GEOM90018 Spatial Databases A3 - Network Analysis

Objectives

By the end of this practical you should be able to:

1. create and retrieve information about objects based on their spatial relations in networks; and
2. perform specific network analysis tasks and queries.

This practical should require *two weeks* to complete.

Overview

This practical will introduce you to the basics of spatial analysis of networks in SQL. We will use the spatial capabilities of PostGIS DBMS. Basic spatial operations in networks will be covered.

Spatial networks are sets of nodes connected by edges, such as might be used to represent cities connected by roads or railways. In network models, the Euclidean distance between nodes does not have to equal the network distance. Edges (links) in the network may be directed or undirected. In addition to connectivity, networks in a spatial DBMS may have geometric information associated with them (for instance, geographic coordinates of the nodes or the geographic shape of edges).

Recommended Reading for Assignment

This assignment requires a good understanding of networks and the associated costs for the traversal when making routing decisions. It is highly recommended to visit the following articles that give an excellent overview about PGRouting (this is the core package that will be used for Assignment 3 and it is already deployed in the PostgreSQL instance for our lab sessions) and network cost arguments used in the routing algorithms. Note that PGRouting function signatures and parameters change over different versions. The database instance that you use for the lab activities has the latest version 3.0.0-rc1. Formulate the output using the latest API versions only.

1. PGRouting Reference Manual (3.0.0)
2. PG Routing Workshop
3. Understanding Cost Arguments for PG Routing

Exercise P 2.1 - Network data entry

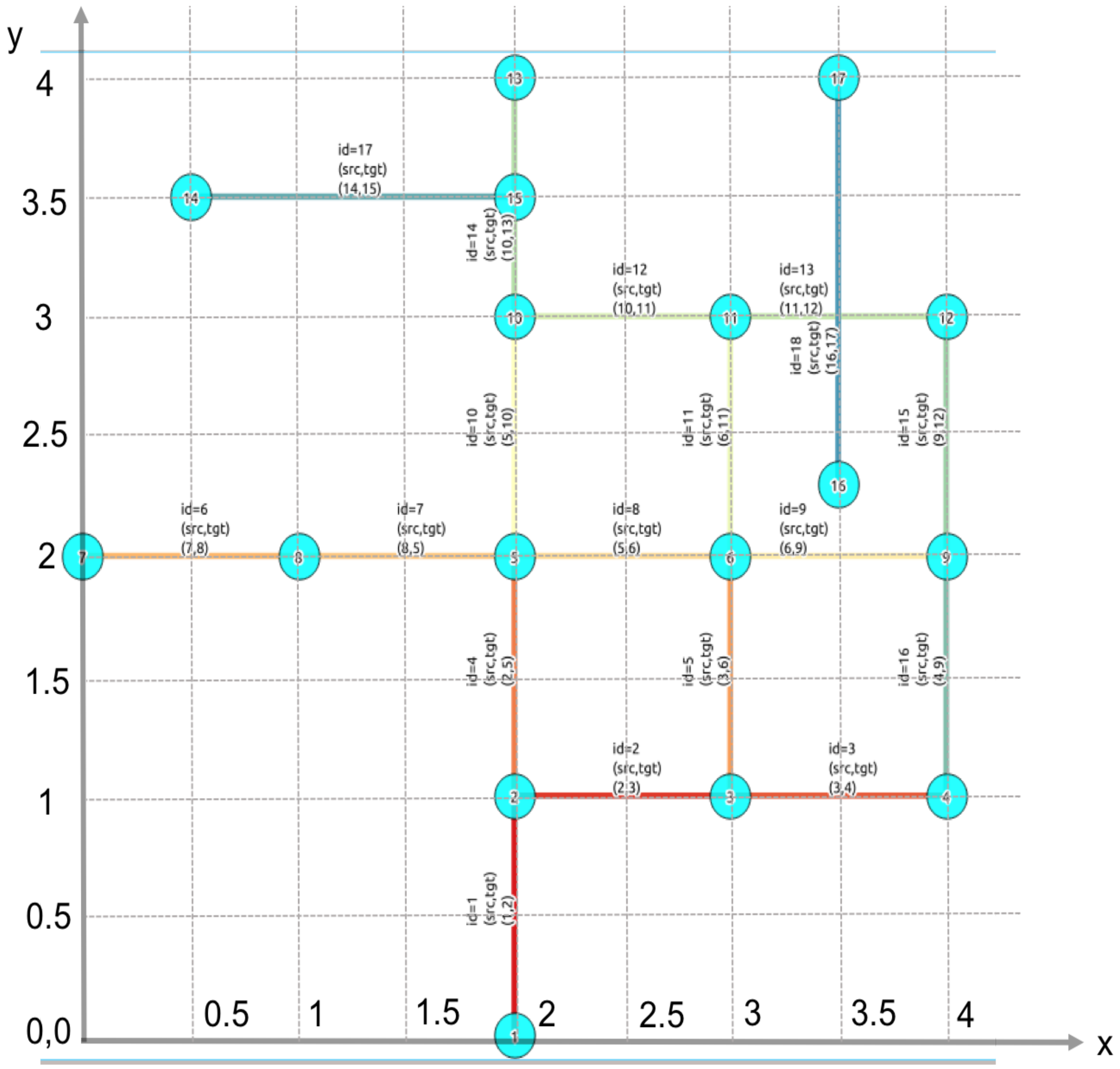
Spatial networks are sets of nodes connected by edges, such as cities connected by roads or railways. An edge is in essence a pair of nodes. As a result, in the PostGIS extension PG_ROUTING (we use version 2.6, which you can check using SELECT pgr_version(). Make sure you look for the appropriate documentation version as well, functions do change), a network is in its simplest form an edge table with "source" and "target" fields (the identity of a pair of nodes) and a "cost" field (the weight associated with that edge). The geometry associated with the nodes does not necessarily have to accord with the underlying network topology. For example, the Euclidean distance between nodes is not necessarily equal to the weight associated with the edge connecting those nodes. In this exercise, we will create a simple network, fill it with data, and perform some basic analysis.

1. Login to the course database server using the same connection string and tools as in previous practicals.
2. *Edge table creation:* As mentioned above, a PostGIS network is essentially a table with source, target, and cost fields, but also potentially with a number of other fields:

```
CREATE TABLE edge_table (
id serial,
dir character varying,
source integer,
target integer,
cost double precision,
reverse_cost double precision,
x1 double precision,
y1 double precision,
x2 double precision,
y2 double precision,
to_cost double precision,
rule text,
the_geom geometry(Linestring) );
```

   What do you think the other attributes are for? Reverse_cost? x1, y1, x2, y2? the_geom? Familiarize yourself with the documentation on pgrouting and find out what each of the different attributes are for.

3. *Populate edge table:* We will now insert data into the edge table. We will construct the network below:



The following example shows the code necessary to create the first six edges. Using this pattern along with the network above, create the remaining edges in the table.

```
INSERT INTO edge_table (cost,reverse_cost,x1,y1,x2,y2,source,target) VALUES (1, 1, 2, 0, 2, 1, 1 ,2);
INSERT INTO edge_table (cost,reverse_cost,x1,y1,x2,y2,source,target) VALUES (1, 1, 2, 1, 3, 1 ,2 ,3);
INSERT INTO edge_table (cost,reverse_cost,x1,y1,x2,y2,source,target) VALUES (1, 1, 3, 1, 4, 1, 3, 4);
INSERT INTO edge_table (cost,reverse_cost,x1,y1,x2,y2,source,target) VALUES (1, 1, 2, 1, 2, 2, 2, 5);
INSERT INTO edge_table (cost,reverse_cost,x1,y1,x2,y2,source,target) VALUES (1, 1, 3, 1, 3, 2, 3, 6);
INSERT INTO edge_table (cost,reverse_cost,x1,y1,x2,y2,source,target) VALUES (1, 1, 0, 2, 1, 2, 7, 8)
...
```

Exercise P 2.2 - Routing

1. *Find some shortest routes*: The most basic algorithm for computing shortest paths is Dijkstra's algorithm. The pgrouting function for computing Dijkstra's algorithm is called pgr_dijkstra. The basic mechanism for calling the routing function is a little different to previous SQL functions we have used. First, the template (see pgr_dijkstra for documentation. Not the use of the boolean modifier - what does it do?):

```
SELECT seq, node, edge, cost FROM pgr_dijkstra('SELECT id, source, target, cost FROM edge_table', 3, 5, false);
```

   Let's look at that a little bit more closely. The pgr_dijkstra function takes the following arguments:

```
pgr_dijkstra(text sql, integer source, integer target, boolean directed);
```

   The first argument is an SQL query. The precise details of the query vary with the different routing algorithms, but for Dijkstra's algorithm in pgrouting a table containing just the edge ids, the source and target node ids, and the cost for each edge is required. Then the source and target node for routing from and to is required (i.e., in the example above, routing from node 3 to node 5). Finally come two Boolean arguments identifying whether the network is directed (true if an edge from node a to node b does not necessarily imply that there exists and edge from node b to node a) and whether the network has reverse costs (true if the cost of traveling along an edge from node a to node b could be different to the cost of traveling from node b to node a). When you run the routing query above, you should get the following output:

seq	node	edge	cost
1	3	2	1
2	2	4	1
3	5	-1	0

   Look at the table and the network above. Can you work out what the answer to your query means? Ask a demonstrator if you are having difficulties. Now do the following exercises to make sure you are understanding the output.

   1. Rerun the query between serveral different pairs of nodes, including 7 and 10, 12 and 3, 4 and 13. Each time check the answer carefully to make sure it accords with what you expected. If it doesn't, then investigate and correct the problem.

2. Now adapt your query to compute only the total cost of each shortest path. Then change it again to list only the path itself, in terms of the sequence of nodes followed.
3. Now run the shortest path query between nodes 14 and 11, and from 6 to 16. Is the answer what you expected? Understand what the correct answer should be and why you get that answer.
4. Switch the cost of the edge from nodes 2 to 3 to -1. Now find the shortest route from nodes 1 to 3. What do you notice? What does cost -1 mean?
5. Now find the shortest route from 3 to 1, but with reverse edge costs enabled.

1. *Add edge geometry:* Notice that all of these shortest paths computed operate purely on the topology of the network, they required no information about the geometry. (Try to display the table in QGIS.) Although Dijkstra's algorithm does not require geometry, the A* algorithm does. (Why?) Run the following A* query:

```
SELECT seq, node, edge, cost FROM pgr_astar( 'SELECT id, source, target, cost, x1, y1, x2,
y2 FROM edge_table', 1, 5);
```

   Note that even though the geometry of the edge end nodes is required for A*, this is not structured as geometry data type. Work out how to add geometry to the edges using the the_geom column. HINT: you will need to construct the geometry from the x and y fields. The PostGIS functions st_makeline and st_point can be used to do this automatically in one UPDATE statement. When you are done, check in QGIS your network has been correctly created.

3. *Advanced routing function*: Try the following challenges:
   1. Compute the shortest paths from node 3 to both nodes 1 and 10 in one step using a single pass of the pgr_dijkstra function.
   2. Compute using an all pairs shortest path algorithm (e.g., pgr_floydWarshall) the costs for paths between every pair of nodes in the network. (And check this carefully. What do you notice?)
   3. Use the k shortest paths algorithm (pgr_ksp) to compute the three shortest paths between nodes 11 and 1.
   4. Now work out how to display the shortest path generated by one of your queries in QGIS.
4. *Apply to some real networks*: There is a small portion of a real road network, the network around the University of Melbourne, included in the spatial data schema, tables spatial.unimelb_edges and spatial.unimelb_nodes. You can apply the skills you now have to routing in this real road network. Have a look at the structure of these tables, and load the data into OpenJump or QGIS to view it. Selecting a suitable origin and destination node, and adapt the routing examples you've already encountered to find the shortest path between those two locations. Note that the routing itself may be a little slower as networks become larger. Use the techniques you developed in 3 above to display the shortest path in QGIS.
5. *Note on pgr_createTopology*: You may notice in some documentation there is a function pgr_createTopology that is often used on a raw network to find intersections between edges and build the network topology from those intersections. This function is not available to you, because of security restrictions in this version of pgrouting. Instead, we will always create our topology directly using the edge table;

Assessment A3 - Network Analysis

Preliminaries

1. This assignment is worth 8% of your final class mark.
2. The due date for the assignment is clearly stated on course web site, accessible via the LMS.

3. NO LATE ASSIGNMENTS WILL BE ACCEPTED. It is your responsibility to ensure you are aware of the assignment due date posted on the course web site.

Submission

Your task is to answer the 8 questions in the "assignment" section below. You should submit one plain text file (named <studentno>.txt where <student_no> is your student number). You must submit your file online using the course web site (accessible via the LMS). No other submission method is acceptable.

Marking

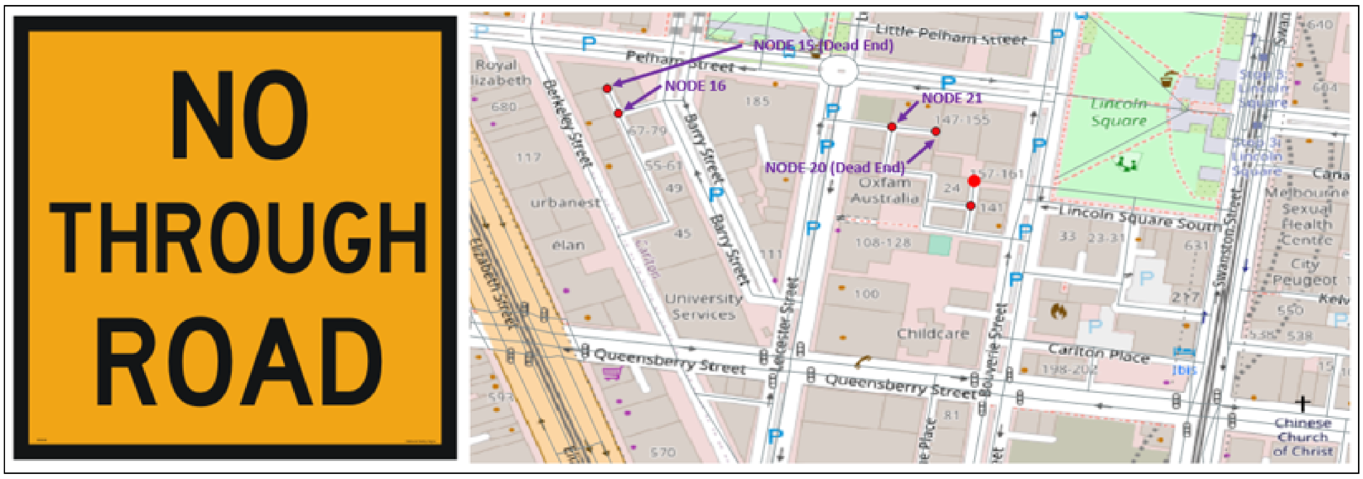
- Your submission will be marked out of **8**, with up to one mark awarded per question.
- A perfect answer will be awarded **1 mark**. A near-perfect answer (almost exactly right, but perhaps with some slight mistake or missing step) may be awarded **half a mark**. Anything else will be awarded **0 marks**.
- **Marks** will be **deducted** for any incorrect submissions (e.g., using a Word or HTML document instead of a plain text file; files that contain other than 8 answers with one answer or comment per line; answers not in the correct order; answers that contain non-SQL parts, such as question numbers; or SQL queries lacking correct termination with a semicolon, ";").
- It is not possible to get negative marks (e.g., an assignment score of 1 out of 8 with 2 marks deducted for incorrect submission will be a final mark of 0, not -1).

Assignment

This assignment concerns the network in the immediate vicinity of the University of Melbourne, found in the spatial.unimelb_edges table. Using that data, your assignment is to provide SQL queries to answer the following 8 questions. Note that the "car" field in unimelb_edges is non-zero when cars are allowed to travel along that edge. Other fields similarly provide information about other modes of transport.

1. Imagine you are at the entrance of the Queen Victoria Market (node 1335) and you need to meet a friend at the entrance of the Union House (node 1811). Your task is to analyse the shortest paths for three different modes of transport: car, bike, and foot. For each mode of transport print out the minimum distance and number of edges you need to traverse to get to your destination.
2. How many nodes can be reached in the network by walking at least 2.4 kms but no more than 3 kms from Node 550? List the destination node and the distance (in metres). Order the results by the highest distance first. Consider the network as an undirected graph for your calculations.
3. How many nodes in the network can you reach and return from if you walk for 5 minutes at a constant speed of 10km/h from the Union House entrance (node 1811)? To clarify, nodes need to be walked to and walked back from to the starting node 1811 in the specified time. Print only the total count of such nodes.
4. What is the difference in travel time [mm:ss] between onward and return journeys from node 252 to node 857 in the unimelb network if you are travelling by car at a constant speed of 30km/h?
5. How many nodes are completely disconnected from Node 34? In other words, there should be no graph that exists between these nodes and 34 in the current dataset. (To achieve this, you can use the pgr_drivingdistance function by applying certain heuristics or solution based on your understanding of the spatial distribution of data). Consider the network as undirected. You can use QGIS to visualize your results.
6. Based on visitor's feedback, University of Melbourne has recently decided to enhance the driving experience within its campus, by putting up appropriate signages near a node, when driving further ahead from this node will be a potential dead end (node) ahead. This is to prevent users from accidentally driving into a road that is a dead end, when they are navigating inside the campus. To understand this better, in the example below, a "NO THROUGH ROAD" signage at Node 16 indicates to the driver the path ahead is a dead end (ending with Node 15). Similarly, a signage at Node 21 (indicating a dead end ahead ending with Node 20).
   Considering that the network is undirected and traversed by car, list all the nodes near which the signage should be erected and the corresponding dead-end node. The output should only be for

those nodes, having IDs between 1900 and 2000. For our example above, the output would be displayed as two rows, with data (21,20) and (16,15).



7. Extending the scenario described in Question 6, University of Melbourne wants to pay special attention to the maintenance of all roads segments (edges), that either (i) terminate at a dead-end node, or (ii) disconnects a sub-set of nodes from the university network. This is to ensure that during emergency scenarios, there are no nodes or sub-sets of nodes orphaned, (not traversable from any other node in the university network) in case the existing road gets damaged. How many such road segments (edges) exist in the university network ("unimelb_edges" table) that satisfy conditions (i) and (ii)? what is the total length of these road segments (in kilometres). Use only the geometry column in "unimelb_edges" table to compute the length.
8. A tank truck has travelled from the Gatehouse Street (node 586) to the 7-Eleven petrol station (node 1870) traversing the shortest path via College Crescent (node 655) and Rathdowne Street (node 1388). Due to the malfunction in the tank, the truck was leaking petrol on its way contaminating everything in the 20m radius. Your task is to report the total size of contaminated area in square meters.