euclidean distance, into the problem of community detection in an unweighted graph. As an alternative we could also work with a weighted graph, which would give a more straightforward representation of our heavily weighted problem. However, as far as dealing with the unweighted case is less expensive from the computational point of view we accepted the challenge of mapping the weight information into the structure of an unweighted graph.
One important decision to be made before the algorithm starts is the selection of a metric to measure the demographic similarity between two GUs.

Algorithm 1 Our algorithm

Assume as given a similarity measure and an input matrix where rows codify the geographic units (GUs).

Phase I Selection of variables, geographic units and the processing space.

Phase II Computation of the set of closest neighbours of each GU, according to the distance information.

- Step 1 Computation of the Delaunay triangulation of the set of GUs. This creates the triangulated graph, G_t, where each node represents a GU.
- Step 2 Edge and node elimination in G_t.
- Step 3 To every node, *i* in G_t, assign the set of its closest neighbours $Neigh_d(i)$.

Phase III Computation of the similarity graph, G_s

- Step 1 For every node, *i*, compute its pairwise similarity with every member of $Neigh_d(i)$.
- Step 2 For each node, *i*, construct the set, $Neigh_s(i)$ (s stand for similarity), with the *k* most similar neighbours taken among the members of $Neigh_d(i)$
- Step 3 Add the edge $i \sim j$ to G_s if $j \in Neigh_s(i)$ and $i \in Neigh_s(j)$

Phase IV Parallel computation of the edge betweenness centrality of every edge in G_s

Phase V Computation of the partitioning of G_s that maximize a given scoring metric after eliminating some of the edges with the highest betweenness values.

There exists a wide choice of metrics for this purpose, among them the Euclidean, cosine and correlation functions are the most popular. In this paper we use the Euclidean and cosine functions for clustering the GUs and the correlation function for clustering the demographic variables.
In what follows the details of the method, which is outlined in Algorithm 1, are discussed.

2.1 SELECTION OF VARIABLES, GUs AND THE SPACE

Let us assume that an input data matrix is given, where each row contains information about one of the GUs, whereas the columns represent the demographic and geographic variables. There are many reasons that make convenient to work with a subset of the variables and/or a subset of the available GUs.

The selection of one or another group of variables for processing gives different perspectives on the data, which often is useful in interactive exploratory data analysis. Besides, the reduction in the number of variables decreases the arithmetic complexity of the computation of the similarity function. It is worth noting, that in our case this gain is not so important because as it will be shown below, our method makes a linear amount of pairwise similarity computations with respect to the number of GUs.

Based on the above said we can conclude that any automatic variable selection method can be plugged-in in the Phase I of the algorithm. Another kind of techniques that can be utilized at this stage is the transformation of the space where the data reside, which involves principal component analysis and other projection algorithms. It is expected that the transformation will lead to the computation of better similarity values as far as they will be more robust in the presence of noise.
From the perspective of their geographical positions, a subset of the available GUs may be chosen when we want to concentrate ourselves on the study of a particular area of the given region or when the computation of the whole region is expensive according to the available equipment. We shall present examples of both situations in the section dedicated to the experiments. Another very important perspective on the same issue arises when solving the clustering problem for a subset of GUs resulting from a complex logical database query with respect to the demographic variables.

In summary, Phase I is responsible for producing an input matrix that better fulfils the requirements of the clustering task at a given moment.

2.2. CREATING THE UNWEIGHTED SIMILARITY GRAPH

Recall that in our problem, a cluster in the desired solution is expected to be compact, which means that each GU is geographically close to a large fraction of the GUs in the same cluster. At the same time, the cluster should be homogeneous as to the similarity measure being used, meaning each GU has rather high similarity values with many members of its cluster. In other words, the clustering can be seen as a two-objective optimization problem or as a single objective one with the restriction of getting compact solutions. We take the later approach and as a first step create a graph based on the distance information.

Without loss of generality, let us assume that the second and third columns of the input matrix contain the latitude and longitude of each GU (these columns were not processed in Phase I). It is worth noting that we are not considering the radius, which means that all GUs are assumed to be geometrically equal with centres at the given coordinates. In some cases this assumption could be too strong and somehow misleading. However, it is adequate for the purposes of this paper as it is explained below.

The Delaunay triangulation (DT) is a sparse graph structure that codifies proximity relationships in a point set. Thus, the first thing we do in Phase II is the construction of the unweighted graph G_t -a DT of the set of GUs centres. The DT is connected to the Voronoi diagram of the centres, which is a

partition of the space into areas, one for each GU centre, so that the area for a GU centre consists of that region of the space that is closer to it than to any other GU centre. An edge in the DT connects two GU centres if and only if the Voronoi areas containing them share a common boundary. Hence, edges of the DT capture spatial proximity. At this point it is worth noting that for those cases where the actual area of each GU is included completely inside of the Voronoi cell of its centre, the DT seems to be adequate to model the distance restriction of our problem. Regarding computational efficiency, many O (n log n) time and O (n) space algorithms exist for computing the DT of a planar point set.

In the Step 2 we eliminate a certain amount of nodes and edges to get a further reduction of the graph (the first was performed with the input data in Phase I). Optionally, we eliminate GUs in regions of high density to reduce the cost of the algorithm. This is accomplished as follows. We compute a DT of the whole set and then select the GUs belonging to triangles which areas are not below the Q area quantile. A new triangulation is performed with the chosen points.

Despite the fact that edges of the DT capture spatial proximity there are some edges that might be far apart, for example the edges between the GUs in the convex hull. Therefore, we remove all edges larger than the largest one found in a set that contains for each GU the smallest among its attached edges. We term G_t the connected graph obtained after this procedure.

As the final step of Phase II we compute the set of closest neighbours of the i-th GU, $Neigh_d(i)$ (where the subscript indicates the distance measure). Considering the way G_t was constructed, it is reasonable to define as neighbours all GUs that can be reached in at most L steps from the given one. In fact, in all the experiments of this paper we use L = 2.

At this point the Phase III of the algorithm starts. The idea is simple, we formulate a k-nearest neighbours problem with respect to the similarity function. Fortunately, we only need to compute the similarity between each GU and its neighbours according to $Neigh_d(i)$ (Step 1). For each GU, i, construct the set, $Neigh_s(i)$ (s stand for similarity), with the k most similar neighbours taken among the members of $Neigh_d(i)$ (Step 2).

Algorithm 2 Community Detection by Girvan and Newman
While the network is not empty, Step 1 Recalculate the betweenness score of every edge. Step 2 Remove the edge with the highest score.

In the final Step 3 the unweighted graph, Gs, is constructed. The following holds:

edge $i \sim j \in G_s$ if and only if $(j \in Neigh_s(i)) \wedge (i \in Neigh_s(j))$.

It is our fundamental claim that in the topology of the unweighted graph G_s there is enough information about our complex clustering problem. If we assume that, and we do, it is reasonable to accept as clusters all the connected sub-graphs with high connection density. Therefore, what we need at this point is a method to detect these communities of nodes. This is a common problem in complex networks research.

2.3. EDGE BETWEENNESS CENTRALITY

The shortest path edge betweenness centrality, counts the number of shortest paths that run along an edge in a graph. The highest values of betweenness are likely to be associated with bridges-edges -those that separate regions of densely connected vertexes. Thus, by removing the bridges the communities can be detected. Unfortunately, when there are several bridges between two regions, they compete for the best scores, hampering an accurate evaluation of the measure. This

problem has been solved with Algorithm 2, hereafter called GN algorithm because was developed by Girvan and Newman [8, 10].

The fastest known version of the GN algorithm for unweighted, undirected graphs (m edges and n vertexes) has worst case time complexity O($m^2 n$), whereas for sparse graphs O(n^3). The high computational cost of the GN clustering algorithm has been an obstacle to its use on relatively large graphs. In fact, the computational cost of the algorithm is already prohibitive when the input is a graph with a few thousand edges. However, in [12] a parallel implementation of the algorithm was reported. As far as it allows users to analyse larger networks on a distributed cluster of computers, we decide to evaluate it in our problem.
This is accomplished in the Phase IV of the algorithm. The output at this stage is the order of removal of the edges, which implicitly defines a hierarchical tree. Each node of the tree is a subset of GUs.

2.4. PARTITIONING THE SIMILARITY GRAPH

The last Phase of our algorithm is responsible for deciding where to cut the hierarchical tree to determine the clusters. There are different ways to accomplish this task. Here we use a recently introduced quality measure for graph clustering called modularity [10]. A high modularity indicates that there are more intra-cluster and less inter-cluster edges than would be expected by chance. Note that this differs from partitioning a graph minimizing the number of inter-cluster edges.
Assuming that a_{ij} represents the fraction of all edges that link the vertices in cluster i to the vertices in cluster j, the modularity is then defined as

$$0 \leq Q = \sum_{i=1}^{K} \left(a_{ii} - \left(\sum_{j=1}^{K} a_{ij} \right)^2 \right) \leq 1$$

The higher the value, the stronger the clustering structure in the network. For a random decomposition Q approaches 0.
The version of the algorithm reported in this paper outputs the clustering with maximum modularity and the ordered list of edge removals. Optionally, the user can take the list of edges and, by gradually removing them from the graph, he or she can create a merge matrix or dendrogram from which a different clustering can be explored.

3. CLUSTERING THE DEMOGRAPHIC VARIABLES

The very same complex networks approach can be utilized to cluster the demographic variables before and after running Algorithm 1. In the first case, the aim is the selection of variables and GUs in Phase I, whereas in the second it is helpful for labelling the clusters of GUs.
Here we use 1 - *corr*(X, Y) as a measure of similarity between the variables X and Y. Note that *corr* stand for correlation (Pearson or Spearman) and no missing values are allowed to ensure *corr*(X, Y) ≤ 1. Strong correlated variables are assumed to belong to the same cluster unless the corresponding edge has a high betweenness value, which is a sufficient condition for separating variables.

The unweighted similarity graph is constructed by attaching to each variable its k-most similar variables and adding the edge ‹X, Y›, if the relation holds in both directions. Once the graph is built the phases IV and V of Algorithm 1 can be executed.

4. A PARALLEL SYSTEM FOR ZONIFICATION

The term interactive high performance computing (IHPC) has to do with a system's ability to provide access to parallel hardware to run programs written in a very high language. The most common example is to accelerate Matlab by running the programs in a computer cluster. Most IHPC systems

come with several mathematical libraries. We are interested in statistical and complex networks libraries. ViPoC (Virtual Interactive Portable Cluster [13, 14]) is one experimental IHPC that fulfils our requirements.
We are building an experimental open source parallel tool for demographic zonification including a priori and posterior statistical analysis.
We pursue five aims: 1) allow rapid prototyping via high level languages like R and Matlab; 2) large collection of statistical and machine learning tools; 3) cheap parallel infrastructure; 4) smooth integration with existing Windows Matlab technology; and 5) web interface with the system. ViPoC provides all these things.

4.1. A SHORT OVERVIEW OF VIPOC

ViPoC is a flexible portable cluster because it can be used in a laptop, in a small office network or in a big organization managing a large cluster. It controls two sub-networks, one is formed by dedicated nodes behind a switch and the other may have a mixture of dedicated and no dedicated nodes -some of them may be virtual- located at the intranet of the organization. Therefore, it is easy to build a low budget cluster.
ViPoC supports the R and Octave languages, among many others. The later is a clone of Matlab, whereas the former has many statistical and machine learning functions ready to use. This is an important characteristic for the system we want to build: a large choice of statistical methods can be used in the a priori and posterior analysis of the zone design problem.
ViPoC comes with three so-called IHPC servers. They provide socket access to Octave, R and the system itself. For example, from Matlab running in the user's desktop computer it is possible to issue the following commands:

callViPoC (`command', vipocIP) (1)
callViPoCOctave (`command', ListArgs, vipocIP) (2)
callViPoCR ('command', ListArgs, vipocIP) (3)

where vipocIP is the ip-address of the ViPoC server and ListArgs is a list of arguments that depend on the concrete command. For example, the call

callViPoC('mpirun -np 4 pgn AGEBs.pebc > getclusters.m', vipocIP)

executes in 4 processors the parallel program pgn. This program is a modification of the code reported in [12]. It takes the file AGEBs.pebc (the list of edges in Gs) as input and writes to the standard output the results (in form of a Matlab program) which is redirected to the file getclusters.m.
In ViPoC the working directory in the user's desktop computer can be mapped to the user's working directory in the cluster (via the Samba protocol). Therefore, the Matlab function getclusters.m can be called from the user's Matlab program, obtaining in this way the computed clusters.

Let us assume that the matrix AGEBdata is a Matlab variable in the user's desktop computer that contains the geographic and demographic information of each GU. That matrix is usually the input to algorithm 1, but sometimes we may want to do some pre-processing. Figure 2 presents a fragment of code that computes the principal components of the matrix, calling the R server of ViPoC (see [13] for syntactic details).

```
1 if Preprocessing

2  callViPoCR ( ' open ' );

3  callViPoCR ( ' assign ', 'X ' , AGEBdata( : , 4: end ) );

4  callViPoCR ( ' eval ', ...

5                sprintf ( ' pc <- prcomp(X, scale=TRUE, ...
```

Figure 2: Fragment of Matlab (or Octave) code that uses the ViPoC R server to compute the principal component of the matrix AGEBdata (each row represents one GU). Although we have removed the ip-address for the sake of space, this code can be executed from the user desktop computer.

This fragment of code corresponds to Phase I of algorithm 1. Notice that the same approach can be utilized after running our algorithm to do some statistical post-processing using R.
At this point it should be clear for the reader that our approach to the construction of the system consists in using different languages and systems to maximize reusability. In the current version we proceeded as follows. Phase I is implemented following the ideas outlined in Figure. 2. In the step 1 of Phase II, a very efficient C++ program obtained from the Internet was used. The remainder of Phase II and Phase III was programmed in Matlab (Octave). This part of the program can be run in the user's desktop computer in Matlab or Octave, or completely in the cluster in Octave. The parallel computation of Phase IV is accomplished as explained above; it returns the best clustering according to the modularity measure. Other reasonable good clustering can be obtained, again via the ViPoC R server, using the complex network package igraph [15]. The final post-processing is accomplished in the same way. It is worth noting that R also has the capacity to run parallel code.

Before we conclude this short presentation of the use of ViPoC as a computational platform for our system we once again stress the following fact. We are interested in providing a system that researchers and practitioners can use in their available hardware: from a single laptop to a large cluster. In the later case, it is also advisable to have a web interface for remote access. ViPoC provides this option.

5. EXPERIMENTAL RESULTS

The aim of this section is simply to illustrate the possibilities of our method. Let us take the data for the metropolitan area of the valley of Toluca. We study 466 GUs described with a vector of 181 demographic variables according to the XII National Population Census [16].
In the first experiment we use all GUs and variables (without any pre-processing) and the Euclidean distance as the demographic similarity measure. The distance graph was built as explained above using two levels of the triangulated graph. For the construction of the similarity graph we took the 15 most similar GUs, which produced a sparse graph with 2937 edges. The algorithm found the nine clusters shown in Fig. 3 (bottom).

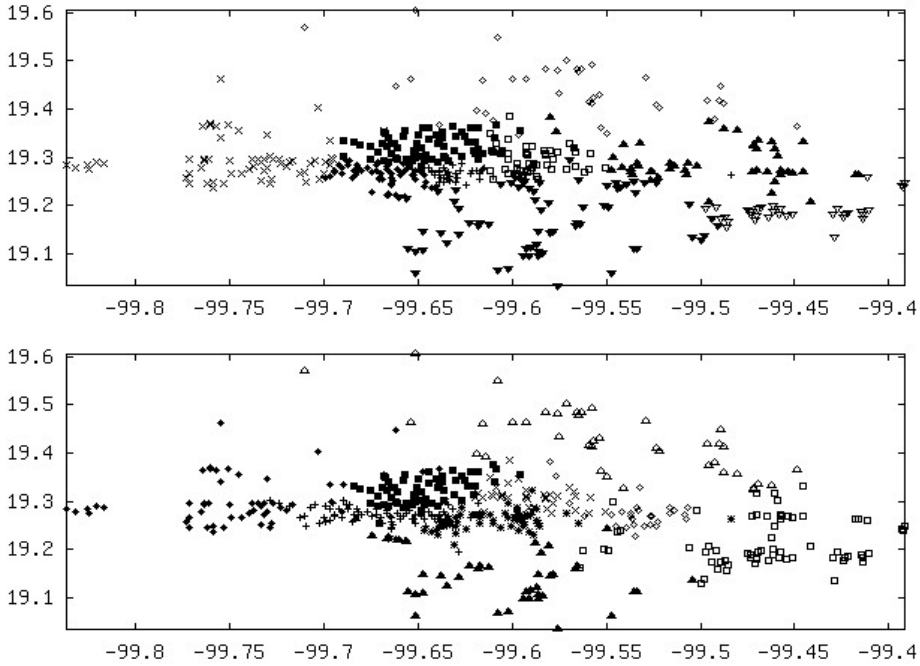


Figure 3: Clustering results for Toluca using the whole set of 181 demographic variables. The top figure shows a clustering where only geographic distances were used, whereas for the bottom figure the Euclidean distances between the vectors of demographic variables

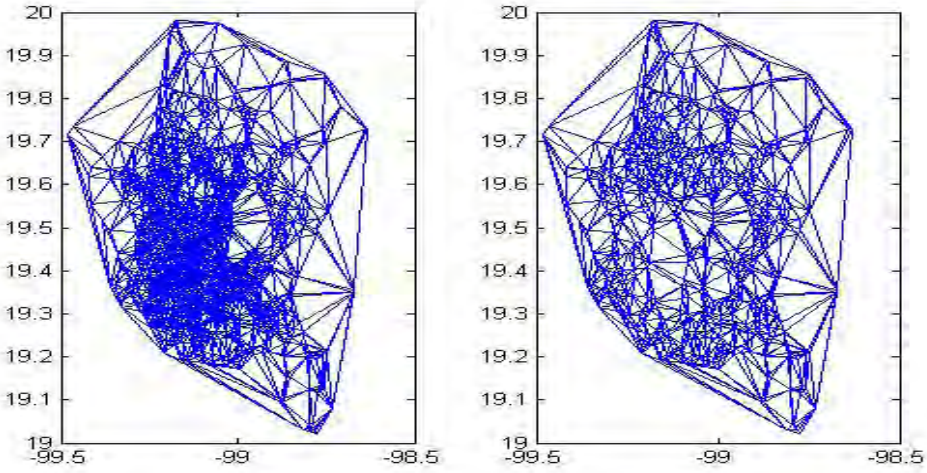


Figure 4: Reducing the number of GUs for more efficient yet approximate clustering. (left) Delaunay triangulation of 4292 GUs in the Metropolitan Valley of Mexico. (right) Triangulation of the GUs belonging to triangles, whose areas are not below the 0.90 area quantile. The number of GUs drops to 836 Gus.

For comparison purposes, in Fig. 3 (top) we show the clustering obtained using exclusively the geographic distances. To accomplish this, we applied the betweenness method to the distances' graph, which has 4289 edges. The important thing to notice is that the bottom partitioning can not be trivially obtained from the top one.

For example, the cluster represented by squares at the bottom of the figure contains GUs from three different clusters of the top clustering (three different triangles). Note that only one of these clusters is completely included in the clusters of squares. Thus, the idea of clustering with one measure and then with the other does not seem to be a good one. Our method achieves a reasonable compromise between the two measures, because the first one is utilized to constraint the solution space in such a way that the optimization of the second does not violate the requirements of the first one.

When dealing with larger problems like the one presented in Fig. 1 (left) the issue of obtaining a response in a reasonable computational time arises.

In an interactive scenario, like the one we are trying to construct, there are different heuristics that can be applied. Just to give an example we have shown in Fig. 4 a simple method that consists in reducing the number of GUs for more efficient yet approximate clustering. This is accomplished by doing a two step triangulation. Only the GUs belonging to triangles which areas are not below the 0.90 area quantile in the first triangulation are considered in the second triangulation. Note that the triangulation algorithm has complexity O (n log n). In the example the number of considered GUs drops from 4292 to 835.

A second strategy, which of course can be combined with heuristics, is the use of parallel or distributed programs. The size of the manageable problem will depend on the available computational power.

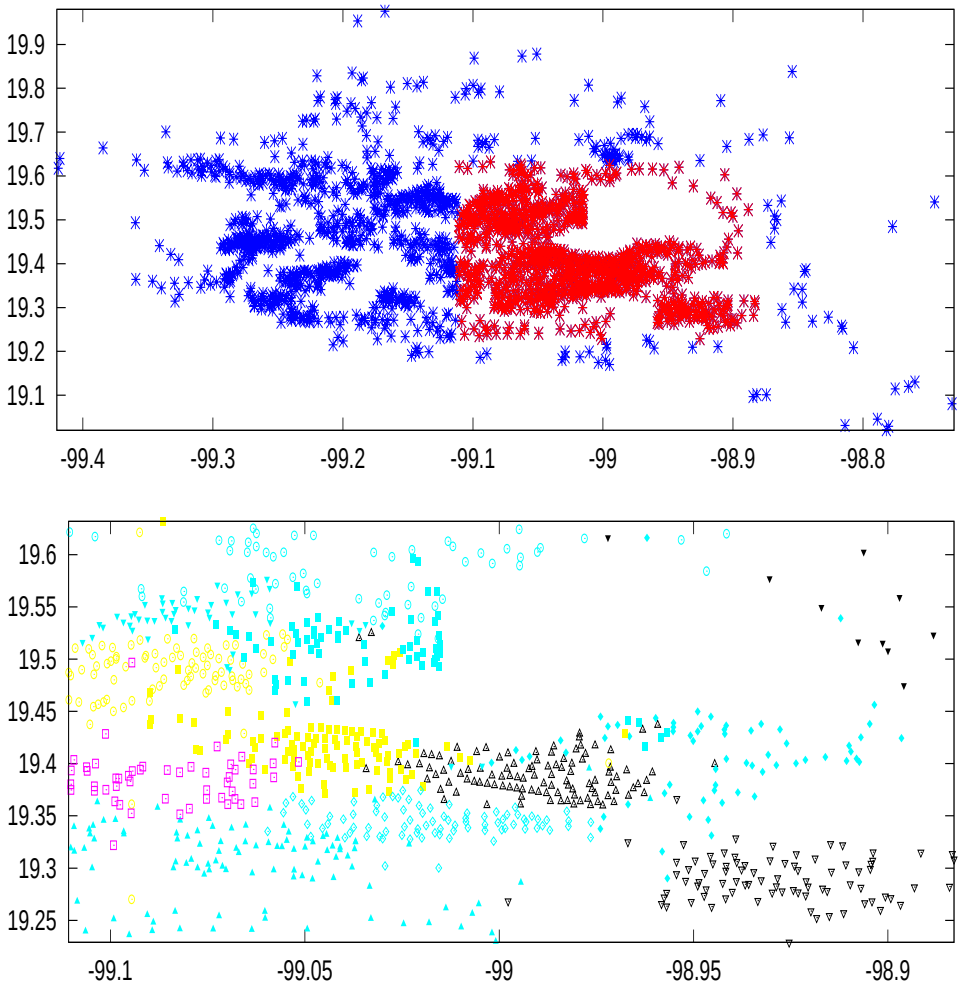


Figure 5: Clustering results for a rectangular area in the valley of Mexico. All demographic variables were used. The plot at the top shows in red the chosen area whereas the bottom plot presents the obtained clusters.

Table 1: Time (seconds) for different number of processors. Parallel computation of the best partitioning of a similarity graph with 875 vertices and 5573 edges, constructed for the rectangular area shown in Fig. 5.

Processors	1	2	3	4	5	6	7
Time (secs)	854.14	689.75	524.19	498.61	476.72	486.17	525.06

Nevertheless, we always can restrict ourselves to the analysis of small areas as far as we are working in an interactive scenario. For example, in Fig. 5 (top) the rectangular area that has been chosen for processing is shown in red.
We use ViPoC to solve this problem in parallel. Table 1 shows the results of a small experiment. We took four clients with P-IV 2.4 GHz processors supporting hyper-threading technology in a 800 MHz bus and 512 MB of RAM. The clients were connected through a Fast-Ethernet network switch to a laptop where ViPoC was running. The table shows the time to complete the Phases IV and V of our algorithm.
The data to be clustered is the rectangular area mentioned above: 875 Gus and 5573 edges. Notice that an approximate time reduction from 14 to 8 minutes is achieved when 5 processors were used.
This experiment has shown an unexpected and interesting issue. Notice that starting from six processors the computational time increases. This is in contrast with the linear speedup of up to 32 processors published in [12]. We have found that this algorithm does not scale well with fast-ethernet networks. We assume that the reported results were obtained with much faster network technologies. The problem is that the implementation of the betweenness algorithm (due to the known recalculation step [8]) needs a certain amount of inter-processor communication that becomes prohibitive already for six processors in slow networks. This is a great obstacle for the purposes of our project because we are specially targeting low cost hardware. Fortunately, in [11] a method (differential betweenness) whose parallel implementation does not require such a heavy inter-processor communication has been introduced. We will study this method in a forthcoming paper.

7. CONCLUSIONS

In this paper we have reported the results of an ongoing project aimed to build an open source system for dealing with the zone design problem. In particular, we are interested in demographic data.
The novelty of our approach is based on the use of a graph partitioning technique popular in the area of complex networks research: community detection by using the betweenness centrality metric. We have shown that even a simplified model (we map the problem to an unweighted graph in contrast to a weighted one) is able to capture enough information about the clustering structure of the data.
We are building the system on top of an interactive high performance computing platform that acts as an interface between parallel hardware of different computational power and the problem software solution. This strategy should make the system useful for researchers and practitioners with low budget and at the same time should also be able to control a large cluster. What is important is that in both scenarios the user sees the same enviroment.
Our results confirm two things: 1) the proposed method is a promising alternative to previous efforts in this area; and 2) the viability of the construction of an open source system with these purposes.
We believe that our approach can be used with similar results in other zone design problems.

REFERENCES

[1] Bernábe B., Desarrollo de un modelo para la determinación de zonificación óptima. PhD thesis, División de Estudios de Postrado de la Facultad de Ingeniería DEPFI, UNAM, Investigación de Operaciones, 2006. Proyecto de tesis doctoral.
[2] Duque, J. C., Church, R. L., and Middleton, R. S., Exact models for the regionalization problem, In Western Regional Science Association Annual Meetings, Santa Fe 2006.
[3] Juan Carlos D., Raúl Ramos, and Jordi Suriñach., Supervised Regionalization Methods: A Survey, Vol. 30, International Regional Science Review 2004, pp. 195-220.
[4] Pizza E., Murilo A., Trejos J., Nuevas Técnicas de Particionamiento en Clasificación Automática, Revista de Matemáticas Teoría y Aplicaciones, ISSN 1409-2433, 1999, pp. 51-66.
[5] Bacao F., Lobo V., Painho M., Applying Genetic Algorithms to Zone Design, Soft Computing - A Fusion of Foundations, Methodologies and Applications, Vol. 2, Issue 5, ISSN:1432-7643, Springer-Verlag Heidelberg 2005, pp. 341-348.
[6] Bernábe L. B., Ramírez R. J., Espinosa R. J., Evaluación de un algoritmo de recocido simulado con superficies de respuestas, Revista de Matemáticas Teoría y Aplicaciones, Vol. 16, No. 1, ISSN 1409-2433, Enero/Julio 2009, pp. 159-177
[7] Bernábe B. and López R., Proceso de Análisis de Correlaciones para Identificar Patrones de Asociación Sobre Datos Censales, Research on Computing Science, ISSN 1665-989, CIC – IPN 2005.
[8] M. Girvan and M. E. J. Newman., Community structure in social and biological networks, Vol 99, arXiv:cond-mat/0112110 v1, In National Academy of Sciences, December 2002, pp. 7821-7826.
[9] M. E. J. Newman., Analysis of weighted networks. Physical Review E, 70(056131), Nov 2004. arXiv:cond-mat/0407503 v1, 20 Jul 2004.
[10] M. J. E. Newman., Fast algorithm for detecting community structure in networks, Physical Review E, 69(066133), 2004. arXiv:cond-mat/0309508 v1, 22 September 2003
[11] A. Ochoa and L. Arco., Differential Betweenness in Comples Networks Clustering, In Progress in Pattern Recognition, Image Analysis and Applications, 13 Iberoamerican Congress on Pattern Recognition, Havana Cuba, Springer Verglag LNCS 5197, September 2008, pp. 227-234.
[12] Qiaofeng Yang and Stefano Lonardi., A parallel edge-betweenness clustering tool for protein-protein interaction networks, Int. J. Data Mining and Bioinformatics, 1(3), 2007.
[13] O. Ochoa., ViPoC una nueva herramienta de software libre para computación interactiva de alto rendimiento, PhD thesis, ICIMAF. Havana. Cuba, adviser: Alberto Ochoa, 2009.
[14] Omar Ochoa Rodríguez and Alberto Ochoa Rodríguez., ViPoC: un clúster virtual interactivo y portátil, Revista Ingeniería Electrónica, Automática y Comunicaciones, Vol. 27, No. 3, 2007.
[15] Gabor Csardi and Tamas Nepusz., The igraph software package for complex network research, InterJournal Complex Systems 1695 2006.
[16] Instituto Nacional de Estadística, Geografía e Informática (INEGI). http://www.inegi.gob.mx.

.

Bibliografía

[Aarts89] Aarts, E. y Korst, J.: Simulated Annealing and Boltzmann Machines, Wiley (1989).

[Aldenderfer84] Aldenderfer, M. S., Blashfield, R K.:. Cluster Analysis. Series: Quantitative Applications in the Sociai Sciences. California Sage Publications, Inc. (1984).

[Anja96] Anja Struyf & Mia Hubert & Peter Rousseeuw: Clustering in an Object-Oriented Environment. Journal of Statistical Software, American Statistical Association, vol. 1(i04) (2006).

[Altman97] Altman, M.: The Computational Complexity of Automated Redistricting: Is Automation the Answer? Rutgers Computer and Technology Law Journal, 23(1) 81-142 (1997).

[Anderberg73] Anderberg, M. R.: Cluster analysis for applications. New York Academic Press. (1973).

[Bação05] Bação, F., Lobo V., Painho, M.: Applying genetic algorithms to zone design. Springer Verlag, Soft Computing - A Fusion of Foundations, Methodologies and ApplicationsEditor Springer Berlin / Heidelberg, ISSN 1432-7643 volume 9 number 5 341-348 (mayo 2005).

[Bação05a] Bação, F., Caeiro, S., Painho, M., Goovaerts, P., Costa, H.: Delineation of estuarine management units: Evaluation of an automatic procedure. Libro Geostatistics for Environmental Applications Editor Springer Berlin Heidelberg, ISBN 978-3-540-26533-7, pp. 429-442 (2005).

[Barr95] Barr, R. S., Golden, J.P., Resende, M. G., W.R. Stewart W.R.: Designing and Reporting on Computational Experiments with Heuristics Methods. Journal of Heuristics, Kluwer Academic Publisher 9–32 (1995).

[Bailey94] Bailey, K. D.: Typologies and Taxonomies.: An introduction to Classification Techniques. London: Sage Publications, Inc. (1994).

[Blanco07] Blanco, V.: Bases de Gröbner Parciales y programación combinatoria multiobjetivo. IV Seminario de Geometría Teórica. Jarandilla de la Vera (Cáceres), 15-18 Noviembre, (2007).

[Melián03] Melián Belén, Moreno Pérez J., Moreno Vega J.: Metaheuristics: A global view. Revista Iberoamericana de Inteligencia Artificial., ISSN: 1137-3601 No.19 pp. 7-28 (2003).

[Bernábe04] Bernábe, L. B., López, S.: Statistical Classificatory Analysis Applied to Population Zones. 8th. World Multiconference on Systemics, Cybernetics and Informatics (2004).

[Bernábe05] Bernábe, L. B., López S.: Proceso de Análisis de correlaciones para identificar patrones de asociación sobre datos censales. Research on Computing Science, ISSN 1665-989 CIC-IPN, (2005).

[Bernábe06] Bernabe, L. B, Zamora, E., Aguirre V.: Desarrollo de un algoritmo de clasificación con restricciones sobre zonas geográficas. CORE 2006, IPN Geographic Information Systems. ISBN 970-36-0332-7, México (mayo 2006).

[Bernábe06a] Bernábe, L. B., Vara, A., Alcocer, Z.: Compactness Classification for Geographic Zones. Electrical and Electronics Engineering, 2006 3rd International Conference 1–4 (2006).

[Bernábe06b] Bernábe, L. B., Osorio, M. A., Duque J. C.: Clasificación Sobre Zonas Geográficas: Un Enfoque de Optimización Combinatoria para el Problema de Regionalización. XIII CLAIO Congreso Latino-Iberoamericano de Investigación Operativa (2006).

[Bernábe06c] Bernabe L. B., Zamora, E.: Clasificación Compacta y Homogénea para Datos Poblacionales XVI. Escuela Nacional de Optimización y Análisis Numérico 2006. Revista CiBiyT, ISSN: 1870 056X (2006).

[Bernábe08] Bernabe, L. B., Duque, J. C., Ramirez, R. J., Osorio, M. A.: Classification over Geographical Zones: A Combinatorial Optimization Approach to the Regional Partitioning Problem. Electronics, Communications and Computers, 2008. CONIELECOMP 2008, 18th International Conference. Volume 3 Issue 5 70-74 (2008).

[Bernábe09] Bernábe, L. B., Ramírez R. J., Espinosa, R. J.: Evaluación de un algoritmo de recocido simulado con superficies de respuestas. Revista de Matemáticas Teoría y Aplicaciones, ISSN: 1409-2433 volume 16 number 1, 159-177 (2009).

[Bernábe09a] Bernábe, L. B., Ramírez, R. J., Espinosa R. J., Osorio M. A., Aceves, R. R.: An Adjusted Variable Neighborhood Search Algorithm applied to the Geographical Clustering Problem Research in Computer Science issn 1870-4069, 42: 113-125 (2009).

[Bernábe09c] Bernábe, B. L., Osorio, L., Espinosa, R., Ramirez, R., Aceves, R.: A comparative study of Simulated Annealing and Variable Neighborhood Search for the Geographic Clustering Problem. The 2009 International Conference on Data Mining (DMIN'09) (2009).

[Caballero06] Caballero, R., González M., Guerrero F., Molina J., Paralera, C.: Un Método Interactivo para elegir una solución sobre una amplia Frontera Eficiente. XIV Jornadas Asepuma Badajoz (2006).

[Campbell78] Campbell, Paul J.: The Origin of “Zorn's Lemma. Historia Mathematica (Elsevier) 5 (1): 77–89 (1978).

[Coladas09] Coladas Luis Uría: Soluciones no dominadas en problemas multiobjetivo. Trabajos de Estadística y de Investigación Operativa. Editor Springer Berlin/Heidelberg. ISSN 0041-0241, volume 32, number (2009).

[Coello05] Coello Coello Carlos: Introducción a la Optimización Multiobjetivo Usando Metaheurísticas julio 2005, CINVESTAV-IPN Sección de Computación, (2005).

[Correa 01] Correa ES, Steiner MTA, Freitas AA, Carnieri C, (2001) A genetic algorithm for the P-median problem. In: Proc. 2001 Genetic and Evolutionary Computation Conf. (GECCO-2001),Morgan Kaufmann, pp. 1268-1275 (2001).

[Cruz04] Nareli Cruz Cortés: Sistema inmune artificial para solucionar problemas de optimización. Tesis doctoral en Ciencias en la Especialidad de Ingeniería Eléctrica con Opción en Computación, dirigida por Carlos A. Coello Coello, Cinvestav-IPN, (2004).

[DelosCobos96] De los Cobos, S., Perez, S., Gutierrez, A., Ordorica, M.: Métodos Heurísticos de optimización en la gran industria nacional. UAM, Depto. Matemáticas (1996).

[Diday72] Diday, E.: Optimisation en classification automatique et reconnaissance des forms. Note Scientifique IRIA No. 6 (1972).

[Duque07] Duque, J. C., Ramos, R., Suriñach, J.: Supervised Regionalization Methods: A Survey. International Regional Science Review 195-220 (2007).

[Duque04] Duque, J. C.: Design of homogeneous territorial units. A Methodological Proposal and Applications. PhD dissertation, University of Barcelona (2004).

[Ester96] Ester, M., Kriegel, H., Sander, J., XU, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with noise. Published in Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (1996).

[Everittt93] Everittt, B. S.: Cluster Analysis. Great Britain: Edward Arnold (1993).

[Ferland90] Ferland, J. A., Guénette, G.: Decision support system for a school districting problem. Operations Research, 38:15-21 (1990).

[Fleischmann98] Fleischmann, B. J., Paraschis, N.: Solving a large scale districting problem: a case report. Computers and Operations Research. 15:521-533 (1998).

[Forgy65] Forgy, E. W.: Cluster analysis of multivariate data: efficiency versus interpretability of classifications, Biometrics 21 (1965).

[Fotheringham91] Fotheringham, A. S., Wong, D. W. S.: The modifiable areal unit problem in multivariate statistical analysis, Environment and Planning A, 23, 1025-44, (1991).

[Garey79] Garey, M., Johnson, D.: Computers and Intractability. A Guide to the Theory of NP-Completeness. Freemann, San Francisco (1979).

[Gierz03] Gierz, G., Hofmann, K. H., Keimel, K., Lawson, J.D, Mislove, M. and. Scott, D. S.: Continuous Lattices and Domains, In Encyclopedia of Mathematics and its Applications, Vol. 93, Cambridge University Press, ISBN 0-521-80338-1 (2003).

[Glover86] Glover, F.: Future Paths for Integer Programming and Links to Artificial Intelligence. Computers and Operations Research 13, 533-549 (1986).

[Gonzalez] González Torres Raúl Ernesto: curso propedéutico de matemáticas discretas. Cinvestav-Guadalajara.

[Gordon96] Gordon, A. D.: A survey of constrained classification. Computational Statistics & Data Analysis, 17–29 (1996).

[Gordon96] Gordon, A. D.: Classification (Second edition). Chapman and Hall/CRC, Boca Raton. 256 pp (1999).

[Gower86] Gower, J. C., Legendre, P.: Metric and euclidean properties of dissimilarity coefficients, Journal of Classification, 3, 5-48, (1986).

[Günter01] Günter Rudolph and Alexandru Agapie. A partial order approach to noisy fitness function. In A Zalzala and et al., editors, Proceedings of the 2000 congress on evolutionary computation (CEC 2000), volume 2, pages 1010–1016, Piscataway (NJ). IEEE Press. (2001).

[Gutierrez91] Gutiérrez Andrade Miguel Angel: La Técnica de Recocido Simulado y sus Aplicaciones, Tesis Doctoral, División de Estudios de Posgrado de la Facultad de Ingeniería, UNAM, México, 1991.

[Haining94] Haining, R., Wise, S., Blake, M.: Constructing regions for small-area analysis–material deprivation and colorectal cancer. Journal of Public Health Medicine, 16: 457-469 (1994).

[Handl06] Handl, J., Knowles, J.: Multiobjective cluster and conglomerado validation. In Multiobjective machine learning edited by Yaochu Jin. Springer Series on Computational Intelligence 16:21-47, (2006).

[Hansen96] Hansen P., Mladenovic, N.: Variable neighborhood search, Les Cahiers du GERAD 96-49 (1996).

[Hansen03] Hansen P., Mladenovic, N.: Variable neighbourhood search. In Fred Glover and Gary A. Kochenberger editors, Handbook of Metaheuristics, Kluwer (2003).

[Hansen03a] Hansen P., Mladenovic N., Pérez M. J.: Variable Neighbourhood Search. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. No.19 ISSN: 1137-3601 pp. 77-92 (2003).

[Hartigan01] Hartigan: Hartigan's (1975) Classic Textbook. Redux Publicación, Journal of Classification, Editor Springer New York. Subject Collection Matemática y Estadística. ISSN 0176-4268 volume 18 number 1 (2001).

[Hess71] Hess, S.W., Samuels, S.A.: Experiences with a sales districting model: criteria and implementation, Management Science 18 (4) 41-54 (1971).

[Hess65] Hess, S.W., Weaver, J.B., Siegfeldt, H.J., Whelan, J.N., and Zitlau, P.A.: Non-partisan political redistricting by computer. Operations Research, 13:998–1008, (1965).

[Horn95] Horn, M. E. T.: Solution techniques for large regional partitioning problems. Geographical Analysis 27: 230–48 (1995).

[Kalcsics05] Kalcsics, J., Nickel, S., Schröder, M.: Toward a unified territorial design approach: Applications, algorithms, and GIS integration, TOP 13 (1) 1–56 (2005).

[Kaufman87] Kaufman, L., Rousseeuw, P.J.: 1987. Clustering by means of medoids. Statistical Data Analysis based on the L1 Norm, North-Holland, Amsterdam 405-416 (1987).

[Keith92] Keith Devlin: The Joy of Sets, Springer Verlag, 1992

[Kirkpatrick83] Kirkpatrick, C. D., Gelatt, M. P., Vecchi.: Optimization by Simulated AnnealingSCIENCIE 220: 4598 (1983).

[Kohonen82] Kohonen T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics, 43: p. 59-69 (1982).

[Kung75] Kung, H. T., Luccio, F., Preparata, F. P.: On Finding the Maxima of a Set of Vectors, Journal of the ACM (JACM), v.22 n.4, p.469-476, Oct. (1975).

[Laguna 2006] Laguna, M., Marti, R., Caballero, R., Molina, J.: Multiobjective Clustering with Metaheuristic Technology. Metaheuristics for Data Mining Problems, INFORMS Computing Society/Heuristic Search (2006).

[Lara03] Lara López Adriana: Un estudio de las Estrategias Evolutivas para problemas Multiobjetivo. Tesis de Maestria en Ciencias en la especialidad de Ingeniería Eléctrica Opción Computación, Codirectores Dr. Carlos A. Coello Coello y Dr. Alin Carsteanu, Cinvestav (2003).

[López05] López Jaimes Antonio: Diseño de un algoritmo evolutivo multiobjetivo paralelo. Tesis de Maestria en Ciencias en la especialidad de Ingeniería Eléctrica, Opción Computación, Dirigida por Dr. Carlos A. Coello Coello (2005).

[Macmillan94] Macmillan, B. P. T.: 'Optimization modelling in GIS framework: the problem of political redistricting', in Fotheringham, Sand Rogerson, P(eds.), Spatial analysis and GIS, London etc.]: Taylor & Francis, pp 221-46 (1994) .

[Mark02] Mark, J. V., Katherine, S. P., Jennifer B.: A New Partitioning Around Medoids Algorithm (2002).

[MacQueen66] MacQueen, J.: Some methods for classfication and analysis of multivariate observations», Proceedings of 5th Symposium of Mathematics, Statistics and Probability, vol. 1, pp. 281-297. Universtity of California Press. Berkeley (1966).

[Martin97] Martin, D.: From enumeration districts to output areas: experiments in the automated creation of a census output geography. Population Trends, 88:36-42 (1997).

[Martin98] Martin, D.: 2001 Census output areas: from concept to prototype. Population Trends, 94:19-24 (1998).

[Mehrotra98] Mehrotra, A., Johnson, E. L., Nemhause, G. L.: An optimization based heuristic for political districting. Management Science, 44(8):1100–1114, (1998).

[Montgomery91] Montgomery, D.: Design and Analisis of Experiments. Ed. Wiley 2ª Edition (1991).

[Murtagh91] Murtagh, F.: A survey of algorithms for contiguity-constrained cluster and related problems. Computer Journal 82-88 (1991).

[Murtagh95] Murtagh, F.: Interpreting the Kohonen self organizing feature map using continuity-constrained cluster. Pattern Recognition Letters 16, 399-408 (1995).

[Murthy96] Murthy CA, Chowdhury N In search of optimal conglomerados using genetic algorithms. Pattern Recognition Letters, 17:825-832 (1996).

[Navernioukó3] Navernioukó Igor: Multiobjective Graph Clustering with Variable Neighbourhood Descent. A thesis submitted in partial fulfilment of the requirements for the degree of master of science. The University of British Columbia The Faculty of Graduate Studies Computer Science (2003).

[Ochoa09] Ochoa, A., Bernábe B., Ochoa, O.: Towards a parallel system for demographic zonification bases on complex networks. Journal of Applied Research and Technology, (august 2009) 7(2): 217-231 (2009).

[Openshaw84] Openshaw, S.: The modifiable areal unit problem, Concepts and Techniques inModern Geography, 38 (GeoBooks, Norwich) (1984).

[Openshaw92] Openshaw, S.: Some suggestions concerning the development of artificial intelligence tools for spatial modelling and analysis in GIS. The Annals of Regional Science, 26, 35-51 (1992).

[Openshaw95] Openshaw, S., Wymer C.: Classifying and regionalizing census data.In Census users handbook, ed. S. Openshaw, Cambridge, UK, GeoInformation International. 239-70 (1995).

[Openshaw95a] Openshaw, S., Rao, L.: Algorithms for reengineering 1991 Census Geography. Environment and Planning A 27:425-446 (1995).

[Openshaw99] Openshaw, S., Alvanides, S.: Applying geocomputation to the analysis of spatial distributions, In: Longley PA, Goodchild MF, Maguire DJ, Rhind DW, (eds) Geographical Information Systems Principles, Techniques, Management and Applications, Wiley, Chichester, pp 267-282 (1999).

[Ohsumi83] Ohsumi, S.: Population assessment of Baird's beaked whales in the waters adjacent to Japan. Rep. Int. Whal. Commn. 33:633-641 (1983).

[Paquete07] Paquete, L., Schiavinotto, T., and Stützle T.: On local optima in multiobjective combinatorial optimization problems Publicación Annals of Operations Research. Editor Springer Netherlands. ISSN 0254-5330, volume 156, number 1 83-97 (2007).

[Pareto96] Pareto, Vilfredo. Cours D'Economie Politique. F. Rouge, (1896).

[Pelta00] Alejandro Pelta David: Algoritmos heurísticos en bioinformática. Tesis doctoral, Dirigida por José Verdegay y Armando Blanco. Depto. de ciencias de la computación e inteligencia artificial. Universidad de Granada (2000).

[Perruchet83] Perruchet, C.: Constrained agglomerative hierarchical-classification. Pattern Recognition 16: 213–17 (1983).

[Pizza99] Pizza, E., Murilo, A., Trejos, J.: Nuevas técnicas de particionamiento en clasificación automática. Revista de Matemáticas Teoría y Aplicaciones, issn: 1409-2433, 51-66 (1999).

[Régnier] Régnier, S.: Sur quelques aspects mathématiques des problémes de classification automatique", ICC Bulletin 4 & Math. Sci. Hum82 (1965).

[Rios08] Ríos Insua David.: Sobre soluciones optimas en problemas de optimización multiobjetivo. Trabajos de Investigación Operativa. Editor Springer Berlin/ Heidelberg. ISSN 0213-8204, volume 2, number 1 49-67 (2008).

[Rousseeuw97] Rousseeuw, P. J., Hubert M., Struyf A.: Clustering in an object-oriented environment. Journal of Statistical Software (1997) 02-10.

[Saha06] Saha, S., Bandyopadhyay, S.: Application Of Multiobjective Optimization For Data Clustering, In the Proceedings of First International Conference On Emerging Applications of IT, Kolkata. , Elsevier publication, pp. 275-278, (2006).

[Sameeh08] Sameeh U., Fakhri K., Jin-Myung W.: Non-dominated Sorting Evolution Strategy-based k-means Clustering Algorithm for Accent Classification, IEEE International Conference on Pattern Recognition (2008).

[Sánchez06] Sánchez Larios Hérica: Modelado de funciones de distancia asiméticas y no uniformes. Tesis Doctoral en Investigación de Operaciones, UNAM. Director Dr. Servio Tulio Guillen B. (2006).

[Shirabe05] Shirabe, T.: A Model of Contiguity for Spatial Unit Allocation, Geographical Analysis 37(1), pp. 2-16. (2005).

[Sokal63] Sokal, R. R., y Sneath P. H.: Principles of Numerical Taxonomy. W.H. Freeman. San Francisco (1963).

[Tavares07] Tavares P. F., Figueira, J. R., Mousseau, V., Roy, B.: Multiple criteria districting problems: The public transportation network pricing system of the Paris region. Publicación Annals of Operations Research. Editor Springer Netherlands, ISSN 0254-5330 volume 154 number 1 169-92 (2007).

[Toda83] Toda, M., Kubo, R., Saitô, N., Statistical Physics, Springer-Verlag, (1983).

[Trejos09] Trejos, Z. J.: Modelos de Clasificación Automática, Centro de Investigación en Matemática Pura y Aplicada (CIMPA), Universidad de Costa Rica, (2009).

[Vicente05] Vicente, E., Rivera L., Mauricio D.: Grasp en la resolución del problema de cluster., ISSN: 1815-0268 2(2) 16-25 (2005).

[Vickrey61] Vickrey, W.: On the prevention of gerrymandering. Political Science Quarterly 76: 105–10 (1961).

[Williams95] Williams, J.C.: Political Redistricting - A Review. Papers in Regional Science 74(1): 13-39 (1995).

[Wise97] Wise, S. M., Haining, R. P., and Ma, J.: Regionalisation tools for exploratory spatial analysis of health data. In Recent developments in spatial analysis: Spatial statistics, behavioural modelling, and computational intelligence, ed. Manfred M. Fischer and Arthur Getis, 83–100. New York: Springer (1997).

[Zamora06] Zamora, A. E.: Implementación de un algoritmo compacto y homogéneo para la clasificación de zonas geográficas agebs bajo una interfaz gráfica. Tesis de Ingeniería en Ciencias de la Computación BUAP FCC, Asesor B. Bernábe (2006).

[Zoltners83] Zoltners, A., Sinha, P.: Towards a unified territory alignment: A review and model. Management Science, (1983) 1237-1256 (1983).

[Web1] http://casoilresource.lawr.ucdavis.edu/drupal/node/340)

[Web2] http://www.cs.cinvestav.mx/~emoobook/nodom/nonodom.html

[Web3] http://www.inegi.gob.mx

[Web4] http://delta.cs.cinvestav.mx/˜ccoello/EMOO/