Universidad de las Ciencias Informáticas

Facultad 3

Algoritmo de agrupamiento geoespacial basado en autómatas celulares

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores: Nailee Vidal Abreu

Bárbaro Enrique Martínez Fernández

Tutor(es): Ing. Yadian Guillermo Pérez Betancourt

Ing. Roger Godofredo Rivero Morales

Ing. Liset González Polanco

La Habana, Junio de 2018

AGRADECIMIENTOS

Queremos agradecer a nuestros familiares, amigos y a todas las personas que de alguna forma han contribuido a nuestro proceso de formación. Gracias formar parte de nuestras vidas, por su apoyo y por su cariño.

Nailee y Bárbaro.

DEDICATORIA

A nuestras familias.

Nailee y Bárbaro.

DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores de la presente tesis que lleva como título: “Algoritmo de agrupamiento geoespacial basado en autómatas celulares” y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Firma del Autor Firma del Autor

_____ _____

Firma del Tutor Firma del Tutor

_____

Firma del Tutor

RESUMEN

La estratificación de territorios es un proceso de clasificación que posibilita la separación de los elementos de un universo en estratos, que presentan desigualdades entre sí. Las técnicas de agrupamiento fueron identificadas como unas de las más utilizadas y con mejores resultados para realizar la clasificación de los datos en este proceso. En la literatura científica consultada se evidencia que el proceso de estratificación presenta limitaciones debido al insuficiente tratamiento a la dependencia espacial y no se emplean las medidas de similitud adecuadas.

El objetivo del presente trabajo es diseñar un algoritmo de agrupamiento basado en autómatas celulares y medidas de similitud geométricas para la estratificación de territorios que facilite la obtención de información precisa para la toma de decisiones en salud. Para ello se realizó un análisis del panorama actual de la estratificación de territorios, los algoritmos de agrupamiento y las medidas de similitud existente. Se generaron los artefactos definidos en la metodología seleccionada y se realizaron las pruebas de software correspondientes. Para valorar los resultados de la solución propuesta se aplicó un caso de estudio, se aplicó el algoritmo sobre un conjunto de datos sintéticos y se analizaron los resultados experimentales.

Se obtuvo como resultado un sistema que permite integrar datos de naturaleza espacial y temática para el análisis y construcción de estratos. La solución se basa en los modelos autómatas celulares y las medidas de similitud geométricas y contribuye a la obtención de información precisa en para la toma de decisiones en las entidades de salud.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad y con el auge de las nuevas tecnologías se generan mayores cantidades de datos que es necesario tratar para obtener información que pueda ser útil a las diferentes organizaciones. Para las empresas u organizaciones los datos son materia prima que permiten encontrar patrones. Estos favorecen la interpretación de fenómenos o sucesos, por ejemplo: qué gustos tiene un usuario, saber si le puede otorgar un préstamo, qué producto se vende más por temporadas, cuál es el perfil de personas que ven una determinada película, o cuáles son las causas de una enfermedad (Oded y Lior 2005).

Se procesan también grandes volúmenes de datos geoespaciales que cuentan con atributos que proveen información asociada a una localización en el espacio. Además, se almacenan metadatos que contienen descripciones de los datos. Mediante el procesamiento de los datos geográficos pueden reconocerse numerosos patrones de los cuales se puede obtener información útil, relevante y de gran importancia para las organizaciones. La información geoespacial puede estar asociada con la cartografía, en el logro de objetivos específicos concernientes a operaciones de análisis y gestión de datos geoespaciales (Cangrejo Aljure y Agudelo 2011a).

Actualmente se dispone de bases de datos espaciales en constante crecimiento, mediante su tratamiento puede obtenerse información útil. Una forma de obtener esta información es aplicando búsquedas orientadas a identificar tipos de información como pueden ser objetos y relaciones espaciales. Las búsquedas pueden involucrar altos niveles de complejidad computacional tanto espacial como temporal. Las bases de datos espaciales soportan operaciones eficientes para la realización de tareas comunes como búsquedas por vecindad y uniones espaciales. Estas no almacenan explícitamente patrones o reglas que determinan las relaciones espaciales entre los objetos y algunas características no espaciales (Ester, Kriegel y Jörg 2001).

Una característica fundamental a tener en cuenta en el análisis de datos geoespaciales es la autocorrelación espacial (AE). Es un procedimiento intrínsecamente geográfico que aporta información acerca del comportamiento de los datos georreferenciados a diferentes escalas, en particular el tipo de asociación existente entre unidades espaciales vecinas. A pesar de su creciente importancia en el marco del análisis exploratorio de datos geoespaciales, su utilización es reducida en aplicaciones prácticas. Estos continúan recurriendo a los coeficientes tradicionales de correlación y a la estadística descriptiva (Celemín 2010).

La evolución de la informática y la electrónica ha dado paso al surgimiento de Sistemas de Información Geográfica (SIG), que constituyen poderosas herramientas para el tratamiento de datos

geoespaciales. Estas herramientas consideran las componentes espacial, temática y temporal de los datos georreferenciados. Como herramienta, dan soporte a la visualización, consulta, edición, análisis espacial y socialización. Los SIGs utilizan capas para manejar la geometría asociada a los datos y la ubicación espacial de dicha geometría que es almacenada en una Base de Datos Espacial (SDB) (Cangrejo Aljure y Agudelo 2011b).

Los SIG pueden definirse como: sistemas informáticos, con la capacidad de visualizar y analizar información georreferenciada de forma versátil e intuitiva. Su uso en la rama de la salud ha cobrado cada día mayor utilidad. Emplearlos contribuye al fortalecimiento de la capacidad de análisis en materia de salud pública y epidemiológica, brindando información de utilidad a la toma de decisiones (Rojas 1998). Por otra parte, facilitan la identificación de la ubicación geográfica de establecimientos de salud y grupos de población que presentan mayor riesgo y que por tanto requieren de mayor atención preventiva, curativa o de promoción de la salud (Morales y Torres 2015).

La acumulación de información geoespacial producto del desarrollo de los sistemas informáticos, y en especial de los SIG, propicia la aplicación de técnicas de minería de datos geoespacial (MDG) para la extracción de conocimientos que asistan a la toma de decisiones. La aplicación de la MDG conduce al descubrimiento de conocimiento implícito sobre la información de los datos geográficos. El descubrimiento de conocimientos sobre datos geoespaciales utilizando la minería de datos geoespacial es un proceso mucho más complejo que utilizando la minería de datos tradicional ya que no solo tiene en cuenta la componente temática, sino también la componente espacial. Si bien el espacio es un componente importante, no siempre se le da la importancia requerida (Cangrejo Aljure y Agudelo 2011a).

En estudios epidemiológicos se han introducido algunas aproximaciones de la minería de datos para determinar relaciones espaciales de determinadas variables. Dentro de los enfoques más utilizados destacan las técnicas de agrupamiento. Las técnicas de agrupamiento fueron identificadas en la literatura científica consultada (López Caviedes 2004; Yenisei Bombino Companioni 2005; Erik Limón 2012) como unas de las más utilizadas y con mejores resultados para realizar la clasificación de los datos en los procesos de estratificación de territorios. Estas técnicas se encargan de descubrir una estructura dentro de un conjunto de datos dividiéndolo en subconjuntos.

La estratificación es originada por las unidades agregadas denominadas estratos que presentan similitudes entre sus elementos y diferencias entre estos y los elementos de diferentes estratos. Desde la década de los 90 se ha incorporado al esquema de estratificación el enfoque epidemiológico de riesgo como base para la toma de decisiones. La bibliografía revisada coincide

en la necesidad de aplicar los conceptos de estratificación epidemiológica de riesgo en el estudio de eventos sanitarios para la toma de decisiones en salud(García Pérez y Alfonso Aguilar 2002).

La estratificación es la separación de datos en categorías o clases que permite aislar la causa de un problema, identificando el grado de influencia de ciertos factores en el resultado de un proceso. Es utilizada para comprender de manera detallada la estructura de un grupo de datos, lo cual permitirá identificar las causas del problema y llevar a cabo las acciones correctivas convenientes (Morales y Torres 2015).

Las principales aproximaciones a la incorporación del espacio en estudios estratificados utilizan técnicas clásicas de minería de datos que, dentro del contexto presenta algunas limitaciones:

- Las medidas de similitud empleadas están enfocadas a los datos temáticos, sin tener en cuenta la componente espacial, esto viola la primera ley de la Geografía, todo está relacionado, pero objetos cercanos están más relacionados que objetos distantes.

- La dependencia espacial se conoce como autocorrelación y suele ignorarse en la estratificación, pues se parte de supuestos de una distribución independiente al analizar los datos, lo que puede producir hipótesis o modelos inexactos e inconsistentes.

- Considerar la continuidad del espacio trae consigo que estas técnicas no sean efectivas, muchas asumen los datos como discretos.

A partir de la situación problemática planteada se formula el siguiente **problema a resolver**:

El insuficiente tratamiento a la dependencia espacial y las medidas de similitud empleadas en la estratificación de territorios, limitan la obtención de información precisa.

Se plantea como **objeto de estudio** el proceso de estratificación de territorios centrando su **campo de acción** en los algoritmos de agrupamiento.

Para darle solución al problema anterior se plantea como **objetivo general:** diseñar un algoritmo de agrupamiento basado en autómatas celulares y medidas de similitud geométricas para la estratificación de territorios que facilite la obtención de información precisa para la toma de decisiones en salud. Este se desglosa en los siguientes **Objetivos específicos:**

- Construir el marco teórico referencial de la investigación, relacionado con la minería de datos espaciales y el proceso de estratificación de territorios.
- Diseñar un algoritmo de agrupamiento basado en autómatas celulares.
- Incorporar el algoritmo propuesto al complemento de estratificación basado en indicadores de salud.
- Verificar la solución informática propuesta aplicando diferentes pruebas y métricas.

En el desarrollo del presente trabajo de investigación fueron utilizados los **Métodos Científicos** siguientes:

- Histórico-Lógico: permitió realizar un estudio de los principales conceptos asociados a la minería de datos espaciales y establecer entre ellos relaciones lógicas.

- Análisis y Síntesis: se utiliza para identificar y analizar las diversas funcionalidades de los SIG que pueden ser aplicadas al proceso de estratificación territorial y su posterior síntesis, conforme a las necesidades de Cuba en el sector de la salud.

- Método de la modelación: la modelación es el método mediante el cual se crean abstracciones con el objetivo de explicar la realidad. El modelo como sustituto del objeto de investigación es semejante a él, existiendo una correspondencia objetiva entre el modelo y el objeto, siendo el investigador quien elabora dicho modelo. El modelo es el eslabón entre el sujeto y el objeto intermedio. La condición principal de la modelación es la relación entre el modelo y el objeto que se modela. La necesidad práctica para la cual se ejecuta la modelación y la posible solución del problema de investigación da la medida en que se logra dicha relación, la que es determinada por el sujeto escogiendo una alternativa de acuerdo con su criterio.

- Método de la observación: la observación científica es la percepción planificada dirigida a un fin y relativamente prolongada de un hacho o fenómeno. Es el instrumento universal del científico, se realiza de forma consciente y orientada a un objetivo determinado.

- Método experimental: históricamente al experimento se le ha atribuido una importancia decisiva en la demostración del vínculo causal entre dos fenómenos, llegando a considerarse solamente como científicas las demostraciones que se realizaban por vía experimental.

Estructuración del trabajo

El trabajo consta de tres capítulos que cubren la fundamentación teórica, diseño del algoritmo, implementación y prueba, además de las conclusiones, recomendaciones, referencias bibliográficas, glosario de términos y anexos.

Capítulo 1. Fundamentos teóricos de la minería de datos espaciales: En este capítulo se presentan un conjunto de elementos teóricos relacionados con los algoritmos de agrupamiento para lograr un mayor entendimiento del trabajo a desarrollar. Se realiza una valoración comparativa entre los algoritmos existentes y su relación con el proceso de estratificación de territorios. Además, se analizan las tecnologías, herramientas informáticas y metodologías a utilizar en el proceso de desarrollo del software.

Capítulo 2. Algoritmo de agrupamiento geoespacial basado en autómata celulares: En este capítulo se realiza una descripción general de la solución propuesta, se especifican los requisitos funcionales y no funcionales que se tendrán en cuenta para la implementación de la misma, y se detallan aspectos relacionados con su diseño y arquitectura. Se especifican los patrones del diseño aplicados y los artefactos derivados de la metodología de desarrollo de software seleccionada.

Capítulo 3. Verificación de la propuesta de solución: Este capítulo describe la etapa de implementación, se elaboran y documentan las pruebas realizadas a la solución propuesta para demostrar el correcto funcionamiento de la misma.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS DE LA MINERÍA DE DATOS ESPACIALES

Introducción

En este capítulo se presentan los elementos teóricos que conforman el marco referencial relacionado con el objeto de estudio. Se realiza un análisis crítico de los algoritmos de agrupamiento existentes y su relación con el proceso de estratificación de territorios. Se analizan las tecnologías, herramientas informáticas y metodologías a utilizar en el proceso de desarrollo del software.

1.1 Descubrimiento de conocimiento en datos geoespaciales

En los últimos años, el desarrollo tecnológico tanto en el área de cómputo como en la de transmisión de datos, ha hecho posible que se gestionen de una mejor manera el almacenamiento y manejo de grandes volúmenes de datos. Las empresas de hoy se mueven en entornos altamente competitivos y de cambio continuo. La dinámica del mercado conduce a la necesidad de contar con la información adecuada en el momento indicado para que los directivos puedan tomar las decisiones de negocio apropiadas. Por ello, han comprendido que los grandes volúmenes de datos que residen en sus sistemas pueden, y deben, ser analizados y explotados para obtener nuevo conocimiento(Morgado García, Ponce de León Lima y Rosete Suárez 2017).

Debido a la tecnología moderna, como el Sistema de Posicionamiento Global (GPS), la capacidad de comunicación inalámbrica de potentes computadoras pequeñas y sensores en el campo, cantidades cada vez mayores de datos geoespaciales están disponibles en forma digital en la actualidad. Datos de los cuales mediante su procesamiento se puede obtener información útil, relevante y de gran importancia para las organizaciones lo cual se ha convertido en un elemento clave en los procesos organizacionales (Dueñas-Reyes 2009).

1.1.1 Datos Geoespaciales

Los datos geoespaciales o geodatos son variables asociadas a una localización en el espacio, se reconocen con el término de datos georreferenciados. Presentan dos tipos de propiedades fundamentales las geométricas y las descriptivas las cuales tienen como características fundamentales las siguientes:

Propiedades geométricas: un dato geoespacial puede representarse utilizando formato ráster o datos vectoriales, los cuales pueden ser expresados mediante tres tipos de objetos espaciales puntos, líneas y polígonos. Los puntos, que se encuentran determinados por las coordenadas terrestres medidas por latitud y longitud, pueden representar ciudades, accidentes geográficos

puntuales, empresas. Las líneas, que son objetos abiertos que cubren una distancia dada y comunican varios puntos o nodos, aunque debido a la forma esférica de la tierra también se le consideran como arcos. Las líneas telefónicas, carreteras y vías de trenes son algunos ejemplos de líneas geográficas. Los polígonos que son figuras planas conectadas por distintas líneas u objetos cerrados que cubren un área determinada, países, regiones o lagos son ejemplos de estos. Se realiza la referencia a un lugar mediante el uso del nombre de lo que representa dicho polígono. El lugar, la forma y la extensión forman uno de los pilares de la información que proporcionan los geodatos (Wilson 2015; Taha 2016; Pérez Bentancourt et al. 2018).

Propiedades descriptivas: los geodatos, además de sus propiedades geométricas, contienen las características de lo que representan (número de trabajadores o productividad de una empresa, distribución de la población de un municipio, tipo y extensión de los usos del suelo de un territorio, capacidad de un canal por solo mencionar). Estas características pueden ser muy diversas, y van desde valores numéricos, a documentos gráficos en formatos de multimedia.

Los datos geoespaciales se caracterizan por su naturaleza georreferenciada y multidireccional. La posición relativa o absoluta de cualquier elemento sobre el espacio contiene información valiosa, pues la localización debe considerarse explícitamente en cualquier análisis. Por multidireccional se entiende a que existen relaciones complejas no lineales, es decir que un elemento cualquiera se relaciona con su vecino y además con regiones lejanas. Es decir, todos los elementos se relacionan entre sí, pero existe una relación más profunda entre los elementos más cercanos. Como destacaba Tobler: en la geografía, todo tiene que ver con todo, pero objetos cercanos están más relacionados que objetos distantes. Es lógico pensar que, si aumenta la distancia entre puntos, la autocorrelación tiende a decrecer cuando estamos en presencia de autocorrelación positiva (Korte 2001; Taha 2016).

Existen métodos y técnicas para la exploración y análisis de datos que, aunque permiten obtener información relevante entre cantidades considerables de estos, presentan limitaciones para analizar datos geográficos, por lo que han surgido herramientas y métodos de procesamiento analítico geoespacial para datos geoespaciales (Dueñas-Reyes 2009).

1.1.2 Proceso del KDD

En (Fayyad, Piatetsky Shapiro y Smyth 1996) definen el proceso de KDD como: "Un proceso no trivial de identificación de información válida, novedosa, potencialmente útil y entendible de patrones

comprensibles que se encuentran ocultos en los datos". Está compuesto por cinco fases, las cuales se describen a continuación:

1. Integración y recopilación de datos: En esta fase se determinan las fuentes de información que pueden ser útiles y dónde conseguirlas.

2. Selección, limpieza y transformación: Incluye detectar los outliers, los datos faltantes o perdidos, construir nuevos atributos y numerizar o discretizar los atributos.

3. **Minería de datos**: El objetivo de esta fase es producir nuevo conocimiento. Para ello se construye un modelo de los patrones y relaciones entre los datos que pueden usarse para hacer predicciones, para entender mejor los datos o para explicar situaciones pasadas. Incluye:

- Elegir el tipo de modelo.
- Determinar qué tipo de tarea de Minería es el más apropiado.
- Elegir el algoritmo de Minería que resuelva la tarea y obtenga el tipo de modelo que se está buscando.

4. Evaluación e interpretación: Existen diferentes medidas de evaluación de los modelos: precisos, comprensibles (inteligibles) e interesantes (útiles y novedosos).

5. Difusión: Después que el modelo es construido y validado, este puede utilizarse en disímiles finalidades, y durante este proceso debe medirse su evolución. De esta manera se sabrá si debe ser reevaluado, re-entrenado y posiblemente reconstruido completamente.

En la minería de datos, una de las técnicas más relevantes es la clasificación de los objetos de acuerdo con características similares. La ubicación geográfica constituye un criterio de vital importancia para esta tarea. Para ello se trata la dependencia espacial entre los objetos siendo importante tener en cuenta la relación entre los vecinos en el conjunto de datos (Dueñas-Reyes 2009).

1.1.3 Dependencia espacial

Dependencia espacial conocida también como autocorrelación espacial (AE) se refiere a la relación entre los datos georreferenciados debido a la naturaleza de la variable bajo estudio y el tamaño, forma y configuración de las unidades espaciales(Rosales Tapia y Quintero Pérez 2012). Según (Vilalta 2005) es la concentración o dispersión de los valores de una variable en un mapa.

Dicho de otra manera, la AE refleja el grado en que objetos o actividades en una unidad geográfica son similares a otros objetos o actividades en unidades geográficas próximas (Goodchild 1987). Este tipo de autocorrelación prueba la primera ley geográfica de Tobler (1970).

La dependencia espacial de un objeto se puede definir como el hecho de estar presente, lo que implica que tiene un lugar y una forma, considerando sus relaciones, causas y consecuencias. Al ser así, puede encontrarse sujeto a la acción de un agente, existe una relación entre ambos, y ésta puede ser considerada como una amenaza. La dependencia espacial de dicho objeto con los elementos a su alrededor se analiza a partir de las relaciones espaciales existentes entre ellos (Rosales Tapia y Quintero Pérez 2012).

Las relaciones espaciales son conceptos que surgen de la interacción entre el espacio y los eventos que en él ocurren, así como todas sus combinaciones. Existen nueve tipos de relaciones espaciales y cada una tiene su propio conjunto de técnicas de análisis (Gatrell 1983; Miller y E. Wentz 2003; Morales Manilla 2007).

Estos tipos de relaciones se organizan con base en la mayor o menor dominancia, sea de las propiedades del espacio o de las propiedades de los eventos (Figura 1). En este contexto los tres grupos de relaciones espaciales son: relaciones métricas, relaciones topológicas y relaciones de organización (Rosales Tapia y Quintero Pérez 2012).

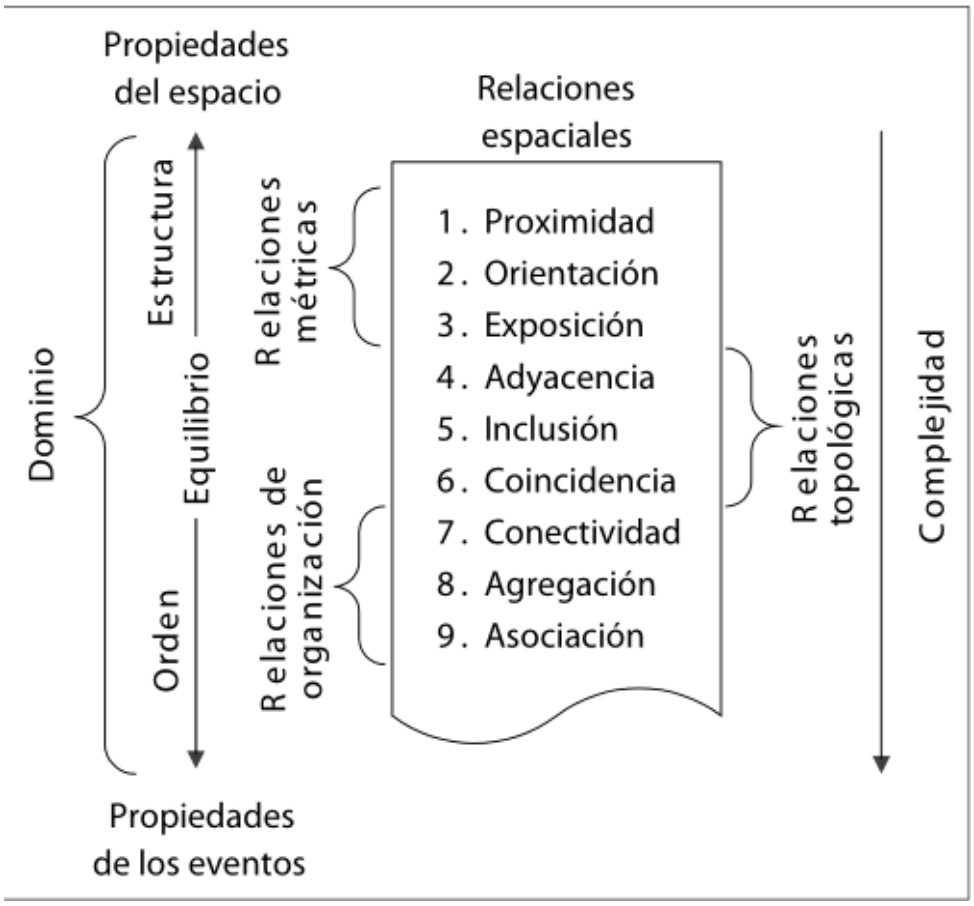

Figura 1. Relaciones espaciales. Fuente (Rosales Tapia y Quintero Pérez 2012).

La técnica más antigua y típica para la detección y medición de la AE es el coeficiente I de Moran, indicador estadístico que indica si existe autocorrelación espacial entre todas las unidades geográficas de la muestra. Esta técnica ha sido utilizada en la investigación en México sobre desarrollo económico regional (Vilalta 2003; Martínez 2004) y comportamiento electoral (Aguayo Telléz y Medellín Mendoza 2014; Vilalta 2004).

El diseño es similar al coeficiente de correlación de Pearson. Sus valores varían entre -1 y +1, donde el primer valor significa una autocorrelación negativa perfecta (perfecta dispersión) y el segundo una autocorrelación positiva perfecta (perfecta concentración); el cero significa un patrón espacial totalmente aleatorio. La diferencia entre los dos coeficientes de Moran y Pearson se basa en que en el primer caso la asociación de valores en el conjunto de datos está determinada por una matriz de distancias o contigüidad que predefine los valores vecinos (los valores para el cómputo del coeficiente) (Vilalta 2005; Brown, Wood y Griffith 2017).

La ecuación 1 del coeficiente I de Moran es:

$$I = \frac{n}{\sum_{i=1}^{i=n}\sum_{j=1}^{j=n} W_{ij}} \bullet \frac{\sum_{i=1}^{i=n}\sum_{j=1}^{j=n} W_{ij}(x_i - \bar{x})(x_j - \bar{x})}{\sum_{i=1}^{i=n}(x_i - \bar{x})^2}$$

Donde n es el número de las unidades (es decir, áreas o puntos) en el mapa, Wij es la matriz de distancias que define si las áreas o puntos geográficos, i y j, son o no vecinos.

1.1.4 Minería de datos geoespacial

La minería de datos es un proceso de búsqueda de información relevante en grandes volúmenes de datos, semejante a la que podría realizar un experto humano. La amplia difusión de información espacial producto del desarrollo de los SIG ha favorecido la explotación de los datos con el objetivo de encontrar conocimiento de manera automatizada. La complejidad de los tipos de datos existentes en Sistemas Gestores de Bases de Datos Geoespaciales y las estructuras de datos que las soportan limitan la utilización de técnicas tradicionales de minería de datos, lo que propicia la aparición de nuevas técnicas que de conjunto forman la minería de datos geoespaciales (Pérez Betancourt y González Polanco 2013).

La minería de datos se ha venido adaptando dentro de las empresas con el fin de realizar exploración y análisis de datos enfocados en el descubrimiento del conocimiento. Dada la importancia que la información geoespacial está tomando, surge la minería de datos geoespacial. Este proceso permite descubrir patrones útiles e inesperados dentro de los datos. Las técnicas de minería de datos geoespacial se aplican para extraer conocimiento a partir de grandes volúmenes de datos que pueden ser de tipo geoespacial y no geoespacial, los de tipo geográfico han tomado mayor relevancia, pues dan un valor agregado al análisis. La característica especial acerca del análisis geoespacial es que “el lugar hace la diferencia”; por lo tanto, la ubicación de los eventos necesita ser integrada en el análisis (Dueñas-Reyes 2009; Vaswani y Karandikar 2017).

1.2 Métodos para el descubrimiento de conocimiento en datos geoespaciales

El descubrimiento de conocimiento sobre datos geoespaciales utilizando la minería de datos geoespaciales es mucho más complejo, ya que no solo tiene en cuenta los patrones correspondientes a datos cartográficos y numéricos de la minería de datos tradicionales, sino que también se ocupa de las múltiples relaciones topológicas de los objetos como son intersecciones, superposiciones, adyacencias y disyunciones entre otros. Además de la distinción, localización e identificación de los objetos también trata las relaciones de orientación espacial y distancia entre

los objetos del ambiente y el objeto observado (Cangrejo Aljure y Agudelo 2011; Pérez Betancourt y González Polanco 2013). A continuación, se describen los métodos utilizados en la minería de datos espaciales:

- Basados en la generalización: requieren de la implementación de jerarquías de conceptos, bien temática o espacial. Dentro de las temáticas se incluyen los datos no espaciales; de ellos se colectan sus características más importantes para la búsqueda, se caracterizan por regiones y se agrupan como datos no espaciales generalizados. Para el caso de los espaciales esta generalización puede ser presentada como la partición en regiones y su posterior fusión dependiendo de los atributos espaciales de los datos.

- Basados en el reconocimiento de patrones: son utilizados en la clasificación de información que pueden ser imágenes de satélites, fotografías, textos o cualquier fuente de datos:

- De agrupamiento: permiten agrupar los objetos de una base de datos en grupos llamados conglomerados, conformados por elementos tan similares como sea posible.

- De exploración de asociaciones espaciales: permiten descubrir reglas de asociación espacial que relacionen a uno o más objetos espaciales.

- Mediante el uso de aproximación y agregación: permiten descubrir conocimiento a partir de las características representativas de los objetos.

1.3 Algoritmos de agrupamiento

Los algoritmos de agrupamiento no se usan sólo en minería de datos. Aunque su principal función es la de agrupar objetos según semejanzas. El funcionamiento de la mayoría de estos algoritmos está basado en la optimización de una función objetivo, que normalmente es la suma ponderada de las distancias a los centros. Estas funciones pueden variar, y muchas veces los distintos algoritmos de reconocimiento de patrones se distinguen principalmente en la definición de sus funciones objetivo a optimizar. Uno de los pasos en los algoritmos de agrupamiento es el de asignar a cada objeto una medida de semejanza, con el fin de determinar a cuál de los grupos detectados pertenece el objeto en cuestión. Esta medida de semejanza entre objetos de un conjunto de datos se basa normalmente en el cálculo de una función de distancia o similitud (Benítez 2005; Vaswani y Karandikar 2017).

Las técnicas de agrupamiento se encargan de descubrir una estructura dentro de un conjunto de datos $D = \{xi, \ldots, xn\}$, dividiéndolo en subconjuntos que muestren ciertas coherencias, es decir, los objetos pueden dividirse en grupos que contienen muestras similares dentro de un mismo grupo o clúster, más similares entre sí que a las muestras de otros grupos; definiendo las muestras parecidas como una noción de similitud o de distancia entre muestras (Jain, Murty y Flynn 1999; Wang et al. 2016).

Los algoritmos de agrupamiento son atractivos para la tarea de identificación de clases. Sin embargo, la aplicación a grandes bases de datos espaciales plantea los siguientes requisitos para algoritmos de agrupación: (1) requisitos mínimos de conocimiento de dominio para determinar los parámetros de entrada, porque los valores apropiados a menudo no se conocen de antemano cuando se trata de bases de datos grandes; descubrimiento de conglomerados con forma arbitraria, porque la forma de los conglomerados en las bases de datos espaciales puede ser esférica, prolongada, lineal, alargada; buena eficacia en bases de datos grandes, es decir, en bases de datos significativamente más que unas pocas mil objetos (Ester et al. 1996).Los algoritmos de agrupamiento pueden clasificarse en: basados en partición, jerárquico, densidad, cuadrícula y modelo (Martínez-González et al. 2017; Vaswani y Karandikar 2017).

1.3.1 Algoritmos basados en partición

Los algoritmos de partición construyen una partición de una base de datos agrupando los objetos en un conjunto de k grupos, k es un parámetro de entrada para estos algoritmos. Es decir, se requiere algún conocimiento de dominio que desafortunadamente no está disponible para muchas aplicaciones. El algoritmo de partición generalmente comienza con una partición inicial y luego utiliza una estrategia de control iterativo para optimizar una función objetivo. Cada grupo está representado por el centro de gravedad del clúster (algoritmos k-means) o por uno de los objetos del clúster ubicado cerca de su centro (algoritmos k-medoid). En consecuencia, los algoritmos de partición utilizan un procedimiento de dos pasos. Primero, determinan k representantes minimizando la función objetivo. En segundo lugar, asignan cada objeto al clúster con su representante "más cercano" al objeto considerado. El segundo paso implica que una partición es equivalente a un diagrama voronoi y cada clúster está contenido en una de las celdas voronoi. Por lo tanto, la forma de los datos de todos los grupos encontrados por un algoritmo de partición es convexa, lo cual es muy restrictivo (Ester et al. 1996; Wang et al. 2015).

K-means es un algoritmo muy conocido y muy usado, por su eficacia y robustez. Su nombre hace referencia al número K de clases o grupos a buscar, que debe definirse con antelación. Se comienza

seleccionando K objetos al azar del conjunto total y asignándolos como patrones o centroides de las K clases que se van a buscar. A continuación, se calculan todas las distancias de todos los objetos restantes a todos los K centroides, y se asigna la pertenencia a cada objeto al clúster que tenga más cercano. Luego se recalcula el centroide de cada clúster, como la media de todos los objetos que lo componen, buscando minimizar el valor de una función de coste, que es un sumatorio de todos los sumatorios de las distancias euclídeas de los objetos de cada clase al centroide de su respectiva clase. Los dos pasos anteriores se repiten sucesivamente hasta que los centros de todos los grupos permanezcan constantes, o hasta que se cumpla alguna otra condición de parada (Benítez 2005).

PAM(Partitioning Around Medoids) es una extensión del algoritmo K-means, en donde cada grupo o clúster está representado por un medoide en vez de un centroide (Benítez 2005). Fue desarrollado para encontrar k clúster, el enfoque de PAM es determinar un objeto representativo para cada clúster. Este objeto representativo, llamado medoid, está destinado a ser el objeto ubicado más al centro del clúster. Una vez que se han seleccionado los medoides, cada objeto no seleccionado se agrupa con el medoide al que es más similar. Todos los valores de semejanza se dan como entradas para PAM. Finalmente, la calidad de una agrupación (es decir, la calidad combinada de los medoides elegidos) se mide por la disimilitud promedio entre un objeto y el medoide de su agrupación. Para encontrar los k medoides, PAM comienza con una selección arbitraria de k objetos. Luego, en cada paso, se realiza un intercambio entre un objeto seleccionado y un objeto no seleccionado, siempre que dicho intercambio resulte en una mejora de la calidad del agrupamiento (Raymond T. Ng y Jiawei Han 2002).

CLARA (Aplicaciones de Clustering Large) se basa en el muestreo. En lugar de encontrar objetos representativos para todo el conjunto de datos, CLARA dibuja una muestra del conjunto de datos, aplica PAM en la muestra y encuentra los medoides de la muestra. El punto es que, si la muestra se dibuja de manera suficientemente aleatoria, los medoides de la muestra se aproximarían a los medoides de todo el conjunto de datos. Para obtener mejores aproximaciones, CLARA extrae múltiples muestras y ofrece la mejor agrupación como salida. Aquí, para mayor precisión, la calidad de un clúster se mide en función de la disparidad promedio de todos los objetos en el conjunto de datos completo, y no solo de esos objetos en las muestras. Complementario a PAM, CLARA funciona satisfactoriamente para grandes conjuntos de datos (por ejemplo, 1.000 objetos en 10 clústeres). CLARA es más eficiente que PAM para valores grandes de n (Raymond T. Ng y Jiawei Han 2002). Este algoritmo está indicado para bases de datos con gran cantidad de objetos, y su principal motivación es la de minimizar la carga computacional, en detrimento de una agrupación óptima y precisa (Benítez 2005).

CLARANS (Clustering Large Applications based on Randomized Search) es más eficiente que los algoritmos existentes PAM y CLARA, los cuales motivan el desarrollo de CLARANS. Y el cálculo de la similitud entre dos polígonos utilizando la distancia de separación entre los rectángulos isotéticos de los polígonos (Raymond T. Ng y Jiawei Han 2002). Es un método k-medoide mejorado. En comparación con los antiguos algoritmos k-medoid, CLARANS es más eficaz y más eficiente. Por último, la agrupación con el coeficiente máximo silueta es elegido como el agrupamiento "natural". Desafortunadamente, el tiempo de ejecución de CLARANS es prohibitivo en bases de datos grandes. CLARANS supone que todos los objetos que se agrupan pueden residir en la memoria principal al mismo tiempo, lo que no es válido para grandes bases de datos. En primer lugar, el foco es lo suficientemente pequeño para ser residente en memoria y segundo, el tiempo de ejecución de CLARANS en los objetos del foco es significativamente menor que su tiempo de ejecución en toda la base de datos (Ester et al. 1996).

1.3.2 Algoritmos jerárquicos

Los algoritmos jerárquicos crean una descomposición jerárquica de la base de datos. Esta descomposición jerárquica está representada por un dendrograma. Un dendrograma es un árbol que iterativamente divide la base de datos en subconjuntos más pequeños hasta que cada subconjunto consiste en un solo objeto. Cada nivel del dendrograma representa un nodo del árbol. De este modo, se produce un conjunto de clústeres anidados organizados como un árbol jerárquico. En dicha jerarquía, cada nodo del árbol representa un grupo de la base de datos. El dendrograma puede ser creado desde las hojas hasta la raíz (enfoque aglomerativo) o desde la raíz hasta las hojas (enfoque divisivo) fusionando o dividiendo grupos en cada paso. En contraste con los algoritmos de partición, los algoritmos jerárquicos no necesitan k como una entrada. Sin embargo, se debe definir una condición de finalización que indique cuándo se debe finalizar el proceso de fusión o división. Un ejemplo de una condición de terminación en el enfoque aglomerativo es la distancia crítica entre todos los grupos de Q. Hasta ahora, el principal problema con los algoritmos de agrupación jerárquica ha sido la dificultad de derivar los parámetros apropiados para la condición de terminación. Esta es un valor de distancia crítica que es lo suficientemente pequeño como para separar todos los clústeres "naturales" y, al mismo tiempo, lo suficientemente grande como para que ningún grupo se divida en dos partes (Ester et al. 1996).

Ejcluster es un algoritmo jerárquico que deriva automáticamente una condición de terminación. Su idea clave es que dos puntos pertenecen al mismo grupo si puede caminar desde el primer punto hasta el segundo por un paso "suficientemente pequeño". Ejcluster sigue el enfoque divisivo. No requiere ninguna entrada de conocimiento de dominio. Además, los experimentos muestran que es

muy efectivo para descubrir clústeres no convexos. Sin embargo, el costo computacional de Ejcluster es O (n2) debido al cálculo de distancia para cada par de puntos. Esto es aceptable para aplicaciones como el reconocimiento de caracteres con valores moderados para n, pero es prohibitivo para aplicaciones en bases de datos grandes (Ester et al. 1996).

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) almacena para cada clúster un triplete de datos que contiene el número de objetos que pertenecen a ese grupo. El valor de la suma de todos los valores de los atributos de todos los objetos pertenecientes al grupo, y la suma de los cuadrados de los atributos de los objetos que pertenecen al clúster. Con esta información construye un árbol de grupos llamado CF-tree (Cluster Features tree). En cada nodo se indica el número de grupos que pertenecen a esa ramificación y cuáles son sus características. El procedimiento del algoritmo BIRCH es el siguiente:

• Generar un CF-tree inicial, leyendo los datos y asignándolos a una rama o a otra. Si la distancia entre un objeto nuevo y los anteriores se hace mayor que cierto parámetro T, se crea una rama nueva.

• Revisar el árbol creado para ver si es demasiado grande, y moldearlo modificando el valor del parámetro T. Si el valor de este parámetro se aumenta, las ramas del árbol se juntan al no haber distinción de grupos.

• Aplicar algún procedimiento de clustering, como el K-means, sobre la información contenida en los nodos de cada nivel.

•Redistribuir los datos según los centroides descubiertos en el paso anterior, logrando un mayor refinamiento en el agrupamiento.

Las principales desventajas del algoritmo BIRCH son su secuencialidad, lo cual puede provocar asignación a distintos clúster de objetos replicados, colocados en distintos lugares de la secuencia, y la fuerte dependencia del parámetro T, de forma que una mala elección de este valor puede generar la creación de falsas agrupaciones, o radicaciones duplicadas, o la asignación de objetos a un mismo nodo, cuando deberían estar en nodos distintos (Benítez 2005).

CURE es un algoritmo jerárquico muy eficiente y robusto (González 2010). El mismo utiliza una política mixta para el cálculo de la distancia entre dos grupos en cada iteración. Esta política es una especie de mezcla entre la política de centroides (donde la distancia entre dos clústeres es la distancia entre sus centros de gravedad) y la llamada política del Minimum Spanning Tree (MST) (donde la distancia entre dos grupos es igual a la distancia mínima entre dos puntos, uno en cada

grupo). Se basa en la selección de más de un elemento representativo de cada clúster. Como resultado, CURE es capaz de detectar grupos con múltiples formas y tamaños. Es un algoritmo de tipo aglomerativo, que comienza considerando todos los objetos como grupos independientes. A partir de ahí combina sucesivamente los objetos, agrupándolos en clúster. De cada uno de estos grupos, almacena los objetos extremos, desplazándolos hacia el centro del clúster mediante un factor de acercamiento que es el valor medio de todos los elementos que componen el grupo (Benítez 2005).

1.3.3 Algoritmos de agrupamiento basados en densidad

Los algoritmos basados en densidad encuentran un número de grupos comenzando por una estimación de la distribución de densidad de los nodos correspondientes. Se basan en la idea de que los objetos que forman una región densa se deben agrupar en un solo grupo. Estos algoritmos usan diversas técnicas para determinar dichos grupos. Pueden ser por grafos, basadas en histogramas, kernels, aplicando la regla K-NN, empleando los conceptos de punto central, borde o ruido(D. Pascual, F. Pla y S. Sánchez 2007). Con estos algoritmos se pueden detectar fácilmente y sin ambigüedades agrupaciones de puntos y puntos de ruido que no pertenecen a ningún conglomerado. La razón principal por la que se reconocen los clústeres es que dentro de cada clúster se tiene una densidad típica de puntos que es considerablemente más alta que fuera del clúster. Además, la densidad dentro de las áreas de ruido es menor que la densidad en cualquiera de los grupos. La idea clave es que, para cada punto de un clúster, la vecindad de un radio dado debe contener al menos un número mínimo de puntos, es decir, la densidad en el vecindario debe exceder algún umbral. La forma de un vecindario está determinada por la elección de una función de distancia para dos puntos p y q, denotada por dist (p, q). Por ejemplo, al usar la distancia de Manhattan en el espacio 2D, la forma del vecindario es rectangular. Se tiene en cuenta que este enfoque funciona con cualquier función de distancia para que se pueda elegir una función adecuada para alguna aplicación determinada (Ester et al. 1996; Lv et al. 2016).

Si p es un punto núcleo, éste forma un clúster junto a los otros puntos (núcleo o no) que sean alcanzables desde él. Cada clúster contiene al menos un punto núcleo; los puntos que no sean núcleos pueden hacer parte de un clúster. Pero estos corresponden a la "periferia" dado que no es posible alcanzar más puntos desde estos. La relación de ser alcanzable no es simétrica. Por definición ningún punto puede ser alcanzable desde un punto que no sea núcleo, sin importar la distancia a la que se encuentre. Es decir, un punto que no sea núcleo puede ser alcanzable, pero nada puede ser alcanzado desde éste. Dos puntos p y q están conectados densamente si existe un punto o tal que ambos p y q sean directamente alcanzables desde o. La relación estar densamente

conectado es simétrica. Un clúster, satisface por lo tanto dos propiedades. 1 Todos los puntos del clúster están densamente conectados entre sí.2 Si un punto A es densamente alcanzable desde cualquier otro punto B del clúster, entonces A también forma parte del clúster. Un punto que no sea alcanzable desde cualquier otro punto es considerado ruido (Ester et al. 1996).

DENCLUE (Density-based Clustering) usa el concepto de las funciones de influencia para catalogar la influencia que cada objeto ejerce sobre los elementos cercanos. Estas funciones de influencia son similares a las funciones de activación usadas para redes neuronales: superado cierto valor umbral de distancia entre objetos (distancia euclídea), la salida cambia de un estado a otro, normalmente entre un estado inactivo (0) y otro activo (1). El valor umbral viene definido por funciones de activación, como la gaussiana o la sigmoidal. La densidad se computa como la suma de todas las funciones de influencia de todos los objetos. Los clústeres se determinan mediante la detección de los máximos locales de densidad. Se consigue así un algoritmo de agrupamiento robusto, capaz de manejar datos ruidosos o erróneos (Benítez 2005).

DBSCAN requiere dos parámetros: una distancia e y el número mínimo de puntos requeridos para que una región se considere densa. El algoritmo comienza por un punto arbitrario que no haya sido visitado. La e-vecindad de este punto es visitada, y si contiene suficientes puntos, se inicia un clúster sobre el mismo. De lo contrario, el punto es etiquetado como ruido. El punto en cuestión puede pertenecer a otra vecindad que lo incluya en el clúster correspondiente. Si un punto se incluye en la parte densa de un clúster, su e-vecindad también forma parte del clúster. Así, todos los puntos de dicha vecindad se añaden al clúster, al igual que las e-vecindad de estos puntos que sean lo suficientemente densas. Este proceso continúa hasta construir completamente un clúster densamente conectado. Entonces, un nuevo punto no visitado se visita y procesa con el objetivo de descubrir otro clúster o ruido. DBSCAN visita cada punto de la base de datos, posiblemente varias veces (como candidatos a diferentes clústeres), ejecuta exactamente una consulta por cada punto (Ester et al. 1996).

OPTICS puede verse como una generalización de DBSCAN para múltiples rangos, reemplazando el parámetro e por el radio máximo de búsqueda. La motivación para la realización de este algoritmo se basa en la necesidad de introducir parámetros de entrada en casi todos los algoritmos de agrupamiento existentes que en la mayoría de los casos son difíciles de determinar. Además, en conjuntos de datos reales no existe una manera de determinar estos parámetros globales. El algoritmo OPTICS trata de resolver este problema basándose en el esquema del algoritmo DBSCAN creando un ordenamiento de la base de datos para representar la estructura del agrupamiento

basada en densidad. Puede hacer una representación gráfica del agrupamiento incluso para conjuntos de datos grandes (D. Pascual, F. Pla y S. Sánchez 2007).

1.3.4 Algoritmos basados en cuadrículas

Los algoritmos basados en cuadrícula son en su mayoría más simples que la interpolación; los enfoques estocásticos abarcan los métodos geoestadísticos («algoritmo basado en cuadrículas (grillas) - Schlumberger Oilfield Glossary» 2000). Se basan en una estructura de rejilla de múltiples niveles. Todo el espacio se cuantifica en un número finito de celdas en las que se realizan operaciones para la agrupación. La información resumida sobre el área cubierta por cada celda se almacena como un atributo de la celda. La principal ventaja de este enfoque es su rápido tiempo de procesamiento. Sin embargo, la información resumida conduce a la pérdida de información. La agrupación basada en rejillas reduce significativa la complejidad computacional, especialmente para agrupar conjuntos de datos muy grandes. El enfoque de clustering basado en rejillas difiere de los algoritmos de clustering convencionales en que no se refiere a los puntos de datos sino al espacio de valores que rodea los puntos de datos(«12. Grid-Based Clustering Algorithms» 2007). En general, un algoritmo de agrupamiento basado en cuadrículas típico consiste en los siguientes cinco pasos básicos (Grabusts y Borisov 2002).

1. Crear la estructura de la grilla, es decir, dividir el espacio de datos en un número finito de celdas.

2. Cálculo de la densidad celular para cada celda.

3. Clasificación de las celdas según sus densidades.

4. Identificar centros de clústeres.

5. Recorrido de celdas vecinas.

STING (Statistical Information Grid) particiona el espacio según niveles, en un número finito de celdas con una estructura jerárquica rectangular. De cada celda extrae la información de los objetos que allí encuentra. Esta es: media, varianza, mínimo y máximo de los valores y tipo de distribución de los objetos encontrados. Con cada nivel se vuelven a particionar las celdas, construyendo un árbol jerárquico a semejanza del algoritmo BIRCH. Acabada la partición del espacio hasta el nivel de detalle deseado, los clústeres se forman asociando celdas con información similar mediante consultas especializadas (Benítez 2005).

CLIQUE (Clustering In Quest) también realiza particiones del espacio según niveles, pero en esta ocasión cada nivel nuevo es una dimensión más, hasta alcanzar las n dimensión o características de los objetos. La estructura de partición es en forma de híper-rectángulos. El funcionamiento es el siguiente. Comienza con una única dimensión, y la divide en secciones, buscando las más densas, o aquellas donde se encuentran más objetos. A continuación, incluye la segunda dimensión en el análisis, particionando el espacio en rectángulos, y buscando los más densos. Luego sigue con cubos en tres dimensiones, y así sucesivamente. Cuando acaba con todas las características o dimensiones de los objetos, se definen los clústeres y las relaciones entre ellos mediante semejanza de densidades y otra información extraída, en todos los niveles o dimensiones (Benítez 2005).

1.3.5 Algoritmos basados en modelos

Se formula una hipótesis para cada uno de los conglomerados y la idea es encontrar el mejor ajuste de ese modelo para cada grupo. A menudo se basan en la suposición de que los datos se generan mediante una mezcla de distribuciones de probabilidad subyacentes (Joshi, Samal y Soh 2009).

1.3.5.1 Algoritmos basados en grafos

Los algoritmos basados en grafos utilizan grafos como representación de conocimiento. La fase de preparación de datos incluye una transformación de los datos a un formato de grafo(Cortez y Pro 2011). El objetivo principal es el de descubrir relaciones topológicas ocultas entre los datos. Matemáticamente, un grafo es un objeto formado por unos vértices y conjuntos de segmentos que conectan pares de vértices, análogamente los datos constituyen los nodos del grafo y las relaciones entre ellos se representan con los vértices (Benítez 2005; Wang et al. 2015; Janssens y Rozenberg 1982).

1.3.5.2 Algoritmos de agrupamiento basados en autómatas celulares

A partir de casi medio siglo atrás, mucho antes de que existiera la vida artificial, el pionero de la informática John Von Neumann investigó la cuestión del origen de la vida y trató de diseñar una máquina capaz de reproducirse. La idea en el diseño de auto-reproducción llevó a Von Neumann a inventar un sistema denominado: Autómatas Celulares, capaces de construir cualquier autómata a partir de un conjunto apropiado de instrucciones codificadas (López Salinas 2011; Adwan et al. 2013).

En la actualidad los autómatas celulares, desempeñan un papel importante en muchas áreas de las matemáticas, como la teoría de la representación, el análisis armónico, la teoría de grupos geométricos, la teoría de la probabilidad y los sistemas dinámicos. Dos nociones matemáticas

aparentemente no relacionadas son las de un grupo y la de un autómata celular. Sin embargo, estos últimos pueden ser utilizados para la representación y clasificación de grupos mediante matrices. Los autómatas celulares lineales sobre un grupo G con un alfabeto de dimensión infinita d sobre un campo K pueden representarse mediante matrices d × d con entradas en el anillo de grupo K (Fawcett 2008; Ceccherini-Silberstein y Coornaert 2010; Khomami, Rezvanian y Meybodi 2018).

Hormigas celulares combina los conocimientos de la agrupación basada en hormigas en el campo de la minería de datos y los autómatas celulares en el campo de la vida artificial con los principios de mapeo de datos del dominio de visualización de datos. Funciona en una maya cuadriculada bidimensional con celdas generalmente blancas o negras, donde la hormiga es la cabeza lectora e interpretadora que se desplaza de acuerdo a los estados que va encontrando hacia cualquiera de los puntos cardinales sobre ésta red, siguiendo un conjunto de reglas diseñadas especialmente para guiar su comportamiento (López Salinas 2011; Moere, Clayden y Dong 2006).

Autómatas celulares

Los Autómata Celulares se consideran sistemas dinámicos extendidos, que consisten de un número considerable de componentes simples idénticos con conectividad local. Y son definidos como un sistema que evoluciona en tiempo y espacio discretos, compuesto de una colección de celdas discretas y deterministas, ordenadas en fila, en forma de matriz o en tres dimensiones, que actualizan sus estados a lo largo del tiempo en base a los estados que tenían las celdas vecinas en el momento inmediato anterior, en otras palabras, el estado siguiente de una celda es determinado por el estado actual de ella y de sus celdas vecinas, donde cada celda se comporta como un autómata de estado finito. Un autómata finito es un modelo de sistema con entradas y salidas discretas, consiste en un conjunto finito de estados y un conjunto de transiciones entre esos estados. La entrada y el estado actuales del autómata determinan su conducta en el instante siguiente (López Salinas 2011; Adwan et al. 2013).

Un autómata celular (A.C.) es un modelo matemático para un sistema dinámico que evoluciona a pasos discretos. Es adecuado para modelar sistemas naturales que puedan ser descritos como una colección masiva de objetos simples que interactúan localmente unos con otros.

Se puede describir a un A.C. como una tupla de objetos caracterizados por los siguientes componentes:

- Una rejilla o cuadriculado (lattice) de enteros infinitamente extendida, y con dimensión de Z+. Cada celda de la cuadrícula se conoce como célula.

- Cada célula puede tomar un valor en Z a partir de un conjunto finito de estados k.
- Cada célula, además, se caracteriza por su vecindad, un conjunto finito de células en las cercanías de la misma.

Definición formal: Un AC se define por un cuádruple (C, Q, V, f), donde C es un espacio celular o red que puede tener n dimensiones n> = 1, cuando tiene dos dimensiones:

$$C = \{(i, j), 1 <= i <= r, 1 <= j <= c\}. (1)$$

Q es un conjunto finito de todos los estados posibles de las celdas; V es un vecindario finito y f es la función de transición local o regla local. Cada vez que las reglas se aplican a todas las celdas, se considera una generación (Pérez Betancourt, Rodríguez Puente y Kaunapawa Mufeti 2015).

Una celda es un elemento único de un espacio celular, la unidad más pequeña del espacio. El espacio celular es un espacio de rejilla formado por celdas y cada celda está en uno de varios estados predefinidos. La regla local es la regla que rige la transición entre estados. De particular importancia es determinar las reglas de transición más apropiadas para el fenómeno estudiado (Pérez Betancourt, Rodríguez Puente y Kaunapawa Mufeti 2015).

La definición de estado finito de una celda se denomina "local", ya que solo utiliza el entorno como su entrada. Vecindario se refiere a las celdas que rodean a una celda en particular, y tienen la capacidad de influir en el próximo estado de esa célula. La elección del vecindario influye en el comportamiento del espacio celular (Pérez Betancourt, Rodríguez Puente y Kaunapawa Mufeti 2015).

La selección de un vecindario apropiado depende de la relación entre los elementos. Los tipos más importantes de vecindarios son el vecindario Von Neumann y el vecindario Moore (Figura 2).

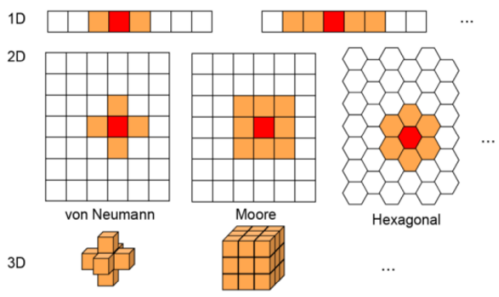

Figura 2. Los tipos de vecindad más importantes. Fuente (Pérez Betancourt, Rodríguez Puente y Kaunapawa Mufeti 2015)

Los autómatas celulares pueden ser usados para modelar numerosos sistemas físicos que se caractericen por un gran número de componentes homogéneos y que interactúen localmente entre sí. De hecho, cualquier sistema real al que se le puedan analogar los conceptos de "vecindad", "estados de los componentes" y "función de transición" es candidato para ser modelado por un A.C (Ahangaran, Taghizadeh y Beigy 2017).

Las características de los autómatas celulares harán que dichos modelos sean discretos en tiempo, espacio o ambos (dependiendo de la variante de la definición de A.C. que se use). Algunos ejemplos de áreas en donde se utilizan los autómatas celulares son:

- Modelado del flujo de tráfico y de peatones.
- Modelado de fluidos (gases o líquidos).
- Modelado de la evolución de células o virus como el VIH.
- Modelado de procesos de percolación.

Componentes de un Autómata Celular

Un espacio n dimensional (usualmente 2D) dividido en un número de subespacios conocidos como células, denominado malla. En el caso usual, la subdivisión es regular (Teselación Homogénea) Cada célula puede estar en un estado, perteneciente a un conjunto nito (o numerable) o de estados. Una configuración inicial, consistente en la distribución de estados de cada celda del autómata en t0.

Un criterio de vecindad determinado por las posiciones relativas de las células consideradas vecinas a una célula dada. Las reglas de evolución dicen cómo se darán los cambios de estado en cada celda, dependiendo del estado anterior propio y de su vecindad. Un Reloj de Cómputo conectado a todas las células, el cual generará los pulsos que aplican las reglas de evolución (Hawick 2013).

Comparación entre los algoritmos de agrupamiento

A continuación, se realiza una comparación entre los algoritmos de agrupamiento más destacadas en la literatura científica consultada. Atendiendo a las clasificaciones de basados en partición, jerárquicos y basados en densidad. Teniendo en cuenta los parámetros de entrada, la medida de similitud empleada y el tratamiento que estos le dan a la componente espacial de los datos. Las medidas de similitud empleadas en los algoritmos comparados están enfocadas a los datos temáticos, sin tener en cuenta la componente espacial. La dependencia espacial, que se conoce

como autocorrelación, suele ser ignorada en el caso de los algoritmos basados en partición y algoritmos jerárquicos. Los algoritmos basados en partición y los basados en densidad necesitan que se les especifiquen algunos parámetros de entrada. En la tabla 1 se muestra más detalladamente.

Tabla 1. Tabla comparativa de los algoritmos de agrupamiento. Fuente (Elaboración propia)

Algoritmos	**Autocorrelación Espacial**	**Parámetros de entrada**	**Medida de similitud empleada**
K-means	no se tiene en cuenta	k cantidad de clúster	temática
BIRCH	no se tiene en cuenta	no tiene	temática
DBSCAN	se tiene en cuenta	e distancia p cantidad mínima de puntos	temática

Los algoritmos basados en densidad tratan los puntos ruidosos y tienen en cuenta la componente espacial de los datos. Crean grupos demasiado densos, no varían la densidad entre los conglomerados y crean clústeres con forma esférica lo cual es muy restrictivo. A partir de la comparación realizada se decide diseñar un algoritmo de agrupamiento basándose en los modelos matemáticos autómatas celulares. Estos modelos tienen en cuenta la vecindad de los objetos en el espacio.

1.3.6 Agrupamiento de polígonos

Cualquiera de los diferentes algoritmos de agrupamiento mencionados anteriormente puede ser utilizado para el agrupamiento de polígonos en el espacio. Todos los algoritmos tienen sus ventajas y sus desventajas a la hora de realizar dichas agrupaciones. En este caso se identifica el enfoque basado en densidad como el más apropiado para agrupar polígonos. A los algoritmos basados en densidad no les es necesario conocer la cantidad de clústeres por adelantado, tal como se requiere en los algoritmos de partición, ni tienen la necesidad de almacenar información resumida como en los algoritmos basados en la cuadrícula. Los polígonos en el espacio geográfico y en muchos otros dominios responden naturalmente al enfoque basado en la densidad. Por ejemplo, en el espacio geográfico, tenemos un conjunto de polígonos contiguos y otro conjunto de polígonos ubicados lejos del primer conjunto. En una escala mayor, estos dos conjuntos pertenecerán a un grupo cada uno,

correspondiendo así a los grupos formados donde la densidad del objeto es alta (Gewali y Manandhar 2018; Day et al. 2017).

1.4 Medidas de similitud geométrica

Para medir lo similares (o disimilares) que son los individuos existe una enorme cantidad de índices de similaridad y de disimilaridad o divergencia. Todos ellos tienen propiedades y utilidades distintas y habrá que ser consciente de ellas para su correcta aplicación al caso que nos ocupe («Criterios de similitud. Similitud, divergencia y distancia» 2017).

Una vez hemos hecho una adecuada selección de las variables a considerar, cada uno de los individuos sujetos al análisis nos vendrá representado por los valores que tomen estas variables en cada uno de ellos. Este es el punto de partida de la clasificación. Para clasificar adecuadamente los individuos deberemos determinar lo similares o disimilares (divergentes) que son entre sí, en función de lo diferentes que resulten ser sus representaciones en el espacio de las variables

La mayor parte de estos índices serán o bien, indicadores basados en la distancia (considerando a los individuos como vectores en el espacio de las variables) (en este sentido un elevado valor de la distancia entre dos individuos nos indicará un alto grado de disimilaridad entre ellos); o bien, indicadores basados en coeficientes de correlación; o bien basados en tablas de datos de posesión o no de una serie de atributos.

Criterios basados en distancias como indicadores de disimilaridad

Se da, en general, el nombre de distancia o disimilaridad entre dos individuos i y j a una medida, indicada por d(i,j). Esta mide el grado de semejanza, o a mejor decir de desemejanza, entre ambos objetos o individuos, en relación a un cierto número de características cuantitativa y / o cualitativas. El valor de d(i,j) es siempre un valor no negativo, y cuanto mayor sea este valor mayor será la diferencia entre los individuos i y j.

Toda distancia debe verificar, al menos, las siguientes propiedades:

- (P.1) $d(i, j) > 0$ (no negatividad)
- (P.2) $d(i, i) = 0$
- (P.3) $d(i, j) = d(j, i)$ (simetría)

Diremos que una distancia es euclidiana cuando pueda encontrarse un espacio vectorial de dimensión igual o inferior a la dimensión del espacio de las variables en el que podamos representar a los individuos por puntos cuya distancia euclídea ordinaria coincida con la distancia utilizada.

Es decir, si existe un espacio vectorial Rm, con m < n (siendo n el número de variables consideradas para representar a los individuos) y dos puntos de ese espacio, Pi y Pj de coordenadas: Pi = (Pi1, Pi2 ,... Pi m) y Pj = (Pj1, Pj2 ,..., Pj m) verificándose que la distancia que estamos considerando entre los individuos i y j es igual a la distancia euclídea entre los puntos Pi y Pj en Rm; esto es: Si d (i,j) = (Pi - Pj), diremos que la distancia d (i,j) es euclidiana.

Cuando la distancia es euclidiana se verifica además que:

- (P.4) d (i, j) < d (i, t) + d (j, t) (desigualdad triangular)
- (P.5) d (i, j) > 0 " i ¹ j

Cualquier distancia que verifica la propiedad P.4 es llamada distancia métrica. Cumpliéndose, en consecuencia, que las distancias euclidianas son un subconjunto de las distancias métricas.

Si además de verificar la propiedad P.4 una distancia verifica la propiedad:

- (P.6) d(i,j) < max [(d(i,t), d(j,t)] (desigualdad triangular ultramétrica) se dice que la distancia es ultramétrica.

Existe una gran cantidad de distancias e indicadores de disimilaridad y no se puede disponer de una regla general que nos permita definir una disimilaridad conveniente para todo tipo de análisis. De las propiedades de que goce, de la naturaleza de las variables utilizadas y de los individuos estudiados y de la finalidad del análisis dependerá la adecuada elección de una u otra.

Como medidas de similitud geométrica se tienen tres funciones a partir de los criterios de: distancia en el espacio entre los objetos, la conectividad y el criterio del tamaño. (Manandhar 2016; Wang, Zhang y Fu 2016)
Para los datos temáticos la distancia euclidiana ponderada determinada por la ecuación:

$$d(X_i, X_j) = \sqrt{\sum_{k=1}^{p} W_k (X_{ik} - X_{jk})^2}$$

Para resolver el problema de la similitud de dos objetos en cuanto a su conectividad se obtiene la longitud de los lados comunes. Partiendo de que estos están conectados si tienen al menos un lado en común, de esta manera objetos vecinos son más similares.

$$\delta(P_i, P_j) = 1 - \frac{min(p_{P_i}, p_{P_j}) - x}{max(p_{P_i}, p_{P_j}) - x}$$

Donde p es el perímetro del objetos y x es la longitud de los lados en común en los objetos P_i y P_j.

La similitud según el criterio del tamaño se determina a partir de la ecuación:

$$\delta(P_i, P_j) = 1 - \frac{min(A_{P_i}, A_{P_j})}{max(A_{P_i}, A_{P_j})}$$

Donde A representa el área de los objetos.

1.5 Estratificación de territorios basada en agrupamiento

La estratificación se define como un conjunto de analogías que dan lugar a subconjuntos de unidades agregadas, denominadas estratos. Un estrato, por tanto, es un conjunto de unidades que presentan uno o varios parámetros, que los hacen similares entre sí y a la vez se diferencia de unidades correspondientes a otros estratos. Es decir, que en cada estrato existe una igualdad interna con diferencias o desigualdades externas (Batista Moliner et al. 2001a; Betancourt, Polanco y Rodríguez 2017).

Podrían citarse otras definiciones, pero en esencia, la estratificación es la separación de datos en categorías o clases que permite aislar la causa de un problema, identificando el grado de influencia de ciertos factores en el resultado de un proceso (Morales y Torres 2015).

Este procedimiento forma parte del proceso integrado de diagnóstico-intervención-evaluación, que como parte del enfoque epidemiológico de riesgo, es una estrategia útil para obtener un diagnóstico objetivo de acuerdo con el cual planificar las actividades de prevención y control de las distintas enfermedades, y sirve de base para categorizar metodológicamente e integrar áreas geoecológicas y grupos poblacionales de acuerdo a factores de riesgo (García Pérez y Alfonso Aguilar 2002).

La estratificación como proceso integrado de diagnóstico-intervención-evaluación, debe seguir los siguientes pasos:

1. Determinación del problema a estudiar.

2. Identificación y medición de las variables.

3. Aplicación del procedimiento de definición de estratos.

4. Identificación de los territorios y estratos más afectados.

5. Determinación de los posibles factores asociados al comportamiento.

6. Selección de intervenciones y adecuación de los servicios para la ejecución de las mismas.

7. Identificación de los indicadores de evaluación.

8. Ejecución de las intervenciones.

9. Evaluación de todo el proceso.

10. Monitoreo y ajuste de acuerdo con los problemas detectados.

Usos de la estratificación:

- Comprender de manera detallada la estructura de un grupo de datos, lo cual permitirá identificar las causas del problema y llevar a cabo las acciones correctivas convenientes.
- Examinar las diferencias entre los valores promedios y la variación entre diferentes estratos, y tomar medidas contra la diferencia que pueda existir.

1.5.1 Estratificación territorial en temas de salud

La estratificación territorial es un proceso que permite separar espacialmente los elementos representativos de los territorios (Batista Moliner et al. 2001a). Su principal utilidad es identificar regiones de un país determinado en las cuales las condiciones de vida desiguales estén relacionadas con diferentes problemas de salud. En Cuba, su principal utilidad es identificar áreas con mayores necesidades de salud. Esto se hace con la finalidad de ofrecer a cada territorio de manera justa los recursos que realmente necesita y efectuar acciones específicas ante cada situación (Betancourt, Polanco y Rodríguez 2017; Batista Moliner et al. 2001b).

Un elemento clave para la aplicación de esta metodología es la precisión al evaluar los límites político-administrativos que demarcan los territorios en sus diferentes niveles (localidad, municipio, provincia, país) y su relación con la distribución de los problemas de salud. En este sentido, es importante destacar que los fenómenos y condiciones que afectan la salud responden a los factores

que los originan y no necesariamente se distribuyen según esos límites territoriales (Barcellos y Buzai 2006).

En el análisis de las condiciones de vida se evidencia esta dificultad, sin embargo, la posibilidad de identificar de inmediato cuáles son las áreas geográficas que tienen peores condiciones de vida, así como la posibilidad de analizar cómo se presentan en estas los diferentes problemas de salud, resulta de extrema importancia para el Sistema de Salud; de manera que se utiliza la estratificación como una metodología muy eficaz para poner en evidencia estas desigualdades (Delgado Acosta et al. 2015).

En la literatura científica consultada (Pérez Betancourt, González Polanco y Fables Rodríguez 2017) se identificaron las técnicas de agrupamiento como unas de las más utilizadas y con mejores resultados para realizar la clasificación de los datos en los procesos de estratificación de territorios. También se identificaron los Sistemas de Información Geográfica como herramientas para realizar el proceso de estratificación, que serán analizadas posteriormente.

1.6 Herramientas, lenguajes y tecnologías a utilizar

En todo proceso investigativo es necesaria la utilización de sistemas de soporte que permitan organizar, facilitar, agilizar y automatizar las tareas generadas durante el transcurso de la investigación. En el proyecto de investigación Técnicas de Programación se potencia el uso de las herramientas, lenguajes y tecnologías empleadas que se describen a continuación y son de vital importancia para una correcta realización.

1.6.1 Lenguaje de modelado

UML es el acrónimo de Lenguaje Unificado de Modelado, este es el lenguaje estándar para visualizar, especificar, construir y documentar los artefactos de un sistema, utilizándose para el modelado del negocio y sistemas de software (Schefer-Wenzl y Strembeck 2013). También ofrece un estándar para describir los modelos, incluyendo aspectos conceptuales como procesos de negocio, funciones del sistema, expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

1.6.2 Herramientas CASE

CASE es el acrónimo de Computer Aided Software Engineering, las herramientas CASE son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y

desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (González 2010).

Visual Paradigm

Es una herramienta de diseño UML y herramienta CASE UML diseñada para ayudar al desarrollo de software. Ofrece un paquete completo necesario para la captura de requisitos, la planificación del software, la planificación de pruebas, el modelado de clases y el modelado de datos (Started 2010).

Las principales características de la herramienta son:

- Soporta las últimas versiones del UML.
- Posee un poderoso generador de documentación y reportes en formato PDF, HTML y MS Word.
- Proporciona soporte para varios lenguajes en la generación de código e ingeniería inversa como: Java, C++, CORBA IDL, PHP, Ada y Python.
- Disponibilidad en múltiples plataformas (Windows, Linux)
- Capacidades de ingeniería directa e inversa.

Se selecciona Visual Paradigm for UML en su versión 8.0 como herramienta para el modelado UML, esta permite trabajar de forma colaborativa, hacer un trabajo organizado y ágil. Posibilita la realización de los diagramas necesarios para el desarrollo y mejor entendimiento de la aplicación. Permite realizar ingeniería inversa a partir del código fuente. Al ser seleccionado el lenguaje de modelado UML, es conveniente tener en cuenta su vinculación con Visual Paradigm, resaltando que este último presenta abundantes tutoriales de UML y demostraciones interactivas.

1.6.3 Lenguaje de programación

Los lenguajes de programación son un conjunto de símbolos junto a un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Constan de un léxico, una sintaxis y una semántica (Louden 2004)

Partiendo de las características de la aplicación, se hace necesaria la selección de un lenguaje mediante el cual se pueda cumplir con los requisitos propuestos. Actualmente existen muchos

lenguajes para el desarrollo de aplicaciones, surgidos a partir de las tendencias y necesidades de los escenarios. El análisis se centró fundamentalmente en el lenguaje Python (Duque 2011).

Python

Se trata de un lenguaje interpretado o de script, con tipado dinámico, multiplataforma y orientado a objetos, que permite la programación imperativa, funcional y orientada a aspectos (Duque 2011).

Se seleccionó Python en su versión 2.7.5 porque su sintaxis es simple, clara y sencilla logrando de esta manera que los programas elaborados en este lenguaje parezcan pseudocódigo. Además, el tipado dinámico, el gestor de memoria, la gran cantidad de librerías disponibles y la potencia del lenguaje, entre otros, hacen que desarrollar una aplicación en Python sea sencillo y rápido.

Es importante tener en cuenta que al seleccionar QGIS como el software que soportará la integración de la solución, el lenguaje de programación más eficiente y conveniente para utilizar es Python; este SIG a partir de su versión 0.9 trae soporte del lenguaje Python que junto con el módulo PyQT4 entrega una solución óptima al desarrollo de plugins e interfaces gráficas de usuario.

PyQt

PyQt es un conjunto de enlaces Python para la biblioteca gráfica Qt. El módulo está desarrollado por la firma británica Riverbank Computing y se encuentra disponible para Windows, GNU/Linux y Mac OS bajo diferentes licencias. PyQt se distingue por su sencillez, por poseer un número importantes de herramientas que gestionen su manipulación y por su posibilidad de adecuarse a las distintas plataformas de software.

Utilizando PyQt en su versión 4.0 en el desarrollo de la herramienta informática, se puede crear una interfaz visual sencilla y sin muchos contratiempos, ya que PyQt posee los componentes visuales necesarios para su desarrollo, así como una abundante documentación y ejemplos.

Qt Designer

Qt Designer es una herramienta que permite acelerar el desarrollo de interfaces multilenguaje debido a que genera un archivo XML cuyo contenido es el formato de dicha interfaz, pudiéndolo convertir con los programas pertinentes a cada lenguaje. Esta herramienta provee características muy poderosas como la previa visualización de la interfaz, soporte para widgets y un editor de propiedades con gran variedad de opciones.

En correspondencia con la elección anterior de PyQt, se ha decidido emplear Qt Designer en su versión 4.7.4 como elemento que soportará el diseño de las interfaces. Su utilización permite la creación de las interfaces visuales de la aplicación de forma sencilla, además de la fácil manipulación de las variables de configuración de cada una de ellas.

1.6.4 Entorno de desarrollo integrado

Un entorno de desarrollo integrado (IDE, por sus siglas en inglés) es una herramienta que permite a los desarrolladores de software escribir sus programas en uno o más lenguajes. Consiste básicamente en una plataforma en la que se integran un editor de código, un compilador 7, un depurador 8 y una interfaz gráfica de usuario (Entornos de programación 2012).

Pycharm

Pycharm es un editor de código inteligente que proporciona soporte de primera clase para los lenguajes de programación: Python, JavaScript, CoffeeScript, TypeScript, HTML/CSS, AngularJS y Node.js, y otros menos utilizados. Pycharm funciona en las plataformas Windows, Mac OS y Linux con una única clave de licencia, también ofrece un espacio de trabajo con colores personalizables y atajos de teclado.

La decisión de seleccionar como IDE, Pycharm en su versión 3.4, está dada a que ofrece auto-completación inteligente de código, comprobación de errores sobre la marcha, soluciones rápidas y fácil navegación en el proyecto. Además, Pycharm mantiene la calidad del código bajo control con chequeos, asistencia a pruebas, refactorizaciones inteligentes, y una serie de inspecciones, lo que ayuda a escribir un código limpio y fácil de mantener (Jetbrains 2014).

1.6.5 Gestor de base de datos

Los Gestores de Bases de Datos (GBD) permiten crear y mantener una base de datos, además actúan como interfaz entre los programas de aplicación y el sistema operativo. El objetivo principal es proporcionar un entorno eficiente a la hora de almacenar y recuperar la información de las bases de datos.

Estos softwares facilitan el proceso de definir, construir y manipular bases de datos para diversas aplicaciones (Cobo 2007).

PostgreSQL

PostgreSQL es un sistema de GBD objeto-relacional, de propósito general, multiusuario y de código abierto, que soporta gran parte del estándar SQL 9 y ofrece modernas características como

consultas complejas, disparadores, vistas, integridad transaccional, control de concurrencia multiversión. Puede ser extendido por el usuario añadiendo tipos de datos, operadores, funciones agregadas, funciones ventanas y funciones recursivas, métodos de indexado y lenguajes procedurales (POSTGRESQL-3 GLOBAL DEVELOPMENT GROUP 2014).

Fue seleccionado PostgreSQL en su versión 9.0, teniendo en cuenta que es un GBD multiplataforma y de código abierto. Además, se valoró la existencia de la extensión PostGIS para permitir el trabajo con datos espaciales.

PostGIS

Para añadir soporte a PostgreSQL de objetos geográficos se utilizó la herramienta PostGIS en su versión 2.1.5. Este módulo convierte la base de datos objeto-relacional PostgreSQL en una base de datos espacial para su utilización en SIG.

PostGIS incluye un conjunto de operaciones para realizar consultas espaciales muy bien optimizadas por sus índices R-Tree 10 y su integración con el planificador de consultas de PostgreSQL. Utiliza las librerías Proj4 11 para dar soporte a la transformación dinámica de coordenadas y la librería GEOS 12 para realizar operaciones de geometría. Utiliza bloqueo a nivel de fila, permitiendo a múltiples procesos trabajar con las tablas espaciales concurrentemente y asegurando la integridad de los datos (PostGIS 2014).

PgAdmin

Como aplicación gráfica para gestionar el GBD PostgreSQL se utilizó la herramienta PgAdmin III en su versión 1.20.0. PgAdmin está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. Soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor y un agente para lanzar scripts programados. La conexión al servidor puede hacerse mediante conexión TCP/IP 13 o Unix Domain Sockets (en plataformas Unix), y puede encriptarse mediante SSL 14 para mayor seguridad (Robinson 2011).

1.6.6 Metodología de desarrollo

El desarrollo de un software no es una tarea fácil, se debe contar con un proceso bien detallado, para esto se necesita aplicar una metodología que sea capaz de llevar a cabo el control total del producto. Las metodologías de desarrollo de software surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un

producto de software. Dichas metodologías pretenden guiar a los desarrolladores, sin embargo, los requisitos de un software son muy variados y cambiantes, y se ha dado lugar a que exista una gran variedad de ellas (Letelier 2006).

Las metodologías se dividen en dos grupos, tradicionales (pesadas) y ágiles (ligeras). Las tradicionales, se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requiere una extensa documentación, pretendiendo prever todo de antemano, además dependen de un equipo de desarrollo bastante grande. En las ágiles es más importante lograr que un producto de software se desarrolle con la calidad requerida, que realizar una buena documentación. En este tipo de metodología el cliente está presente en todo momento y colabora con el proyecto, que posee un equipo de desarrollo pequeño (Letelier 2006).

Programación extrema

Programación extrema (XP, por sus siglas en inglés) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Además, se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (Joskowicz 2008).

Características de la metodología XP (Beck 2000):

- XP es una metodología “liviana” que no tiene en cuenta la utilización de elaborados casos de uso y la generación de una extensa documentación.
- XP tiene asociado un ciclo de vida y es considerado a su vez un proceso.
- La tendencia de entregar software en espacios de tiempo cada vez más pequeños con exigencias de costos reducidos y altos estándares de calidad.
- XP define Historias de Usuario (HU) como base del software a desarrollar, estas historias las escribe el cliente y describen escenarios sobre el funcionamiento del programa. A partir de las HU y de la arquitectura perseguida se crea un plan de liberaciones entre el equipo de desarrollo y el cliente.

Fases de la metodología XP:

- Planificación: Durante esta etapa se lleva a cabo el proceso de identificación y confección de las HU

- Diseño: Durante esta etapa se crea un diseño evolutivo que va mejorando incrementalmente y que permite hacer entregas pequeñas y frecuentes de valor para el cliente, basado principalmente en el desarrollo de las tarjetas Clase-Responsabilidad- Colaboración (CRC).

- Desarrollo: En esta fase se realiza la implementación de las HU que fueron seleccionadas por cada iteración. Al inicio se lleva a cabo un chequeo del plan de iteraciones por si es necesario realizar modificaciones. Como parte de este plan se crean tareas de ingeniería para ayudar a organizar la implementación exitosa de las HU.

- Pruebas: Esta fase permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones. XP divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñadas por los programadores, y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñada por el cliente final.

El ciclo de desarrollo consiste en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.

2. El programador estima el esfuerzo necesario para su implementación.

3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.

4. El programador construye ese valor de negocio.

5. Vuelve al paso 1

A partir del estudio de XP, se concluye que responde a las necesidades principales de tiempo, entorno y cantidad de programadores, e incluye al cliente como parte fundamental del equipo de desarrollo. Además, se preocupa más en el avance exitoso del producto que en generar una documentación detallada del mismo, siendo capaz de adaptarse a los cambios de requisitos en cualquier punto del ciclo de vida del proyecto. Estos elementos demuestran que es una

metodología factible para guiar el proceso de desarrollo de la solución, por lo que se decide incluirla en la propuesta.

1.7 Conclusiones parciales

- Con el desarrollo de este capítulo se obtuvo un mejor dimensionamiento del problema a partir del análisis de los principales conceptos asociados a la solución.
- El estudio de los algoritmos de agrupamiento facilitó identificar los más utilizados y eficientes.
- Se decidió diseñar un algoritmo de agrupamiento basándose en medidas de similitud geométricas y en los modelos matemáticos autómatas celulares.
- Estos modelos tienen en cuenta la vecindad de los objetos en el espacio de manera que tratan la autocorrelación espacial cumpliendo con la primera ley de la geografía.
- Para la implementación del software fue seleccionado un conjunto de herramientas y tecnologías basadas en licencias de software libre, encaminado a obtener un producto de alta independencia tecnológica y utilizable en diferentes plataformas.

CAPITULO 2: ALGORITMO DE AGRUPAMIENO GEOSPACIAL BASADO EN AUTOMATAS CELULARES

Introducción

En este capítulo se presenta una propuesta de solución del algoritmo de agrupamiento geoespacial basado en autómatas celulares y se describe cada una de las fases que lo conforman. Se especifican los requisitos de software y se obtienen los artefactos correspondientes a las fases de planificación y diseño de la metodología seleccionada.

2.1Propuesta de algoritmo

En la Figura 3 se muestra un esquema que representa los componentes del algoritmo que se propone, a continuación, se expone en detalle cada uno de los componentes de este algoritmo, así como las relaciones existentes entre los mismos.

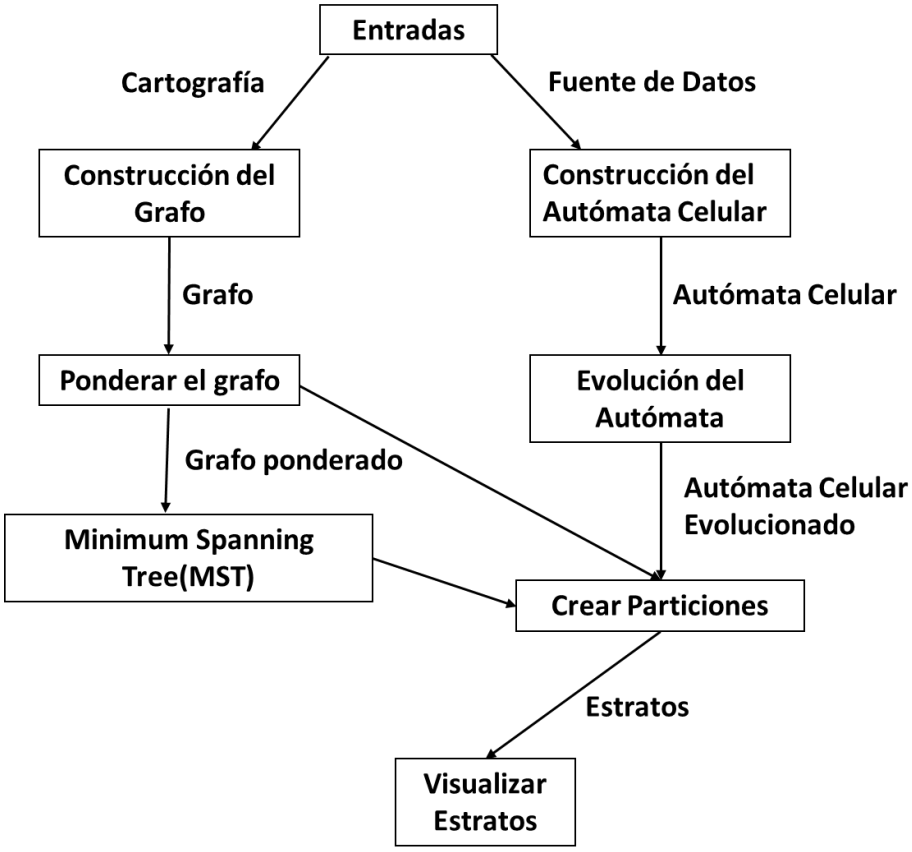

Figura 3. Modelo conceptual del algoritmo. Fuente: (Elaboración propia)

Se tiene como **entradas** del algoritmo tanto la cartografía como una fuente de datos. Como resultado final o **salidas** los grupos o estratos.

El algoritmo se desglosa en las siguientes fases:

- Construcción de un grafo a partir de la cartografía.
- Ponderación del grafo aplicando el criterio de conectividad.
- Obtención del árbol de cubrimiento mínimo(MST) de este grafo(Opcional).
- Construcción de un Autómata Celular a partir de la fuente de datos.
- Evolución de este Autómata mediante las reglas de evolución definidas.
- Obtención de las particiones según las reglas de partición definidas.

2.1.1 Construcción del grafo de similitud geoespacial

Un grafo, o grafo no dirigido, G = (V, E) se define como un conjunto V finito y no vacío de vértices y un multiconjunto E de aristas, donde cada arista $(v_i, v_j) \in V$ es un par no ordenado de vértices. Opcionalmente una arista puede tener un valor que la identifique y una lista de atributos. Cuando los elementos de E tienen multiplicidad uno, el grafo se denomina grafo simple.

La definición de grafo dirigido es similar a la anterior, con la única diferencia que las aristas son pares ordenados.

Nodos:

Los nodos del grafo representan los territorios los cuales son representados por polígonos. En estos se almacenan las características de los mismos.

Aristas:

Las aristas, en este caso dirigidas, representan las relaciones entre los territorios con lados en común.

Peso:

Representa la similitud geométrica entre nodos enlazados aplicando el criterio de conectividad donde se obtiene la longitud de los lados comunes. Partiendo de que estos están conectados si tienen al menos un lado en común, de esta manera polígonos vecinos son más similares.

$$\delta(P_i, P_j) = 1 - \frac{min(p_{P_i}, p_{P_j}) - x}{max(p_{P_i}, p_{P_j}) - x}$$

Donde p es el perímetro del polígono y x es la longitud de los lados en común en los polígonos P_i y P_j (Pérez Betancourt, González Polanco y Febles Rodríguez 2017).

Algoritmo 1 Construcción del grafo de similitud geoespacial

Entrada: Capa de características

Salida: Grafo de vecindad

1: Inicializar grafo
2: Inicializar un índice espacial vacío (index)
3: Llenar un diccionario con las características almacenadas en la capa de entrada (feature_dict)
4: **para todo** f ∈ feature_dict.values() **hacer**
5: insertar características (f) en el índice (index)
6: adicionar nodo al grafo con los parámetros (id, coordenadas (x;y))
7: **fin para**
8: **para todo** feature ∈ feature_dict.values() **hacer**
9: a = las intersecciones entre el índice espacial y cada feature del feature_dict
10: **para todo** b en a **hacer**
11: Intersecting_f ← feature_dict[b]
12: **si** feature ≠ intersecting_f y no coinciden sus geometrías **entonces**
13: añadir la arista que une los nodos al grafo y ponderarla mediante el algoritmo similitudConectividadGeometrica()
14: **fin si**
15: **fin para**
16: **fin para**

2.1.2 Creación y evolución del autómata celular

Un AC se define por un cuádruple (C, Q, V, f):

- C espacio celular o rejilla formada por un conjunto de células, en cada célula se almacena un objeto del conjunto de objetos $O = \{(Xi, \ldots, Xn)\}$.

- Q conjunto finito de estados en los que puede estar cada célula Q = {∅, ¬∅}.

- V vecindad, se define un radio de vecindad que corresponde al tipo vecindario Moore.
- f es la regla local que define que el autómata evoluciona en función de la distancia.

Algoritmo 2 Evolución del Autómata Celular
Entrada: Datos (espacio, radio, Vecindad = 2)
Salida: Autómata Celular evolucionado

```
1:  Inicializar un iterador (i)
2:  Inicializar un boolean (seguir)
3:  mientras i< longitud(autómata)-2 hacer
4:      si distancia(espacio[i], espacio[i+1]) > distancia(espacio[i],espacio[i+radioVecindad])
        entonces
5:          x ← espacio[i+1]
6:          espacio[i+1] ← espacio[i+2]
7:          espacio[i+2] ← x
8:          i ← i + 1
9:          seguir ←True
10: fin si
11:     i←i+1
12: fin mientras
13: aumentar la generación en 1
14: crear una copia de la generación y almacenarla en la matriz (generaciones)
15: retornar seguir
```

2.1.3 Reglas de partición

Para la obtención de los grupos partiendo de un Grafo de Similitud Geoespacial y la evolución de un Autómata Celular se definieron como reglas de partición las siguientes:

1. ∀ Xi y Xj ∈ O = {x1, ..., xn}, siendo O el conjunto de objetos a analizar. Cada objeto X tiene una vecindad espacial VE = {a1, ..., an} y una vecindad temática VT = {bi, ..., bn}. Si Xi ∈ VE(j) y VT(j) y viceversa entonces pertenecen al mismo grupo.
2. ∀ Xi, Xj, Xk. Si Xi, Xj ∈ A ^ Xj, Xk ∈ B → Xi, Xk ∈ A.

3. ∀ Xi y Xj ∈ O = {x1, ..., xn}, siendo O el conjunto de objetos a analizar. Cada objeto X tiene una vecindad espacial VE = {a1, ..., an} y una vecindad temática VT = {bi, ..., bn}. Si Xi ¬ ∈ VE(j) y VT(j) y viceversa entonces no pertenecen al mismo grupo.

Algoritmo 3 Implementación de las Reglas de Partición

Entrada: Grafo de Similitud Geoespacial (graph), Autómata Celular(CA)

Salida: Los grupos obtenidos

```
Inicializar diccionario vacío (labels)
Inicializar diccionario vacío (grupos)
Obtener una lista del autómata con los identificadores de los objetos(espacio)
Inicializar un contador (k)
para todo i ∈ range (len(espacio)) hacer
    x ← pertenecenGrupo(espacio[i]-1)
    labels[k] ← x
    k← k+1
fin para
k ←0
mientras len (labels) > 0 hacer
    x ← labels.keys()
    z ← labels.pop(x[0])
Si len (z) > 0entonces
    para todo j ∈range (1, len (x)) hacer
        si len (z.intersection (labels.get(x[j]))) entonces
            z ← z.union (z,labels.pop(x[j]))
fin si
        fin para
    grupos [k] ← z
    k ← k+1
fin si
fin mientras
retornar grupos
```

2.2 Análisis de la complejidad algorítmica

En este epígrafe se calcula la complejidad temporal, según el enfoque teórico, de los algoritmos propuestos en el epígrafe anterior. Para el cálculo de la misma se determina la complejidad de cada

paso del algoritmo y se utilizan las reglas de la suma y la multiplicación de la notación asintótica “o grande” (O) para obtener la complejidad de dichos algoritmos.

2.2.1 Análisis del algoritmo 1: Construcción del grafo de similitud geoespacial

A continuación, se muestra la complejidad de cada paso del algoritmo.

- Pasos 1-2 O (1)
- Paso 3 O(n) n la cantidad de features almacenadas en la capa
- Pasos 4-7 O(n)
    - Paso 5 O (1)
    - Paso 6 O (1)
- Pasos 8-16 O(n*z)
    - Paso 9 O (1)
    - Pasos 10-15 O(logn)
        - Paso 11 O (1)
        - Pasos 12-14 O (1)
            - Paso 13 O (1)

Aplicando la regla de la suma se puede concluir que la complejidad del algoritmo es el Max(O(n*z), O(nlogn), O (1)) o sea O(nlogn).

2.2.2 Análisis del algoritmo 2: Evolución del Autómata Celular

A continuación, se muestra la complejidad de cada paso del algoritmo.

- Pasos 1-2 O (1)
- Pasos 3-12 O(n)
    - Pasos 4-10 O (1)
        - Pasos 5-9 O (1)
    - Paso 11 O (1)
- Paso 13 O (1)
- Paso 14 O(n)
- Paso 15 O (1)

Aplicando la regla de la suma se puede concluir que la complejidad del algoritmo es el Max(O(n), O(n), O (1)) o sea O(n).

2.2.3 Análisis del algoritmo 3: Implementación de las Reglas de Partición

A continuación, se muestra la complejidad de cada paso del algoritmo.

- Pasos 1-4 O (1)
- Pasos 5-9 O(n^2)
    - Paso 6 O(n)
    - Pasos 7-8 O (1)
- Paso 10 O (1)
- Pasos 11-23 O(n^2)
    - Pasos 12-13 O (1)
    - Pasos 14-22 O(n^2)
        - Pasos 15-19 O(n^2)
            - Pasos 16-18 O(n)
            - Paso 17 O (1)
    - Pasos 20-21 O (1)
- Paso 24 O (1)

Aplicando la regla de la suma se puede concluir que la complejidad del algoritmo es el Max(O(n^2), O(n^2), O (1)) o sea O(n^2).

2.3 Instanciación de algoritmo en el SIG QGIS

En este epígrafe se instancia la solución en el Sistema de Información Geográfico QGIS y para ello se determinan los requisitos de software. Además, se generan los artefactos resultantes de las fases planificación y diseño de la metodología utilizada.

2.3.1 Requisitos de software

“Un requisito es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste” (Sommerville 2005). La calidad con que se realiza la captura de los requisitos afecta todo el proceso de desarrollo del software repercutiendo en el resto de las fases de desarrollo del mismo. Además, contribuye a tomar mejores decisiones de diseño y de arquitectura.

Requisitos funcionales

Un requisito funcional (RF) define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas. Los requerimientos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone que un sistema debe cumplir. Estos son complementados por los requisitos no funcionales, que se enfocan en cambio, en el diseño o la implementación (Sommerville 2005).

A continuación, se muestran los RF identificados:

- **RF 1**: Obtener capa de características a través del QGIS.
- **RF2**: Construir el grafo de similitud geoespacial.
- **RF3**: Construir árbol de cubrimiento mínimo(MST).
- **RF4**: Importar datos desde una fuente de datos.
- **RF5**: Construir Autómata Celular.
- **RF6**: Evolucionar Autómata Celular.
- **RF7**: Visualizar estratos.

Requisitos no funcionales

Los requisitos no funcionales (RNF) son propiedades o cualidades que el sistema debe tener. Estas propiedades o cualidades se refieren a las características que hacen al sistema estable, usable, rápido, confiable y escalable (Sommerville 2005).

A continuación, se muestran los RNF identificados:

Requisitos de Software

- **RNF 1**: Se debe tener instalada la herramienta QGIS en su versión 2.6 o superior.
- **RNF 2**: Se debe tener instalado el GBD PostgreSQL en su versión 9.0 o superior.
- **RNF 3**: Se debe tener instalado el módulo Postgis en su versión 2.1.5 o superior.

Requisitos de Hardware

- **RNF 4**: La estación de trabajo debe contar con al menos 1,0 GB de Random Access Memory (RAM, por sus siglas en inglés).
- **RNF 5**: La capacidad mínima de espacio en disco debe ser 2.0 GB.

Requisitos de Usabilidad

- **RNF 6**: La aplicación podrá ser usada por cualquier usuario con conocimientos básicos sobre informática

Restricciones de diseño e implementación

- **RNF 9**: Se hace uso de la herramienta QGIS en su versión 2.6 e IDE Pycharm 3.4.
- **RNF 10**: El lenguaje de programación usado para la implementación es Python.

2.3.2 Fase de planificación

La metodología XP define como fase inicial la planificación. Durante esta etapa se lleva a cabo el proceso de identificación y confección de las historias de usuario, así como la familiarización del equipo de trabajo con las tecnologías y herramientas seleccionadas para el desarrollo del software. También el cliente especifica la prioridad en que se deben implementar las historias de usuario, además de una estimación del esfuerzo que costará. El resultado de la fase es un plan de entregas donde se realiza una estimación de las versiones que tendrá el producto en su realización, de manera tal que guíe el desarrollo del mismo (Beck 2000).

Historias de usuario

Las historias de usuarios (HU) es la técnica utilizada en XP para especificar los requisitos del software; en ellas el cliente describe brevemente las características que el sistema debe poseer. Se realiza una por cada característica principal del sistema. El tratamiento de las HU es muy dinámico y flexible. En cualquier momento pueden reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada HU es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (Letelier 2006).

Se identificaron siete HU luego de obtener las principales funcionalidades del sistema. A continuación, en las tablas 2 y 3 se muestra una breve descripción de dos de ellas.

Tabla 2. Historia de usuario: Evolucionar Autómata Celular. Fuente: (Elaboración propia)

Historia de Usuario "Evolucionar Autómata Celular"	
Número:6	**Nombre Historia de Usuario:** Evolucionar Autómata Celular
Usuario: Experto	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Puntos Estimados: 3	**Iteración Asignada:** 2

Programador responsable: Nailee Vidal Abreu, Bárbaro E. Martínez Fernández
Descripción: La aplicación debe ser capaz de evolucionar el autómata, a partir de: • el autómata celular.
Observaciones:

Tabla 3. Historia de usuario: Crear particiones. Fuente: (Elaboración propia)

Historia de Usuario "Crear particiones"	
Número: 7	**Nombre Historia de Usuario:** Crear particiones
Usuario: Experto	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Puntos Estimados: 2	**Iteración Asignada:** 4
Programador responsable: Nailee Vidal Abreu, Bárbaro E. Martínez Fernández	
Descripción: La aplicación debe ser capaz de crear particiones, a partir de: • el autómata celular evolucionado. • el grafo ponderado. • el árbol de cubrimiento mínimo(MST).	
Observaciones:	

Estimación de esfuerzos por historias de usuario

En el presente epígrafe se realiza la estimación del esfuerzo por HU, se hace necesario tener en cuenta que estas deben ser programadas en un tiempo de una a tres semanas. Si la estimación es superior a tres semanas, se divide en dos o más HU. Si es menor de una semana, se combina con otra HU. Estas estimaciones permiten tener una medida de la velocidad del proyecto y ofrecen una guía a la cual ajustarse. Los resultados estimados se muestran en la tabla 4.

Tabla 4. Estimación de esfuerzos por Historia de Usuario. Fuente: (Elaboración propia)

Historia de usuario	**Puntos de estimación (semanas)**
HU 1: Obtener capa de características a través del QGIS.	1

HU 2: Construir el grafo de similitud geoespacial.	2
HU 3: Construir árbol de cubrimiento mínimo(MST)**.**	2
HU 4: Importar datos desde una fuente de datos.	1
HU 5: Construir Autómata Celular.	2
HU 6: Evolucionar Autómata Celular.	3
HU 7: Crear particiones.	2

Plan de iteraciones

Una vez finalizadas las HU se debe crear un plan de iteraciones, indicando cuáles se desarrollarán en cada iteración. En la Tabla 5 se muestra cómo quedó definido el plan de iteraciones para la solución propuesta.

Tabla 5. Plan de duración de las iteraciones. Fuente: (Elaboración propia)

Iteraciones	**Orden de las historias de usuario a implementar**	**Duración de las iteraciones (semanas)**
Iteración 1	Obtener capa de características a través del QGIS. Importar datos desde una fuente de datos.	2
Iteración 2	Construir el grafo de similitud geoespacial. Ponderar grafo. Construir Autómata Celular. Evolucionar Autómata Celular.	3
Iteración 3	Construir árbol de cubrimiento mínimo(MST).	3
Iteración 4	Crear particiones.	2
Total		10

2.3.2 Fase de diseño

La metodología de desarrollo XP plantea prácticas especializadas que accionan directamente en la realización del diseño para lograr un sistema robusto y reutilizable. Se trata en todo momento de conservar su simplicidad, es decir, crear un diseño evolutivo que va mejorando incrementalmente y

que permite hacer entregas pequeñas y frecuentes de valor para el cliente, basado principalmente en el desarrollo de las tarjetas Clase-Responsabilidad-Colaboración (CRC).

Tarjetas Clase-Responsabilidad-Colaboración

Las tarjetas CRC son utilizadas para representar las responsabilidades de las clases y sus interacciones. Estas tarjetas permiten trabajar con una metodología basada en objetos, permitiendo que el equipo de desarrollo completo contribuya en la tarea del diseño. En cada tarjeta CRC el nombre de la clase se coloca a modo de título, las responsabilidades se colocan a la izquierda y las clases que se implican en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente.

Una clase es cualquier persona, evento, concepto, pantalla o reporte. Las responsabilidades de una clase son las cosas que conoce y las que realizan, sus atributos y métodos. Los colaboradores de una clase son las demás clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades (Casas y Reinaga 2008).

En las tablas 6 y 7 se muestran las tarjetas CRC correspondientes a las clases Autómata Lineal y Grafo Vecindad.

Tabla 6. Tarjeta CRC para la clase Autómata Lineal. Fuente: (Elaboración propia)

Clase: Autómata Lineal	
Responsabilidad	Colaboración
• Evolucionar el autómata.	Cellular Automata.

Tabla 7. Tarjeta CRC para la clase Grafo Vecindad. Fuente: (Elaboración propia)

Clase: Grafo Vecindad	
Responsabilidad	Colaboración
• Construir el grafo por vecindad.	Similitud Conectividad.

Arquitectura de software

La arquitectura de software es la definición y estructuración de una solución que cumple con los requisitos técnicos y operativos. Optimiza atributos que implican una serie de decisiones, tales como la seguridad, el rendimiento y la capacidad de administración. Estas decisiones en última instancia, afectan la calidad de la aplicación, el mantenimiento, el rendimiento y el éxito global (Pressman 2005).

Estilo arquitectónico a utilizar

Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software. Estos permiten expresar un esquema de organización estructural esencial para un sistema de software (Pressman 2005). En este trabajo se hace uso del estilo arquitectónico en capas, logrando que el sistema quede organizado y así tener un orden lógico en la programación del mismo.

Arquitectura en capas

La arquitectura en capas se define como una organización jerárquica donde cada capa proporciona servicios a la inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Con esto se logra abstraer las funcionalidades de una capa de manera tal que pueda ser totalmente remplazada sin afectar a las otras, solamente cambiar las referencias de las implicadas en el cambio (Peláez 2009).

En la figura 4 se presenta una imagen de la arquitectura de la solución.

Figura 4. Estilo arquitectónico. Fuente: (Elaboración propia)

Capa de presentación: es la parte de la aplicación con que el usuario interactúa, por lo que deberá cumplir muchos requisitos. Estos requisitos abarcan factores generales como la facilidad de uso, rendimiento, diseño e interactividad. Es importante que la aplicación tenga un buen diseño para apoyar una experiencia de usuario intuitiva, desde el principio, ya que la experiencia del usuario es influenciada por muchos aspectos diferentes de la arquitectura de la aplicación.

Capa de negocio: es donde residen las clases gestoras de la información, se reciben las peticiones del usuario y se envían las respuestas a la capa de presentación. Se nombra capa de negocio porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes del usuario y presentar los resultados obtenidos, y con la capa de acceso a datos, para enviar datos que necesitan persistirse en la base de datos o recibirlos de la misma.

Capa de acceso a datos: está constituida por las clases gestoras del acceso a datos, encargadas de acceder a los mismos y realizan todo el almacenamiento de la información. Esta capa recibe solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Modelo de la vista lógica de la estructura del sistema

El modelo organiza una descripción de la arquitectura de software utilizando la vista lógica. Ofrece soporte a los requerimientos funcionales, lo que el sistema debe proveer en términos de servicios a sus usuarios (Barrera León 2011).

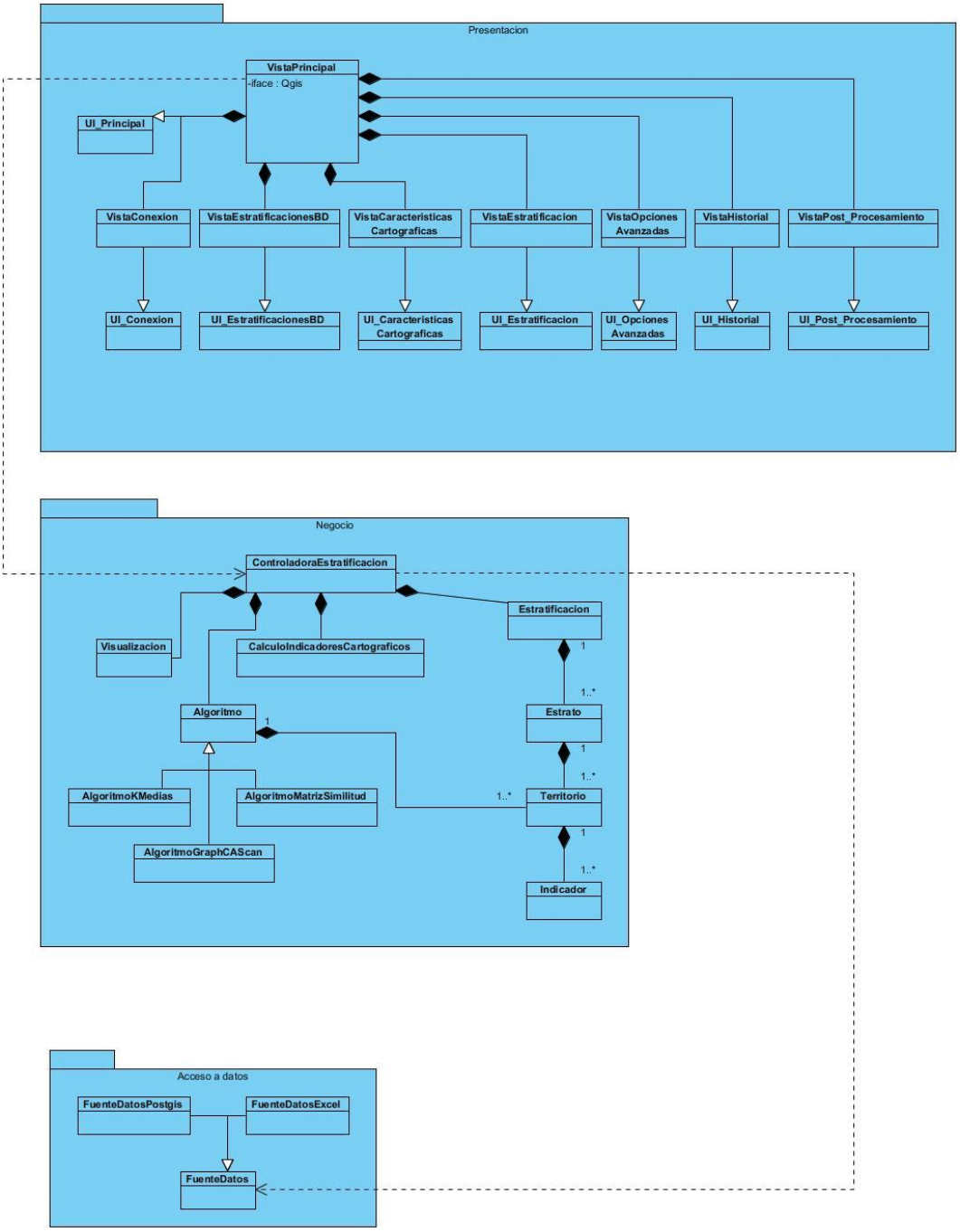

Figura 5. Modelo de la vista lógica de la estructura del sistema. Fuente (Elaboración propia)

2.6 Conclusiones parciales

- El algoritmo de agrupamiento basado en autómatas celulares y medidas de similitud geométricas propuesto integra la componente espacial mediante el tratamiento de la autocorrelación, lo que facilita la obtención de información precisa para la toma de decisiones en salud.

- La creación del grafo de vecindad utilizando medidas de similitud geométrica permite que al comparar los datos solo se comparen los vecinos reduciendo el tiempo de ejecución y haciendo al algoritmo escalable.

- La instanciación de la solución propuesta en el SIG QGIS permitió un mayor entendimiento de las necesidades del cliente.

- Mediante la descripción de las siete HU divididas por iteraciones se logró una mejor organización del trabajo y se establecieron 10 semanas como tiempo de duración para las cuatro iteraciones que se realizan.

- La utilización del estilo arquitectónico en capas permitió una mejor estructuración de la aplicación.

CAPÍTULO 3: VERIFICACIÓN DE LA PROPUESTA DE SOLUCIÓN

Introducción

En el siguiente capítulo se realizan las pruebas de software definidas por la metodología seleccionada, así como la experimentación sobre datos sintéticos. Se realiza un caso de estudio para verificar la validez de los resultados de la solución propuesta y analizan los resultados experimentales.

3.1 Pruebas de software

Cuando se desarrolla una solución informática se deben realizar una gran cantidad de pruebas para verificar que el código esté correcto. Estas pruebas normalmente tienen que ser ejecutadas en varias ocasiones y se ven afectadas por los cambios que se introducen conforme se va construyendo la solución. XP divide las pruebas en dos grupos: pruebas de aceptación, o pruebas funcionales diseñadas por el cliente final, destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida y pruebas unitarias, encargadas de verificar el código y diseñadas por los programadores.

3.1.1 Pruebas de aceptación

Las pruebas de aceptación XP son especificadas por el cliente, y se centran en las características y funcionalidades generales del sistema, que son visibles y revisables por parte del usuario. Estas pruebas derivan de las HU que se han implementado como parte de la liberación del software (Joskowicz 2008).

Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Así mismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una HU no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información (Joskowicz 2008).

Casos de prueba

En las tablas 8 y 9 se muestran los casos de prueba de aceptación aplicados a las HU Obtener capa de características a través del QGIS y Construir el grafo de similitud geoespacial.

Tabla 8. Caso de prueba de aceptación de Obtener capa de características atreves de QGIS. Fuente: (Elaboración propia)

Caso de prueba de aceptación	
Código: HU1_P1	**Historia de Usuario:** 1
Nombre: Obtener capa de características a través del QGIS.	
Descripción: Prueba para validar la funcionalidad obtener capa de características a través del QGIS.	
Condiciones de ejecución: Cargar la cartografía.	
Resultados esperados: Obtiene a través del sistema de información geográfico QGIS las características de la cartografía.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 9. Caso de prueba de aceptación de Construir el grafo de similitud geoespacial. Fuente: (Elaboración propia)

Caso de prueba de aceptación	
Código: HU2_P1	**Historia de Usuario:** 2
Nombre: Construir el grafo de similitud geoespacial.	
Descripción: Prueba para validar la funcionalidad construir el grafo de similitud geoespacial.	
Condiciones de ejecución: Tener como entrada la capa de la cartografía	
Resultados esperados: Crea un grafo de similitud geoespacial a partir de los datos de la cartografía	
Evaluación de la prueba: Prueba satisfactoria.	

Análisis de los resultados

Para validar que el resultado obtenido por el sistema coincide con el resultado esperado por el cliente se diseñaron un total de 7 casos de prueba de aceptación en conjunto cliente-desarrolladores. De este total, 6 arrojaron el resultado esperado mientras que 1 prueba resultó fallida, las funcionalidades que respondían a esta prueba fueron tratadas en la siguiente iteración y al volver a aplicar las pruebas de funcionalidad mostraron un resultado exitoso. Finalmente se obtuvieron un total 7 pruebas satisfactorias de 7 casos de prueba aplicados.

3.1.2 Pruebas de caja blanca

Las pruebas de caja blanca se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al código fuente. Se escogen distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa cerciorándose que se devuelvan los valores de salida adecuados (Pressman 2005).

Las pruebas de caja blanca intentan garantizar que:

- Se ejecutan al menos una vez todos los caminos independientes de cada módulo.
- Se utilizan las decisiones en su parte verdadera y en su parte falsa.
- Se ejecuten todos los bucles en sus límites.
- Se utilizan todas las estructuras de datos internas.

La técnica utilizada dentro de las pruebas de caja blanca fue camino básico. En la figura 7 se enumeran las sentencias de código del método evolucion().

```
def evolucion(self):
    i=0
    seguir=False
    while i<(len(self.espacio)-2):


        if self.distancia(self.espacio[i],self.espacio[i+1]) > self.distancia(self.espacio[i],self.espacio[i+self.getRadioVecindad()-1]):
            x=self.espacio[i + 1]
            self.espacio[i + 1] = self.espacio[i + 2]

            self.espacio[i + 2] = x
            i = i + 1
            seguir=True


        i+=1
    self.setGeneration()
    self.copia()

    return seguir
```

Figura 6. Código del método evolucion(). Fuente: (Elaboración propia)

Luego de haberse construido el grafo se realiza el cálculo de la complejidad ciclomática mediante las tres fórmulas descritas a continuación, las cuales deben arrojar el mismo resultado para asegurar que el cálculo de la complejidad sea correcto.

1. La complejidad ciclomática coincide con el número de regiones del grafo de flujo.
2. La complejidad ciclomática, V(G), de un grafo de flujo G, se define como V(G) = Aristas - Nodos + 2.
3. La complejidad ciclomática, V(G), de un grafo de flujo G, también se define como V(G) = Nodos de predicado (nodos de los cuales parten dos o más aristas) + 1.

A partir del grafo de flujo del método evolucion() que se presenta en la figura 8, la complejidad ciclomática sería:

- Como el grafo tiene tres regiones, V(G) = 3
- Como el grafo tiene 8 aristas y 7 nodos, V(G) = 8 - 7 + 2 = 3
- Como el grafo tiene 2 nodos de predicado, V(G) = 2 + 1 = 3

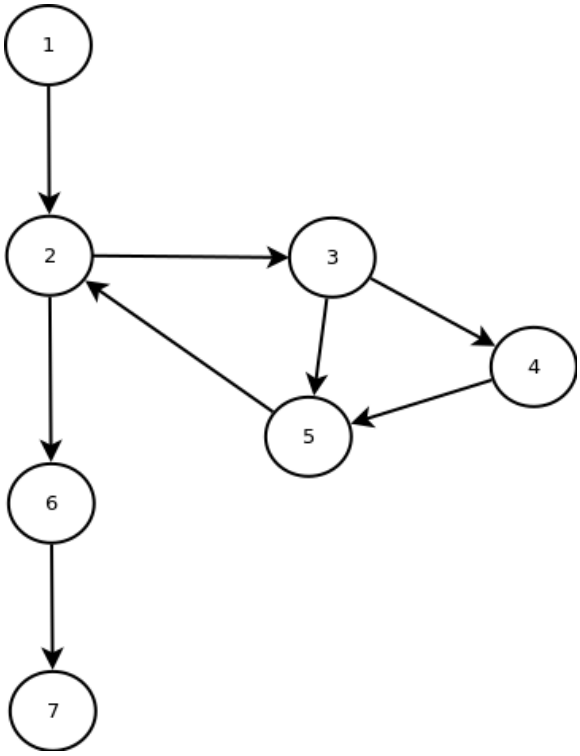

Figura 7. Grafo de flujo del método evolucion(). Fuente: (Elaboración propia)

Dado a que el cálculo de las tres fórmulas anteriormente mencionadas arrojó el mismo resultado se puede plantear que la complejidad ciclomática del método es 3. Esto significa que existen 3 posibles caminos por donde el flujo puede circular. Este valor representa el número mínimo de casos de pruebas para el procedimiento tratado.

Caminos básicos identificados:

- Camino 1: 1-2-6-7
- Camino 2: 1-2-3-5-2-6-7
- Camino 3: 1-2-3-4-5-2-6-7

Para cada camino básico determinado se realiza un diseño de caso de prueba.

Tabla 10. Caso de prueba para el camino básico #1. Fuente: (Elaboración propia)

Caso de prueba para el camino básico #1 (1-2-6-7)	
Descripción	Prueba para comprobar los resultados de la función evolución() en caso que de que la cantidad de elementos de la lista espacio menos dos sea menor o igual al valor de i.
Condición de ejecución	• longitud de self.espacio – 2 < = i
Entrada	• i > = (len(self.espacio) - 2)
Resultado	• seguir = False
Resultado de la prueba	Prueba satisfactoria

Tabla 11. Caso de prueba para el camino básico #2. Fuente: (Elaboración propia)

Caso de prueba para el camino básico #2 (1-2-3-5-2-6-7)	
Descripción	Prueba para comprobar los resultados de la función evolución() en caso que la distancia entre el elemento de la lista de espacio en la posición i y el siguiente sea menor que la distancia entre el elemento de la lista espacio en la posición i y los de su radio de vecindad.
Condición de ejecución	• distancia entre self.espacio[i] y self.espacio[i+1] < distancia entre self.espacio[i] y self.espacio[i+self.getRadioVecindad-1]
Entrada	• self.distancia(self.espacio[i], self.espacio[i+1]) < self.dsitancia(self.espacio[i], self.espacio[i+self.getRadioVecindad-1])
Resultado	• seguir = False
Resultado de la prueba	Prueba satisfactoria

Tabla 12. Caso de prueba para el camino básico #3. Fuente: (Elaboración propia)

Caso de prueba para el camino básico #2 (1-2-3-4-5-2-6-7)	
Descripción	Prueba para comprobar los resultados de la función evolución() en caso que la distancia entre el elemento de la lista de espacio en la posición i y el siguiente sea mayor que la distancia entre el elemento de la lista espacio en la posición i y los de su radio de vecindad.
Condición de ejecución	• distancia entre self.espacio[i] y self.espacio[i+1] > distancia entre self.espacio[i] y self.espacio[i+self.getRadioVecindad-1]
Entrada	• self.distancia(self.espacio[i], self.espacio[i+1]) > self.disitancia(self.espacio[i], self.espacio[i+self.getRadioVecindad-1])
Resultado	• seguir = True

Resultado de la prueba	Prueba satisfactoria

3.2 Experimentación sobre datos sintéticos

Con el fin de valorar los resultados de la solución propuesta se decide aplicarla sobre un conjunto de datos sintéticos donde se realiza un proceso de estratificación a los 168 municipios de Cuba, según la división política administrativa vigente. Se obtuvo una capa vectorial desde la Infraestructura de datos espaciales de Cuba con los polígonos que representan a cada territorio escogido para el análisis.

Se realizó el proceso de estratificación utilizando la medida de similitud geométrica conectividad. Se obtuvieron 107 estratos, 83 de ellos conformados por un solo territorio. Estos resultados demuestran que el algoritmo funciona sobre un conjunto de datos grande, aunque devuelve demasiados estratos aislados. En las figuras 9 y 10 se muestran los resultados.

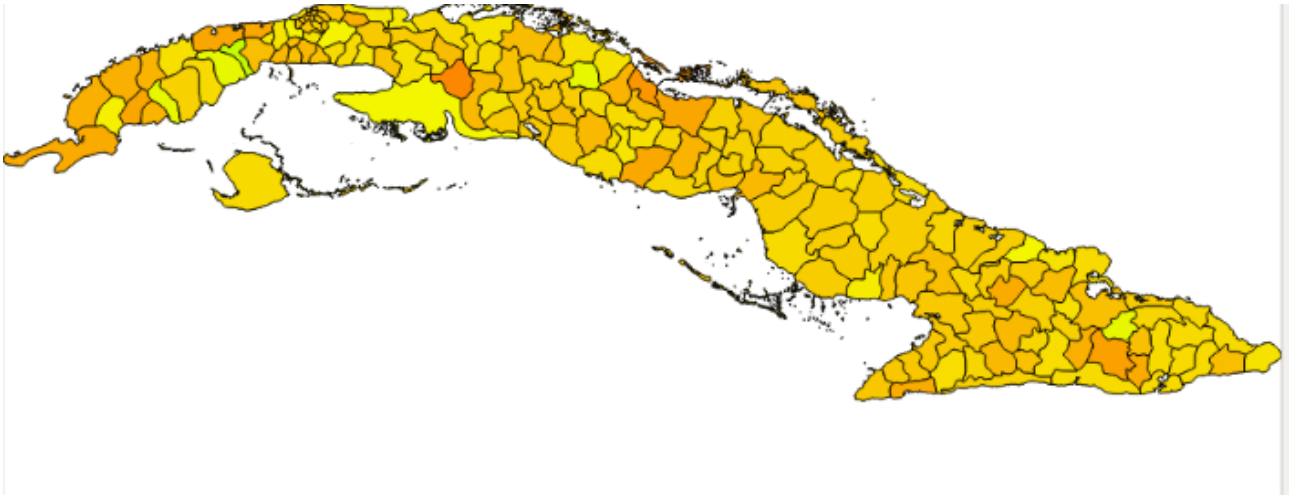

Figura 8. Mapa temático de la estratificación realizada a datos sintéticos. *Fuente: (Elaboración propia)*

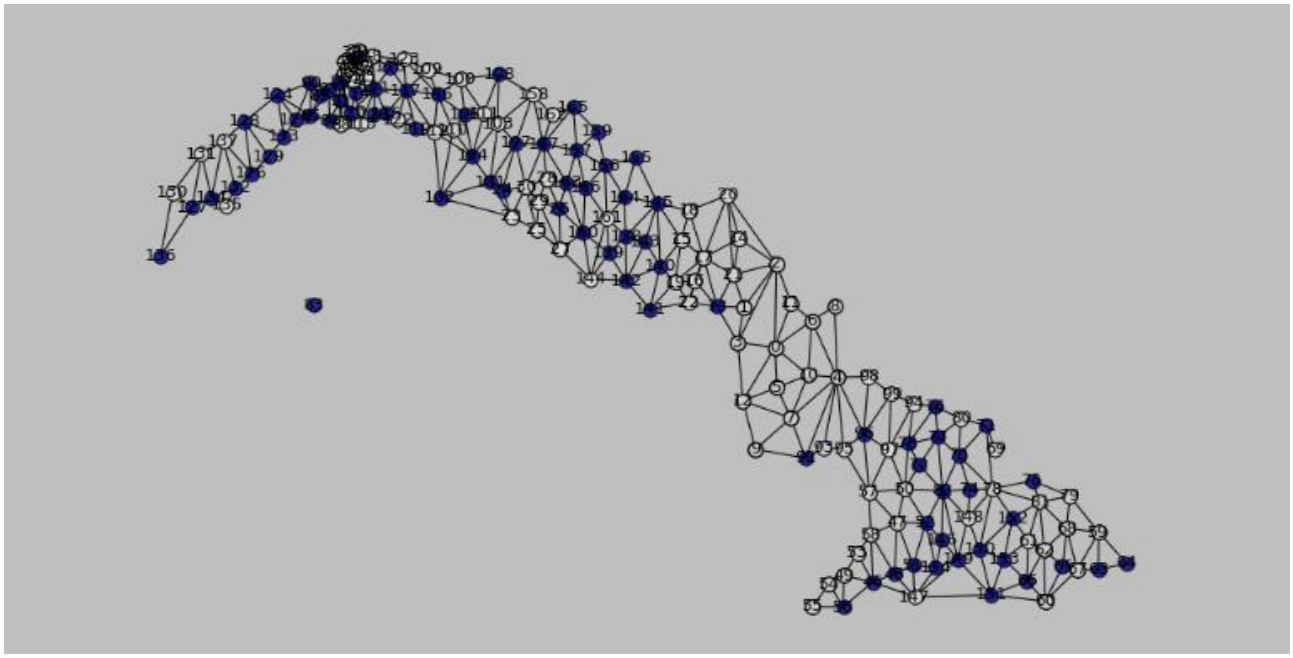

Figura 9. Grafo de similitud geométrica realizada a datos sintéticos. *Fuente: (Elaboración propia)*

3.3 Aplicación a un caso de estudio

Se decide aplicar la solución a un caso de estudio, en correspondencia con el trabajo realizado por (Pérez Betancourt, González Polanco y Febles Rodríguez 2017), donde se realiza un proceso de estratificación a las 15 provincias de Cuba, más el municipio especial Isla de la Juventud según la división política administrativa vigente. Se obtuvo una capa vectorial desde la Infraestructura de datos espaciales de Cuba con los polígonos que representan a cada territorio escogido para el análisis. El algoritmo se ejecuta en un tiempo de ejecución de 0.0010759830 segundos y en una estación de trabajo de 4,0 GB de Random Access Memory (RAM, por sus siglas en inglés), con capacidad de espacio en disco de 500 GB y un microprocesador Intel Pentium @ 1.60 GHz.

A partir de los trabajos reportados en la literatura sobre la relación de las enfermedades con el espacio, se decidió escoger como variables las 10 principales causas de muerte de Cuba en el año 2016. Los indicadores de estas variables por territorios se obtuvieron del Anuario estadístico de salud (MINSAP, 2016). Se seleccionaron los indicadores siguientes:

- Tumores malignos.
- Enfermedades del corazón.
- Enfermedades cerebrovasculares.
- Influenza y neumonía.

- Accidentes.
- Enfermedades crónicas de las vías respiratorias inferiores.
- Enfermedades de las arterias, arteriolas y vasos capilares.
- Diabetes mellitus.
- Lesiones autoinfligidas intencionalmente.
- Cirrosis y otras enfermedades crónicas del hígado.

Aplicación al caso de estudio

Para realizar la clasificación de cada una de las provincias de Cuba se utiliza la herramienta Estratificación de Territorios Basada en Indicadores de Salud y se emplea el algoritmo de agrupamiento desarrollado, GraphCAScan. Las figuras 11 y 12 muestran parte del proceso realizado.

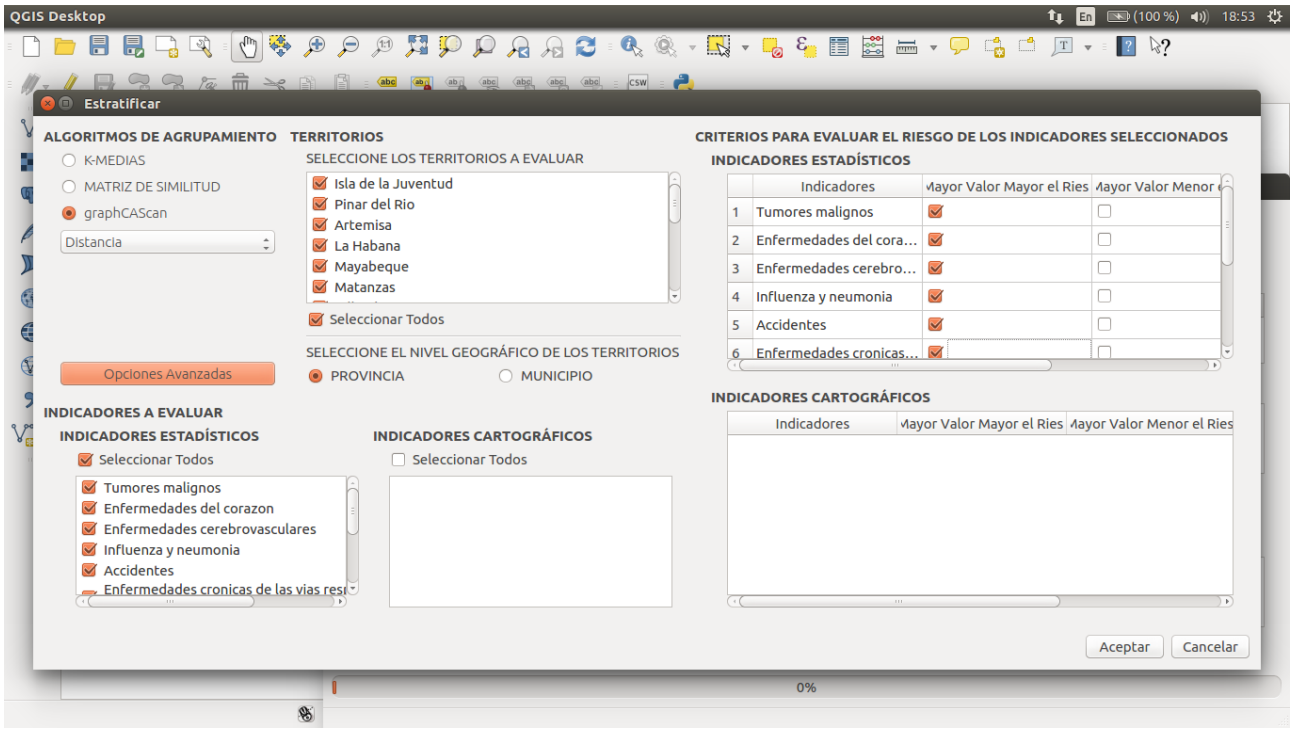

Figura 10. Interfaz de usuario Vista Estratificación. Fuente: (Elaboración propia)

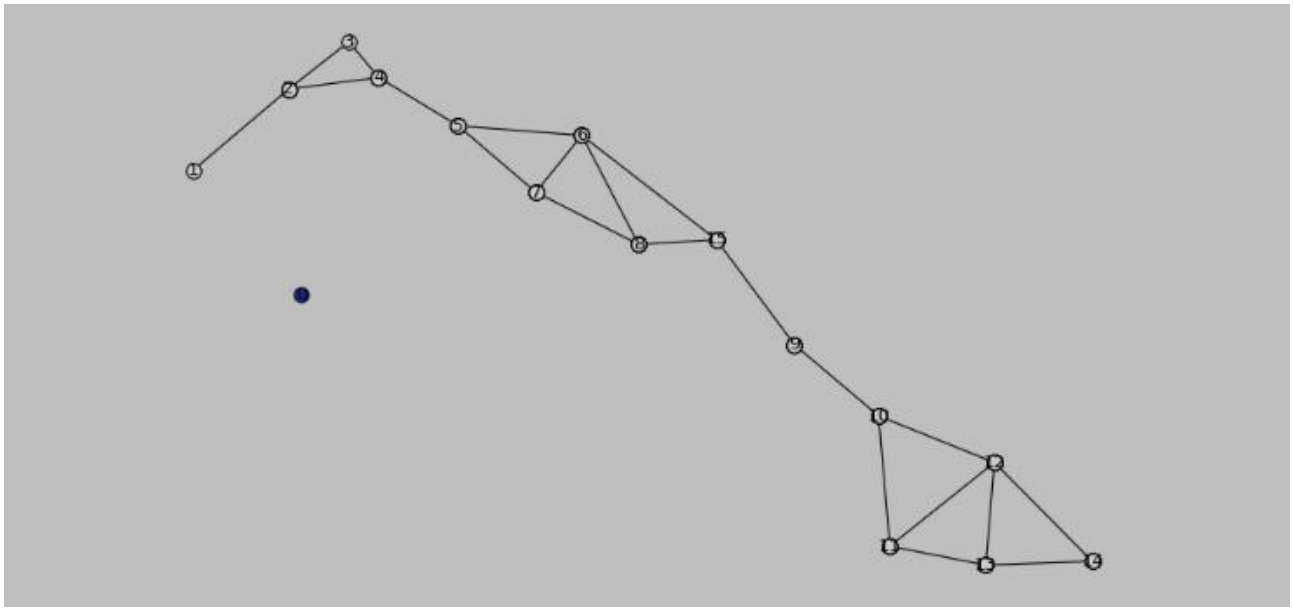

Figura 11 Grafo de similitud geométrica. Fuente: (Elaboración propia)

Resultados de la aplicación del caso de estudio

A continuación, se muestran los resultados de la estratificación de las provincias de Cuba a partir del proceso analítico-estadístico de las variables de salud escogidas. En la figura 13 se presentan en forma de mapa y en la tabla 13 de manera más detallada.

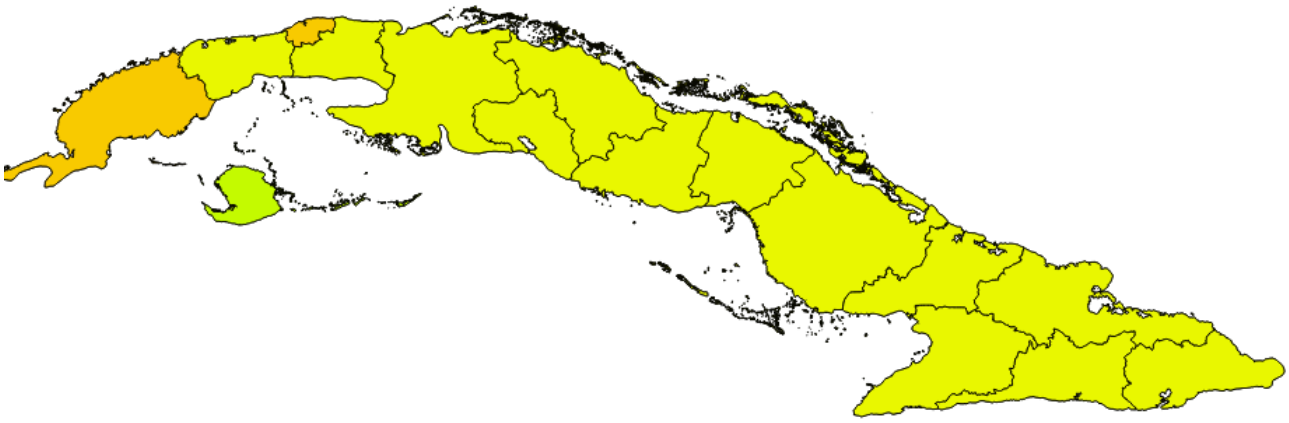

Figura 12. Mapa temático de la estratificación realizada utilizando la herramienta propuesta. Fuente: (Elaboración propia)

Tabla 13. Resultados de la estratificación realizada utilizando la herramienta propuesta. Fuente: (Elaboración propia)

Nombre del estrato	Provincias	Riesgo de salud
Estrato 1	Isla de la Juventud	0.5717

Estrato 2	Artemisa, Mayabeque, Matanzas, Villa Clara, Cienfuegos, Sancti Spíritus, Camagüey, Ciego de Ávila, Las tunas, Granma Holguín, Guantánamo, Santiago de Cuba	0.6406
Estrato 3	Pinar del Río, La Habana	0.7422

Comparación de resultados

Para determinar la similitud existente entre los resultados obtenidos por la herramienta desarrollada y los del proceso realizado en (Pérez Betancourt, González Polanco y Fables Rodríguez 2017), a partir del caso de estudio descrito anteriormente, se diseñó la medida que se describe a continuación para evaluar los estratos resultantes atendiendo a si son lo suficientemente compactos.

Suponiendo que el número final de estratos es k, la medida de exactitud c está dada por la ecuación siguiente.

$$c = \frac{\sum_{i=0}^{k} ti}{n}$$

donde n es el número total de territorios de la estratificación y ti es el número de territorios que aparecen clasificados correctamente en el estrato i. Un territorio estará clasificado correctamente en un estrato si pertenece a la misma componente conexa de los territorios agrupados en ese estrato.

Siempre toma valores entre 0 y 1, correspondiente este último a que los grupos son compactos.

A continuación, se presenta una tabla comparativa entre los resultados del proceso de estratificación realizado utilizando la herramienta desarrollada y el efectuado en (Pérez Betancourt, González Polanco y Fables Rodríguez 2017).

Tabla 14. Tabla comparativa de los resultados de los procesos de estratificación realizados (por estratos). Fuente: (Elaboración propia)

Resultados obtenidos utilizando la herramienta desarrollada	**Resultados del proceso realizado en (Pérez Betancourt, González Polanco y Febles Rodríguez 2017)**
Isla de la Juventud	Artemisa, La Habana, Mayabeque, Villa Clara, Santi Spíritus, Ciego de Ávila.
Artemisa, Mayabeque, Matanzas, Villa Clara, Cienfuegos, Sancti Spíritus, Camagüey, Ciego de Ávila, Las tunas, Granma, Holguín, Guantánamo, Santiago de Cuba	Pinar del Río, Guantánamo
Pinar del Río, La Habana	Isla de la Juventud, Matanzas, Cienfuegos, Camagüey, Las Tunas, Granma, Holguín, Santiago de Cuba

A partir de los resultados obtenidos en ambos procesos de estratificación, se aplica la medida de exactitud descrita inicialmente, obteniéndose una c de 0.87 para GraphCAScan y un c de 0.5 para k-means. Teniendo en cuenta lo anterior se evidencia que el valor de c para el algoritmo GraphCAScan se acerca más a 1. Esto permite verificar la veracidad de los estratos generados por la herramienta desarrollada a partir del caso de estudio propuesto y evidencia que GraphCAScan crea grupos más compactos que los desarrollados por el algoritmo k-means.

3.4 Resultados experimentales

Con el objetivo de validar el algoritmo se decide comparar los resultados del mismo con los resultados del algoritmo k-means ya implementado en el complemento Estratificación de territorios basada en indicadores de salud. Para realizar dicha comparación se realiza el proceso de estratificación a las 15 provincias de Cuba, más el municipio especial Isla de la Juventud utilizando la similitud temática de los datos y la medida de similitud geométrica conectividad. Se comparan los estratos y el riesgo promedio de salud de los mismos. En las figuras 14 y 16 se muestran los resultados en mapas temáticos de GraphCAScan y K-means respectivamente, en las figuras 15 en forma de grafo para GraphCAScan y en tablas 15 y 16 más detalladamente.

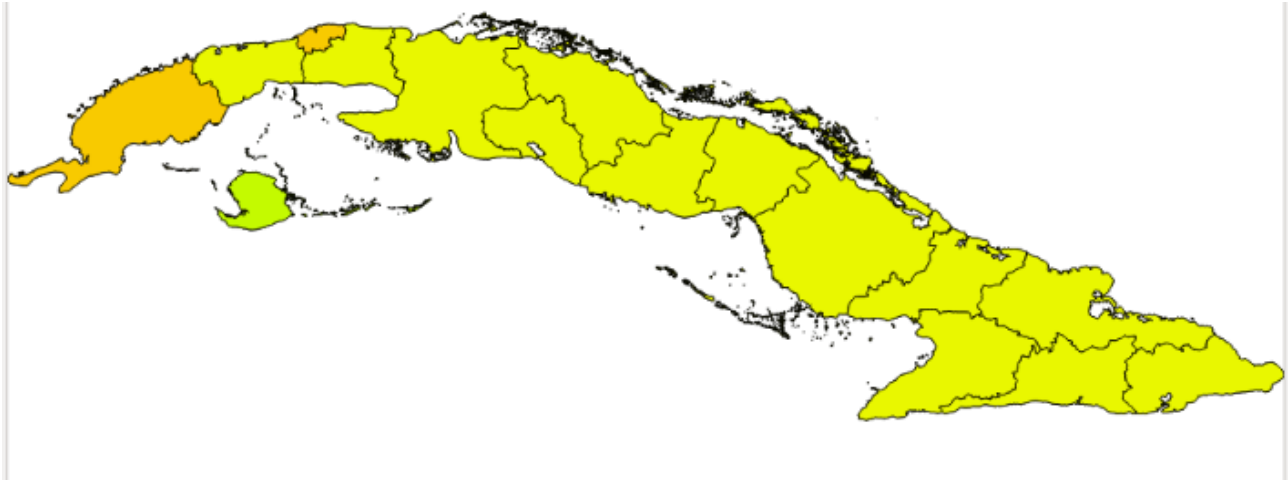

Figura 13. Mapa temático de la estratificación realizada utilizando GraphCAScan. Fuente: (Elaboración propia)

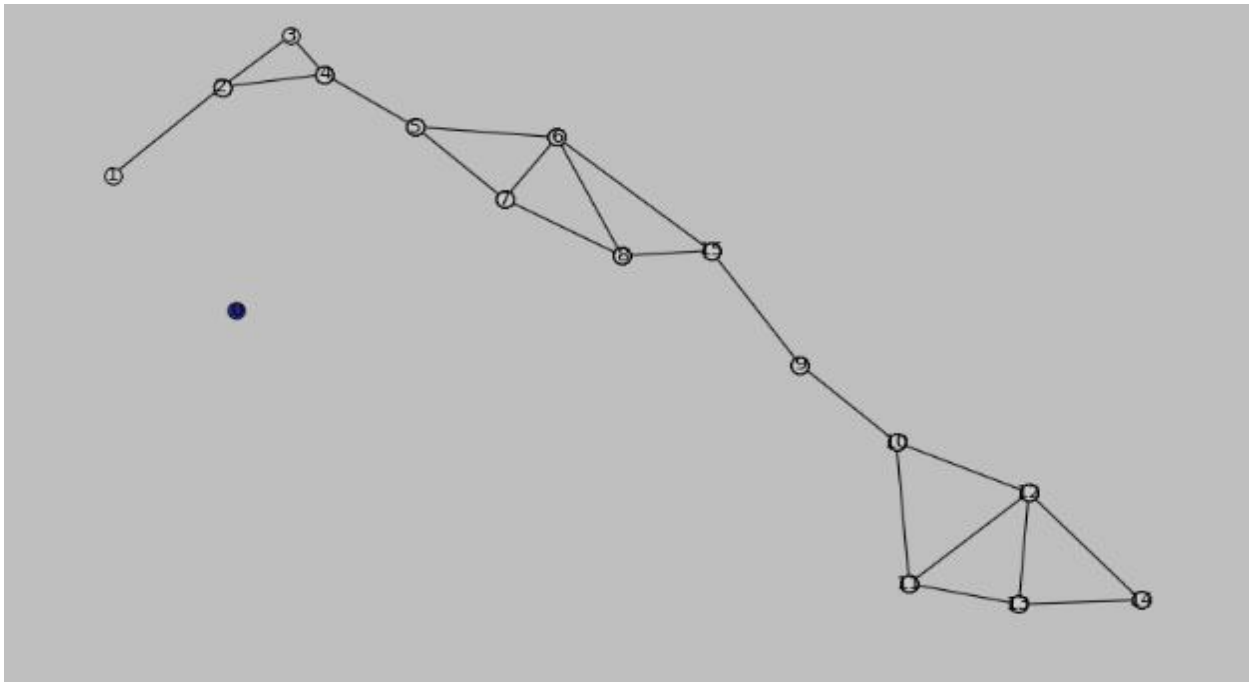

Figura 14. Grafo resultante de la estratificación realizada utilizando GraphCAscan. Fuente: (Elaboración propia)

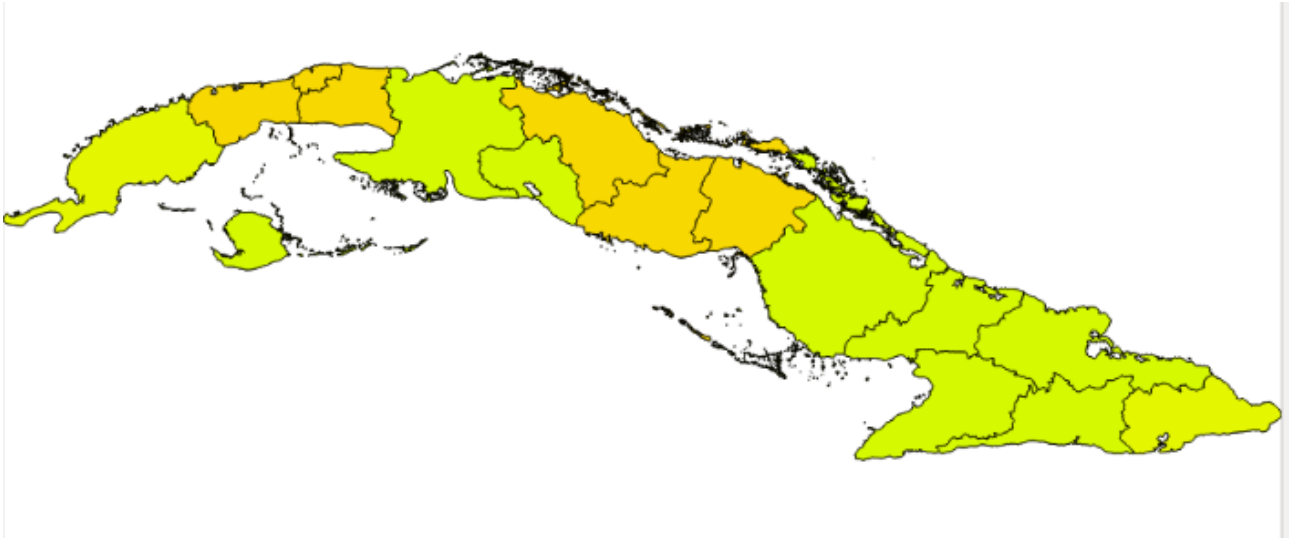

Figura 15. Mapa temático de la estratificación realizada utilizando k-means. Fuente: (Elaboración propia)

Tabla 15. Estratos y riesgo promedio de salud por estratos utilizando GraphCAScan. Fuente: (Elaboración propia)

Estrato	**Utilizando GraphCAScan (Similitud temática y conectividad)**	**Riesgo promedio de salud por estratos**
Estrato 1	Isla de la Juventud	0.5717
Estrato 2	Artemisa, Mayabeque, Matanzas, Villa Clara, Cienfuegos, Sancti Spíritus, Camagüey, Ciego de Ávila, Las tunas, Granma, Holguín, Guantánamo, Santiago de Cuba,	0.6406
Estrato 3	La Habana, Pinar del Río	0.7422

Tabla 16. Estratos y riesgo promedio de salud por estratos utilizando k-means. Fuente: (Elaboración propia)

Estrato	**Utilizando k-means (Similitud temática y conectividad)**	**Riesgo promedio de salud por estratos**
Estrato 1	Artemisa, La Habana, Mayabeque, Villa Clara, Santi Spíritus, Ciego de Ávila.	0.7178
Estrato 2	Pinar del Río, Guantánamo	0.6319
Estrato 3	Isla de la Juventud, Matanzas, Cienfuegos, Camagüey, Las Tunas, Granma, Holguín, Santiago de Cuba	0.6015

Se observa que ambos algoritmos crean tres estratos, GraphCAScan en el primer estrato ubica un solo territorio con un riesgo promedio de salud de 0. 5717, en el segundo estrato 12 territorios con un riesgo promedio de salud de 0.6406 y en el tercer estrato tres territorios con un riesgo promedio de salud de 0.7422. Mientras que k-means agrupa seis territorios en el primer estrato con un riesgo promedio de salud de 0.7178, dos en el segundo estrato con un riesgo promedio de salud de 0.6319 y ocho en el tercer estrato con un riesgo promedio de salud de 0.6015. Con estos resultados se llega a la conclusión de que el algoritmo GraphCAScan crea grupos más compactos en el espacio. También se observa que el algoritmo GraphCAScan presenta menor riesgos de salud por estratos en el primer estrato y mayor riesgo de salud por estratos en el tercer estrato. El algoritmo K-means presenta menor riesgos de salud por estratos en el tercer estrato y mayor riesgo de salud por estratos en el primer estrato.

Métricas

Se realizó una evaluación del comportamiento del agrupamiento según los dos algoritmos utilizados. Los resultados de los grupos construidos por cada algoritmo fueron evaluados a partir de diferentes índices de validación de clúster. En la tabla 17 se muestra el resultado para cada índice, en todos los casos el algoritmo k-means obtiene el mejor desempeño.

Tabla 17. Resultados de la evaluación de índices de validación. Fuente: (Elaboración propia)

Métrica	K-means Temática	GraphCAScan Conectividad
Precision	**0,67**	**0,42**
Recall	**0,63**	**0,12**
F-measure	**0,62**	**0,14**
Jaccard	**0,63**	**0,13**
Rand_score	**0,68**	**0,50**

3.5 Conclusiones parciales

En el presente capítulo se realizaron pruebas de aceptación y de caja blanca, se experimentó sobre un conjunto de datos sintéticos, se aplicó la solución propuesta a un caso de prueba y se analizaron los resultados experimentales para validar y verificar la solución propuesta. A partir de la realización de estas pruebas se llega a las siguientes conclusiones:

- Las pruebas de software realizadas permitieron detectar y corregir las no conformidades de la propuesta de solución.

- La aplicación del algoritmo sobre un conjunto de datos sintéticos demostró que funciona sobre un conjunto de datos grande evidenciando la efectividad de la solución presentada.

- La aplicación del caso de estudio permitió verificar la veracidad de los estratos generados por la herramienta desarrollada y evidenció que GraphCAScan crea grupos más compactos que los desarrollados por el algoritmo k-means.

- Al analizar los resultados experimentales se llegó a la conclusión de que, aunque el algoritmo propuesto crea grupos más compactos en el espacio que el algoritmo K-means, al evaluar los resultados de cada algoritmo con diferentes índices de validación de clúster K-means tiene un mejor desempeño.

Conclusiones generales

Como resultados de la presente investigación se obtuvo un algoritmo de agrupamiento basado en autómatas celulares como propuesta de solución para integrar a la estratificación de territorios utilizando SIG que contribuyen al mejoramiento de la capacidad de gestión de las entidades de salud. En función de los resultados obtenidos se arribó a las siguientes conclusiones:

- La definición del marco teórico referencial de la investigación relacionado con la minería de datos y el proceso de estratificación de territorios basado en indicadores de salud, evidenciaron que las medidas de similitud empleadas en los algoritmos existentes están enfocadas a los datos temáticos, sin tener en cuenta la componente espacial y que la autocorrelación suele ser ignorada por lo que fundamenta la necesidad de desarrollar un algoritmo que tenga esto en cuenta.

- La integración de la solución propuesta al complemento de estratificación basado en indicadores de salud facilitó la realización del proceso de estratificación de territorios utilizando indicadores de naturaleza espacial y temática.

- Con la creación del algoritmo se incorporó el tratamiento de la componente espacial de los datos al complemento de estratificación de territorios lo que permite la obtención de información precisa para la toma de decisiones.

- Las pruebas aplicadas para la verificación de la solución informática y la valoración de los resultados a través de un caso de estudio demostraron que el sistema cumple con los requisitos definidos, garantizando su correcto funcionamiento.

RECOMENDACIONES

- Incorporar otra regla de partición a la solución propuesta que mejore la calidad de los grupos creados y el desempeño del algoritmo en general.

REFERENCIAS BIBLIOGRÁFICAS

12. Grid-Based Clustering Algorithms. *Data Clustering: Theory, Algorithms, and Applications* [en línea], 2007. S.l.: Society for Industrial and Applied Mathematics, ASA-SIAM Series on Statistics and Applied Mathematics, pp. 209-217. [Consulta: 1 abril 2018]. ISBN 978-0-89871-623-8. Disponible en: https://epubs.siam.org/doi/abs/10.1137/1.9780898718348.ch12.

ADWAN, O., HUNEITI, A., AYYAL AWWAD, A., AL DAMARI, I., ORTEGA, A., ABU DALHOUM, A.L. y ALFONSECA, M., 2013. Utilizing an enhanced cellular automata model for data mining. *International Review on Computers and Software* [en línea], [Consulta: 13 mayo 2016]. Disponible en: https://repositorio.uam.es/handle/10486/666491.

AGUAYO TELLÉZ, E. y MEDELLÍN MENDOZA, S.E., 2014. Dependencia espacial de la delincuencia en Monterrey, México. ,

AHANGARAN, M., TAGHIZADEH, N. y BEIGY, H., 2017. Associative cellular learning automata and its applications. *Applied Soft Computing*, vol. 53, pp. 1 – 18. ISSN 1568-4946. DOI https://doi.org/10.1016/j.asoc.2016.12.006.

algoritmo basado en cuadrículas (grillas) - Schlumberger Oilfield Glossary. [en línea], 2000. [Consulta: 1 abril 2018]. Disponible en: http://www.glossary.oilfield.slb.com/es/Terms/g/gridding_algorithm.aspx.

BARCELLOS, C. y BUZAI, G.D., 2006. La dimensión espacial de las desigualdades sociales en salud: aspectos de su evolución conceptual y metodológica. *Departamento de Ciencias Sociales. Universidad Nacional de Luján: Anuario de la División Geografía*, pp. 275–92.

BARRERA LEÓN, L.F., 2011. *Diseñode arquitectura del software*. 2011. S.l.: s.n.

BATISTA MOLINER, R., COUTIN MARIE, G., FEAL CAÑIZARES, P., GONZÁLEZ CRUZ, R. y RODRÍGUEZ MILORD, D., 2001a. DETERMINACIÓN DE ESTRATOS PARA PRIORIZAR INTERVENCIONES Y EVALUACIÓN EN SALUD PÚBLICA. ,

BATISTA MOLINER, R., COUTIN MARIE, G., FEAL CAÑIZARES, P., GONZÁLEZ CRUZ, R. y RODRÍGUEZ MILORD, D., 2001b. Determinación de estratos para priorizar intervenciones y evaluación en Salud Pública. *Revista Cubana de Higiene y Epidemiología*, vol. 39, no. 1, pp. 32–41. ISSN 1561-3003.

BECK, K., 2000. *Extreme programming explained: embrace change*. 2000. S.l.: s.n.

BENÍTEZ, I., 2005. *Técnicas de Agrupamiento para el Análisis de Datos Cuantitativos y Cualitativos*. 2005. S.l.: s.n.

BETANCOURT, Y.G.P., POLANCO, L.G. y RODRÍGUEZ, J.P.F., 2017. Estratificación de territorios basada en indicadores de salud y medidas de similitud geometricas. . S.l.: Edacun-Redipe, Ciencia e innovación tecnológica, ISBN 978-959-7225-27.

BROWN, T.T., WOOD, J.D. y GRIFFITH, D.A., 2017. Using spatial autocorrelation analysis to guide mixed methods survey sample design decisions. *Journal of Mixed Methods Research*, vol. 11, no. 3, pp. 394–414.

CANGREJO ALJURE, D. y AGUDELO, J.G., 2011a. Minería de datos espaciales. ,

CANGREJO ALJURE, D. y AGUDELO, J.G., 2011b. Minería de datos espaciales. ,

CASAS, S. y REINAGA, H., 2008. *Identificación y modelado de aspectos tempranos dirigido por tarjetas de responsabilidades y colaboraciones*. 2008. S.l.: s.n.

CECCHERINI-SILBERSTEIN, T. y COORNAERT, M., 2010. *Cellular Automata and Groups*. S.l.: s.n.

CELEMÍN, J.P., 2010. Autocorrelación espacial e indicadores locales de asociación espacial. Importancia, estructura y aplicación. ,

COBO, Á., 2007. Diseño y programación de bases de datos. . S.l.: s.n.,

CORTEZ, A. y PRO, L., 2011. Descubrimiento de Conocimiento Basado en Grafos. ,

Criterios de similitud. Similitud, divergencia y distancia. [en línea], 2017. [Consulta: 6 junio 2018]. Disponible en: https://www.uv.es/ceaces/multivari/cluster/criterios_de_similitud.htm.

D. PASCUAL, F. PLA y S. SÁNCHEZ, 2007. Algoritmos de agrupamiento. ,

DAY, J., CHEN, Y., ELLIS, P. y ROBERTS, M., 2017. A free, open-source tool for identifying urban agglomerations using polygon data. *Environment Systems and Decisions*, vol. 37, no. 1, pp. 68–87.

DELGADO ACOSTA, H., GONZÁLEZ MORENO, L., VALDÉS GÓMEZ, M., HERNÁNDEZ MALPICA, S., MONTENEGRO CALDERÓN, T. y RODRÍGUEZ BUERGO, D., 2015. Estratificación de riesgo de tuberculosis pulmonar en consejos populares del municipio Cienfuegos. *MediSur*, vol. 13, no. 2, pp. 275–284.

DUEÑAS-REYES, M.X., 2009. Minería de datos geoespaciales en búsqueda de la verdadera información. ,

DUQUE, R.G., 2011. *Python para todos.* 2011. S.l.: s.n.

ESTER, M., KRIEGEL, H.-P. y JÖRG, S., 2001. Algorithms and Applications for Spatial Data Mining. ,

ESTER, M., KRIEGEL, H.-P., SANDER, J. y XU, X., 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. ,

FAWCETT, T., 2008. Data mining with cellular automata. *ACM SIGKDD Explorations Newsletter*, vol. 10, no. 1, pp. 32–39.

FAYYAD, U.M., PIATETSKY SHAPIRO, G. y SMYTH, P., 1996. From data mining to knowledge discovery in databases. ,

GARCÍA PÉREZ, C. y ALFONSO AGUILAR, P., 2002. Estratificación epidemiológica de riesgo. ,

GATRELL, A.C., 1983. Distance and space: a geographical perspective. ,

GEWALI, L.P. y MANANDHAR, S., 2018. Approaches for Clustering Polygonal Obstacles. En: A. in I. SYSTEMS y COMPUTING (eds.), *Information Technology-New Generations*. S.l.: Springer, 558, pp. 887-892. ISBN 978-3-319-54978-1.

GONZÁLEZ, M.A., 2010. Herramientas CASE. ,

GOODCHILD, M., 1987. A spatial analytical perspective on geographical information systems. . S.l.: s.n.,

GRABUSTS, P. y BORISOV, A., 2002. Using grid-clustering methods in data classification. ,

HAWICK, K.A., 2013. Neighbourhood and number of states dependence of the transient period and cluster patterns in cyclic cellular automata. *Proc. 10th Int. Conf. on Scientific Computing (CSC'13). p. CSC7339. No. CSTN-207, WorldComp, Las Vegas, USA (22-25 July 2013), http://www. massey. ac. nz/kahawick/cstn/207/cstn-207. html* [en línea]. S.l.: s.n., [Consulta: 3 diciembre 2015]. Disponible en: http://complexity.massey.ac.nz/cstn/207/cstn-207.pdf.

JAIN, A.K., MURTY, M.N. y FLYNN, P.J., 1999. Data clustering: a review. *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323.

JANSSENS, D. y ROZENBERG, G., 1982. Graph grammars with neighbourhood-controlled embedding. *Theoretical Computer Science*, vol. 21, no. 1, pp. 55-74. ISSN 0304-3975. DOI 10.1016/0304-3975(82)90088-3.

JETBRAINS, I., 2014. *Python IDE & Django IDE for Web developers : JetBrains PyCharm*. 2014. S.l.: s.n.

JOSHI, D., SAMAL, A. y SOH, L. n-Ki at, 2009. Density-Based Clustering of Polygons. ,

JOSKOWICZ, J., 2008. *Reglas y prácticas en eXtreme Programming*. S.l.: s.n.

KHOMAMI, M.M.D., REZVANIAN, A. y MEYBODI, M.R., 2018. A new cellular learning automata-based algorithm for community detection in complex social networks. *Journal of Computational Science*, vol. 24, pp. 413–426. ISSN 1877-7503. DOI 10.1016/j.jocs.2017.10.009.

KORTE, G., 2001. *The Gis Book* [en línea]. fifth. NY, USA: OnWord Press. ISBN 978-0-7668-2820-9. Disponible en: http://books.google.com.cu/books?id=_C6oPvJ5S_EC.

LETELIER, P., 2006. *Métodologías ágiles para el desarrollo de software*. 2006. S.l.: s.n.

LÓPEZ SALINAS, A.M., 2011. *INTRODUCCIÓN A LA VIDA ARTIFICIAL Y AUTÓMATAS CELULARES*. 2011. S.l.: s.n.

LOUDEN, K.C., 2004. Lenguajes de programación: Principios y práctica. Cengage Learning Latin America. ,

LV, Y., MA, T., TANG, M., CAO, J., TIAN, Y., AL-DHELAAN, A. y AL-RODHAAN, M., 2016. An efficient and scalable density-based clustering algorithm for datasets with complex structures. *Neurocomputing*, vol. 171, pp. 9–22.

MANANDHAR, S.K., 2016. Efficient Algorithms for Clustering Polygonal Obstacles. ,

MARTÍNEZ, J.M., 2004. *Convergencia y divergencia regional en México*. S.l.: s.n.

MARTÍNEZ-GONZÁLEZ, B., PARDO, J.M., ECHEVERRY-CORREA, J.D. y SAN-SEGUNDO, R., 2017. Spatial features selection for unsupervised speaker segmentation and clustering. *Expert Systems with Applications*, vol. 73, pp. 27–42.

MILLER, H.J. y E. WENTZ, 2003. Geographic representation and spatial analysis in geographic information systems. *Annals of the Association of American Geographers*, vol. 93.

MOERE, A.V., CLAYDEN, J.J. y DONG, A., 2006. Data clustering and visualization using cellular automata ants. *AI 2006: Advances in Artificial Intelligence* [en línea]. S.l.: Springer, pp. 826–836. [Consulta: 3 diciembre 2015]. Disponible en: http://link.springer.com/chapter/10.1007/11941439_87.

MORALES MANILLA, J.L., 2007. The definition of a minimum set of spatial relationships. ,

MORALES, R. y TORRES, Y., 2015. *Propuesta para la estratificación de territorios basada en indicadores de salud*. S.l.: s.n.

MORGADO GARCÍA, T., PONCE DE LEÓN LIMA, D.A. y ROSETE SUÁREZ, A., 2017. Descubrimiento de conocimiento en bases de datos históricas de una empresa comercializadora. ,

ODED, M. y LIOR, R., 2005. *Data Mining and Knowledge Discovery Handbook*. S.l.: s.n.

PELÁEZ, J., 2009. *Arquitectura basada en capas*. 2009. S.l.: s.n.

PÉREZ BENTANCOURT, Y.G., GONZÁLEZ POLANCO, L., FEBLES RODRÍGUEZ, J.P. y CABRERA CAMPOS, A., 2018. Propuestas para el análisis geoespacial en estudios salubristas. *Revista Cubana de Ciencias Informáticas*, vol. 12, no. 2, pp. 44-57. ISSN 2227-1899.

PÉREZ BETANCOURT, Y.G. y GONZÁLEZ POLANCO, L., 2013. La minería de datos espaciales y su aplicación en los estudios de salud y epidemiología. ,

PÉREZ BETANCOURT, Y.G., GONZALEZ POLANCO, L. y FABLES RODRÍGUEZ, J.P., 2017. Estratificación de territorios basada en indicadores de salud y medidas de similitud geométricas. ,

PÉREZ BETANCOURT, Y.G., GONZÁLEZ POLANCO, L. y FEBLES RODRÍGUEZ, J.P., 2017. Estratificación de territorios basada en indicadores de salud y medidas de similitud geométricas. ,

PÉREZ BETANCOURT, Y.G., RODRÍGUEZ PUENTE, R. y KAUNAPAWA MUFETI, T., 2015. Cellular Automata and its applications in modeling and simulating the evolution of diseases. ,

POSTGIS, 2014. *POSTGIS DEVELOPMENT TEAM*. 2014. S.l.: s.n.

POSTGRESQL-3 GLOBAL DEVELOPMENT GROUP, 2014. *PostgreSQL*. 2014. S.l.: s.n.

PRESSMAN, 2005. *Ingeniería de Software un enfoque práctico*. S.l.: s.n.

RAYMOND T. NG y JIAWEI HAN, 2002. CLARANS: A Method for Clustering Objects for Spatial Data Mining. , vol. 14.

ROBINSON, C., 2011. *Basic introduction into pgAdmin*. 2011. S.l.: s.n.

ROSALES TAPIA, A.R. y QUINTERO PÉREZ, J.A., 2012. Modelo de dependencia espacial aplicado al análisis de la distribución del consumo de alcohol en el campus CU , UNAM. ,

SCHEFER-WENZL, S. y STREMBECK, M., 2013. Modeling context-aware RBAC models for business processes in ubiquitous computing environments. ,

SOMMERVILLE, I., 2005. *Ingenieria del software*. S.l.: s.n.

STARTED, G., 2010. *Software Design Tools for Agile Teams, with UML, BPMN and More*. 2010. S.l.: s.n.

TAHA, A., 2016. Knowledge Discovery In GIS Data. *arXiv preprint arXiv:1601.07241* [en línea], [Consulta: 8 noviembre 2016]. Disponible en: http://arxiv.org/abs/1601.07241.

VASWANI, K. y KARANDIKAR, A.M., 2017. An Algorithm for Spatial Data Mining using Clustering. *International Journal of Computer & Mathematical Sciences*, vol. 6, no. 8, pp. 6. ISSN 2347 – 8527.

VILALTA, C.J., 2003. Una aplicación del análisis espacial al estudio de las diferencias regionales del ingreso en México. ,

VILALTA, C.J., 2004. Sobre la espacialidad de los procesos electorales y una comparación entre las técnicas de regresión OLS y SAM. . 2004.

VILALTA, C.J., 2005. Cómo enseñar autocorrelación espacial. *Economía, Sociedad y Territorio* [en línea], vol. 5. Disponible en: http://www.redalyc.org/articulo.oa?id=11101804.

WANG, J., DENG, Z., CHOI, K.-S., JIANG, Y., LUO, X., CHUNG, F.-L. y WANG, S., 2016. Distance metric learning for soft subspace clustering in composite kernel space. *Pattern Recognition*, vol. 52, pp. 113–134.

WANG, J.-F., ZHANG, T.-L. y FU, B.-J., 2016. A measure of spatial stratified heterogeneity. *Ecological Indicators*, vol. 67, pp. 250–256.

WANG, W., DU, S., GUO, Z. y LUO, L., 2015. Polygonal Clustering Analysis Using Multilevel Graph-Partition: Polygonal Clustering Analysis Using Multilevel Graph-Partition. *Transactions in GIS*, vol. 19, no. 5, pp. 716–736. ISSN 13611682. DOI 10.1111/tgis.12124.

WILSON, J.P., 2015. GIScience Research at the 2015 Esri International User Conference. *Transactions in GIS*, vol. 19, no. 3, pp. 339–341.