PgRouting: un ejemplo

Oscar Javier Puentes Puentes
oskarj84@gmail.com

Versión 0.2

Abril de 2013

El presente documento describe a través de la solución de un problema de rutas la utilización de el componente pgRouting.

Índice de contenido

Introducción

El componente pgRouting es un producto de software libre desarrollado para extender las funcionalidades de enrutamiento de las bases de datos geoespaciales en PostGIS/PostgreSQL.

Descripción del problema

Se desea realizar un ejercicio practico utilizando el componente pgRouting para encontrar la ruta mínima entre dos puntos de la ciudad de Bogotá.

Prerequisitos

- PostgreSQL Server 9.1 (9.1.8)
- PostGIS 2 (2.0.1)
- Apache Server 2 (2.2.22) [opcional si se quiere tener la documentación disponible en una red local]

Instalación

1. Agregar el repositorio al sistema:

```
$ sudo add-apt-repository ppa:georepublic/pgrouting
$ sudo apt-get update
```

Si los comandos no son reconocidos, puede que deba instalar el paquete python-software-properties

2. Instalar los paquetes de pgRouting.

```
$ sudo apt-get install gaul-devel postgresql-9.1-pgrouting
postgresql-9.1-pgrouting-dd postgresql-9.1-pgrouting-tsp postgresql-
server-dev-9.1
```

Si no está instalado el servidor PostgreSQL y PostGIS, el proceso encontrará las dependencias y serán instaladas. La versión que instala para para PostgreSQL es la 9.1.8 y para PostGIS es la 2.0.1. Se debe tener en cuenta que si no se actualizan las fuentes de paquetes a la ultima versión de ubuntugis ,por ejemplo, el sistema instalará la versión 1.5 de PostGIS.

3. Instalar el paquete para importar datos de OpenStreetMap (opcional)

```
$ sudo apt-get install osm2pgrouting
```

Instalación de la información de práctica y configuración de la aplicación

1. Instalar el material de práctica (opcional)

```
$ sudo apt-get install pgrouting-workshop
```

La documentación queda instalada en `/usr/share/pgrouting/workshop/`.

2. Se recomienda copiar los archivos a una en el home y realizar un enlace simbólico al servidor web. Este paso no es necesario pero habilita que la documentación sea accesible en la red a la que estamos conectados.

```
$ cp -R /usr/share/pgrouting/workshop ~/pgrouting-workshop
$ sudo ln -s ~/pgrouting-workshop /var/www/pgrouting-workshop
```

3. Crear la base de datos plantilla sobre la cual se crearan las bases de datos habilitadas para pgRouting. Ubicarse dentro del directorio bin en donde está la información de practica y ejecutar el script create_templates.sh.

```
$ create_templates.sh
```

La ultima versión oficial de pgRouting es soportada por PostGIS 1.5, para la versión 2 de PostGIS no hay soporte oficial todavía. Debido a esto, se deben reemplazar los scripts de configuración originales por los que se encuentran en desarrollo actualmente; estos archivos pueden descargarse de la siguiente dirección: https://github.com/pgRouting/pgrouting/tree/master/core/sql

3. Cargar las funciones del núcleo. Ejecute las siguientes sentencias:

```
$ sudo su postgres
$ createdb routing2
$ psql -d routing2 -c "CREATE EXTENSION postgis;"
$ psql -d routing2 -f /usr/share/postlbs/routing_core.sql
$ psql -d routing2 -f /usr/share/postlbs/routing_core_wrappers.sql
$ psql -d routing2 -f /usr/share/postlbs/routing_topology.sql
```

Desarrollo del ejercicio

Para el ejercicio se utilizó la información del sistema vial de la ciudad de Bogotá que puede descargarse del sitio oficial: http://www.ideca.gov.co/index.php?q=es/content/cat%C3%A1logo-de-datos-geogr%C3%A1ficos-mapa-de-referencia

La información es descargada en formato Shapefile e importada en PostgreSQL a través de alguno de los métodos disponibles para este fin. Hay que tener en cuenta que las geometrías deben cargarse en modo simple.

Los datos cargados tienen la siguiente estructura:

```
svia
(
  gid serial NOT NULL,
  svicodigo character varying(10),
  svisentido character varying(10),
  svisgeom integer,
  shape_leng numeric,
  the_geom geometry(LineString,3116),
  CONSTRAINT svia_pkey PRIMARY KEY (gid)
)
```

Se agregan tres columnas adicionales para iniciar el proceso:

```
ALTER TABLE svia ADD COLUMN source integer;
ALTER TABLE svia ADD COLUMN target integer;
ALTER TABLE svia ADD COLUMN length double precision;
```

Por restricciones de las funciones de pgRouting el nombre de la columna de la geometría debe tener el nombre the_geom, por lo tanto debe cambiarse el nombre.

```
ALTER TABLE svia RENAME COLUMN geom TO the_geom;
```

Para crear los vértices de las vías se utiliza la siguiente función:

```
SELECT assign_vertex_id('svia',0.00001,'the_geom','gid');
```

La tabla se configura de la siguiente manera:

	gid [PK] seria	svicodigo characte	svisentid characte	svisgeom integer	shape_le numeric	the_geon geometry	source integer	target integer	length double pr
1	1	11005497	E-W/W-E	4	0.000718	0102000002	1	2	0.000718
2	2	11006876	W-E	1	0.001981	0102000002	1259	756	0.001981
3	3	10010289	E-W/W-E	4	0.000298	0102000002	179	4374	0.000298
4	4	10006232	E-W/W-E	4	0.000384	0102000002	3063	5475	0.000384
5	5	10006965	N-S/S-N	4	0.000917	0102000002	2682	2629	0.000917
6	6	10000487	E-W/W-E	4	0.000352	0102000002	4240	2985	0.000352
7	7	10000627	N-S/S-N	4	0.000276	0102000002	5517	5548	0.000276
8	8	11010654	N-S/S-N	4	0.001766	0102000002	3	4	0.001766
9	9	11012204	N-S/S-N	4	0.000392	0102000002	5	6	0.000392
10	10	50006408	N-S/S-N	4	0.000682	0102000002	7	8	0.000682
11	11	16000003	E-W/W-E	4	0.001383	0102000002	9	10	0.001383
12	12	50006469	E-W/W-E	4	0.001983	0102000002	11	12	0.001983
13	13	16001171	E-W/W-E	4	0.000425	0102000002	13	14	0.000425
14	14	13000925	E-W/W-E	4	0.001918	0102000002	15	16	0.001918
15	15	13000525	E-W/W-E	4	0.001597	0102000002	17	18	0.001597
16	16	50006965	E-W/W-E	4	0.000436	0102000002	19	20	0.000436

El primer termino corresponde a la tabla en donde se encuentran las geometrías.
El segundo termino corresponde a la tolerancia mínima de la distancia para crear vértices cercanos. Se debe tener en cuenta la escala y la proyección para que los vértices no queden superpuestos.
El tercer termino corresponde al nombre de la columna que contiene la geometría, a pesar de que se puede enviar el nombre como parámetro, la columna debe llamarse 'the_geom' pues internamente ese nombre es usado.
El ultimo termino corresponde a la columna que contiene el identificador geométrico.

Una vez se ejecuta la función las columnas source y target son actualizadas.

Se debe actualizar la longitud de las lineas, esto se realiza con la siguiente instrucción:

```
UPDATE svia SET length = ST_LENGTH(the_geom);
```

Para encontrar la ruta mínima entre dos vértices, se puede utilizar la función shortest_path:

```
CREATE TABLE shortest_path AS
SELECT * FROM shortest_path('
                SELECT gid as id,
                         source::integer,
                         target::integer,
                         length::double precision as cost
                        FROM svia',
                1, 455, false, false);
```

El primer termino es una cadena de texto que contiene una consulta en donde se especifica la información a utilizar.
El segundo y tercer termino corresponden al nodo inicial y nodo final sobre los cuales se debe hallar la ruta.
El tercer termino indica si los arcos tienen dirección
El ultimo termino indica si contiene costos de reversa.

Después de la ejecución se crea una tabla/vista que muestre la ruta encontrada:

```
CREATE TABLE shortest_geom AS
SELECT svia.* FROM
shortest,
svia
WHERE
shortest.edge_id = svia.gid
```

La ruta encontrada después de la ejecución es:

En el ejemplo anterior como se puede notar no se utilizó la dirección de las vías, esta dirección esta almacenada en el campo svisentido en donde se pueden identificar 'S-N', 'N-S', 'W-E' y 'E-W' que corresponden a las vías con un solo sentido y 'S-N/N-S' que corresponden a las vías con doble sentido.

A las vías de un solo sentido se les asigna un valor muy alto en la columna reverse_cost:

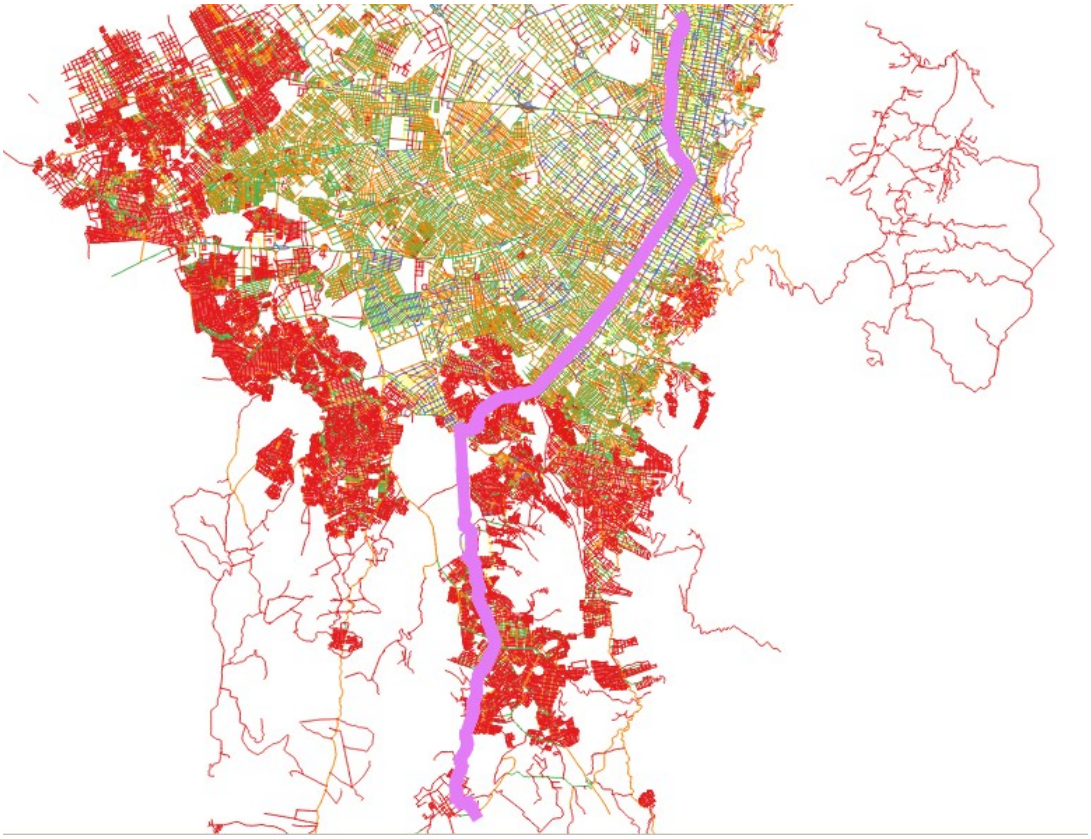
```
UPDATE svia_2 SET reverse_cost = length + 100000 WHERE svisentido =
'E-W' OR svisentido = 'N-S' OR svisentido = 'W-E' OR svisentido = 'S-
N';
```

Adicionalmente hay algunas vías que no tienen sentido, por lo tanto no serán utilizadas dentro de los cálculos:

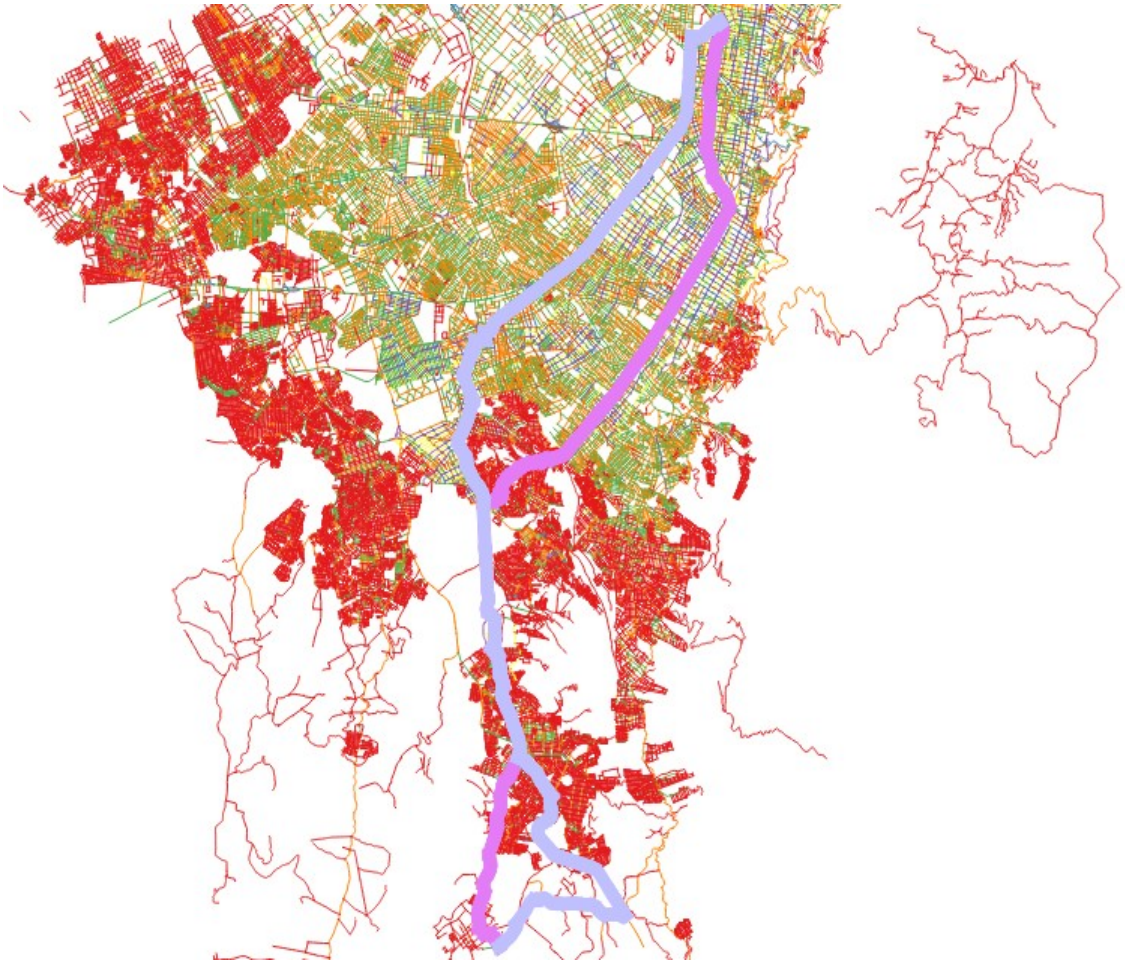
```
CREATE TABLE shortest_path_directed AS
SELECT * FROM shortest_path('
                SELECT gid as id,
                         source::integer,
                         target::integer,
                         length::double precision as cost,
                         reverse_cost::double precision
                        FROM svia_2 WHERE svisentido is not null',
                35, 1814, true, true);

CREATE TABLE shortest_path_directed_geom AS
SELECT svia_2.* FROM
```

```
  shortest_path_directed,
  svia_2
WHERE
  shortest_path_directed.edge_id = svia_2.gid
;
```

Si se ejecutara la función sin tener en cuenta los sentidos de las vías, la solución sería (de color violeta):



Y teniendo en cuenta el sentido (color lila):



Bibliografía

PGROUTING COMMUNITY. pgRouting Workshop. Consultado en http://workshop.pgrouting.org. Marzo 22 de 2013

REGINA O. OBE, LEO S. HSU. PostGIS in action. Ed. Manning Publications Co. 2011

Gregor The Map Guy. http://gregorthemapguy.blogspot.com/2012/07/pgrouting-and-postgis-2.html