USING POSTGRESQL AND FRIENDS FOR A STREET SWEEPING SOLVER PROJECT

James Marca, Activimetrics LLC

March 2020

STREET SWEEPING AND TRASH HAULING

- Pencil and paper routes
- Low-tech solutions
- Objectives:
  - minimize miles traveled
  - minimize vehicles
- The problems are typically NP-Hard

BUSINESS OPPORTUNITY

- Use Google Operations Research Tools (ORTools)
- Solve for best routes
- Initial maps using OpenStreetMap (OSM)

TECHNICAL CHALLENGES

- Need to get OSM data into a routable network
- Need to convert usual network into a LineGraph
- Need to present results

POSTGRESQL AND FRIENDS TO THE RESCUE

POSTGIS

- https://postgis.net/

> *a spatial database extender for PostgreSQL. It adds support for geographic objects allowing location queries to be run in SQL*

PGROUTING

- https://pgrouting.org/

> *extends the PostGIS / PostgreSQL geospatial database to provide geospatial routing functionality*

LOAD OSM DATA

- Use osmium https://osmcode.org/ to extract a city
- Read data into database with osm2pgrouting https://github.com/pgRouting/osm2pgrouting/wiki/Documentation-for-osm2pgrouting-v2.2

EXTRACT A CITY

```
osmium extract -p port-au-prince-poly.osm \
    -o port-au-prince-latest.osm \
    haiti-and-domrep-latest.osm.pbf
```

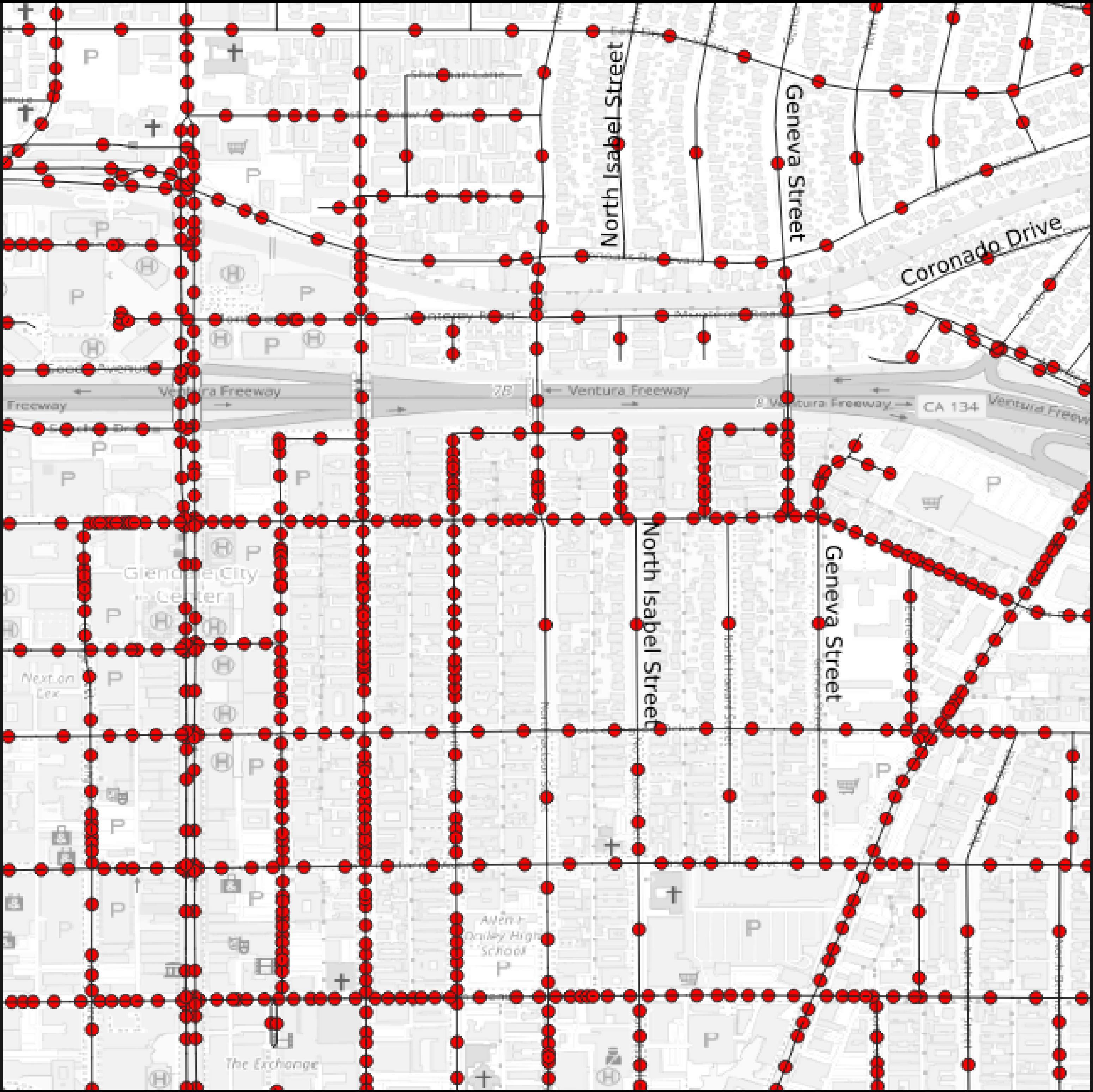
LOAD THE DATA INTO PGROUTING TABLES

```
osm2pgrouting --f data/port-au-prince-latest.osm \
              --conf data/map_config_streets.xml \
              --dbname portauprince \
              --prefix 'portauprince_' \
              --username dbuser \
              --clean
```

CLEAN OSM DATA

TOO MANY SUB-SEGMENTS

- OSM is designed for many things
- some street segments are extraneous
- Example: intersections for service roads create too many segments


North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Sheridan Lane
Goode Avenue
Ventura Freeway
Freeway
Ventura Freeway
CA 134
Ventura Freew
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange

OBJECTIVE: COMBINE SEGMENTS

- Goal is to link up segments
- Need to introspect each node
  - Is it an isolated mid-point?
  - Can it be linked to another segment?
  - But want to keep the breaks at intersections

WITH RECURSIVE

- `WITH` statements are great just to organize long SQL
- But `WITH RECURSIVE` statements are *indispensable* for problems like this
- Allows recursively combining all nodes on a street

STRATEGY

- Each segment has source and target
- Sum up all sources, all targets
- Sources, targets seen once are likely interior nodes

```
  id  |             name              | source | target | one_way |
        cost_s | rev_cost_s
------+-----------------------+---------+---------+----------+--
        ------+------------
  433 | Western Avenue            |      1 |    303 |        0 |
        1.30 |        1.30
 4725 | Glenoaks Boulevard        |      1 |   4061 |        1 |
        2.58 |       -2.58
  299 | Geneva Street             |      2 |    216 |        0 |
        26.01 |       26.01
 1735 | Glenoaks Boulevard        |      2 |   1267 |        0 |
        8.40 |        8.40
```

COUNTS OF SOURCE, TARGET

```
sources(source,count) as (
   select source,count(*) as count
   from glendale_ways group by source
   ),
targets(target,count) as (
   select target,count(*) as count
   from glendale_ways group by target
   )
```

POTENTIAL INTERIOR NODES

Any record with target count **and** source count of 1

```
possible_interiors as (
   select w.*,s.count as scount, t.count as tcount
   from glendale_ways w
   join targets t on (w.target=t.target)
   join sources s on (w.source=s.source)
   where t.count=1 and s.count=1
   ),
```

EXAMPLE RESULT

```
 source | target |          name          | scount | tcount
--------+--------+------------------------+--------+--------
      3 |   1964 | Geneva Street          |      1 |      1
     11 |   5607 | East Colorado Street   |      1 |      1
     15 |   3918 | South Central Avenue   |      1 |      1
     20 |   4529 | Harvey Drive           |      1 |      1
     31 |   2068 | East Mountain Street   |      1 |      1
```

“TRUE” INTERIORS

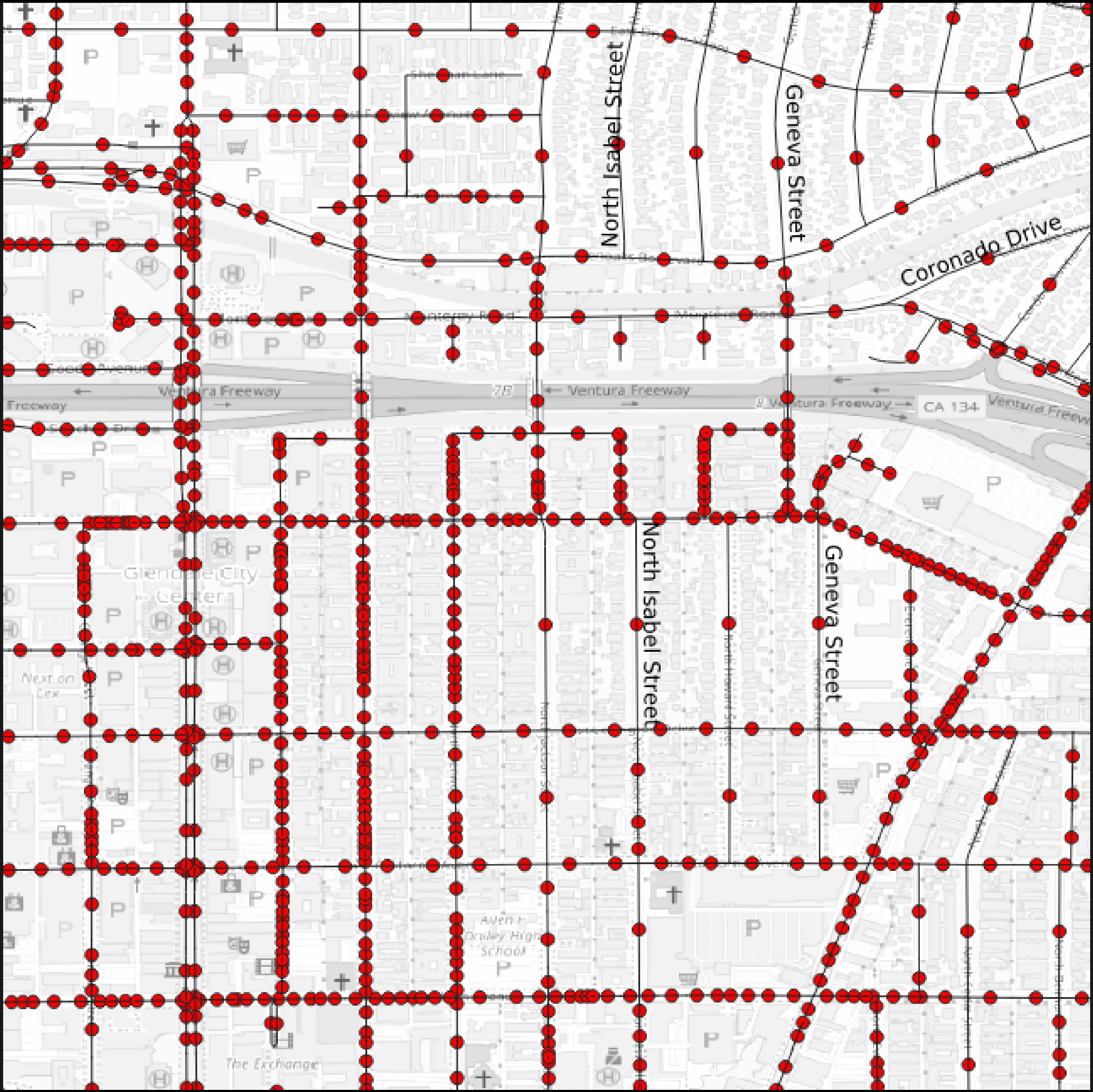
Possible interiors whose source *and* target nodes are *also* possible interior segments

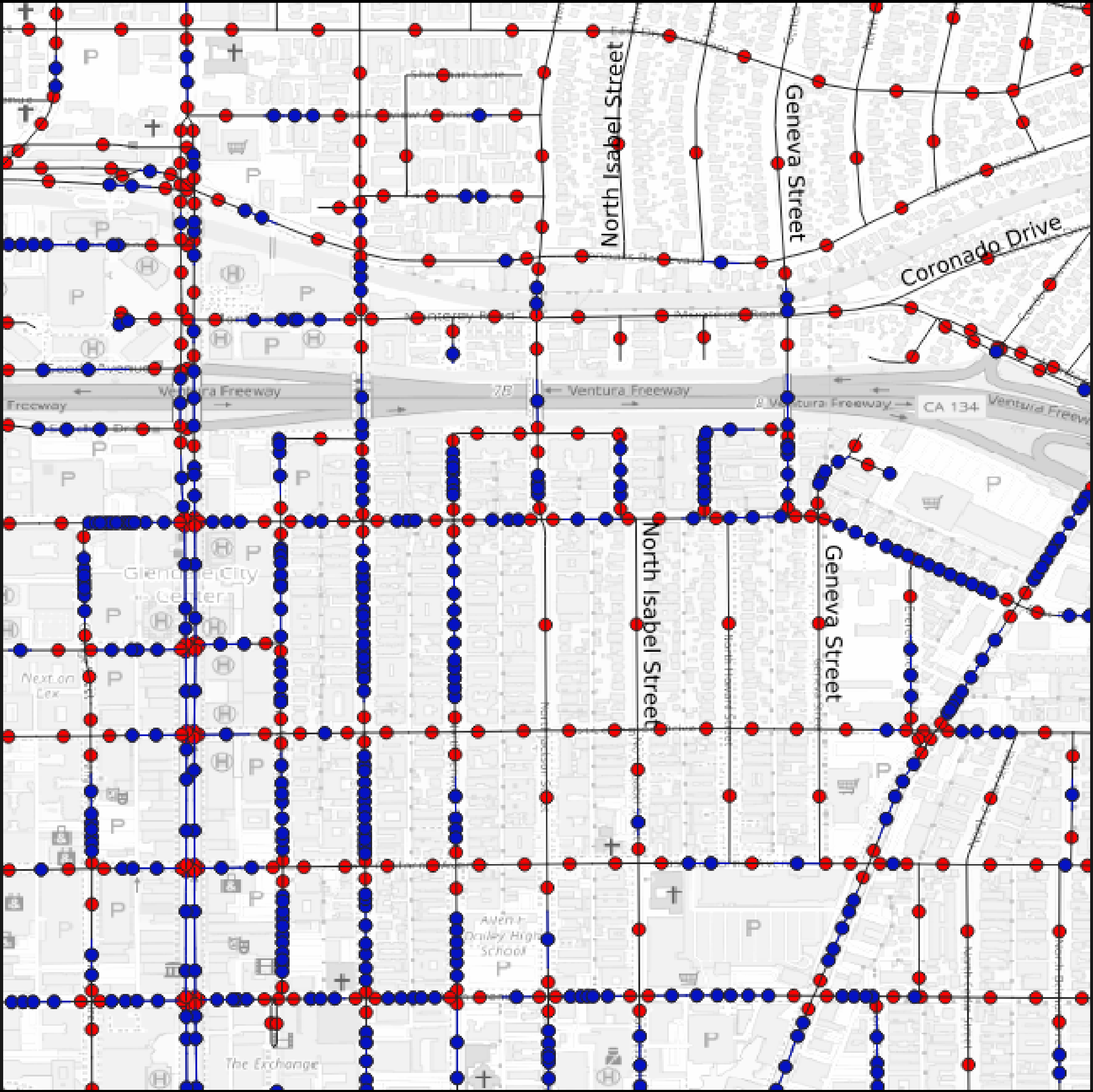
```
interiors as (
   select pi.*
   from possible_interiors pi
   join sources s on (pi.target=s.source)
   join targets t on (pi.source=t.target)
   where s.count=1 and t.count=1
   )
```

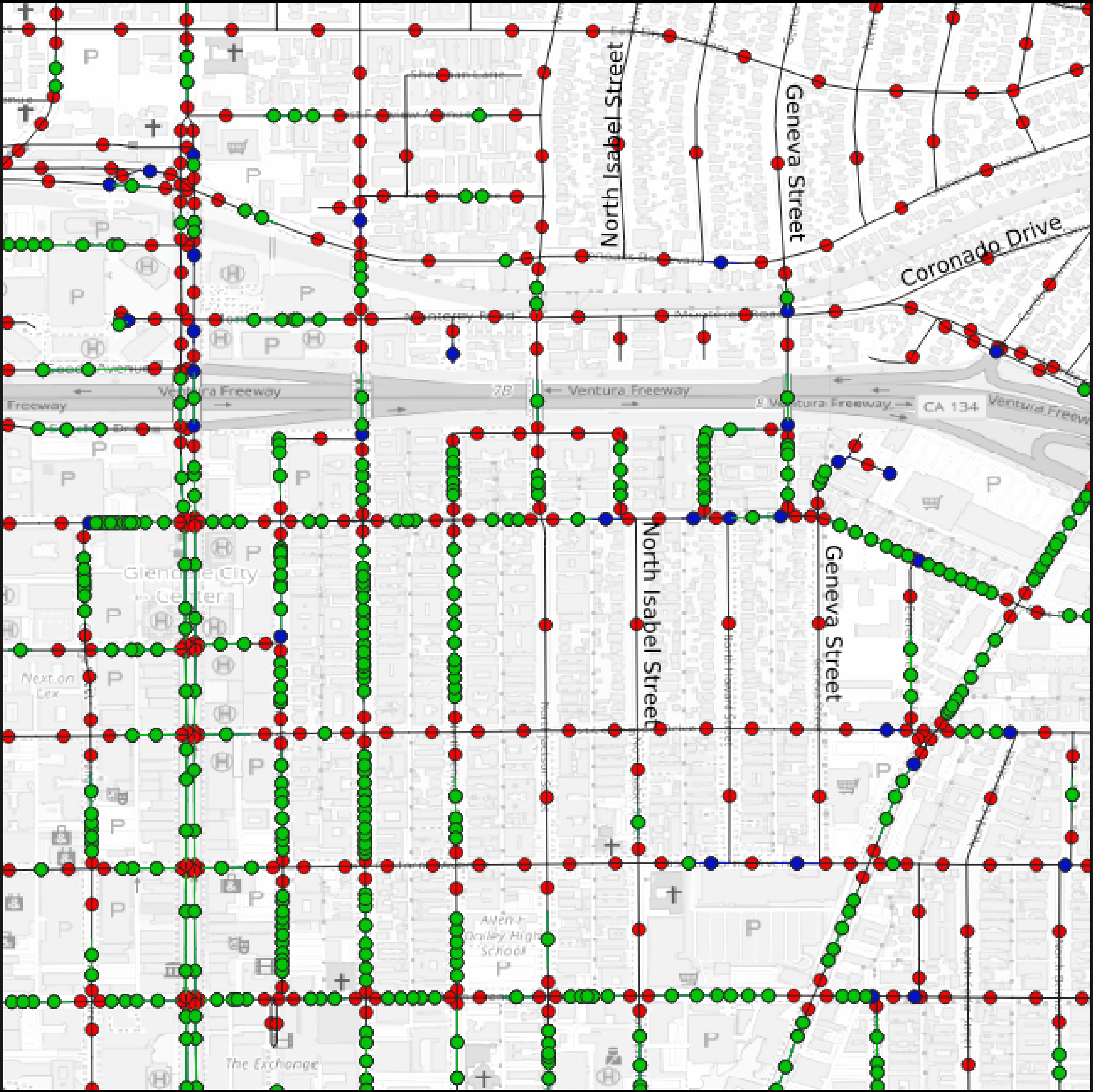
EXAMPLE RESULT

```
  id  | source | target |          name          | scount |
        tcount
------+--------+--------+------------------------+--------+---
        -----
 2509 |     37 |   1986 | North Jackson Street   |      1 |
       1
 2503 |     57 |   1977 | South Pacific Avenue   |      1 |
       1
  398 |    118 |    277 | East Mountain Street   |      1 |
       1
 2424 |    127 |   1891 | Harvey Drive           |      1 |
       1
 5282 |    148 |   4621 | Flintridge Drive       |      1 |
       1
```

BUT MAPS ARE EASIER TO VISUALIZE


North Isabel Street
Geneva Street
Coronado Drive
Sheridan Lane
Monterey Road
Ventura Freeway
CA 134
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange


North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
Ventura Freeway
Freeway
7B
Ventura Freeway
CA 134
Ventura Freew
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange


North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
Freeway
CA 134
Glendale City Center
Next on Cex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange

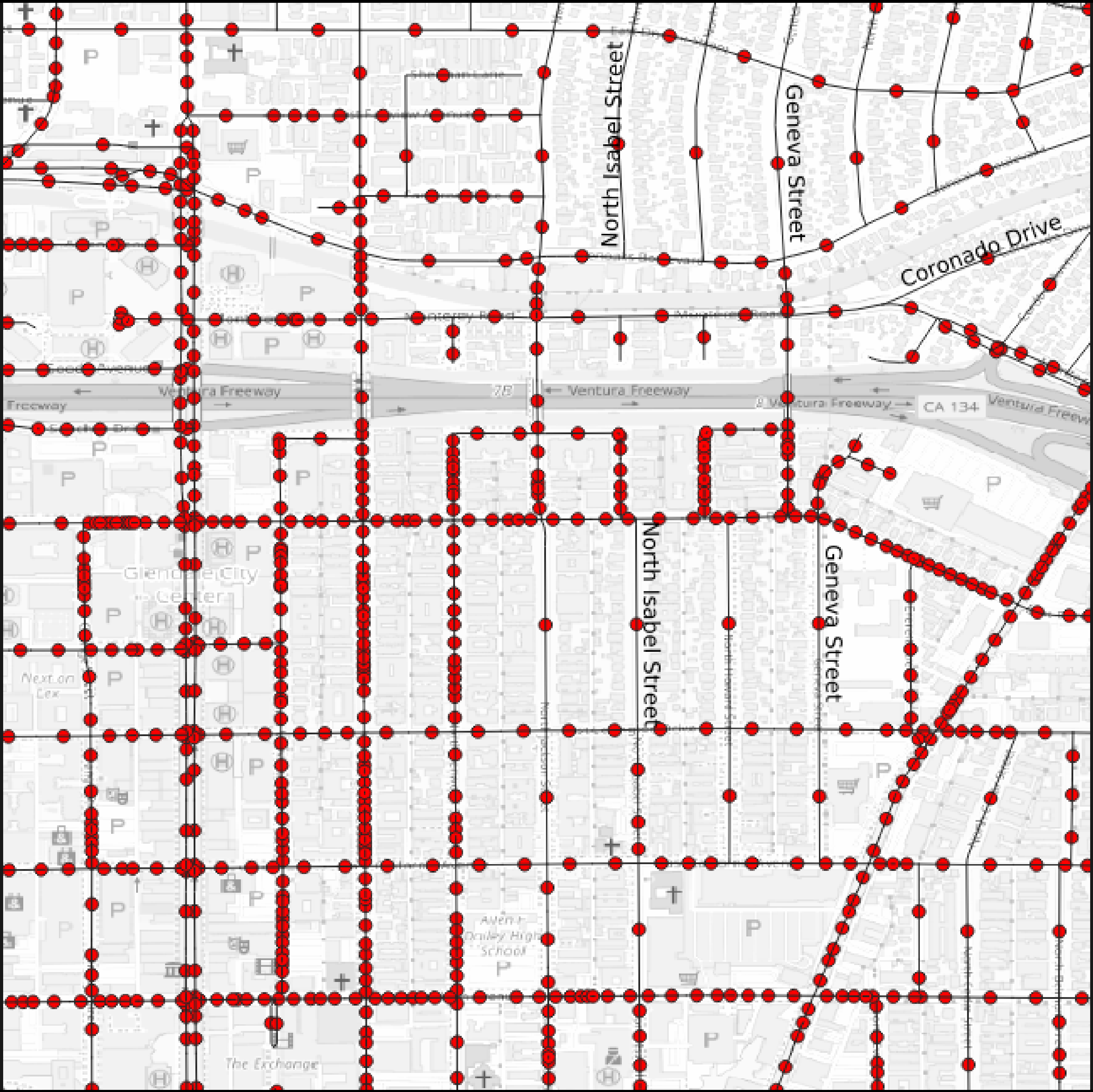
SEQUENCE STARTS

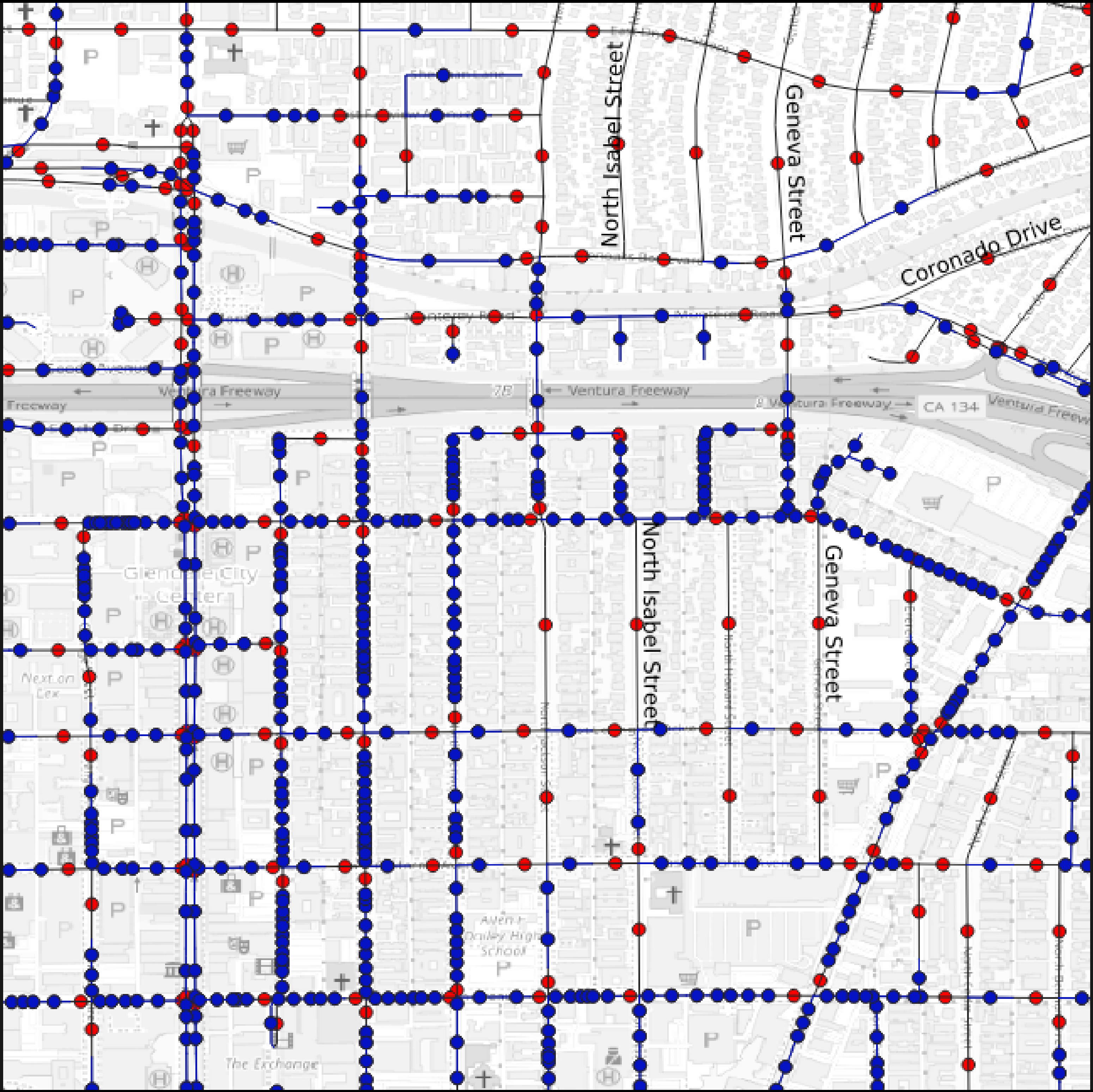
- Interior segments start and end at non-interior segments
- “Starts” are segments with
  - target is unique (count of 1)
  - source is not unique (node is source for lots of segments)
- “Ends” are segments with
  - source is unique (count of 1)
  - target is not unique

FIRST IDENTIFY POSSIBLE STARTS

- A “start” to a chain of isolated segments
- The “target” field has a count of one.

```
possible_starts as (
    select w.*, s.count as scount, t.count as tcount
    from glendale_ways w
    join targets t on (w.target=t.target)
    join sources s on (w.source=s.source)
    where t.count = 1 -- link is only one touching target
    )
```

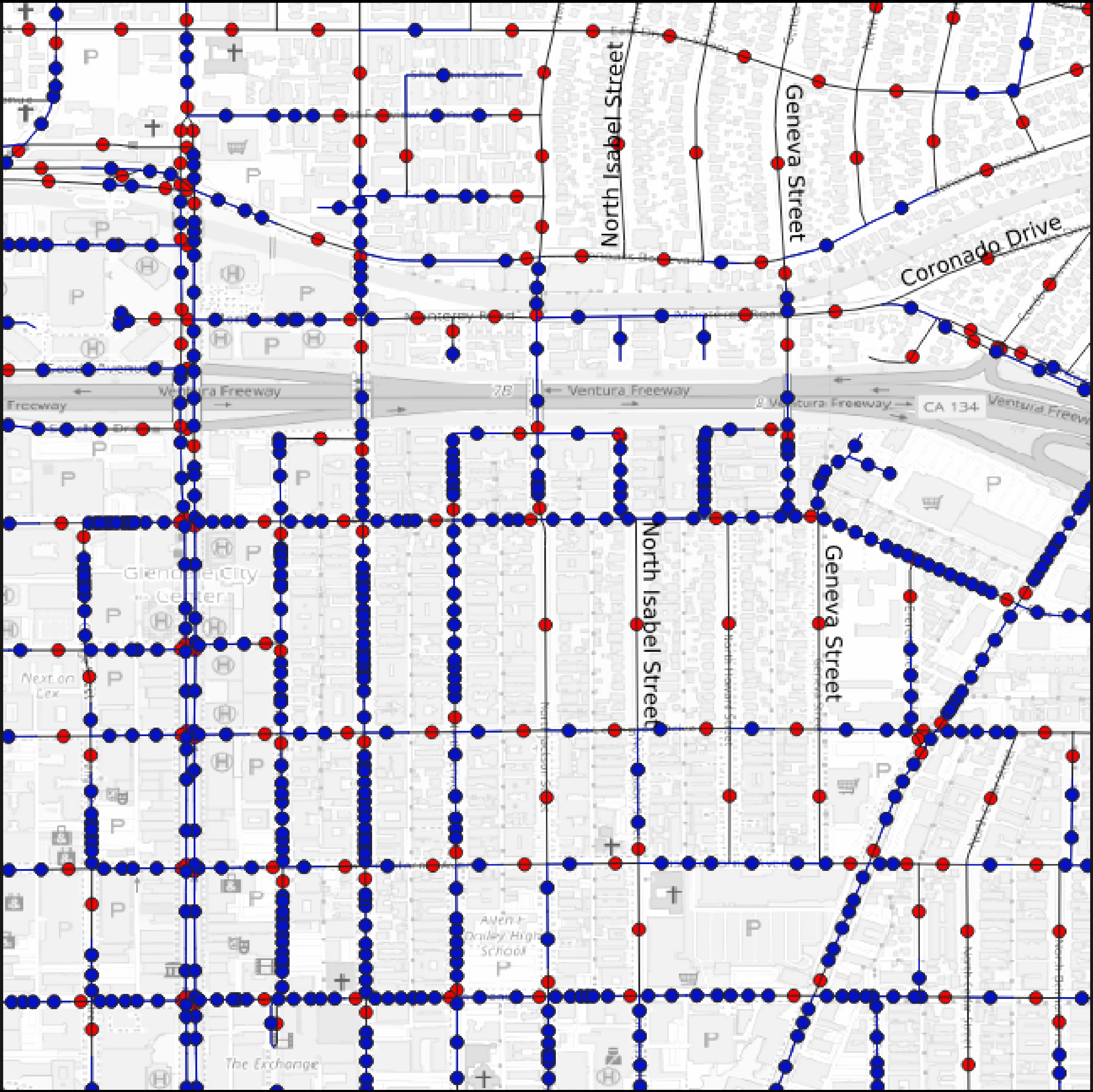

North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Goode Avenue
Ventura Freeway
Freeway
CA 134
Ventura Freew
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange


North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
Freeway
CA 134
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange

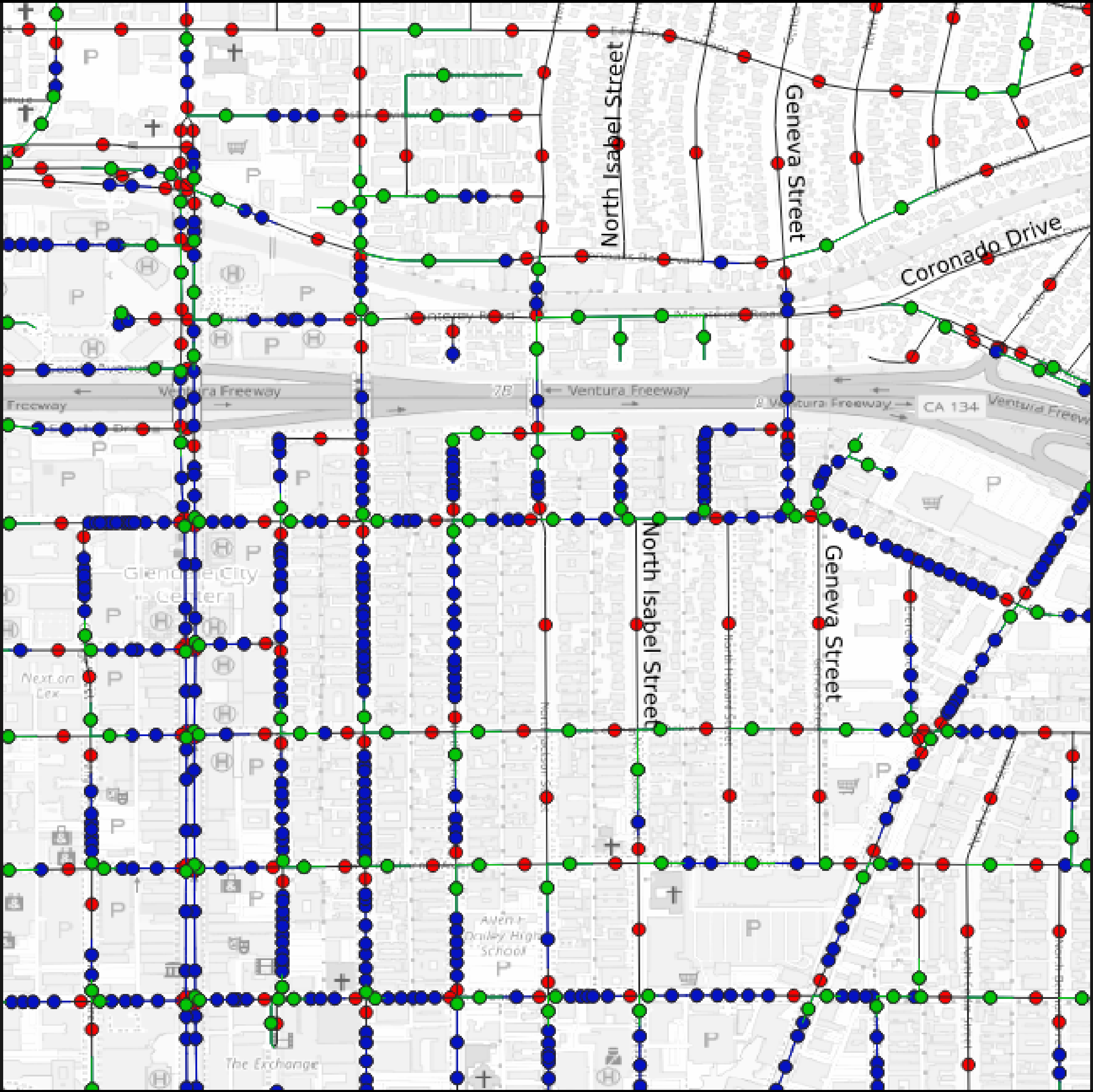
NARROW QUERY DOWN

- Possible starts is too broad
- For actual starts, source node has count > 1

```
starts as (
   select ps.*
   from possible_starts ps
   where ps.scount > 1
   )
```



North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
7B
CA 134
Freeway
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange


North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
Freeway
78
CA 134
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange

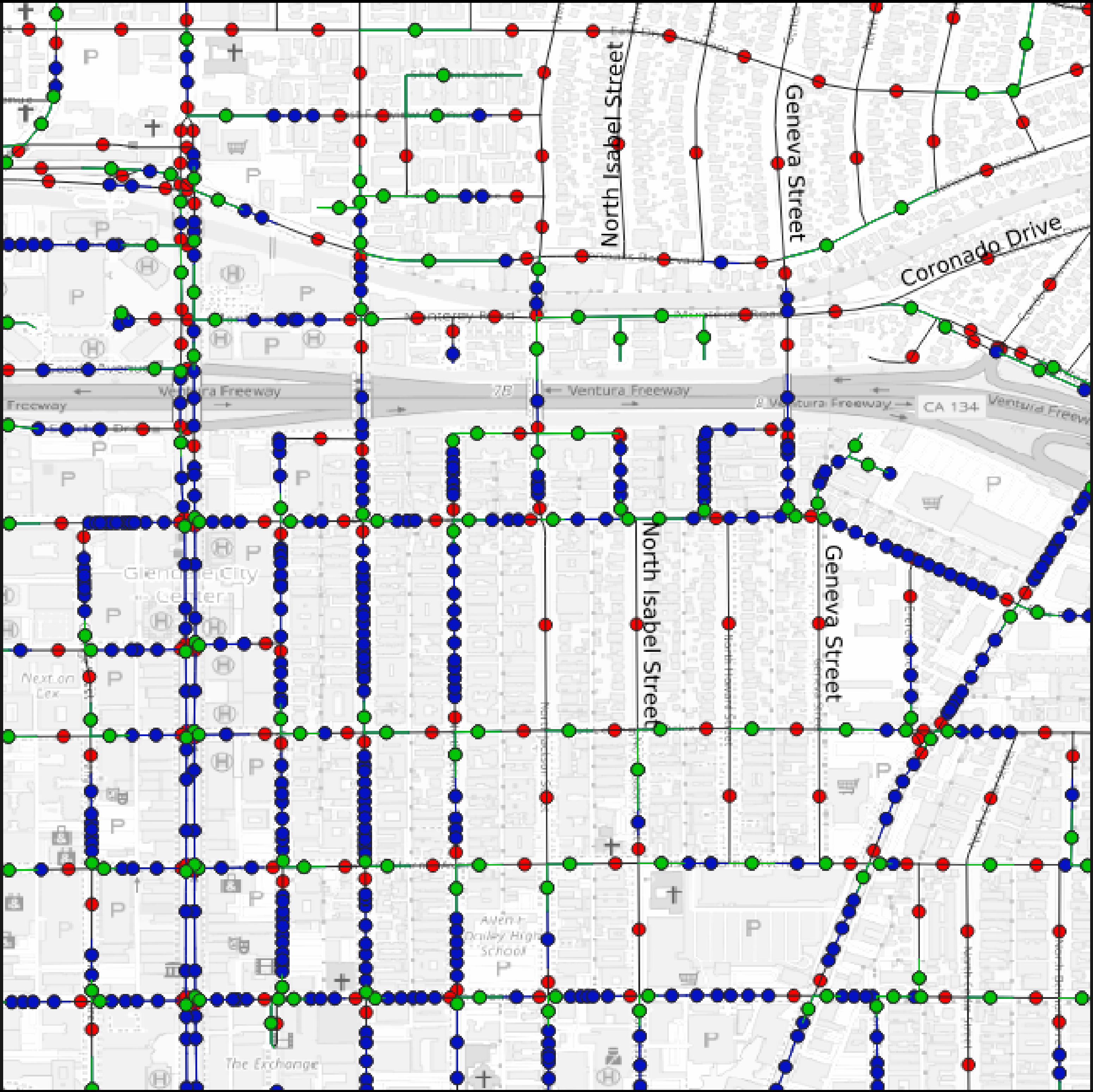
THIS IS TOO NARROW

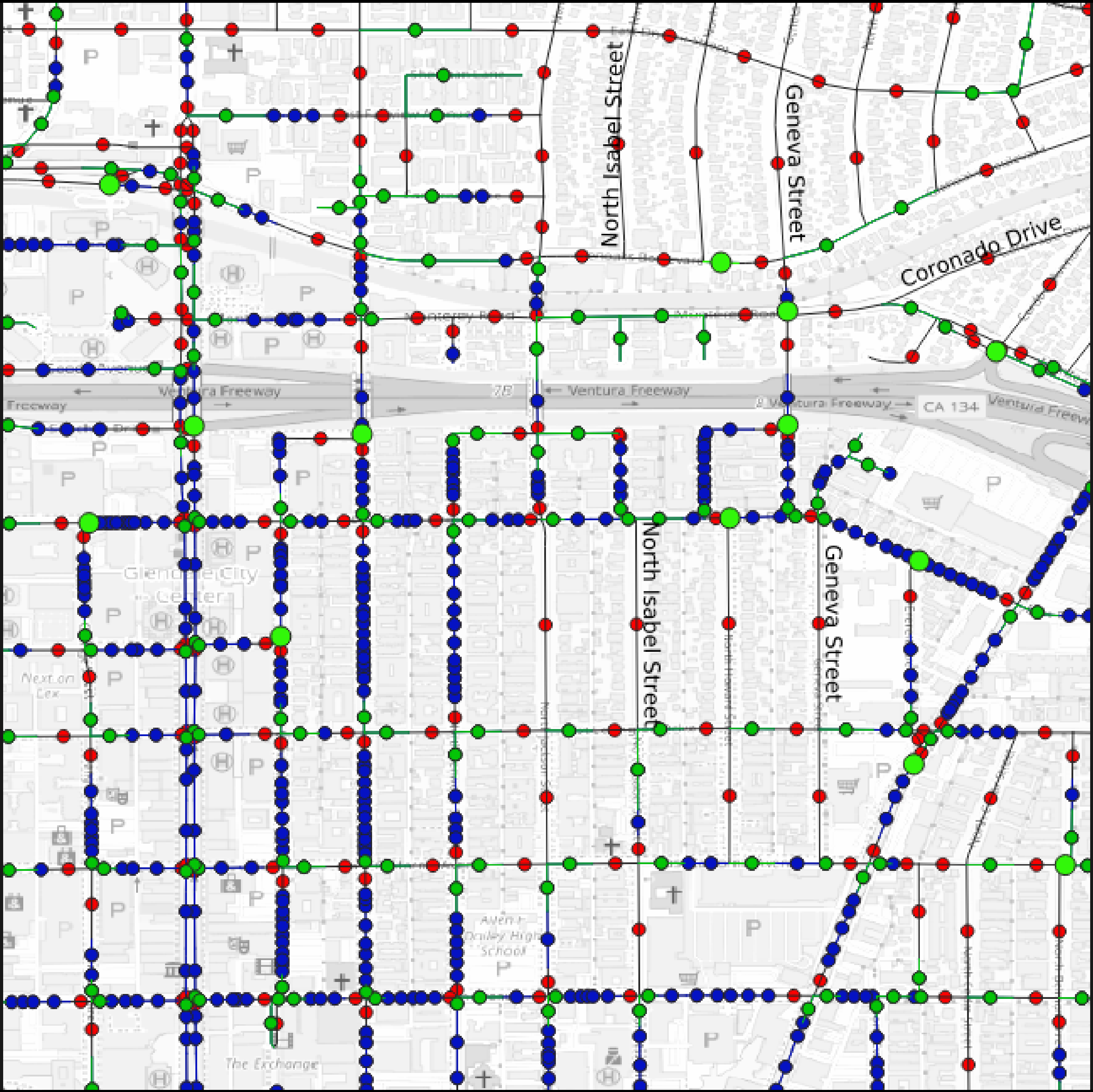
- Some special cases need to be handled
- All of these were worked out one by one
- There are more
- Other cities may have different quirks

A SOURCE IS A TARGET

- Some sources are also targets
- Flow direction is not uniform

```
union
select ps.*
from possible_starts ps
join targets t on (ps.source=t.target)
where ps.scount=1 and  t.count>1
-- more than one link target == ps.source
```

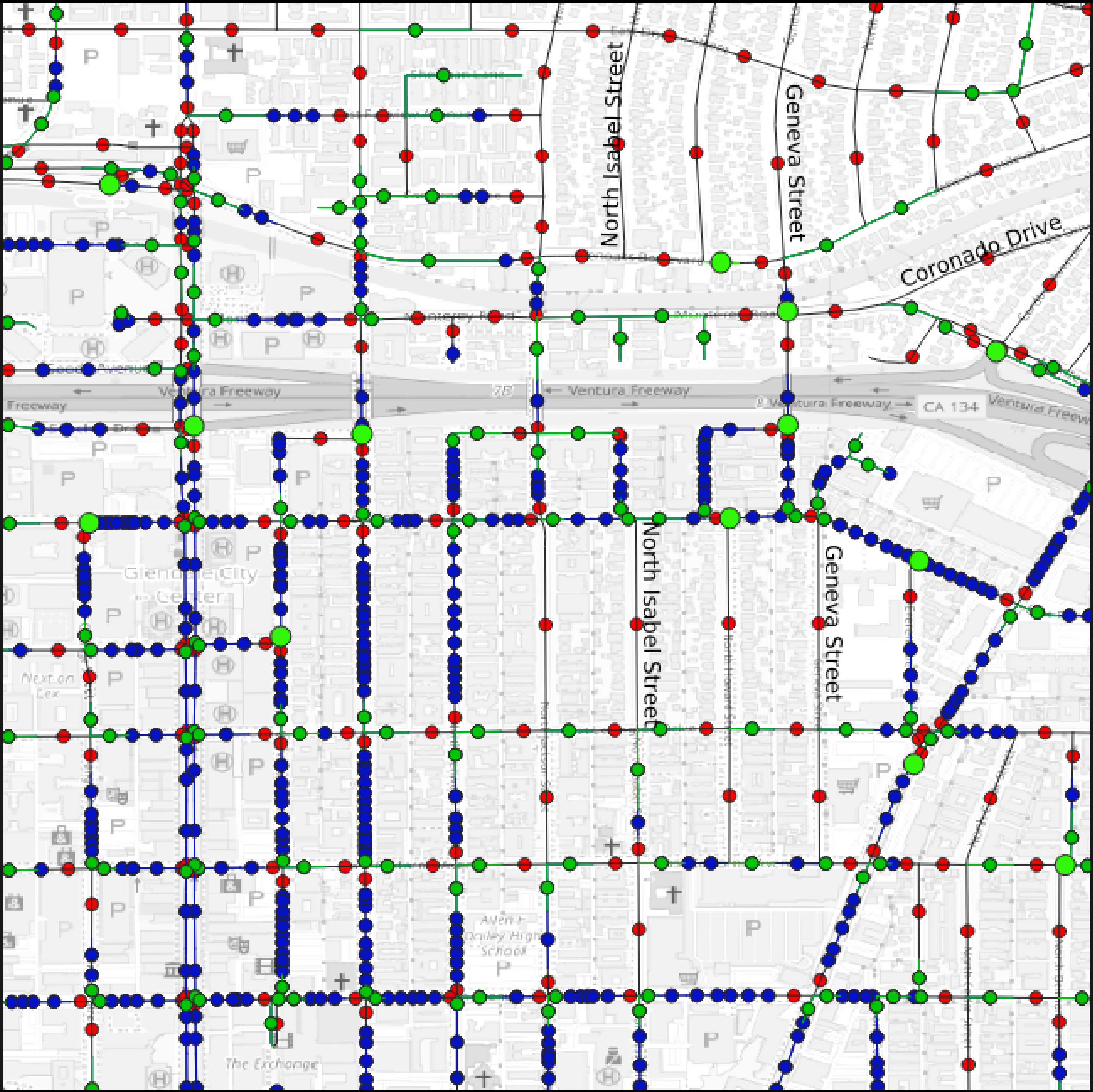

North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
Freeway
Ventura Freeway
CA 134
Glendale City Center
North Isabel Street
Geneva Street
Next on Cex
Allen F. Daily High School
The Exchange

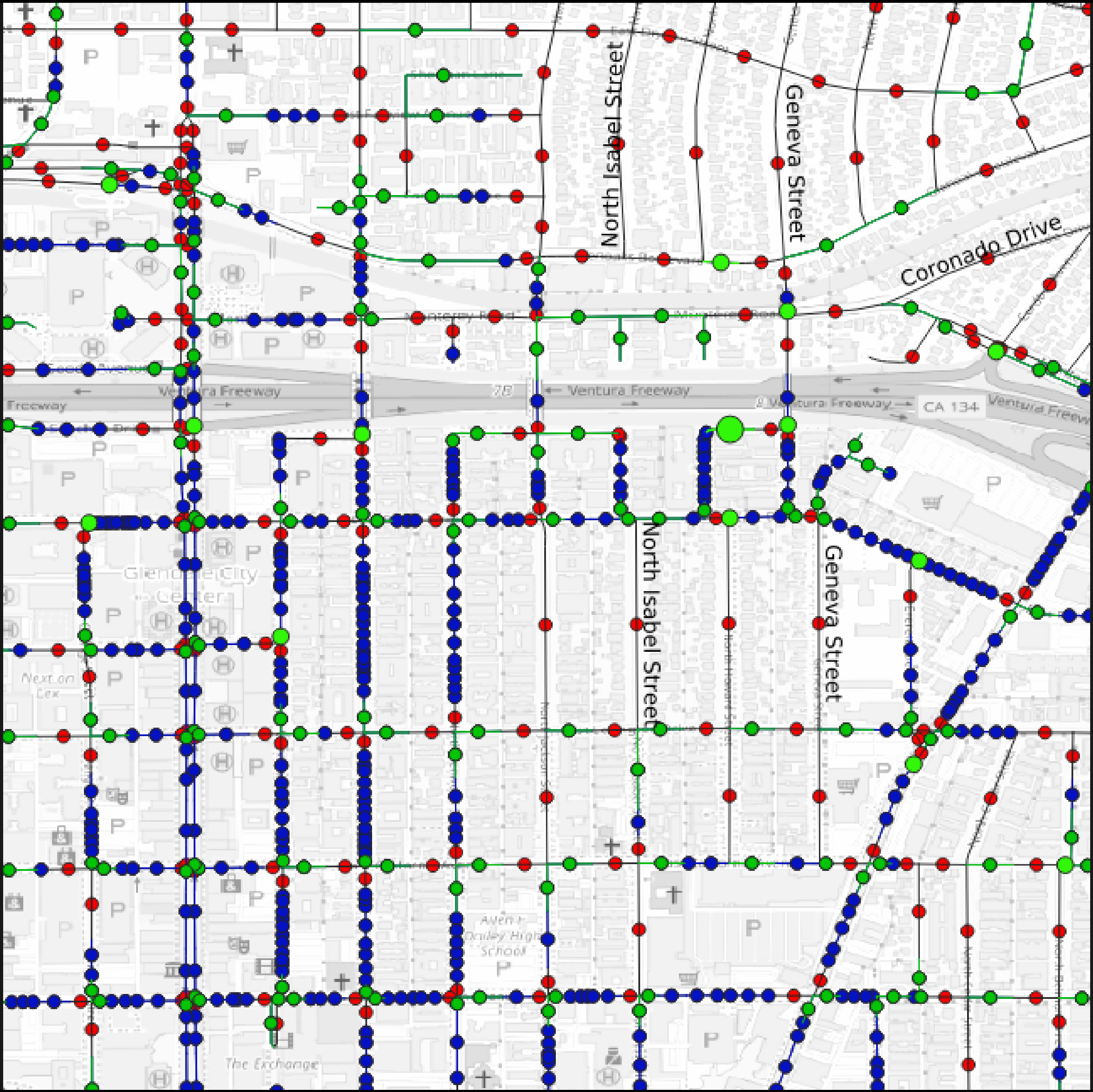

North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
78
CA 134
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange

NAMES CHANGE, ONE PATH

Look at possible interiors to identify a name change

```
union
select ps.*
from possible_starts ps
join possible_interiors pi on (ps.source=pi.target)
where ps.name != pi.name and ps.scount=1 and pi.tcount=1
-- name change of road
```

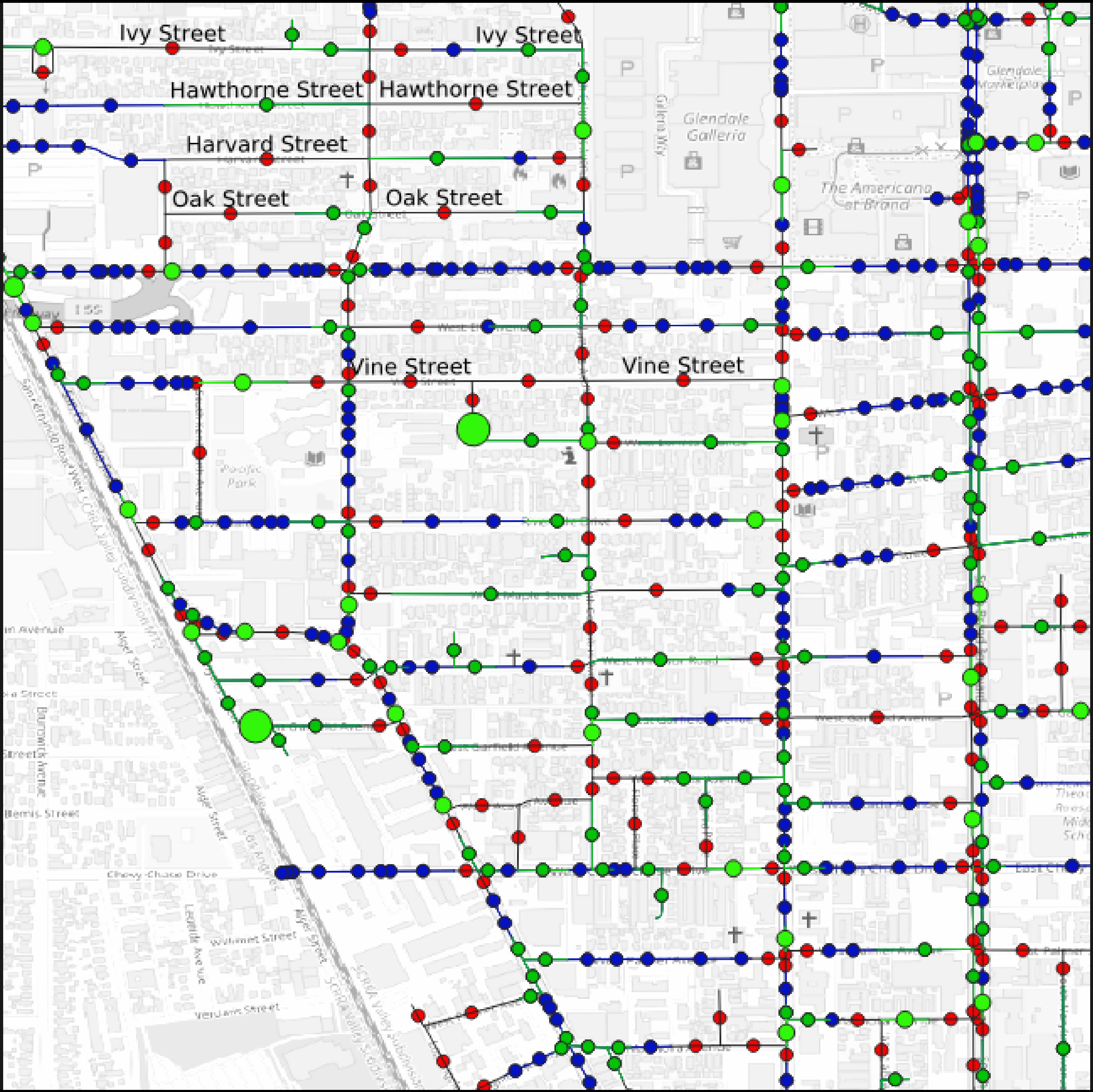

North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
Freeway
7B
CA 134
Glendale City Center
North Isabel Street
Geneva Street
Next on Cex
Allen F. Daily High School
The Exchange


North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
Freeway
7B
CA 134
Glendale City Center
Next on Cex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange

SINGLETONS

Like name change, but not in `possible_interiors` set

```
union
select ps.*
from possible_starts ps
join possible_starts psi on (ps.source = psi.target
                             and ps.name != psi.name)
-- singletons
```


Ivy Street
Ivy Street
Hawthorne Street
Hawthorne Street
Harvard Street
Oak Street
Oak Street
Vine Street
Vine Street
Glendale Galleria
Galleria Way
The Americana at Brand
Glendale Marketplace
Pacific Park
Chevy Chase Drive
Bemis Street
Verdant Street
Wilmer Street
Alger Street
Burchett Avenue
East Chevy Chase Drive

(DATA IS ALWAYS MESSIER THAN ONE WOULD EXPECT)

THE FULL STARTS QUERY

```
starts as (
   select ps.*
   from possible_starts ps
   where ps.scount >1
  union    -- more than one link target == ps.source
   select ps.*
   from possible_starts ps
   join targets t on (ps.source=t.target)
   where ps.scount =1 and  t.count>1
  union    -- name change of road
   select ps.*
   from possible_starts ps
   join possible_interiors pi on (ps.source=pi.target)
   where ps.name != pi.name and ps.scount=1 and pi.tcount=1
  union    -- singletons
```

ENDS QUERY IS SIMILAR

Nothing new, as starts and ends are basically the same

NOW THE RECURSIVE BIT

- recursive calls are broken into two steps
- the first is an initializing step
- the second is the recursive part
- the recursive part is a union with the initializing step
- the recursion needs to have a well-defined stop

INITIALIZATION STEP

```
search_graph(gid, length_m, name, source, target, depth,
             path, segments, cycle, ...) as (

   select g.gid, g.length_m, g.name, g.source, g.target,
   1 as depth,
   array[g.gid] as path,
   st_asewkt(g.the_geom) as segments,
   false as cycle,
   ... -- other stuff
   from ends g
   -- initialize with ends, connect interiors to source
```

INITIALIZATION NOTES

- `gid` is unique identifier for each segment
- `path` is an array of `gid`’s
- Start the recursion from the end
- Push new `gid`’s to the beginning of the array

WHY ST_ASEWKT?

```
st_asewkt(g.the_geom) as segments
```

- Not free to convert geom to text representation
- But union of geoms is pickier
- By combining geoms as text, can preserve their type of LineString

RECURSIVE PART

```
union all
 select g.gid, g.length_m + sg.length_m,
 sg.name, g.source, sg.target,
 sg.depth+1 as depth,
 g.gid || sg.path as path,
 st_asewkt( st_makeline( g.the_geom, sg.segments )),
 g.gid = ANY(sg.path) as cycle,
 ... -- other stuff
 from interiors g -- recurse on interiors
 join search_graph sg on
   (g.target=sg.source -- interior target -> chain source
    and g.name=sg.name)-- but same street name too please
 where sg.depth < 100 and not sg.cycle -- stop guards
```

EXPLANATION

- Start segment grown at `ends`
- Grow segments from the `interiors`
- Creates a list of increasingly long segments

POSTGIS NOTES

```
st_asewkt( st_makeline( g.the_geom, sg.segments ))
```

- `st_makeline()` used to avoid array type error
- Makes a new line for each segment
- Prepends new line bit to growing line
- Whole result is dumped as well known text for next recursive loop

ALTERNATE VERSION

```
ARRAY[g.the_geom] as segments
…
array_prepend(g.the_geom, sg.segments)
            ::geometry(LineString,4326)[],
```

- Cast fixes recursive error re: mismatched array types
- `EXPLAIN ANALYZE` says they’re the same speed:
    - `st_asewkt` 117s vs ARRAY 119s

EXAMPLE RESULTS

```
WITH RECURSIVE …
select gid,name,source,target,depth
from search_graph order by depth desc,name;
 gid  |         name           | source | target | depth
------+-----------------------+--------+--------+-------
 6344 | North Louise Street    |   5686 |    234 |    19
 6179 | North Louise Street    |   5685 |    234 |    18
 5311 | Emerald Isle Drive     |   4635 |    149 |    17
 6326 | North Louise Street    |   5520 |    234 |    17
 5309 | Emerald Isle Drive     |   4650 |    149 |    16
 5280 | Flintridge Drive       |   4620 |    147 |    16
 6327 | North Louise Street    |   5667 |    234 |    16
 5310 | Emerald Isle Drive     |   4648 |    149 |    15
```

NEED TO PICK THE LONGEST

- The longest segment has depth of 19
- Need to choose that one, not the shorter ones
- Next part of WITH RECURSIVE statement picks off longest segments

LONGEST GROUPS

```
gid_paths as (select unnest(sg.path) as node,depth
   from search_graph sg ),
gid_max_depth as (
   select node,max(depth) as depth
   from gid_paths group by node ),

distinct_paths as (
  select distinct path
  from search_graph sg
  join gid_max_depth gm
       on (gm.depth=sg.depth and
           gm.node in (select unnest(sg.path)))
  )
```

MAKE ONE RECORD

- In one step:
    - Pick longest sequence using `distinct_paths`
    - Merge `starts` to add starting node
    - Convert text geom back to binary geom

MERGED SEGMENTS

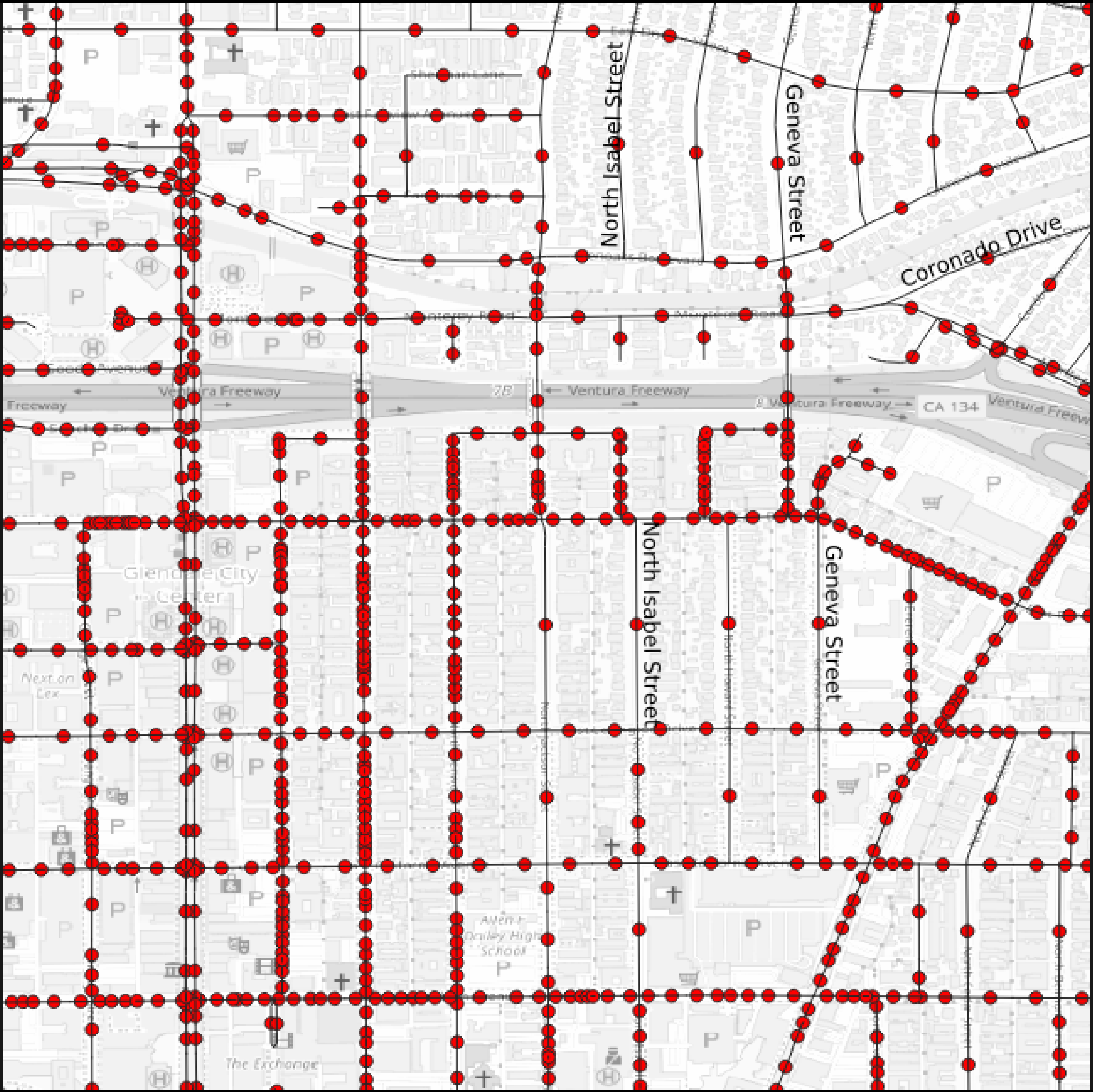
```
segments as (
   select c.name, g.source, c.target, c.depth+1 as depth,
   g.gid || c.path as path,
   ST_SimplifyPreserveTopology(
    ST_GeomFromEWKT(st_asewkt(st_makeline(g.the_geom,
                                          c.segments))),
    0.0000001) as the_geom,  … other_columns …
   from search_graph c
   join distinct_paths dp on (c.path=dp.path)
   join starts g -- add start nodes to chain
        on (g.target=c.source  --start.target == source
            and g.name=c.name) -- same name please
   )
```

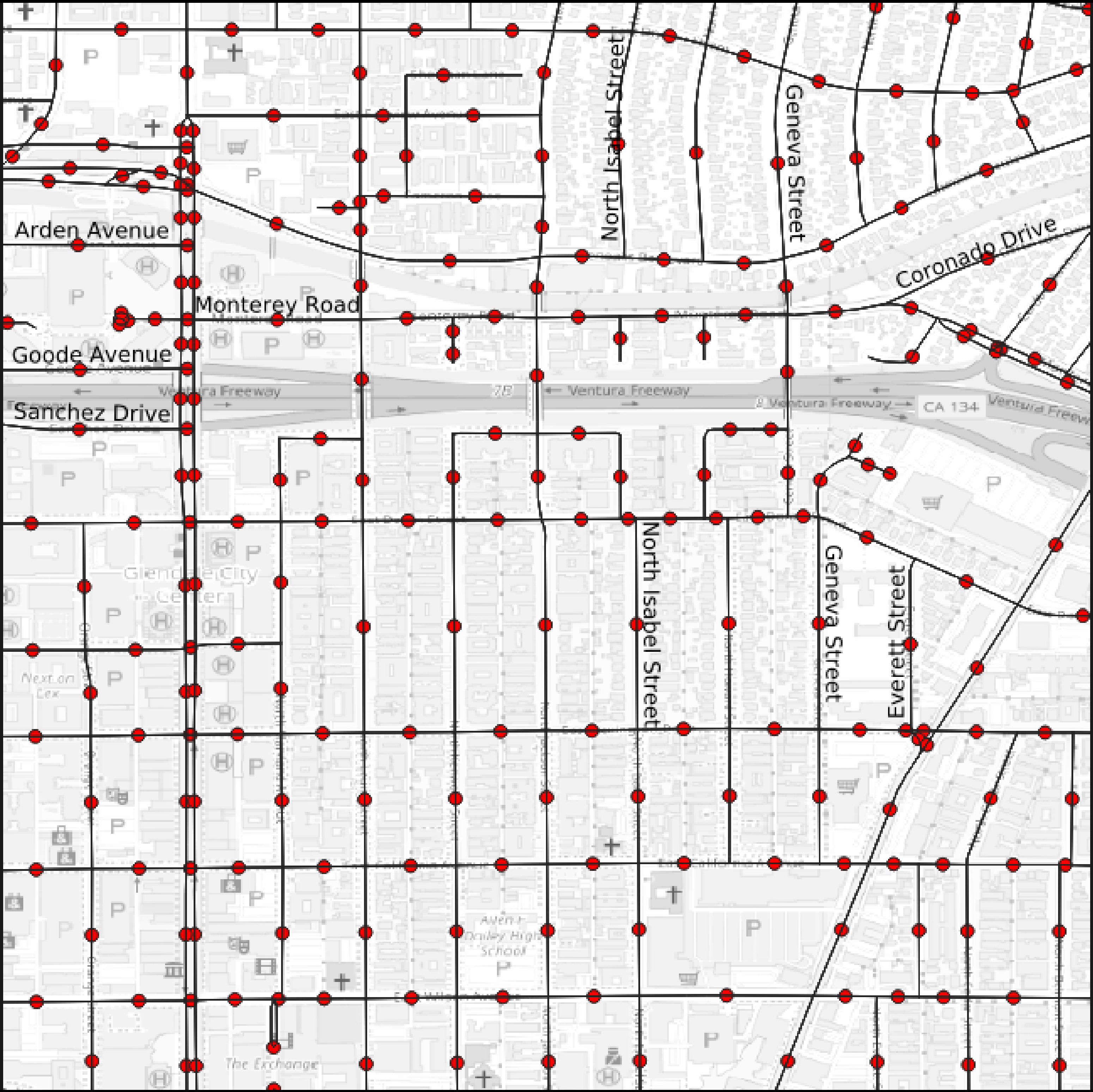
BOOK-KEEPING, AND FINISH UP

- The remaining SQL just tidies up
- Make a new table
  - Start with the old table
  - Drop the components of merged segments
  - Add the new, longer merged segments

```
grouped as (
 select * from keep_ways
union
 select * from new_ways
)
insert into new_glendale_ways ( … )
select … from grouped;
```

FINAL OUTPUT OF SEGMENT-JOINING WORK


North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Goode Avenue
Ventura Freeway
Freeway
CA 134
Ventura Freew
Glendale City Center
Next on Lex
North Isabel Street
Geneva Street
Allen F. Daily High School
The Exchange


Arden Avenue
Monterey Road
Monterey Road
Goode Avenue
Sanchez Drive
North Isabel Street
Geneva Street
Coronado Drive
Monterey Road
Ventura Freeway
7B
Ventura Freeway
Ventura Freeway
CA 134
Ventura Freeway
Glendale City Center
North Isabel Street
Geneva Street
Everett Street
Next on Lex
Allen F. Daily High School
The Exchange

SOME NOTES

- Not all segments are fixed properly
- Reduced number of segments by 40%
  - for Glendale, California
  - went from 7653 links to 4597 links
- Huge impact on problem size
- Absolutely worth the effort to figure this out

CONVERTING STREETS TO CURBS

ONE-WAY AND TWO-WAY STREETS

- OSM data is pretty good about identifying one-way streets
- pgRouting can analyze OSM data and establish forward and backward traversal costs
- But using two-way streets is buggy

CONVERT ALL STREETS TO CURBS

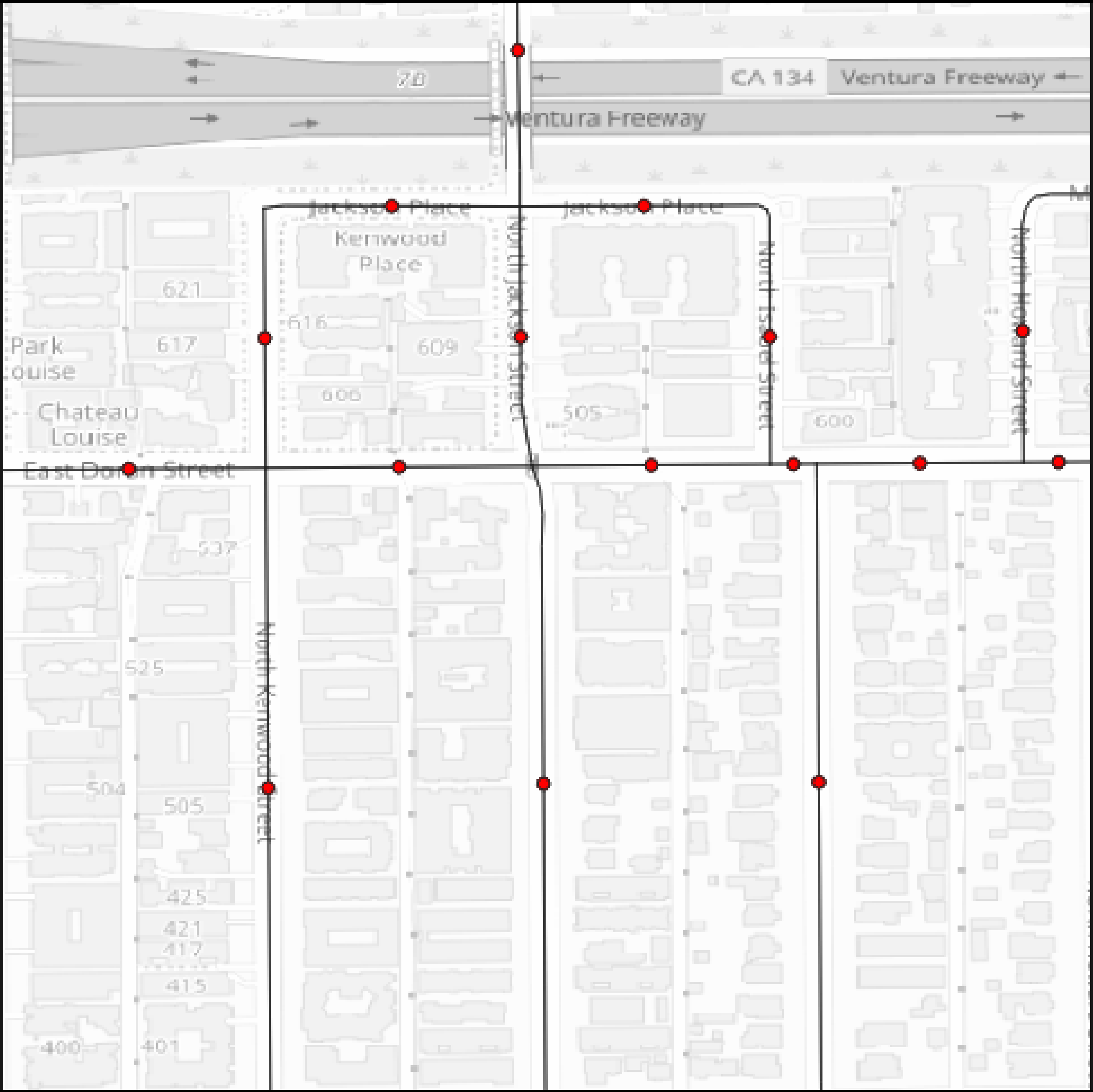
- Curbs are all one-way
- On two-way streets, curb movements are in opposite directions
- On one-way streets, curb movements are in same direction
- Easier to reason about moving from curb to curb

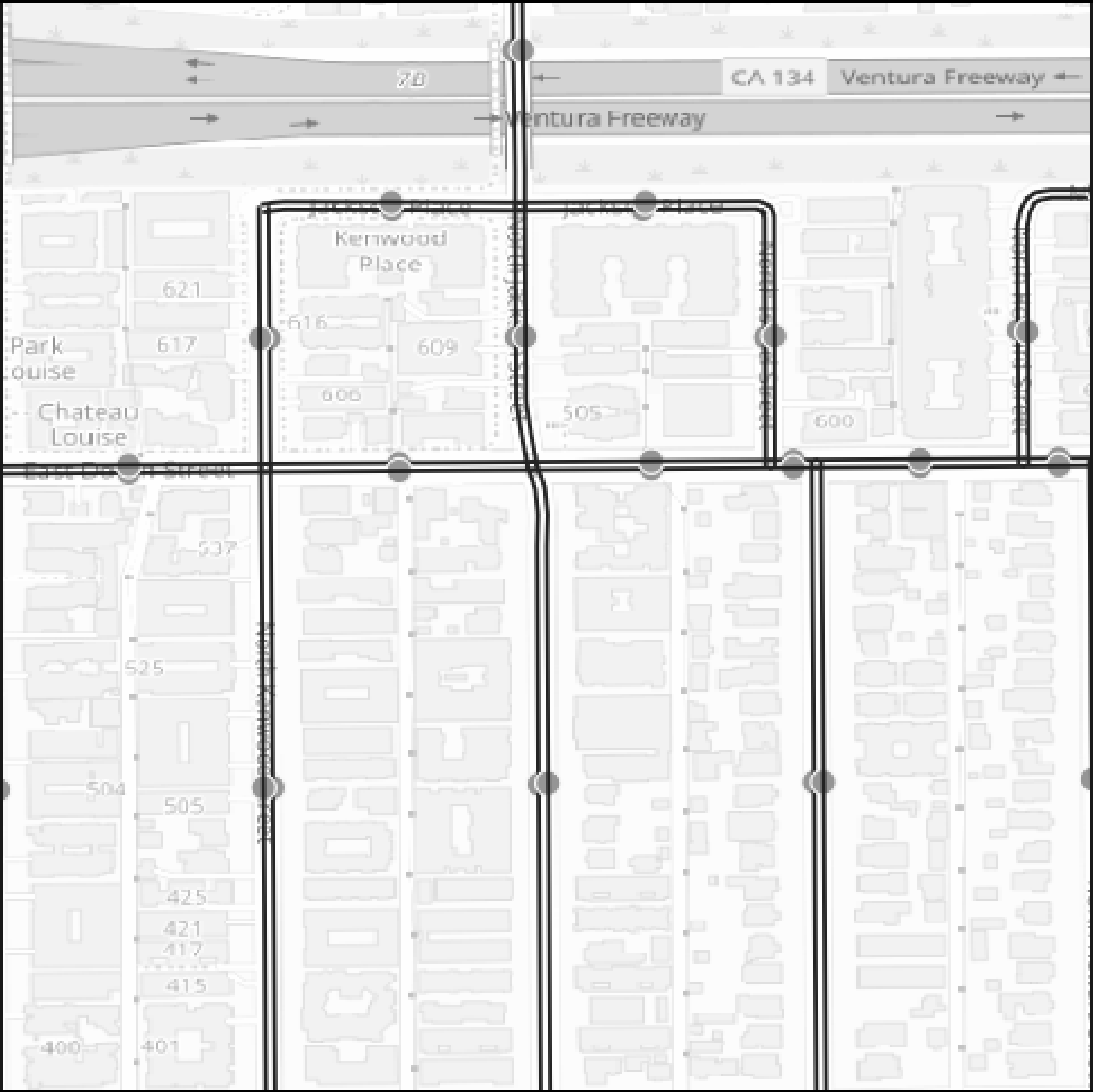
BIG SQL STATEMENT I’M GOING TO TALK ABOUT

```
drop sequence if exists curbgraph_v2_serial;
create sequence curbgraph_v2_serial;

drop table if exists curbs_v2_graph cascade;

with
tform as (
    select id, st_transform(the_geom,32611) as
        geom,reverse_cost
    from new_glendale_ways
    ),
rhs as (
    select ST_Reverse(ST_Transform (
             ST_OffsetCurve(
                geom
```


CA 134
Ventura Freeway
78
Ventura Freeway
Jackson Place
Jackson Place
Kenwood Place
North Jackson Street
North Isabel Street
North Howard Street
621
617
616
609
606
505
600
Park Louise
Chateau Louise
East Doran Street
537
525
North Kenwood Street
504
505
425
421
417
415
400
401

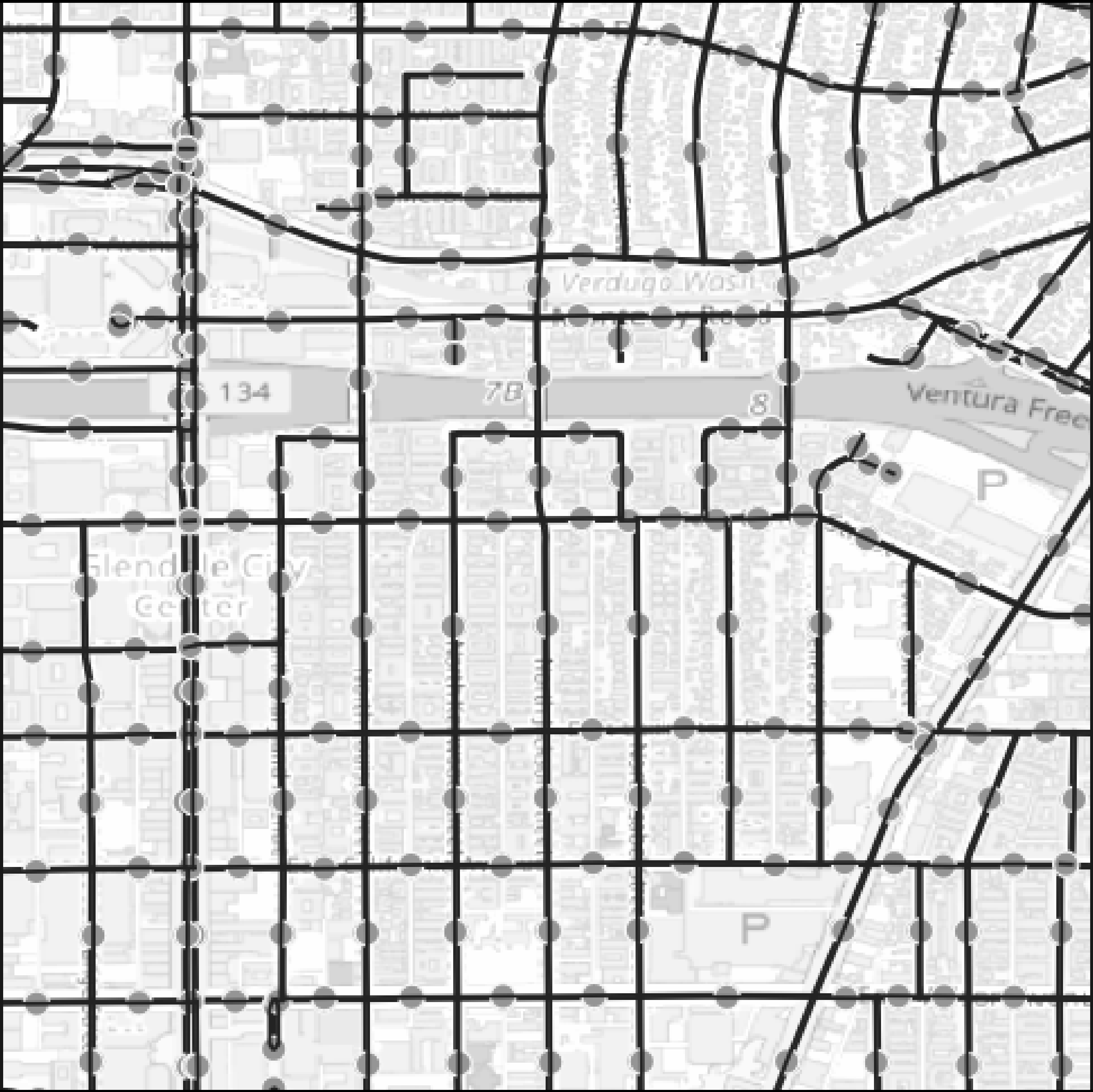

78
CA 134
Ventura Freeway
Ventura Freeway
Jackson Place
Jackson Place
Kenwood Place
621
617
616
609
606
505
600
Park Louise
Chateau Louise
East Doran Street
537
525
504
505
425
421
417
415
400
401

MAKING A LINE GRAPH

WHAT IS A LINE GRAPH

- The usual navigation map:
  - intersections as nodes
  - streets as links between nodes
- Edge covering needs to reach every street
- Convert original graph to line graph
  - streets are nodes
  - links represent legal movements between streets

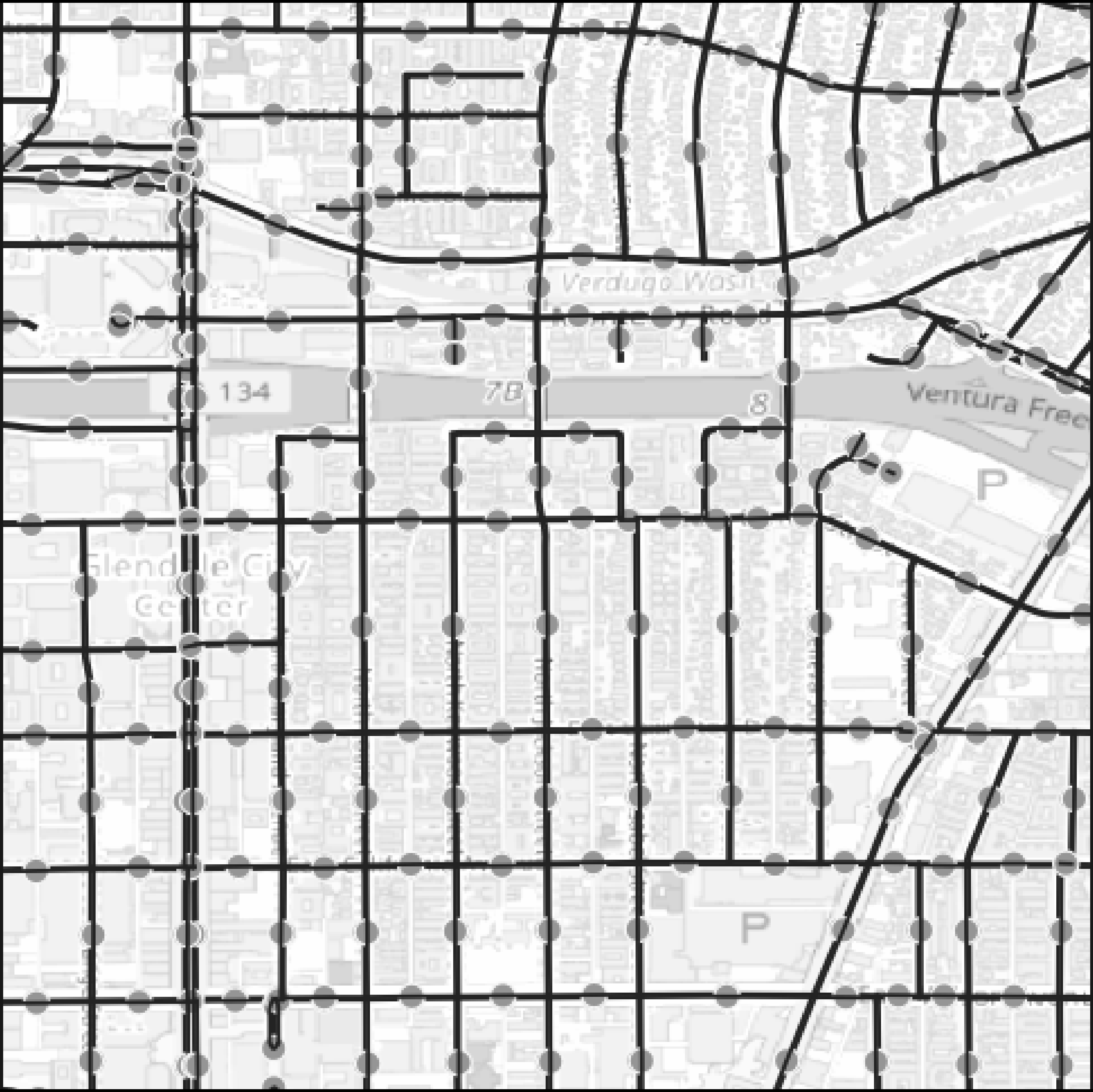
THE CURBS

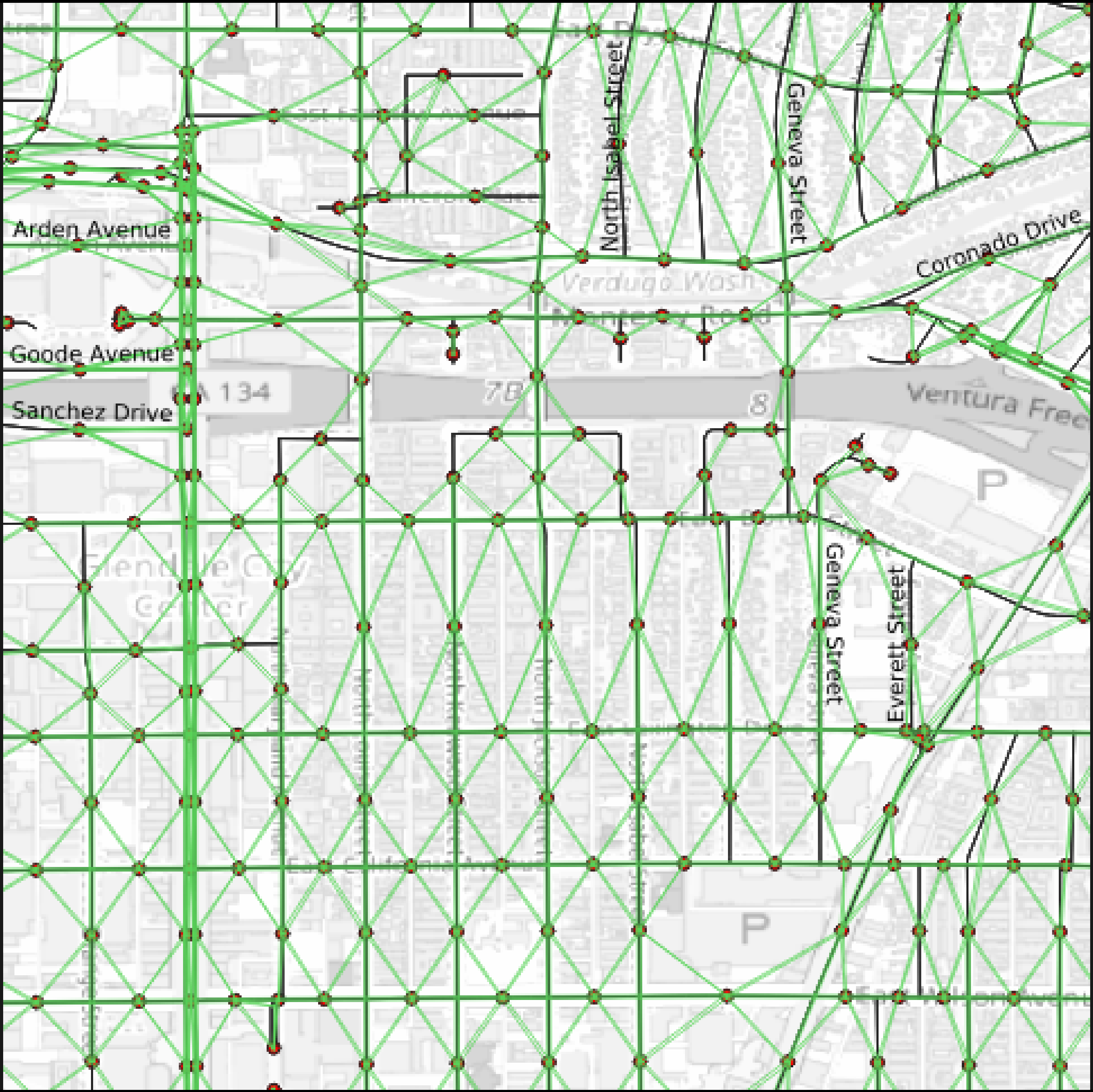

Verdugo Wash
Monterey Road
134
7B
8
Ventura Free
P
Glendale City Center
P

USE PGROUTING TO MAKE LINEGRAPH

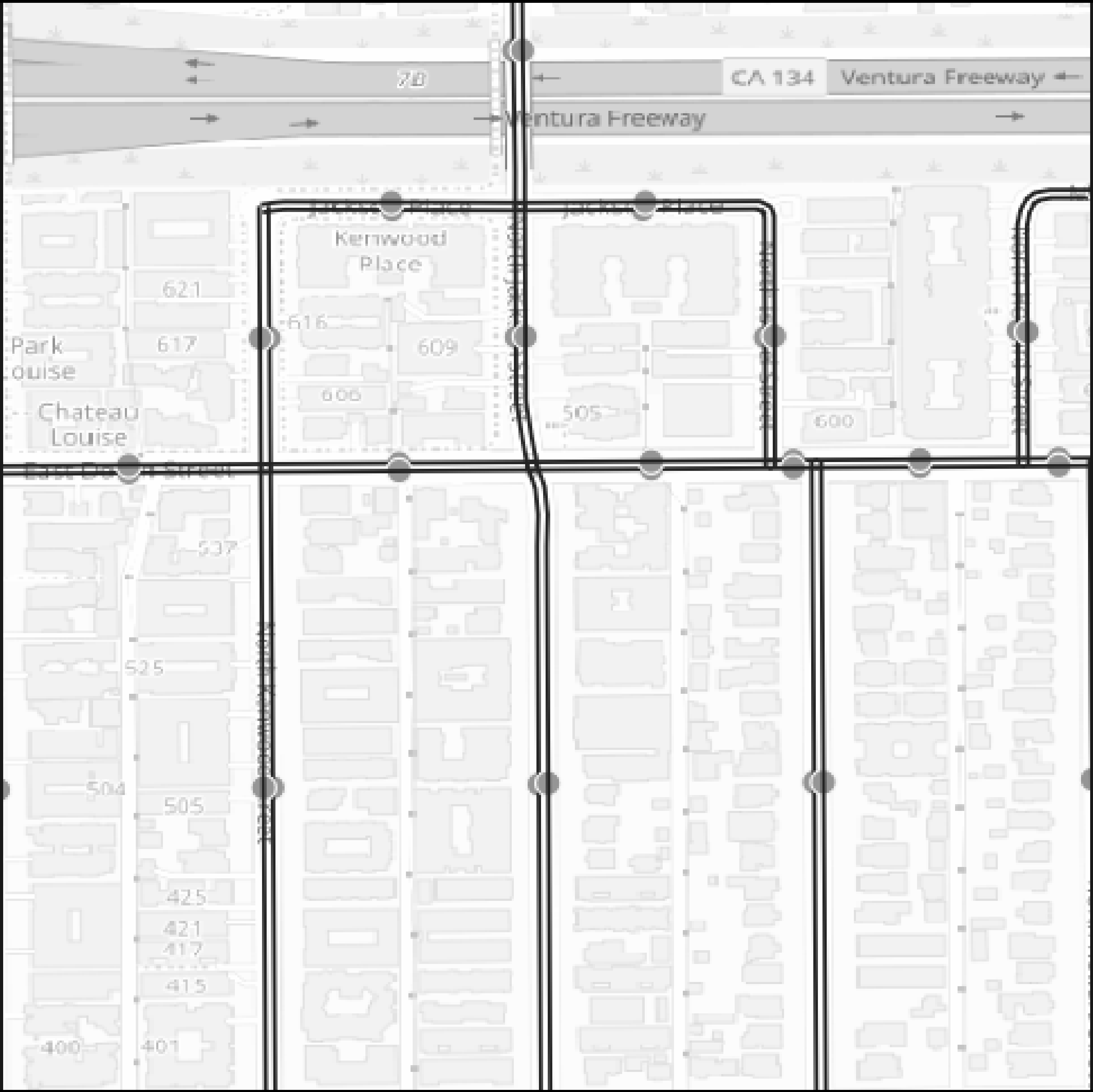
- With curb graph in hand, this is a very easy task

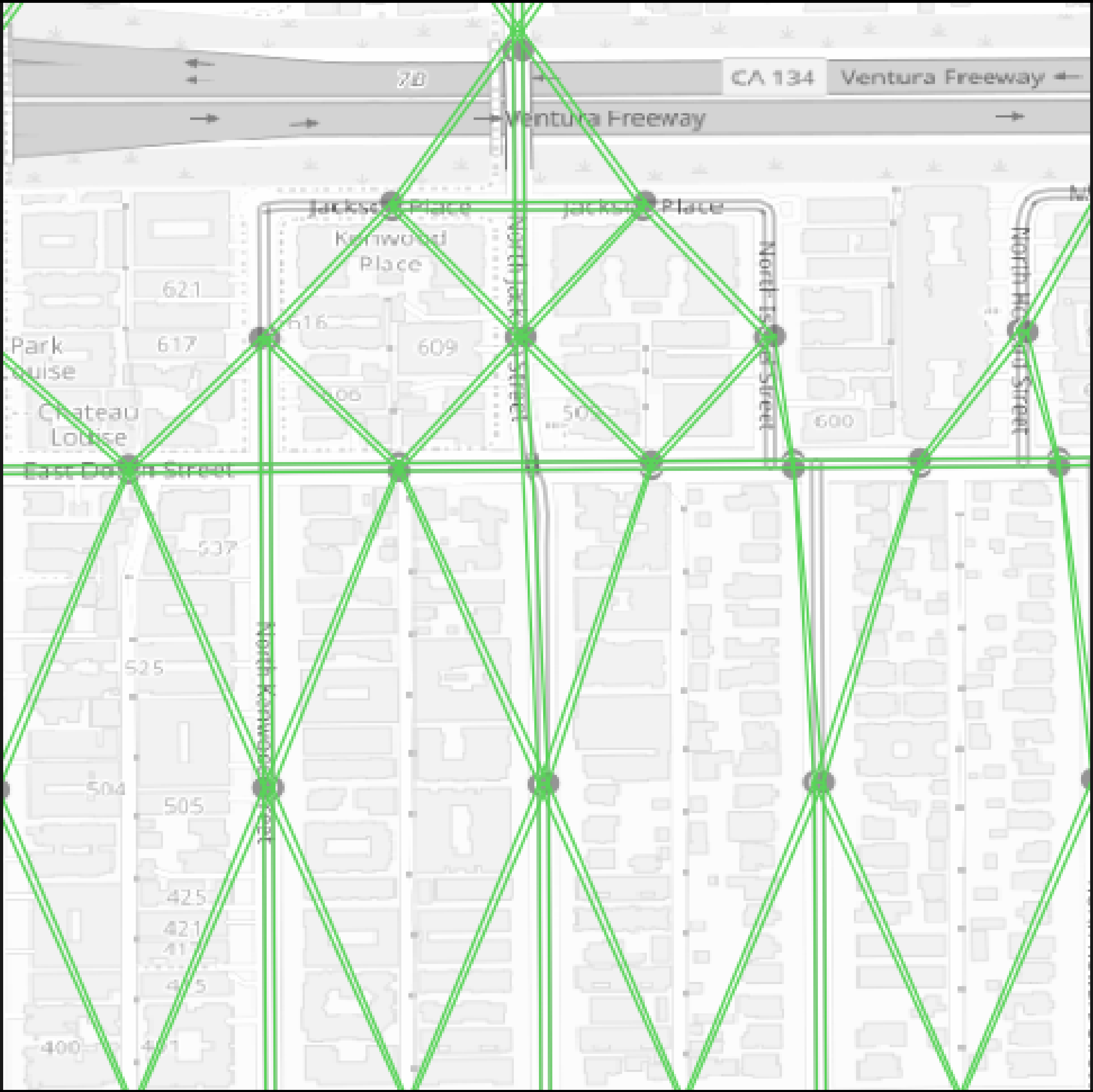
```
drop table if exists curbs_v2_linegraph;
SELECT * into curbs_v2_linegraph FROM pgr_lineGraph(
    'SELECT curbid as id, source, target, cost_s as cost,
        reverse_cost_s as reverse_cost FROM curbs_v2_graph'
);
```


Verdugo Wash
Monterey Road
134
7B
8
Ventura Freeway
P
Glendale City Center
P


Arden Avenue
Goode Avenue
Sanchez Drive
CA 134
7B
8
Ventura Free
Coronado Drive
Verdugo Wash
Monterey Road
North Isabel Street
Geneva Street
Glendale City Center
Geneva Street
Everett Street
P
P

ZOOMING IN ON AN AREA


78
CA 134
Ventura Freeway
Ventura Freeway
Kenwood Place
621
617
616
609
606
505
600
Park
ouise
Chateau Louise
537
525
504
505
425
421
417
415
400
401


78
CA 134
Ventura Freeway
Ventura Freeway
Jackson Place
Jackson Place
Kenwood Place
621
617
616
609
606
507
600
Park Louise
Chateau Louise
North Jackson Street
North Isabel Street
North Rossmore Street
East Doran Street
537
525
North Kenwood Street
504
505
425
421
417
405
400
401

ALL TO ALL DISTANCE MATRIX

THE NEED FOR DISTANCES

- Solver must reach each node (street)
- To do that efficiently, it must know distance between streets
- Goal of solver is to minimize overall travel distance
- Therefore must have all to all travel matrix (or close to it)

NOT HARD, JUST IRRITATING

- pgRouting has an excellent function `pgr_dijkstraCostMatrix()`
- creates a matrix of distances
- but 9,193 nodes means table with 84,511,249 entries
- I run out of RAM

UGLY HACKS

- step through the curb table 3,000 at a time
- grab random bunches of under-represented origins
- rinse and repeat

FIRST, INSERT ALL IMMEDIATE NEIGHBORS

```
truncate new_curbs_linegraph_matrix;
with onesteps as (
  select source as start_vid,
      target as end_vid,
      target_length_m as agg_cost
  from new_curbs_v2_linegraph a
  )
insert into new_curbs_linegraph_matrix
   select * from onesteps
   on conflict do nothing;
-- INSERT 0 28067
```

NEXT, FUNCTION TO STEP THROUGH DATA METHODICALLY

FLESHOUT_2000...

```
create or replace function
        fleshout_2000_curb_linegraph_matrix(starting int)
returns integer as
$BODY$
DECLARE
   i text;
   subsel text := 'SELECT id, source, target, target_length_m
        as cost, reverse_cost FROM new_curbs_v2_linegraph';
   insert_sql text := '';
   check_sql text := '';
   get_one_sql text := '';
   test_sql text := '';
   startid int := 0;
BEGIN
   insert_sql := '
```

WHAT IT DOES

- Loops over data

```
FOR startid IN starting..7000 by 1000 LOOP
    RAISE NOTICE 'populate db starting with %', startid;
    EXECUTE  insert_sql using startid;
END LOOP;
```

- Can pass in `starting` point as function parameter
- Steps forward 1000 each iteration

SQL QUERY BITS

- Query will find 3,000 by 3,000 distance matrix
- (because 3000 is what works on my laptop)

```sql
select distinct source
from new_curbs_v2_linegraph nl
where source > $1
order by source
limit 3000
```

ANOTHER SIMILAR FUNCTION WITH RANDOM

```
with
 low_block (sid) as (
    select source
    from new_curbs_v2_linegraph nl
    where source <3300
    order by random()
    limit 1000
    ),
 mid_block (sid) as (
    select source
    from new_curbs_v2_linegraph nl
    where source >= 3300 and source <= 6600
    order by random()
    limit 1000
    )
```

OR FOCUS ON THE UNDER-REPRESENTED ONES

```
with
 sid_count (sid,cnt) as (
   select start_vid, count(*)
   from new_curbs_linegraph_matrix
   group by start_vid
   order by count
   ),
 lo_block (sid) as (
   select sid from sid_count
   limit 500
   ),
 hi_block (sid) as (
   select sid
   from sid_count
   where cnt > 9000
```

OR GET SMART ABOUT “UNDERREPRESENTED”

```
with
 sid_count (sid,cnt) as (
   select start_vid, count(*)
   from new_curbs_linegraph_matrix
   group by start_vid
   order by count
   ),
 pctl (hicount) as (
   SELECT percentile_cont(0.07) WITHIN GROUP (ORDER BY cnt)
        FROM sid_count
 ),
 lo_block (sid) as (
   select sid from sid_count
   limit 500
   )
```

THE TABLE IS CLOSE ENOUGH

- Each Origin should have 9123 destinations

```
with counts as (
   select start_vid,count(*) as cnt
   from new_curbs_linegraph_matrix group by start_vid)
select count(*),floor(cnt) from counts group by floor(cnt);
 count | floor
-------+-------
   201 |  9190
  2374 |  9191
  6607 |  9192
    11 |  9193
(4 rows)
```

SOLVE THE STREET SWEEPING PROBLEM

OR TOOLS TO THE RESCUE

- OR Tools is great
- But it isn’t PostgreSQL related
- So I’ll talk about it some other time

SOME BENCHMARKS

- My formulation takes about 20 minutes to generate an initial solution
- Can run for hours
- Difficult to get the “shape” of a solution right
- Difficult to visualize the output

SAVE THE GENERATED PATHS

- After solver finishes, generate a list of nodes “swept”
- For deadhead nodes, use pgRouting to find intermediate nodes
    - Deadhead meaning drive without sweeping over several streets to get to a street that needs sweeping
- Gather the list of all nodes each vehicle visits (sweep plus non-sweep)

PYTHON CODE TO SAVE LIST OF NODES TO DB

```python
def sequence_to_table(self,vsequence,table_name):
    sequence = 0
    insert_query_string = """insert into {}
        (veh,sweep,linkid,geom)
    select %s,%s,%s,c.curb_geom as the_geom
    from curbs_v2_graph c
    where c.curbid =%s"""
    insert_query =
        sql.SQL(insert_query_string).format(sql.Identifier(ta

    with self.conn.cursor() as cur:
        cur.execute(
            sql.SQL("drop table if exists
        {}").format(sql.Identifier(table_name)))
```

ASIDE

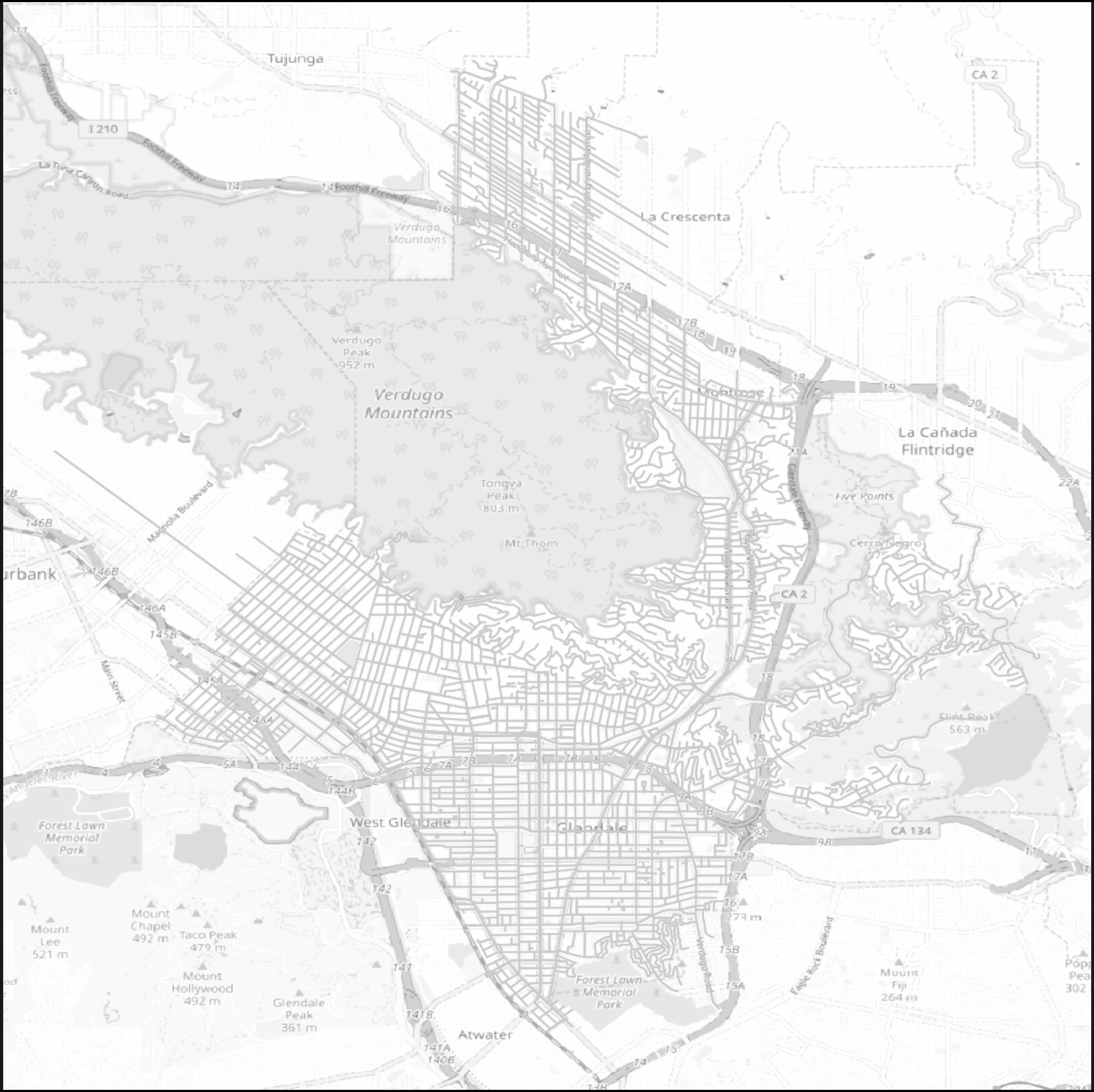
- Do not use Python string formatting to insert strings and variables into your generated SQL
- Doing so is strongly discouraged by psycopg
- Instead use sql.SQL, and pass parameters to execute

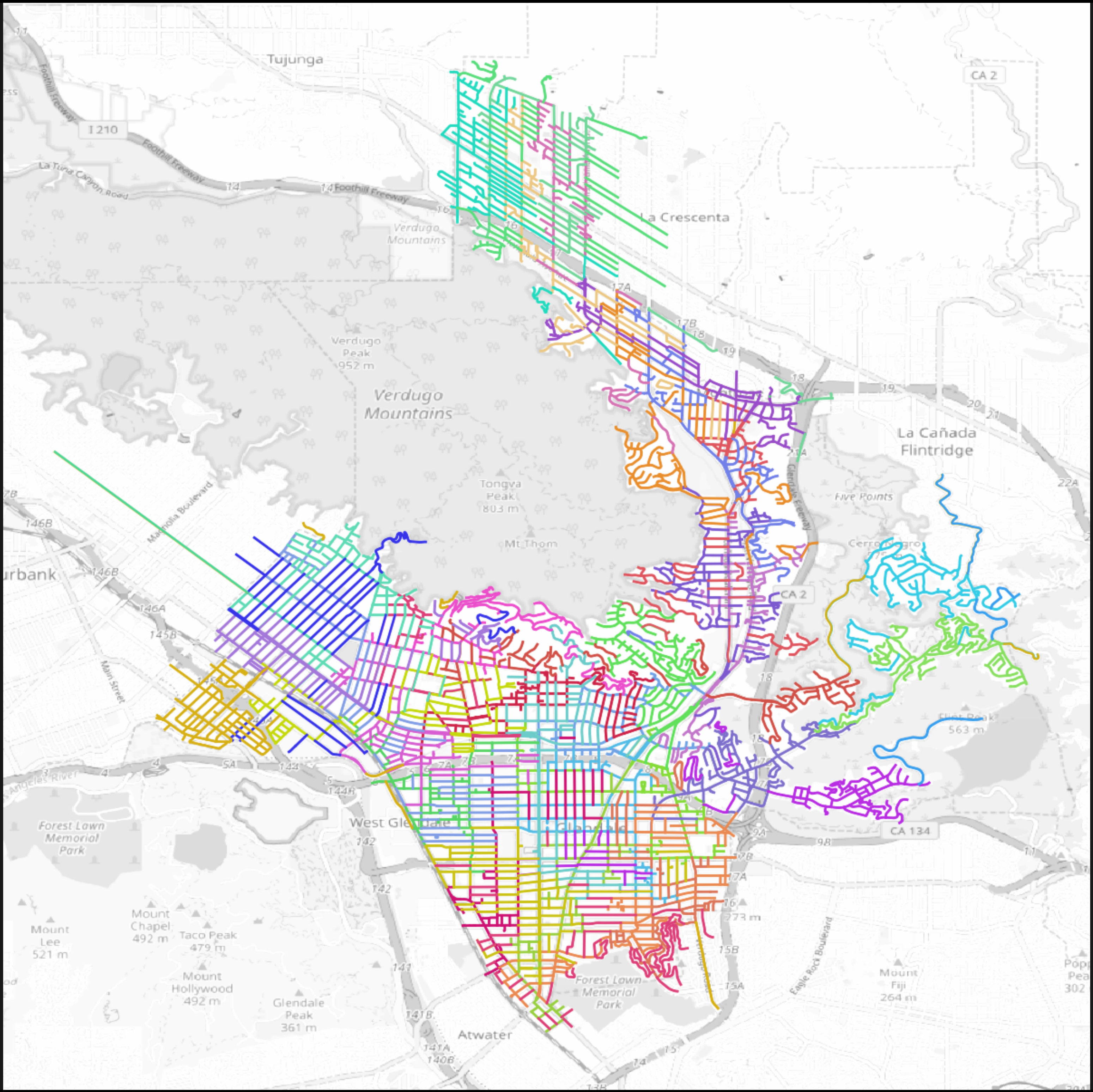
```
sql.SQL("drop table {}").format(sql.Identifier(table_name))
...
cur.execute(insert_query,(veh,sweep,linkid,linkid))
```

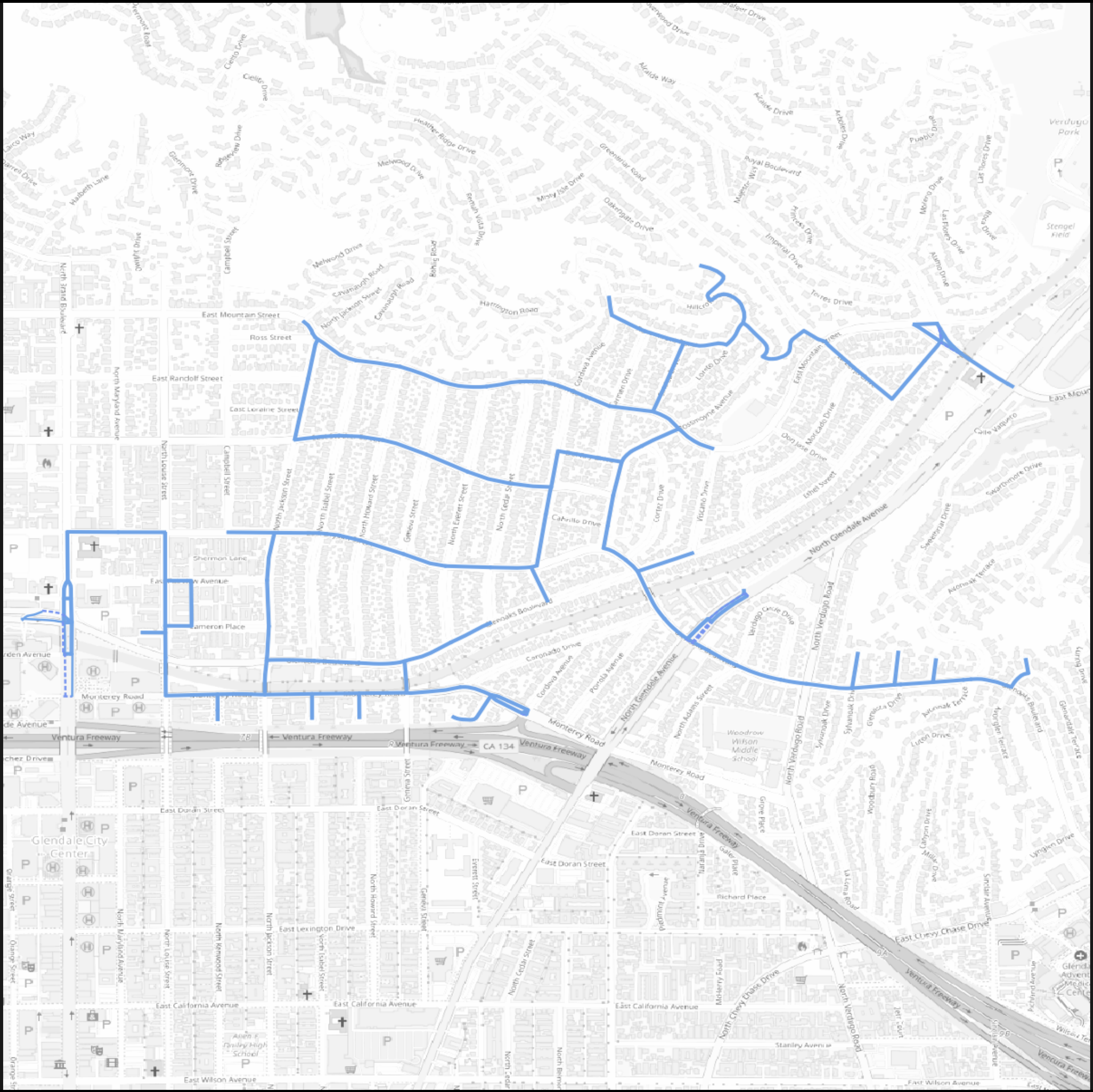
VISUALIZING THE OUTPUT

QGIS PLUS POSTGIS TABLES

- The real reason I included geometry in output table
- QGIS can directly display PostGIS geometry tables


Tujunga
I 210
Foothill Freeway
La Tuna Canyon Road
Verdugo Mountains
La Crescenta
CA 2
Verdugo Peak 952 m
Verdugo Mountains
Montrose
La Cañada Flintridge
Glendale Freeway
Five Points
Tongva Peak 803 m
Mt Thom
Cerro Negro
urbank
Magnolia Boulevard
Main Street
CA 2
Flint Peak 563 m
Los Angeles River
Forest Lawn Memorial Park
West Glendale
Glendale
CA 134
273 m
Mount Lee 521 m
Mount Chapel 492 m
Taco Peak 479 m
Mount Hollywood 492 m
Glendale Peak 361 m
Forest Lawn Memorial Park
Verdugo Road
Eagle Rock Boulevard
Mount Fiji 264 m
Popp Peak 302
Atwater


Tujunga
I 210
Foothill Freeway
La Tuna Canyon Road
14
14 Foothill Freeway
16
Verdugo Mountains
CA 2
La Crescenta
17A
17B
18
19
20
21
La Cañada Flintridge
Verdugo Peak 952 m
Verdugo Mountains
Tongva Peak 803 m
Mt Thom
Glendale Freeway
Five Points
Cerro Negro
22A
CA 2
Burbank
146B
146A
145B
Magnolia Boulevard
Main Street
5A
144
144B
Los Angeles River
West Glendale
142
Glendale
Flint Peak 563 m
CA 134
9A
9B
11
17B
17A
16
273 m
15B
15A
Forest Lawn Memorial Park
Forest Lawn Memorial Park
Mount Lee 521 m
Mount Chapel 492 m
Taco Peak 479 m
Mount Hollywood 492 m
Glendale Peak 361 m
141
141B
141A
140B
Atwater
Eagle Rock Boulevard
Mount Fiji 264 m


East Mountain Street
Ross Street
East Randolf Street
East Loraine Street
North Brand Boulevard
North Maryland Avenue
North Louise Street
Campbell Street
North Jackson Street
North Isabel Street
North Howard Street
Geneva Street
North Everett Street
North Cedar Street
Cabrillo Drive
Sherman Lane
Cameron Place
Monterey Road
Ventura Freeway
CA 134
Glenoaks Boulevard
Coronado Drive
Cordova Avenue
Portola Avenue
North Glendale Avenue
North Verdugo Road
North Adams Street
Verdugo Circle Drive
Woodrow Wilson Middle School
Harrington Road
Heather Ridge Drive
Melwood Drive
Cavanaugh Road
Greenbriar Road
Oakengate Drive
Misty Isle Drive
Alcalde Way
Royal Boulevard
Imperial Drive
Torres Drive
Princess Drive
Cortez Drive
Viscano Drive
Loreto Drive
Cossimoyne Avenue
Mercado Drive
Don Jose Drive
Ethel Street
Calle Vaquero
Sweetbriar Drive
Swarthmore Drive
Verdugo Park
Stengel Field
Sylvanoak Drive
Genelita Drive
Adonmak Terrace
Lucon Drive
East Doran Street
East Lexington Drive
East California Avenue
East Wilson Avenue
Stanley Avenue
Monterey Road
East Chevy Chase Drive
North Chevy Chase Drive
Richard Place
Glendale City Center
Allen F. Daily High School
Orange Street
North Kenwood Street
Glendale Adventist Medical Center

NICE MAPS, BUT …

- The maps are difficult to view
- Routes are on top of each other
- No sense of the movement of the vehicle
- Try animating!
- Helpful blog posts all over (look up geogiffery) (https://medium.com/@tjukanov/geospatial-animations-with-qgis-atlas-995d7ddb2d67)

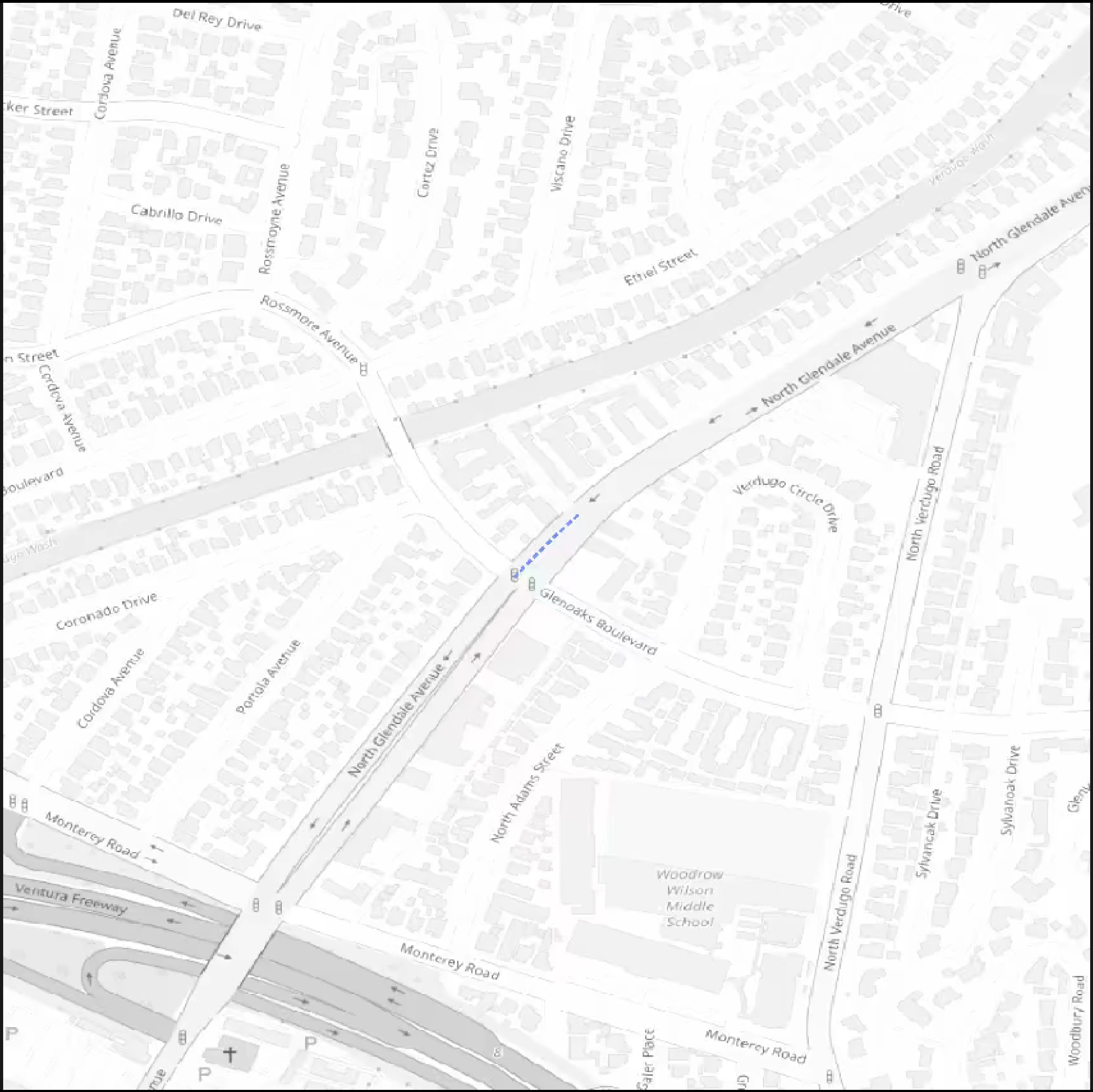
USE QGIS ATLAS FUNCTIONALITY

- Image stack style animation
- Make a print view
- Control the print view with an “atlas”
- Dump thousands of images to a directory
- Use ffmpeg

NAUSEA-INDUCING RESULTS

Animation link:

https://activimetrics.com/images/jittery.webm


Del Rey Drive
Cordova Avenue
ker Street
rive
Cortez Drive
Viscano Drive
Cabrillo Drive
Rossmoyne Avenue
Verdugo Wash
Ethel Street
North Glendale Ave
Rossmore Avenue
n Street
Cordova Avenue
North Glendale Avenue
oulevard
Verdugo Circle Drive
North Verdugo Road
ugo Wash
Glenoaks Boulevard
Coronado Drive
Cordova Avenue
Portola Avenue
North Glendale Avenue
North Adams Street
Sylvanoak Drive
Sylvanoak Drive
Glen
Monterey Road
Woodrow Wilson Middle School
Ventura Freeway
North Verdugo Road
Monterey Road
Monterey Road
Woodbury Road
P
P
P
Saler Place
nue

USE POSTGIS TO MAKE POV LAYER

- Break up the segments into pieces (currently using 25 meters)
- POV table computes spatial centroid over 2 preceding, 10 following segments
- POV table is then used as atlas layer

TIP FROM THE POSTGIS DOCS

- Use `ST_LineSubstring` to break line into N parts
- Each part is from i to i+1, i = [0 .. N-1]
- Use `generate_series` to generate the i values

POSTGIS DOC CODE:

```
SELECT field1, field2,
       ST_LineSubstring(the_geom, 100.00*n/len,
  CASE
    WHEN 100.00*(n+1) < len THEN 100.00*(n+1)/len
    ELSE 1
  END) AS the_geom
FROM
  (SELECT sometable.field1, sometable.field2,
  ST_LineMerge(sometable.the_geom) AS the_geom,
  ST_Length(sometable.the_geom) As len
  FROM sometable ) AS t
CROSS JOIN generate_series(0,10000) AS n
WHERE n*100.00/len < 1;
```

MY MODIFICATIONS

- Construct SQL with WITH statements
- Compute required length of series based on longest road / 25 meters

POSTGIS TRICK

```
st_length(st_transform(geom,32611))
```

- To get meters, transform geometry
- `geom` starts in projection 4326, which is in degrees
- Using `st_length()` on degrees is useless
- By transforming to projection 32611, the `st_length()` call gives meters

METERS TRICK → 326??

- Find your zone https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system#
- Pick the correct SRID

```
select srid,proj4text
 from spatial_ref_sys where srid between 32600 and 32661
 order by srid;
 srid  |    proj4text
-------+---------------------------------------
 32601 | +proj=utm +zone=1 +datum=WGS84 +units=m +no_defs
 32602 | +proj=utm +zone=2 +datum=WGS84 +units=m +no_defs
 32603 | +proj=utm +zone=3 +datum=WGS84 +units=m +no_defs
 32604 | +proj=utm +zone=4 +datum=WGS84 +units=m +no_defs
 32605 | +proj=utm +zone=5 +datum=WGS84 +units=m +no_defs
 …
```

FIND THE LONGEST SEGMENT

```
with
 lengthshare as (
   select id,linkid,veh,sweep,geom,
          st_length(st_transform(geom,32611)) as len
   from solver_output
   order by id
   ),
 maxlen as (
   select max(len) as len  from lengthshare
   ),
```

DETERMINE “MAXITER”

```
maxiter as (
   select (ceil(len/25.00)+1)::int as maxiter
   from maxlen
   )
```

Divide the longest length by 25, and round

USING MAXITER, GENERATE SERIES

```
series as (
   select maxiter, generate_series(1,maxiter) - 1 as n
   from maxiter
   )
```

More flexible than the example code fixing at 10000

SNIP EACH LINE INTO PIECES

```
snipped as (
   select id, id+(n/maxiter::numeric) as frame,
     linkid,veh,sweep,
     st_linesubstring(geom,
          25.00*n/len,
          case
            when 25.00*(n+1) < len then 25.00*(n+1)/len
            else 1
          end) as geom
   from lengthshare l
   cross join series s
   where s.n*25.00/len < 1
   order by frame )
```

FINALLY, SAVE TO NEW TABLE

```
insert into
  solver_output_snipped (id,frame,linkid,veh,sweep,geom)
  select id,frame,linkid,veh,sweep,geom from snipped;
```

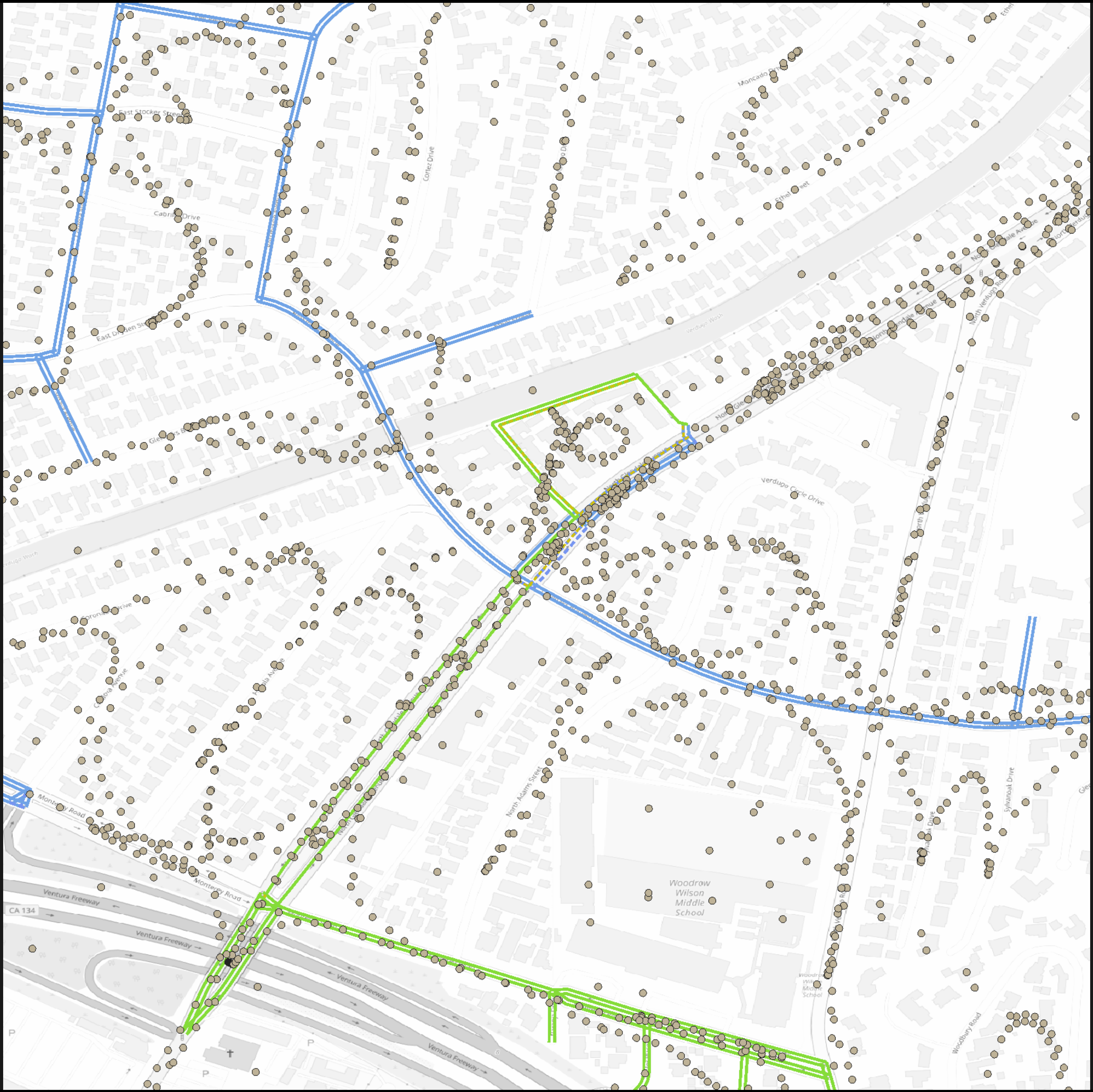
THERE WILL BE JITTER

```
st_linesubstring(geom,25.00*n/len,
                 case when 25.00*(n+1) < len then 25.00*
        (n+1)/len
                 else 1 end) as geom
```

- When the line doesn’t divide into 25 meters exactly, the last segment will be shorter
- Will result in some jitter at end of roads

THE RESULT

- A table of points
- Can be used as the point-of-view
- Centers the atlas window where needed


East Stocker Street
Cabrillo Drive
Cortez Drive
East Dryden Street
Glenoaks Blvd
Moncado Drive
Ethel Street
Verdugo Wash
North Glendale Avenue
North Verdugo Road
Verdugo Circle Drive
Coronado Drive
Cordova Avenue
Hermosa Avenue
North Adams Street
Monterey Road
Ventura Freeway
CA 134
Woodrow Wilson Middle School
Sylvanoak Drive
Woodbury Road
P

BONUS: ARROW HEADS!

- Previous animation just showed current street
- With snipped roads, can show progress along street (every 25m)
- Looks more like a real animation

SMOOTHER ANIMATION

Animation link

https://activimetrics.com/images/smoother.webm

untain Street
Carmen Drive
Rossmoyne Avenue
Nibley Park Tennis Court
Nibley Park
Moncado Drive
East Mountain Street
Don Jose Drive
Del Rey Drive
Cordova Avenue
East Stocker Street
North Cedar Street
Cortez Drive
Viscano Drive
Cabrillo Drive
Rossmoyne Avenue
Ethel Street
Rossmore Avenue
East Dryden Street
Cordova Avenue
Glenoaks Boulevard
Verdugo Wash
North Glendale Avenue
Verdugo Circle Drive
North Verdugo Road
Coronado Drive
Glenoaks Boulevard
Cordova Avenue
Portola Avenue
North Glendale Avenue
North Adams Street
Monterey Road
Woodrow
Road
Sylvanoak Drive

QUESTIONS?

THANK YOU