A*NAV:
A* Navigation for Cyclists with Tired Legs

What does A*NAV do?

- A*NAV is a **web-based routing application** that allows users to **customize routes** based on their own **route preferences**
- It provides a quick and efficient service that finds the best route based on these dynamic preferences
- Results are displayed in an intuitive and user-friendly interface on a visually appealing website

What Factors Does A*NAV Consider?

- Distance
- Incline
- Road Type
- Bike Lane Availability

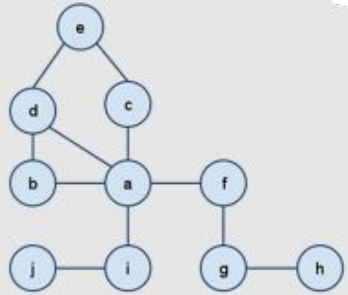
Architecture

Open Street Map

Google Elevation API

PostgreSQL

Cloud-hosted database



Route optimization

API

Frontend UI

Data Pipeline

OSM Bulk Data

OSM XML
Stream Parsing

Google Elevation
API Queries

Database Ingestion

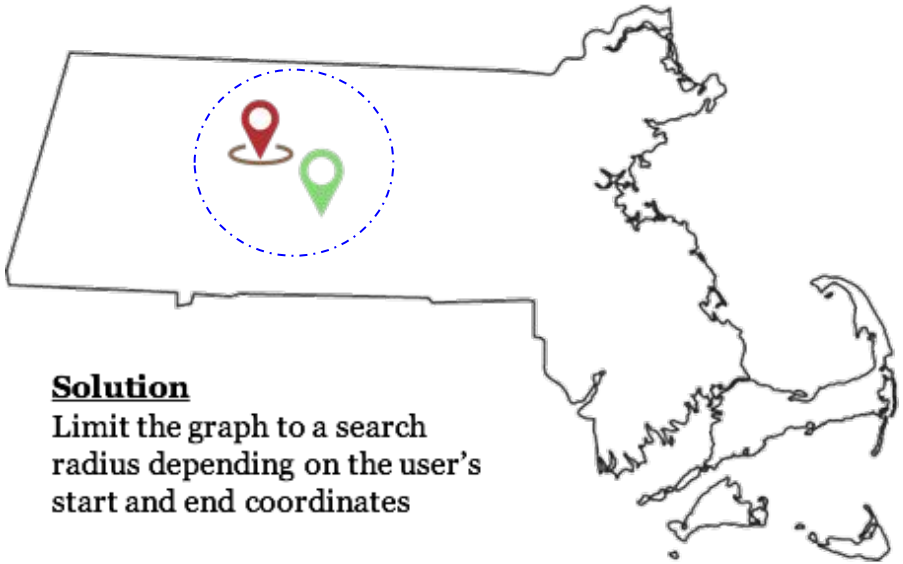
Data Output | Explain | Messages | Notifications | Query History

	id bigint	way_id bigint	start_node_id bigint	end_node_id bigint	highway text
1	16712134	44383526	5262444453	5262444474	track
2	5689408	9119721	67493103	67472267	residential
3	16036780	32819122	4317697821	4317697822	road
4	14451463	44044548	5270810458	71212647	track
5	1703204	8820963	2589923768	2589924003	track

Data Cleaning
and Transformation

Speed Optimization

How do we limit the size of our graph?

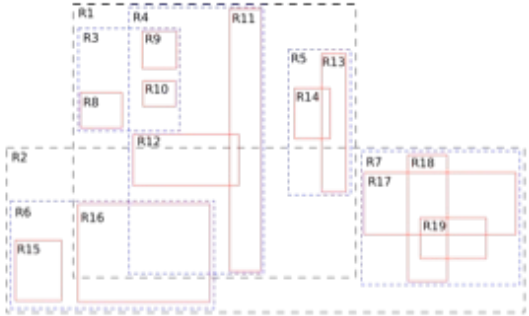


Solution

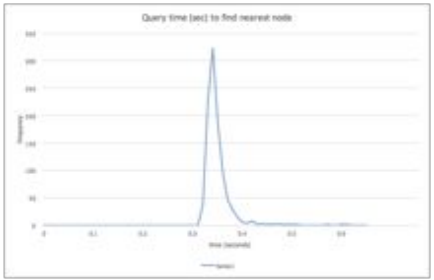
Limit the graph to a search radius depending on the user's start and end coordinates

How do we quickly get data from PostgreSQL?

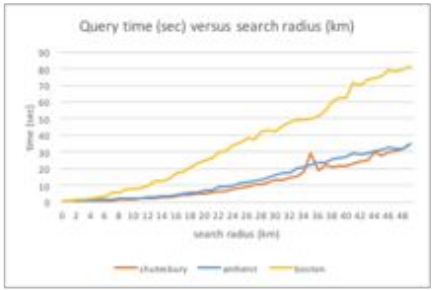
R-Tree Spatial Indexes



Results



0.3 seconds to find nearest node



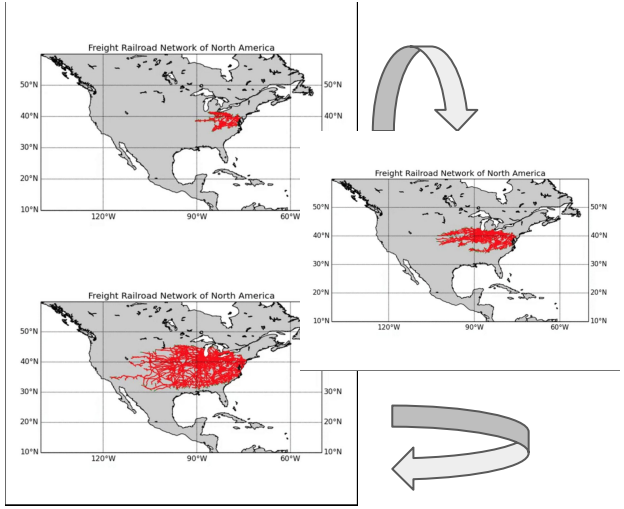
Find nearby nodes in ~seconds

Algorithm

A* Algorithm =
Dijkstra + Heuristic Function

With heuristic function, A* can prioritize the search on edges that are more “likely” to be on the shortest path

We use 2D distance for our heuristic function.

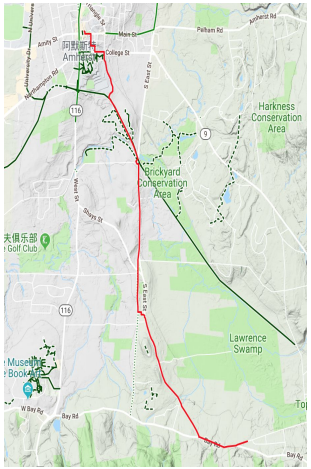


Take User Preference Into Account?

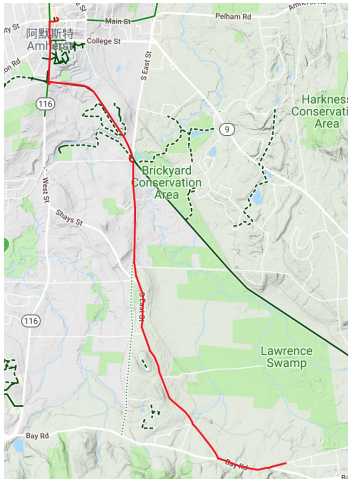
Add cost function to the edge based on preference!

```
incl = incline_multiplier(float(incline))*flatness_pref
cost = float(distance) * multiplier + incl
```

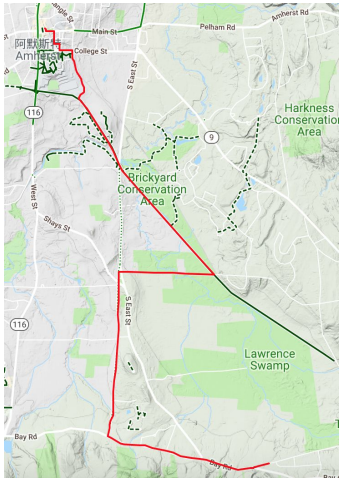
No Pref



Bike Pref

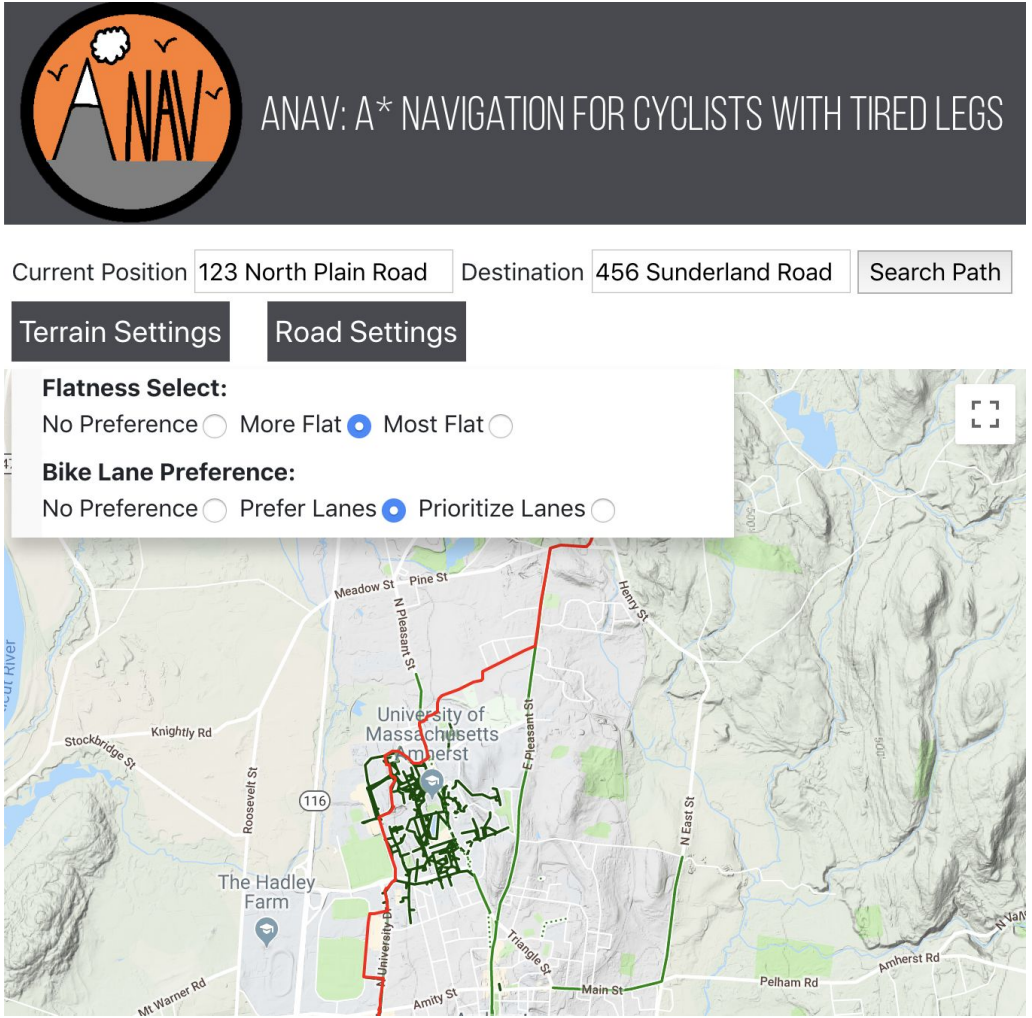


Highway Pref



Frontend

- Familiar, Contextual UI
- Clean Readability
- Embedded Google Map
- Form validation
- Helpful error messages



API

HTML Page

Current Position: 54 Stony Hill Road, Amherst, MA, 01003
Destination: 154 Hicks Way, Amherst, MA, 01003

Flatness Select:
No Preference ○ More Flat ● Most Flat ○

Bike Lane Preference:
No Preference ○ Prefer Lanes ● Prioritize Lanes ○

Motorway Preference:
No Preference ○ Prefer Motorways ● Prioritize Motorways ○

Highway Preference:
No Preference ○ Prefer Highways ● Prioritize Highways ○

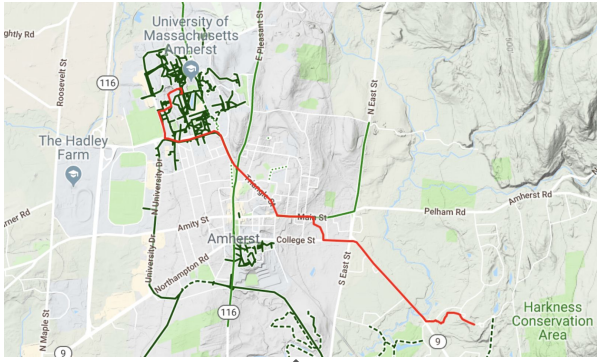
Residential Road Preference:
No Preference ○ Prefer Residential ● Prioritize Residential ○

Flask
web development, one drop at a time

Python (Flask)

```
if request.method == 'POST':
    CurPos = request.form['CurPos']
    Destination = request.form['Destination']
    flatness_val = float(request.values.get('group1'))
    bicycle_val = float(request.values.get('group2'))
    motorway_val = float(request.values.get('group3'))
    highway_val = float(request.values.get('group4'))
    residential_val = float(request.values.get('group5'))
```

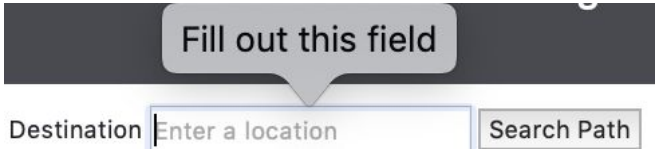
Display Route




```
# Get Optimized route from the optimizer
A = (CurPos_location.latitude, CurPos_location.longitude)
B = (Destination_location.latitude, Destination_location.longitude)
preferences = (flatness_val, bicycle_val, distance_val,
               motorway_val, highway_val, residential_val)

route = graph_utils.optimize(A, B, preferences, debug=True)
```

Call Route Optimizer



Input Parameters Validation