Using AWS and PostgreSQL to make your own routing engine

DC HACK && TELL

MAY 11 2016

Link to project:
https://github.com/ampetr/pgrouting

Intro

- Many free routing API's available
  - Google Maps directions API
  - HERE navigation
  - Mapzen Valhalla
  - OpenStreet Routing Project
  - Graph hopper

- Subject to API limits
- May not return the route itself, (or will be slow to do so).
- May need to know the IDs of edges in route

- Also free graph theory libraries for shortest path
  - igraph (python, R)
  - networkx (python)

- Involve creating a "graph object," still may be slow

Intro

- Today I'll show how to use pgRouting to make an okay-ish routing engine, with the benefit that
  - Can calculate huge number of shortest path routes
  - Built on PostgreSQL database, so routes are efficiently stored
  - No limit on usage
- Okay-ish because
  - Built on OpenStreetMap (imperfect)
  - Does nor know about arbitrary forbidden turns
  - Disallows *any* U-turns
  - Doesn't know about traffic conditions

pgRouting Extension for Postgres

- Amazon web services does have PostgreSQL available through RDS product
- However, it does not support pgRouting extension
- So use regular EC2 instance and install Postgres on it.
  - Instructions here: https://github.com/ampetr/pgrouting
- Shortest path syntax:

One-to-one

```
create table test_route
AS
SELECT seq, id1 AS vertex_id, id2 AS edge_id, cost
    FROM pgr_dijkstra('
                SELECT   edge_id as id,
                         vertex_id_1::int4 as source,
                         vertex_id_2::int4 as target,
                         length_meters::float8 as cost
                        FROM road_edges',
                104815,108913,  true, false); -- " arguments are:
                sql,source_id,target_id,directed,has_reverse_cost"
;
```

One-to-many:

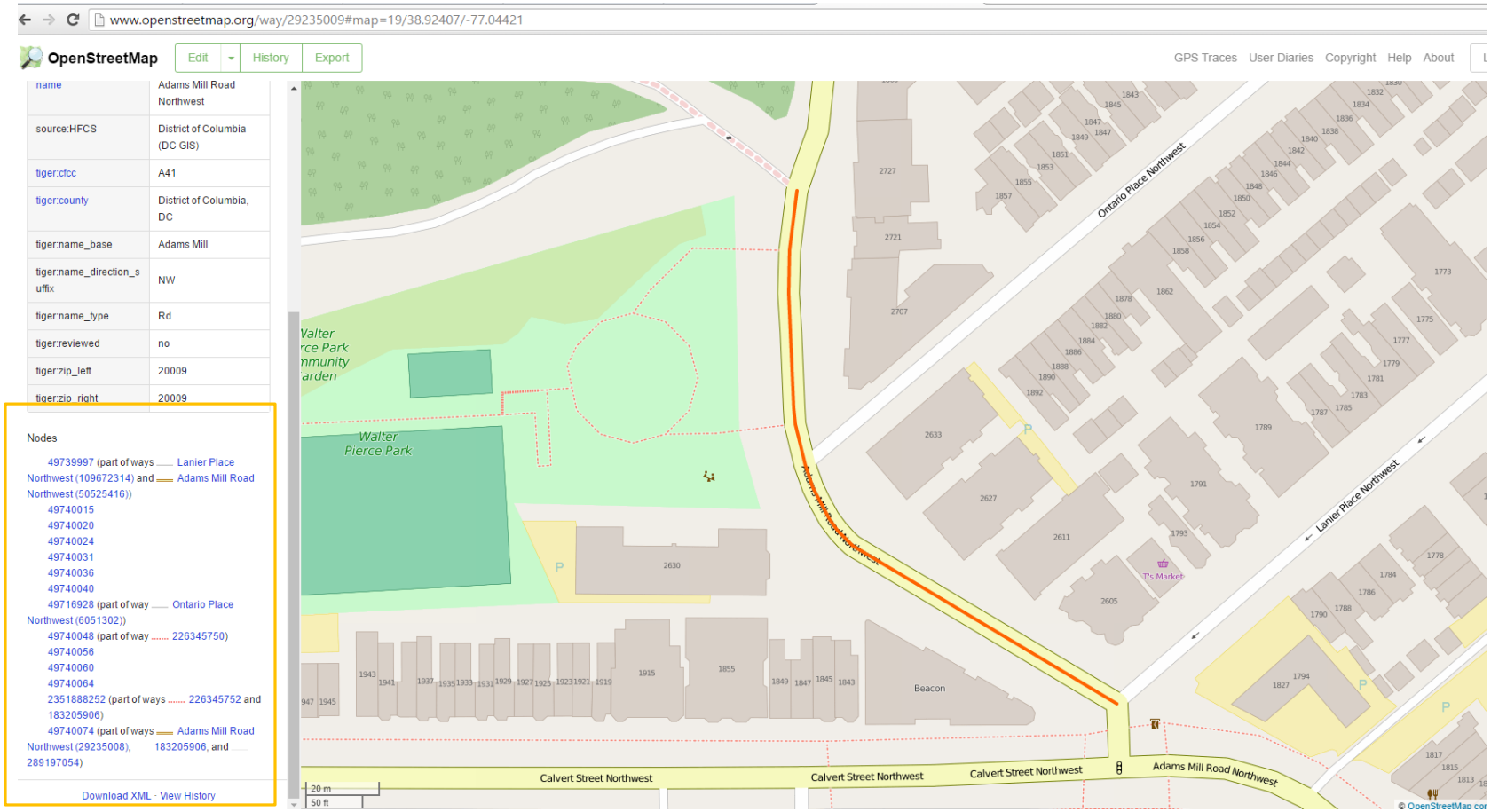
```
create table way_route_o2m
AS
SELECT  seq, id1 AS target, id2 AS edge, cost
    FROM pgr_kdijkstrapath('
                SELECT   edge_id as id,
                         vertex_id_1::int4 as source,
                         vertex_id_2::int4 as target,
                         length_meters::float8 as cost
                        FROM road_edges'
                100499,
                (select array_agg(vertex_id) from road_nodes limit 20),
            true, false);
```

Edges table should have index on edge_id column

OpenStreetMap

- Three types of objects: nodes, ways, relations
- Nodes define the break points in ways
- Ways include roads, but also bodies of water, borders, etc.
- Relations are shapes made of ways

Way-node structure

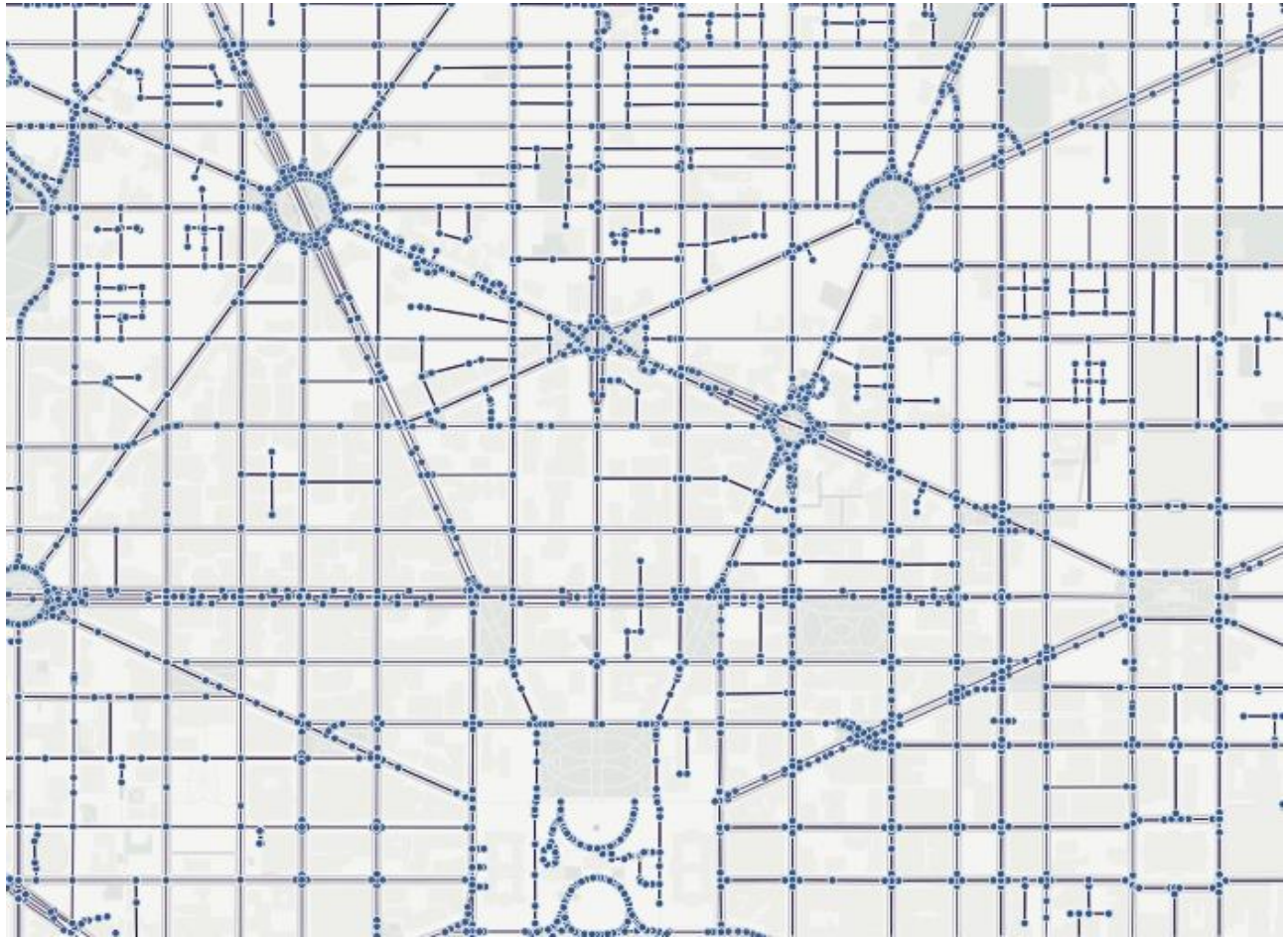


Disclaimer

- pgRouting actually has a built-in function to import OpenStreetMap Data “osm2pgrouting”
- I didn’t know this at the time, so wrote my own
- Now I’m too lazy to learn the built-in function

Import OSM data

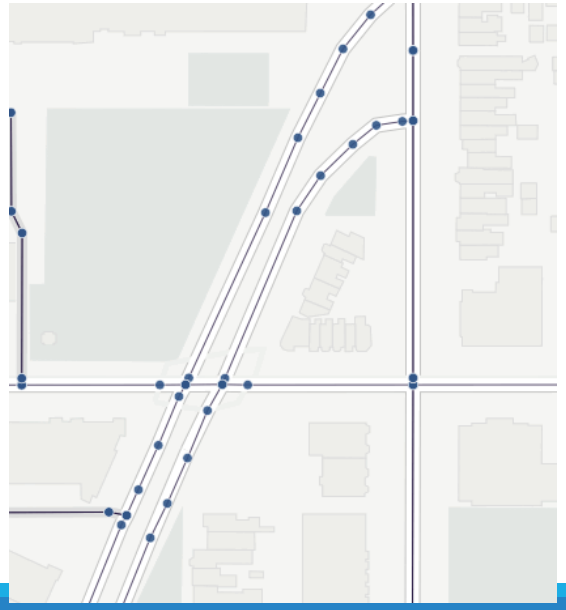
- Fave places to download OSM Data:
  - https://mapzen.com/data/metro-extracts/
  - http://extract.bbbike.org/
- Extract is ".osm" file (xml format)
- Import into postgres database
  - Python code available on github:
  - https://github.com/ampetr/pgrouting
- Parses the XML and extracts the three children: nodes, ways, relations
- Filters to ways that are tagged as roads

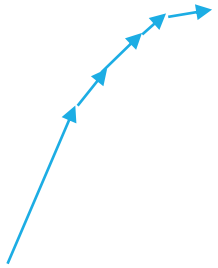


http://bit.do/osm-dc-ways

Create edges table

- Use the way-node relationship to get straight line edges
- Create reverse order way for two-way streets
- Ordering by the node_order field, use the `lead()` function to get the next node in a way
- Use `ST_MakeLine(start_point, end_point)` to create line geometry
- Get line length with `ST_Length(line)`

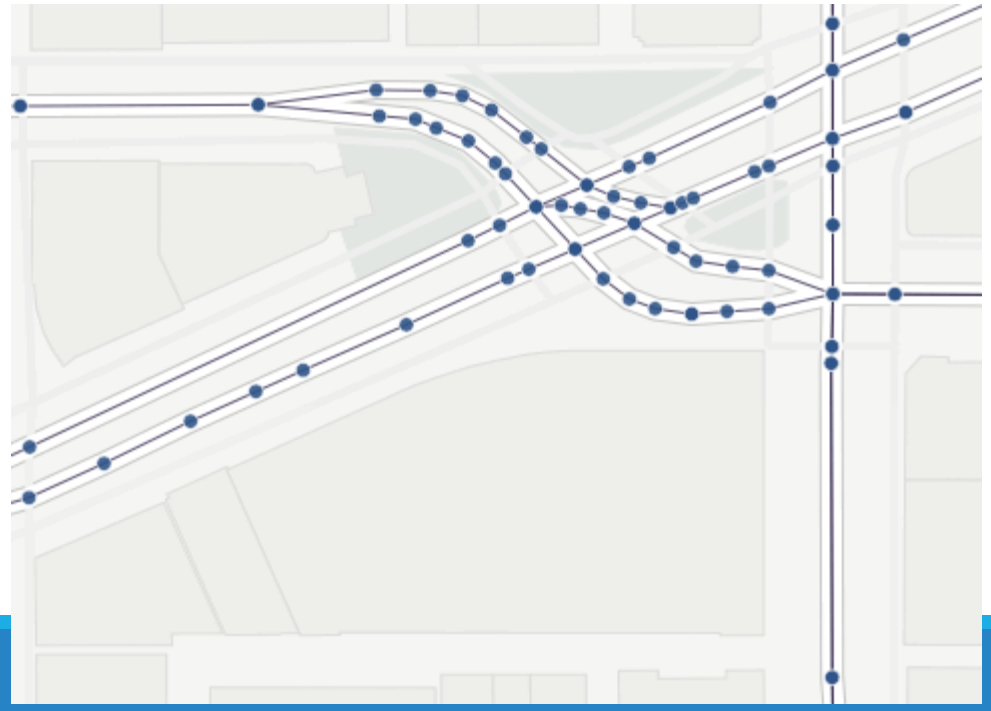




So we’re set?

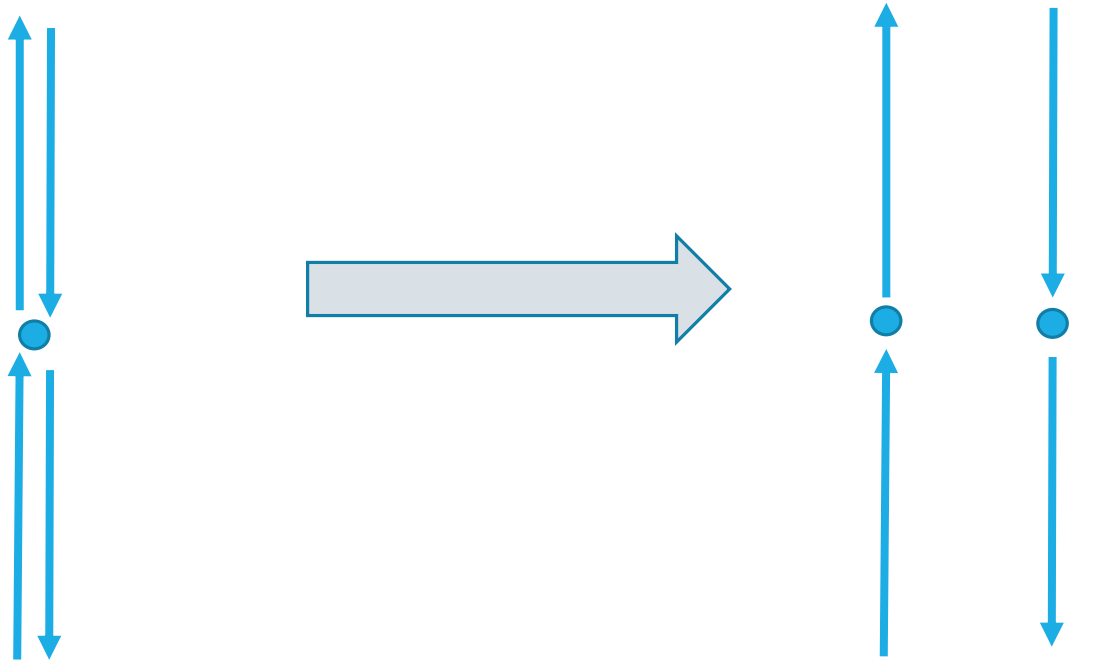
- We have a table of straight line edges, and we know their lengths
- First try

- But remember, this is a directed graph. It doesn’t know about how roads work.
- ➔ You can make a U-turn **anywhere**(not just intersections, **at any node**)



Forbid U-turns

- Identify Uturns:
  - End point of edge A is start point of B, and their bearings are roughly 180 degrees different
  - Create new nodes:



Not finished yet..

Thank you!

- Link to project on github:

Thank you!

Github: https://github.com/ampetr/pgrouting
Twitter: https://twitter.com/1littlevictory
Email: anna@split.us