RCCI Vol. 4, No. 1-2 ENERO- JUNIO, 2010 p. 59-64

Recibido: *Aceptado:*

Aplicación de las gramáticas de grafo en Sistemas de Información Geográfica

Application of graph grammar in Geographic Information Systems

Rafael Rodríguez Puente

Facultad 4, Universidad de las Ciencias Informáticas, Carretera a San Antonio de los Baños km 2 ½, Torrens, Boyeros, Ciudad de La Habana, Cuba. CP 19370
rafaelrp@uci.cu

Resumen

En este artículo se describe un mecanismo para hacer búsqueda de caminos mínimos en sistemas de información geográfica que manipulen mapas con una gran cantidad de objetos geográficos.

Palabras clave: gramáticas de grafo, sistemas de información geográfica, reducción de grafos.

Abstract

This article describes a mechanism to find shortest path in Geographic Information Systems with a large amount of geographical objects.

Keywords: graph grammar, geographic information system, graph reduction.

Introducción

La mayor utilidad de un Sistema de Información Geográfica (SIG) esta estrechamente relacionada con la capacidad que posee éste de construir modelos o representaciones del mundo real a partir de las bases de datos digitales (FERNÁNDEZ 2003), esto se logra aplicando una serie de procedimientos específicos que generan más información para el análisis.

Una de las aplicaciones de un SIG, es la búsqueda de rutas en mapas, o sea, un SIG puede responder a la pregunta: ¿Cómo llego desde el lugar X al lugar Y, de la forma más rápida posible?, esta pregunta es respondida en forma de imagen, o sea, se muestra la ruta para llegar desde el lugar X al Y en un mapa.

En muchas empresas se necesitan los SIG para planificar el transporte de sus productos, por ejemplo, el Instituto Postal Telegráfico de Venezuela (IPOSTEL), necesita un SIG que calcule y le muestre como debe distribuir la flota de vehículos para el transporte de bultos postales, lo cual va a disminuir el tiempo que un paquete demora en llegar al destinatario.

Varios de los sistemas desarrollados, en el caso de la búsqueda de rutas, sólo tienen en cuenta los puntos principales de una ciudad, o las ciudades principales de un país, y por tanto, se obvia la mayoría de los objetos (que pueden ser casas, calles, hospitales, escuelas) del mapa, lo cual se traduce en restricciones sobre los usuarios del Sistema de Información Geográfica.

Estas restricciones están dadas en gran medida, debido a que para resolver estos problemas, debemos transformar el mapa en un modelo matemático, en este caso en un grafo, que nos permita hacer los análisis necesarios: rutas mínimas, problemas de transporte, entre otros. El problema radica en la dimensión que alcanzaría este grafo, ya que en un mapa nos podemos encontrar con varios millones de objetos, por ejemplo, un mapa de Cuba, que es un país pequeño, en el cual haya un objeto geográfico por cada una de las construcciones que hay en el país, cuando se obtenga el grafo que representa este mapa, nos sería prácticamente imposible cargarlo completo en memoria.

El problema que se trata de solucionar, es el de representar un mapa (todos los objetos) a través de un grafo para posteriormente hacer

análisis de redes, particularmente, la búsqueda de rutas mínimas. Por lo cual se presenta una propuesta de algoritmo para reducir un grafo, y a su vez, un algoritmo para hacer búsquedas de caminos mínimos en el grafo reducido con el algoritmo propuesto; lo cual puede ser aplicado posteriormente a los Sistemas de Información Geográfica, ya que para analizar y manipular los datos de un mapa, es conveniente hacerlo a través de un grafo, en el cual, los vértices representan los objetos del mapa, y las aristas las relaciones entre estos.

Con esto se logrará una mayor calidad en los servicios que brinda un SIG, ya que no sólo se utilizaría en la búsqueda de caminos mínimos, sino en cualquier operación que traiga consigo un procesamiento del mapa a nivel de todos los objetos que lo componen.

Es necesario mencionar que se llevó a cabo esta tarea, empleando conceptos de la teoría de grafos, teoría de redes, gramáticas de grafo y la reescritura de grafos, para finalmente obtener dos algoritmos que nos permiten reducir sustancialmente un grafo y además hacer búsquedas de caminos mínimos.

Conceptos

En esta sección examinaremos algunos conceptos referidos a conjuntos de vértices de un grafo que poseen determinadas propiedades. Mediante la búsqueda de dichos conjuntos pueden ser resueltos ciertos tipos de problemas de la vida práctica.

Reducción de un grafo

Un método utilizado con el fin de estudiar de una forma más sencilla las propiedades de un grafo G, consiste en construir, a partir de éste, un grafo más simple denotado Gr, descomponiendo G en subgrafos y considerando éstos como vértices de Gr. Estos vértices se unirán por arcos deducidos de G de acuerdo con ciertas reglas(González, 1975).

Consideremos el grafo G = (V, A), y sea p una propiedad mediante la cual se define una partición P en V, lo cual permite descomponer G en subgrafos de acuerdo con dicha partición. Sean $A_1,..., A_s$, las clases de la partición P.
El grafo reducido $G_r=(C, R)$ según la propiedad p se construye de la manera siguiente:

- Asignamos a cada conjunto A_i, el vértice Ci ϵ C. De esta forma G_r tendrá S vértices.
- $C_i, C_j \epsilon$ R si y solamente si (x, y) A con x ϵ A_i e y ϵ A_j.

Ejemplo, en el grafo de la ilustración 1, si consideramos la propiedad p_1, la cual determina la partición P en la que:
A_1 = {x1, x2}
A_2 = {x5}
A_3 = {x3, x4, x6}
Obtendremos el grafo reducido que se muestra en la ilustración 2 (González, 1975).

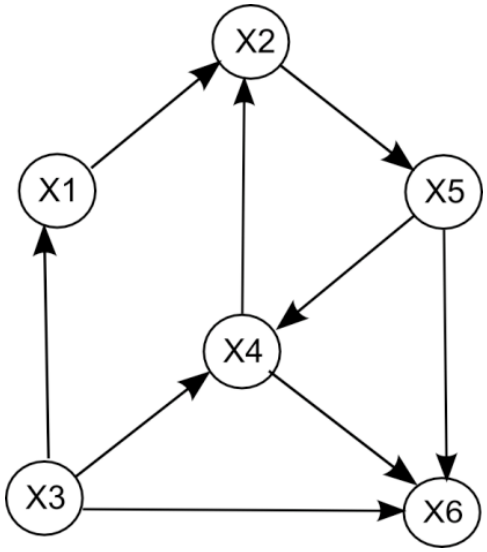

Ilustración 1. Ejemplo de grafo

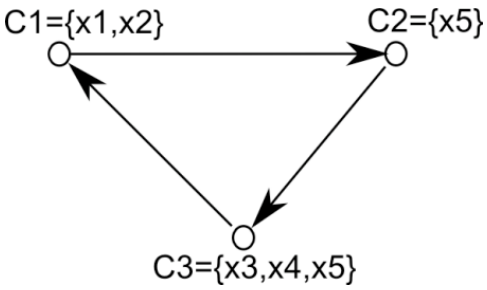

Ilustración 2. Grafo reducido según la propiedad p_1

Resulta evidente que considerando diferentes propiedades para definir la partición, obtendremos diferentes grafos reducidos, por tanto, es importante especificar la propiedad basada en la cual se va a realizar la reducción.

Reescritura de grafos

Una regla de reescritura de grafo es aplicada a un grafo para sustituir un subgrafo por otro (Rozenberg and Salomaa, 1997; Janssens and Rozenberg, 1982; Kreowski, 1990).
La terminología para la reescritura de grafos no está estandarizada, aquí utilizaremos los

- Regla de reescritura:
  - o $g_l ::= g_r$, un subgrafo isomorfo a g_l es sustituido por otro isomorfo a g_r
  - o Información de empotrado, puede ser textual o gráfica, los modelos de engomado (BLOSTEIN D. 1994) la especifican con un isomorfismo de engomado.
  - o Condición de aplicación, son restricciones sobre la aplicación de la regla, es opcional.
  - o Función de transferencia de atributos, es una función que asigna atributos, es opcional.
- G: Grafo al cual le vamos a aplicar la regla.
- gl: subgrafo a ser reemplazado.
- gr: subgrafo utilizado para reemplazar a g_l.
- Resto del grafo: G-g_l.

Una regla de reescritura, provee información de empotrado, la cual es usada para conocer la forma de conectar el grafo gr con el resto del grafo. La principal problemática es convertir las aristas pre-empotradas en aristas post- empotradas (Rozenberg and Salomaa, 1997).

Algunos mecanismos de empotrado, permiten una especificación no restringida de las aristas post-empotradas, aunque también existen mecanismos que si establecen una restricción (Rozenberg and Welzl, 1986).

La selección de un mecanismo nos pone en la disyuntiva siguiente: usamos un mecanismo complejo pero con pocas reglas de escritura, o uno de muchas reglas de escritura que sean simples (Blostein, 1994).

Después de una revisión de los mecanismos de empotrado existentes, aquí aplicaremos una variante del mecanismo de empotrado profundidad 1, utilizando el modelo NCE. La variación que se propone para poder dar solución a la problemática planteada es la de especificar en la información de empotrado el peso de la arista.

Algoritmos propuestos

A continuación se describe un algoritmo para reducir un grafo a través de una propiedad dada, el mismo dará como salida un nuevo grafo, con una reducción significativa en la cantidad de nodos, y con una serie de especificaciones que permitirán, en cualquier momento, obtener el grafo original si es necesario. Posteriormente, se enuncia un algoritmo que permite hacer búsquedas de camino mínimo en grafos reducidos con el algoritmo de reducción enunciado en este trabajo.

Finalmente, se dan dos variantes que permiten reducir la complejidad y el tiempo de respuesta del algoritmo de búsqueda de camino mínimo, la selección de una u otra, está en dependencia de las necesidades específicas y del hardware del que se disponga para ejecutar los algoritmos.

Reducir el grafo

Entrada del algoritmo:

- Un grafo que representa un mapa. G=(V,E)
- Una propiedad p, por la cual se agruparán los vértices del grafo

Salida:

- Un grafo que representa al mapa
- Un conjunto de reglas de reescritura de grafos

El algoritmo está compuesto por los siguientes pasos:

1. Particionamos el conjunto de vértices inicial según una propiedad, o sea, agrupamos los vértices que cumplan la propiedad escogida, los conjuntos de vértices obtenidos los denotaremos por A_i, por ejemplo, en A_1 están todos los vértices que representan objetos que están en el municipio Palma Soriano.
2. Construimos el grafo reducido G_r, donde habrá un nodo por cada conjunto A_i obtenidos en el paso anterior.
3. Por cada A_i, escribimos una regla de reescritura, de la siguiente forma:
   3.1. g_l es un nodo (lo llamaremos nodo no terminal), con una etiqueta que identifique la propiedad seleccionada para particionar el conjunto de vértices, es decir, si la propiedad escogida fuera código postal, un nodo gl pudiera tener la etiqueta 10400. Este nodo agrupará todos los vértices que pertenezcan a la partición A_i.
   3.2. g_r sería el subgrafo dado por $g_r=(A_i, E_i)$, donde $(u, v) \in E_i$ si y sólo si $\exists (u, v) \in E$ y $u, v \in Ai$
   3.3. Como especificación de empotrado, utilizaremos tuplas con la forma (vp, v, c, u), si y sólo si $\exists (u, v) \in E$ de costo c, donde $u \in$ Resto del grafo (G-g_l) y v es un vértice del grafo gr, o sea, $v \in A_i$; vp es el valor de la propiedad que identifica a Ai. Se crearán dos conjuntos formados por tuplas con la forma anterior, uno para las aristas que entran en el nodo gl con etiqueta vp, y otras para las que salen del mismo.
   3.4. Si queremos seguir compactando el grafo, escogemos otra propiedad y volvemos al paso 1.

Búsqueda de rutas mínimas

El primer paso para la búsqueda es definir una función de distancia fd(V1, V2, V), que calcule la distancia mínima desde el vértice V1 hasta el V2 pasando por V, para lo que se deben cumplir las condiciones:

1. V es un nodo no terminal
2. $\exists$ (V1,V1'), (V1',V2)|V1' ϵ V

Para buscar una ruta mínima, podemos aplicar varios algoritmos, en este caso, aplicaremos el algoritmo Dijkstra (Gondran, 1984; Sedgewick, 1983), con una pequeña variación:

1. Sea G = (V, E) un grafo dirigido y etiquetado.
2. Sean los vértices A (origen) y Z (destino)
3. Sea N un conjunto vacío
4. Sea D un vector con tantas dimensiones como elementos tiene V, y que almacena las distancias entre a y cada uno de los vértices de V.
5. Sea P un vector con las mismas dimensiones que D, y que conserva la información sobre qué vértice precede a cada uno de los vértices en el camino.

El algoritmo para determinar el camino de longitud mínima entre los vértices a y z es

1. N={A}
2. Para todos los nodos v:
Si v es adyacente a A:
D[v]=c[A, v]
P[v]=A
En caso contrario:
D[v] = infinito
3. Repetir
Buscar un nodo w que no este en N tal que D[w] es mínimo
N = N U w
D[v] = min (D[v], D[w] + c[w, v])
Si D[v] > D[w] + c[w, v]
P[v]=w
4. Hasta que todos los nodos estén en N

La variación al algoritmo estaría, en que cada vez que calculamos distancia de un nodo V_x a otro nodo V_y adyacente a V_x, donde V_x no es vértice de origen (En el algoritmo de búsqueda de camino mínimo enunciado anteriormente se denotó por la letra A) y además es un nodo no-terminal, la distancia se calcula utilizando la función fd(V_1,V_2,V), definida anteriormente, donde V sería V_x, V_1 sería el vértice que precede a V y lo podemos encontrar en el conjunto P (P[V]) y V_2 sería V_y.

Al terminar este algoritmo, en D[Z] estará guardada la distancia mínima entre A y Z. Por otro lado, mediante el vector P se puede obtener el camino mínimo: en P[Z] estará y, el vértice que precede a z en el camino mínimo; en P[y] estará el que precede a y, y así sucesivamente, hasta llegar al vértice A.

Este algoritmo se puede optimizar más aún, teniendo en cuenta una de las siguientes propuestas:

5. En cada iteración de reducción del grafo, guardamos las distancias entre cada tres vértices (V_1,V_2,V) (Ver definición de la función fd), esto traerá como consecuencia una alta complejidad para encontrar cada posible trío de vértices en cada iteración, pero como la reducción del grafo se realiza una sola vez, nos beneficiaría en la búsquedas de caminos mínimos, que sí se realizan muchas veces, ya que cada vez que apliquemos el algoritmo de búsqueda de camino mínimo, y nos encontremos con un nodo no-terminal, no tendremos que calcular el camino en el subgrafo correspondiente al vértice no-terminal, porque ya lo tendríamos calculado de antemano.
6. Cada vez que se haga una búsqueda de camino mínimo almacenar, de forma persistente, los valores que va tomando la función fd, de forma tal que sólo se haga el cálculo de la función una sola vez, esta variante tiene la ventaja que no aumenta la complejidad de la reducción del grafo y a medida que se ejecuta el algoritmo de búsqueda de camino mínimo, el tiempo de respuesta disminuye.

Conclusiones

Con la implementación y posterior aplicación en un Sistema de Información Geográfica de los algoritmos presentados, se puede lograr una mayor calidad en los servicios que brinda un SIG, ya que no sólo se utilizaría en la búsqueda de caminos mínimos, sino, en cualquier operación que traiga consigo un procesamiento del mapa a nivel de todos los objetos que lo componen.

Al obtener los resultados expuestos en el presente capítulo, podemos afirmar que la teoría de grafos y de reescritura de grafos, está estrechamente vinculada con los Sistemas de Información Geográfica.

El algoritmo planteado permitirá a los Sistemas de Información Geográfica brindar una mayor cantidad de servicios y con más calidad, al poder manipular todos los objetos representados en un mapa utilizando una forma compactada o reducida.

Referencias

BLOSTEIN D., F. H., GRABAVEC A. Practical Use of Graph Rewriting, 1994.

FERNÁNDEZ, T. D. Área de Servicios de Información Geográfica Proyecto UCI-Ciudad Digital, 2003.

JANSSENS, D. and G. ROZENBERG Graph grammars with neighbourhood-controlled embedding, 1982: 55-74.

KREOWSKI H, R. G. On Structured Graph Grammars II Informations Sciences, 1990, 52: 221-246.

LAREDO GONZÁLEZ, P. F. Introducción a la teoría y aplicaciones de las redes. Editorial Pueblo y Educación, 1975. 342 p.

MICHEL GONDRAN, M. M., STEVEN VADJA. Graphs and Algorithms. Jon Wiley & Sons, Inc., 1984. 650 p. 0471-10374-8

ROZENBERG, G. and A. SALOMAA. Handbook of Formal Languages. springer, 1997. p.

ROZENBERG, G. and E. WELZL Boundary NLC graph grammars - basic definitions, normal form, and complexity, 1986: 136-167.

SEDGEWICK, R. Algoritms ADDISON-WESLEY, 1983.

WATERS, N. Harvard, Geography and GIS: A Race to the Top GeoWorld, 2006.