A Sense of Place: Quick and Dirty Accessibility Analysis Using Open Source Network Analysis Tools

Adam Porr

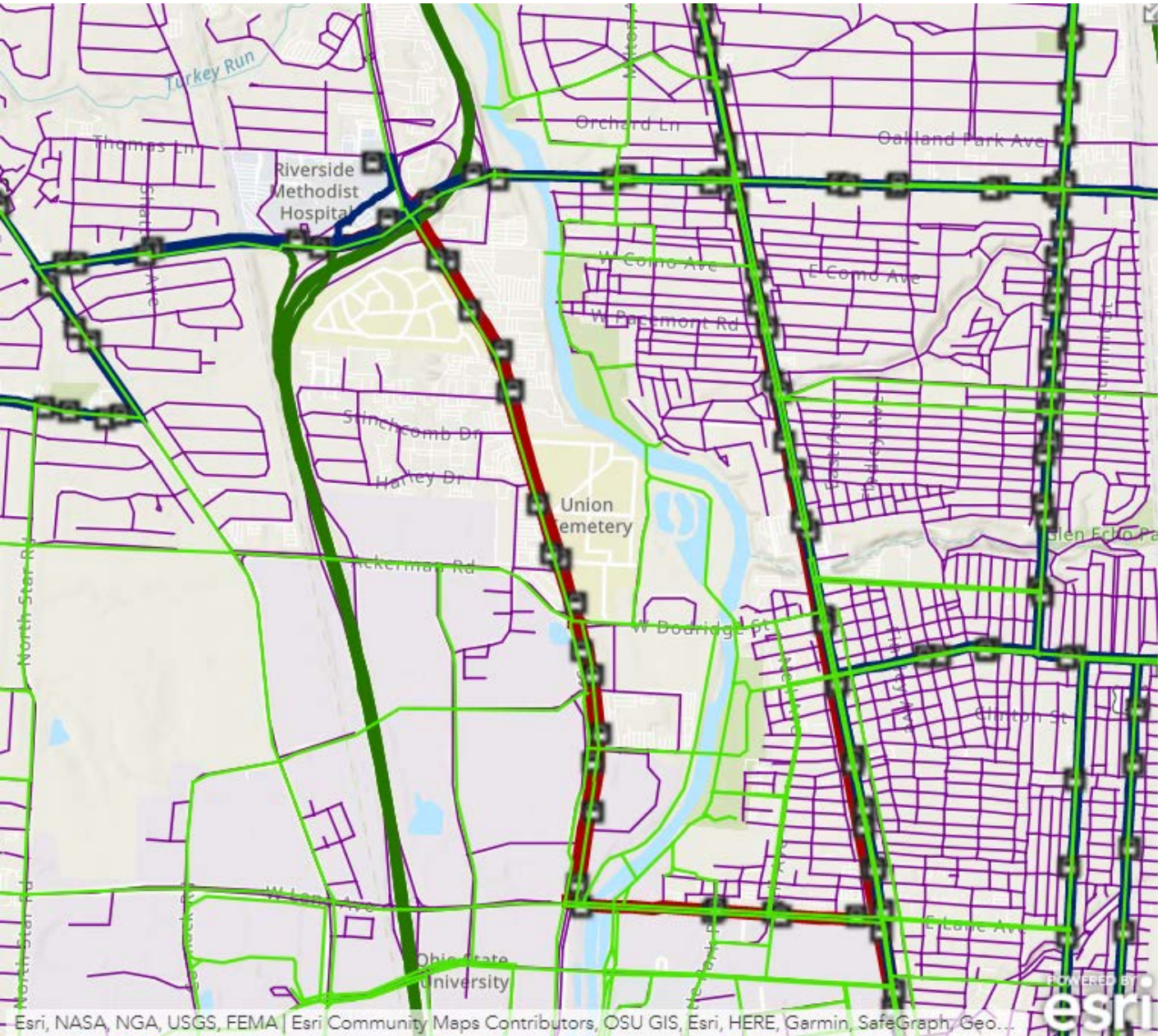
Research & Data Officer

Mid-Ohio Regional Planning Commission

Ohio GIS Conference

September 20 – 22, 2022

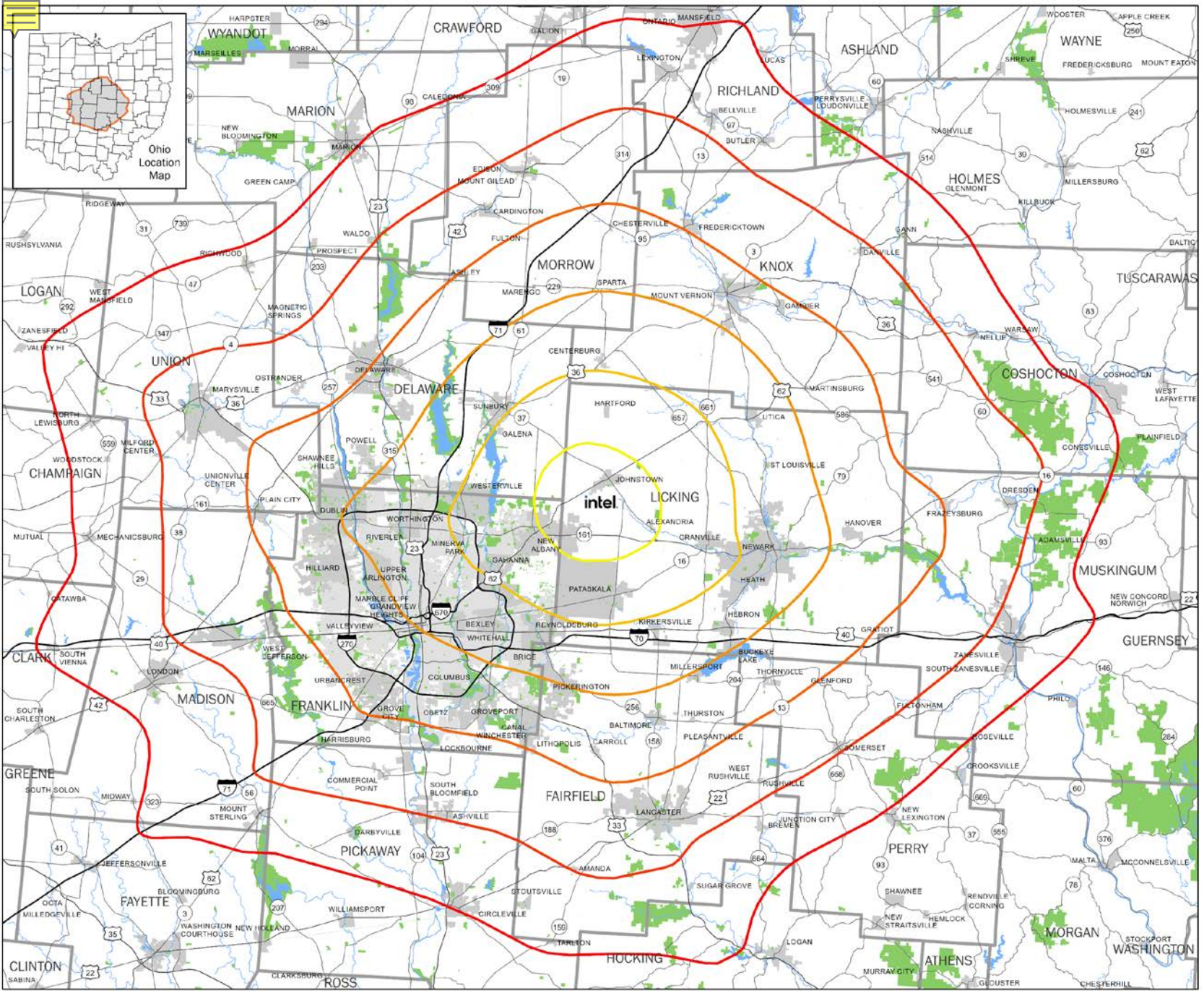
Networks are hard. Multimodal networks are much harder!



- Connectivity
- Costs
- Restrictions
- Data availability, attributes, and integration
- **Also, data and tools can be expensive!**

Sometimes you need foolproof routing or research-grade analysis.

Source: NBC



But sometimes you just need a general sense of a place… and the people and things that are nearby.

Source: Extracted from MORPC Intel Site Driving Times map

Luckily, there are (open source) apps for that!

OpenStreetMap

Network of roads, trails, sidewalks, etc.

Google Transit

Network of transit routes (“GTFS”)

conveyal

“R5” routing engine

(Rapid Realistic Routing on Real-world and Reimagined networks)

One R package to rule them all and on the PC bind them!

That's great, but how does it work?

```r
install.packages("r5r")
library(r5r)
r5r_core <- setup_r5(data_path = "./input_data", verbose=TRUE)
locations = read.csv("./input_data/sample_locations.csv")
ttm_car <- travel_time_matrix(r5r_core = r5r_core,
                          origins = locations[1, ],
                          destinations = locations[1:nrow(locations), ],
                          mode = c("CAR"),
                          departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                          format = "%Y-%m-%d %H:%M:%S"),
                          max_trip_duration = 90)
ttm_bike <- travel_time_matrix(r5r_core = r5r_core,
                          origins = locations[1, ],
                          destinations = locations[1:nrow(locations), ],
                          mode = c("BICYCLE"),
                          departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                          format = "%Y-%m-%d %H:%M:%S"),
                          max_trip_duration = 90)
ttm_transit <- travel_time_matrix(r5r_core = r5r_core,
                          origins = locations[1, ],
                          destinations = locations[1:nrow(locations), ],
                          mode = c("TRANSIT","WALK"),
                          departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                          format = "%Y-%m-%d %H:%M:%S"),
                          max_trip_duration = 90)
write.csv(ttm_car, "output_data/ttm_sample_locations_car.csv", row.names=FALSE)
write.csv(ttm_bike, "output_data/ttm_sample_locations_bike.csv", row.names=FALSE)
write.csv(ttm_transit, "output_data/ttm_sample_locations_transit.csv", row.names=FALSE)
```

Pre-requisites:

1. Install **R**
2. Download OSM extract (a)
3. Download **GTFS** file (b)
4. Create a spreadsheet with your **origin and destination coordinates** with fields "id", "lat", and "lon".

(1) https://download.geofabrik.de/

(2) https://cota.com/data/

That’s great, but how does it work?

```
install.packages("r5r")
library(r5r)
r5r_core <- setup_r5(data_path = "./input_data", verbose=TRUE)
locations = read.csv("./input_data/sample_locations.csv")
ttm_car <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("CAR"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_bike <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("BICYCLE"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_transit <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("TRANSIT","WALK"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
write.csv(ttm_car, "output_data/ttm_sample_locations_car.csv", row.names=FALSE)
write.csv(ttm_bike, "output_data/ttm_sample_locations_bike.csv", row.names=FALSE)
write.csv(ttm_transit, "output_data/ttm_sample_locations_transit.csv", row.names=FALSE)
```

Install and load r5r

That's great, but how does it work?

```r
install.packages("r5r")
library(r5r)
r5r_core <- setup_r5(data_path = "./input_data", verbose=TRUE)
locations = read.csv("./input_data/sample_locations.csv")
ttm_car <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("CAR"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_bike <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("BICYCLE"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_transit <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("TRANSIT","WALK"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
write.csv(ttm_car, "output_data/ttm_sample_locations_car.csv", row.names=FALSE)
write.csv(ttm_bike, "output_data/ttm_sample_locations_bike.csv", row.names=FALSE)
write.csv(ttm_transit, "output_data/ttm_sample_locations_transit.csv", row.names=FALSE)
```

Build network from OSM and GTFS

Note: To understand the network in more detail, see: http://docs.opentripplanner.org/en/latest/RouterConfiguration/

That’s great, but how does it work?

```
install.packages("r5r")
library(r5r)
r5r_core <- setup_r5(data_path = "./input_data", verbose=TRUE)
locations = read.csv("./input_data/sample_locations.csv")
ttm_car <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("CAR"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_bike <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("BICYCLE"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_transit <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("TRANSIT","WALK"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
write.csv(ttm_car, "output_data/ttm_sample_locations_car.csv", row.names=FALSE)
write.csv(ttm_bike, "output_data/ttm_sample_locations_bike.csv", row.names=FALSE)
write.csv(ttm_transit, "output_data/ttm_sample_locations_transit.csv", row.names=FALSE)
```

Load origin and destination coordinates

That's great, but how does it work?

```
install.packages("r5r")
library(r5r)
r5r_core <- setup_r5(data_path = "./input_data", verbose=TRUE)
locations = read.csv("./input_data/sample_locations.csv")
ttm_car <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("CAR"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_bike <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("BICYCLE"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_transit <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("TRANSIT","WALK"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
write.csv(ttm_car, "output_data/ttm_sample_locations_car.csv", row.names=FALSE)
write.csv(ttm_bike, "output_data/ttm_sample_locations_bike.csv", row.names=FALSE)
write.csv(ttm_transit, "output_data/ttm_sample_locations_transit.csv", row.names=FALSE)
```

Compute travel time matrices for several modes (car, bike, transit)

Note: The full list of configurable parameters is available at: https://ipeagit.github.io/r5r/reference/travel_time_matrix.html

That's great, but how does it work?

```
install.packages("r5r")
library(r5r)
r5r_core <- setup_r5(data_path = "./input_data", verbose=TRUE)
locations = read.csv("./input_data/sample_locations.csv")
ttm_car <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("CAR"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_bike <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("BICYCLE"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
ttm_transit <- travel_time_matrix(r5r_core = r5r_core,
                              origins = locations[1, ],
                              destinations = locations[1:nrow(locations), ],
                              mode = c("TRANSIT","WALK"),
                              departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                              format = "%Y-%m-%d %H:%M:%S"),
                              max_trip_duration = 90)
write.csv(ttm_car, "output_data/ttm_sample_locations_car.csv", row.names=FALSE)
write.csv(ttm_bike, "output_data/ttm_sample_locations_bike.csv", row.names=FALSE)
write.csv(ttm_transit, "output_data/ttm_sample_locations_transit.csv", row.names=FALSE)
```

Write the matrices to CSV

That’s great, but how does it work?

```
install.packages("r5r")
library(r5r)
r5r_core <- setup_r5(data_path = "./input_data", verbose=TRUE)
locations = read.csv("./input_data/sample_locations.csv")
ttm_car <- travel_time_matrix(r5r_core = r5r_core,
                          origins = locations[1, ],
                          destinations = locations[1:nrow(locations), ],
                          mode = c("CAR"),
                          departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                          format = "%Y-%m-%d %H:%M:%S"),
                          max_trip_duration = 90)
ttm_bike <- travel_time_matrix(r5r_core = r5r_core,
                          origins = locations[1, ],
                          destinations = locations[1:nrow(locations), ],
                          mode = c("BICYCLE"),
                          departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                          format = "%Y-%m-%d %H:%M:%S"),
                          max_trip_duration = 90)
ttm_transit <- travel_time_matrix(r5r_core = r5r_core,
                          origins = locations[1, ],
                          destinations = locations[1:nrow(locations), ],
                          mode = c("TRANSIT","WALK"),
                          departure_datetime = as.POSIXct("2022-09-12 17:00:00",
                                                          format = "%Y-%m-%d %H:%M:%S"),
                          max_trip_duration = 90)
write.csv(ttm_car, "output_data/ttm_sample_locations_car.csv", row.names=FALSE)
write.csv(ttm_bike, "output_data/ttm_sample_locations_bike.csv", row.names=FALSE)
write.csv(ttm_transit, "output_data/ttm_sample_locations_transit.csv", row.names=FALSE)
```

Total runtime:

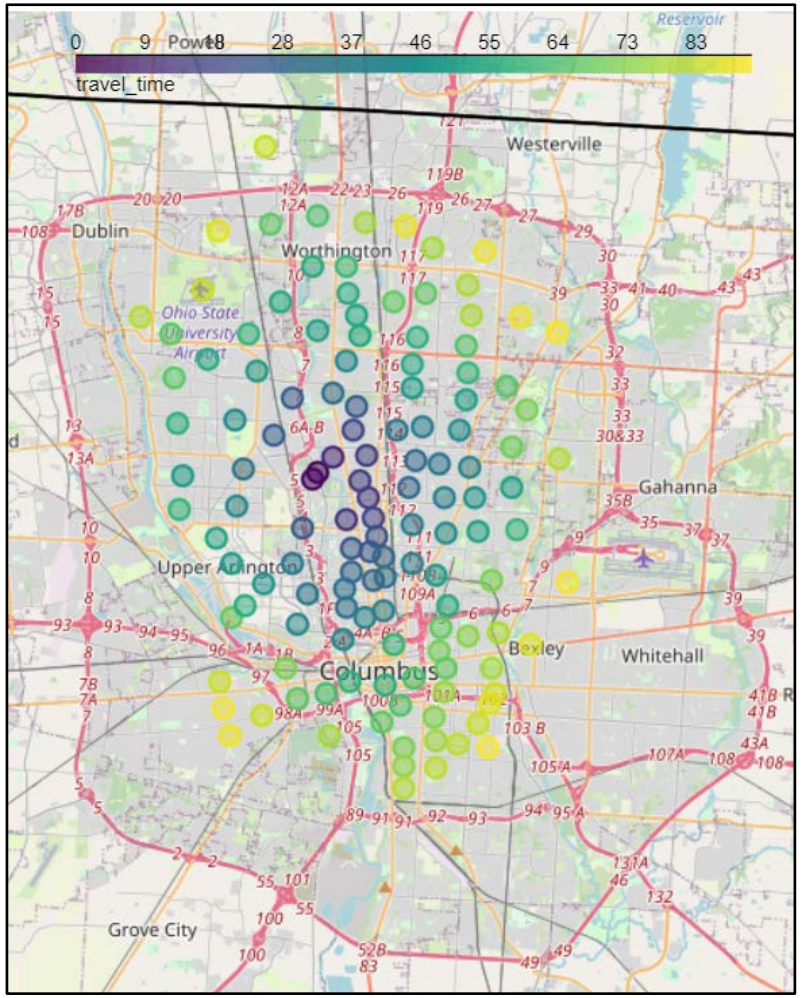
2.02 minutes

(4 cores @ 3GHz, 16GB RAM)

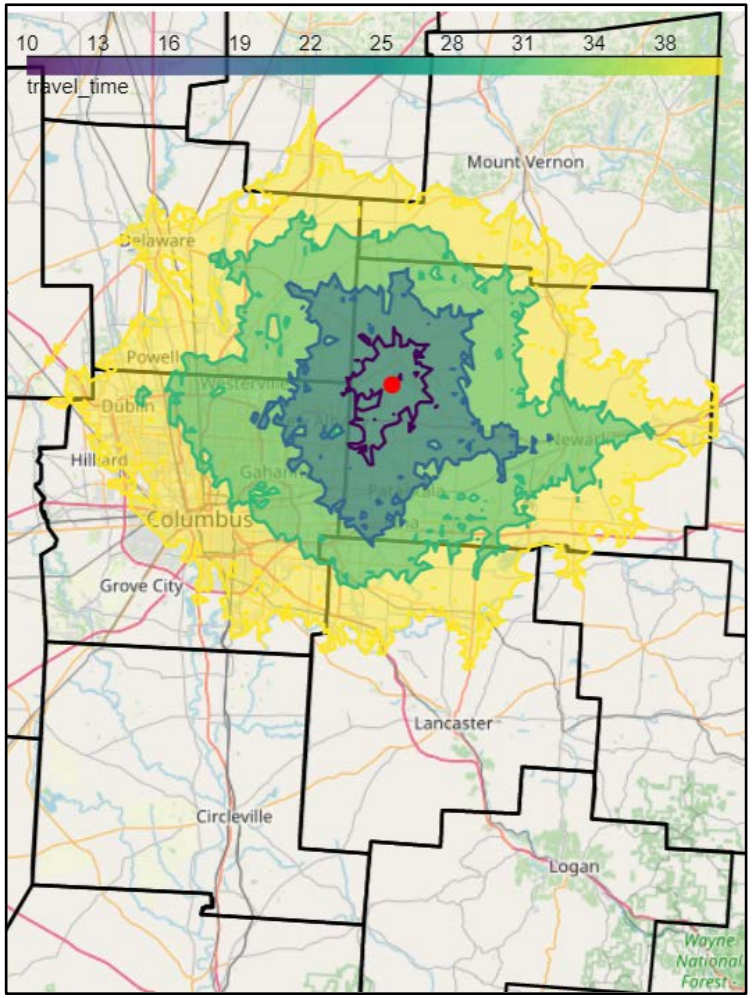
The result doesn’t look very impressive at first…

	A	B	C	D
1	**fromId**	**toId**	**travel_time**	
2	1	161	89	
3	1	352	87	
4	1	450	89	
5	1	451	86	
6	1	452	87	
7	1	453	88	
8	1	551	87	
9	1	552	85	
10	1	553	86	

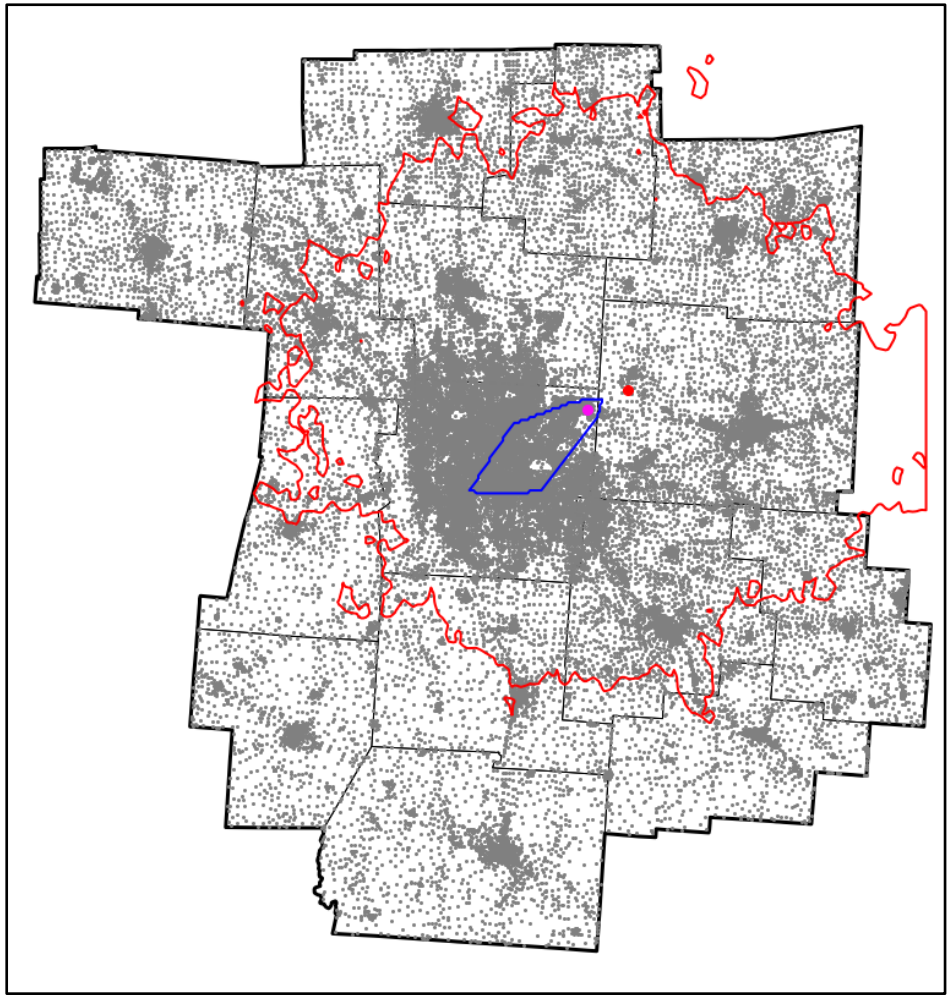
… but with a little GIS work, cool things are possible!



Travel times point-to-point

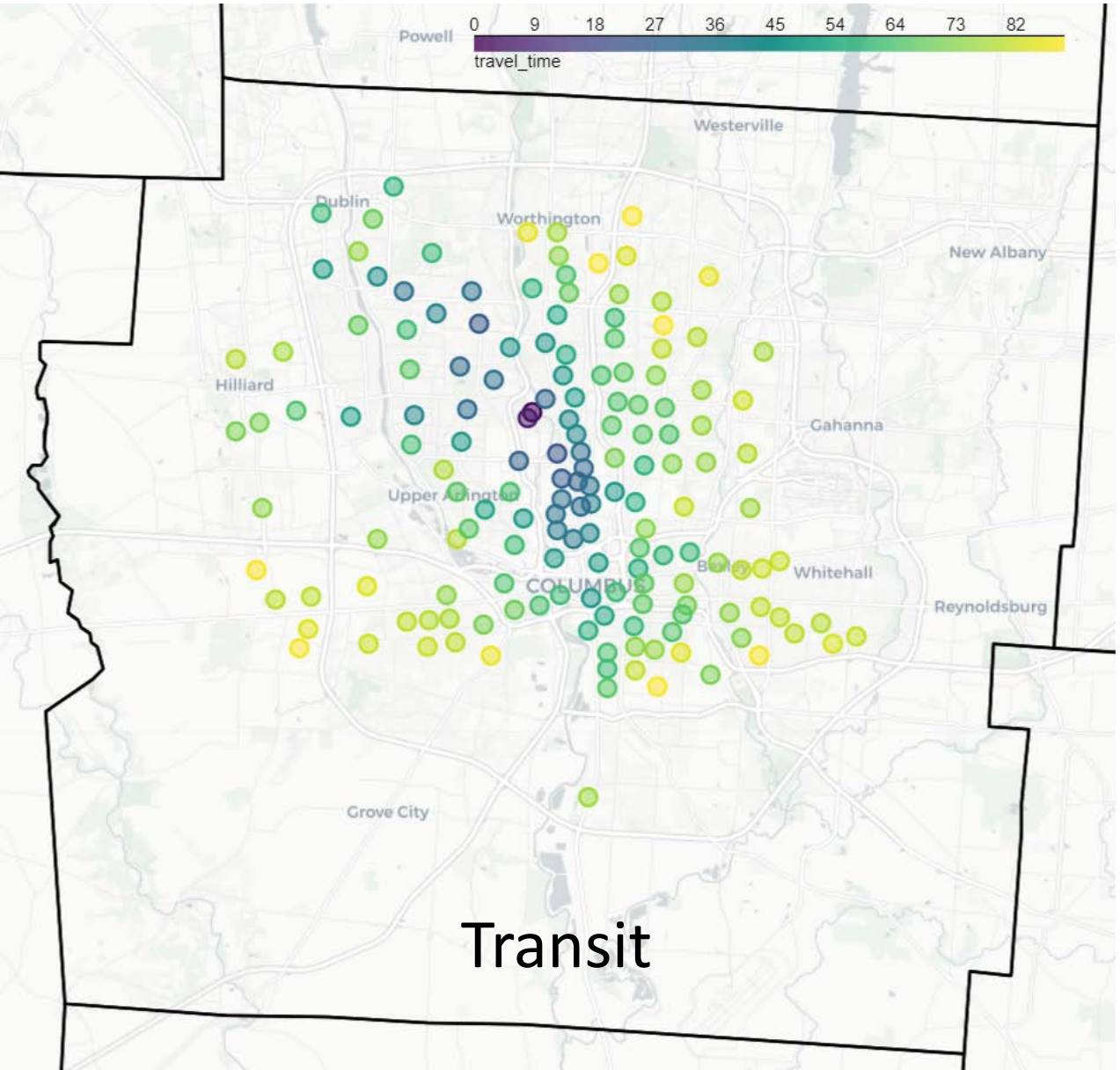


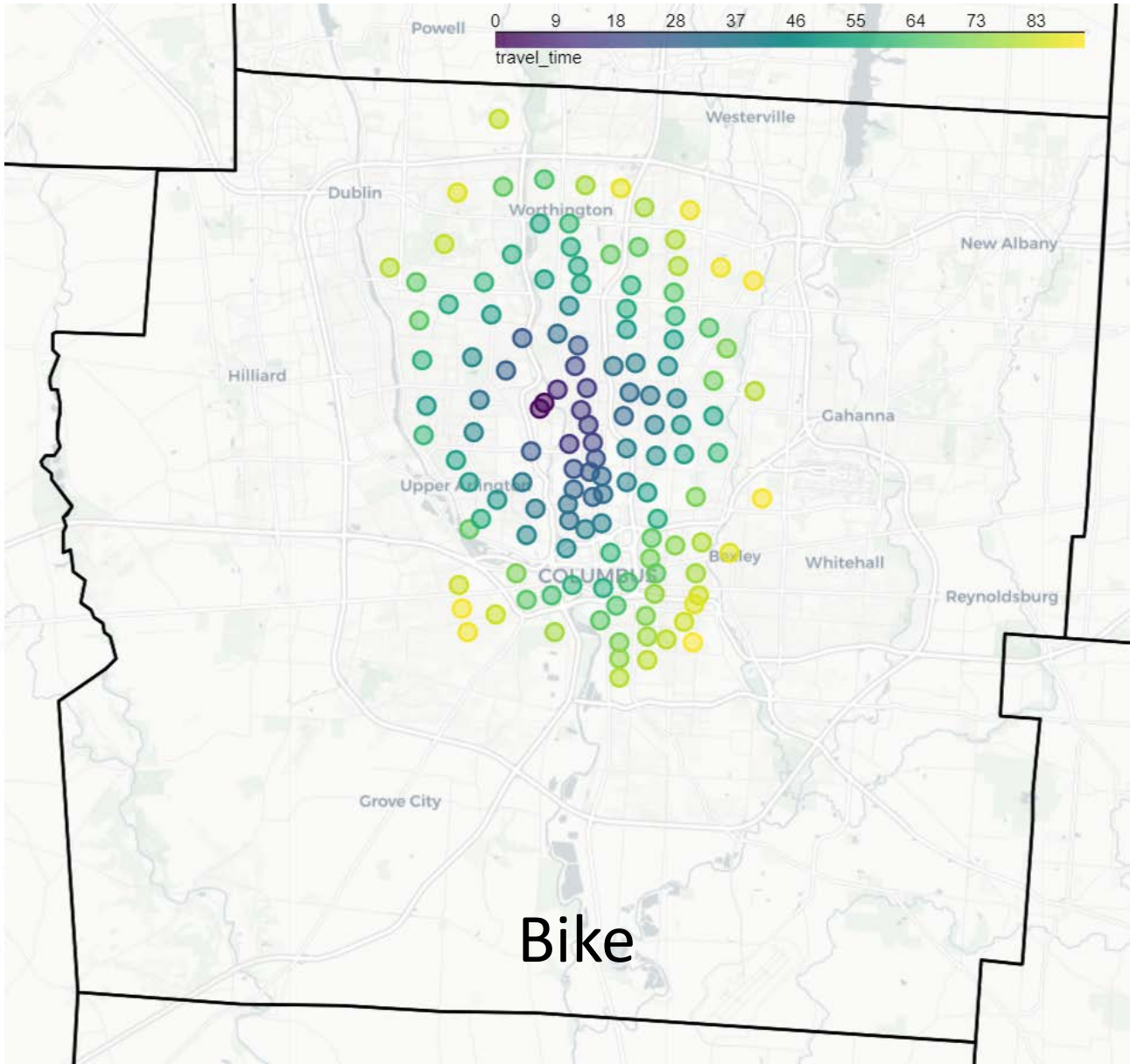
Travel time rings



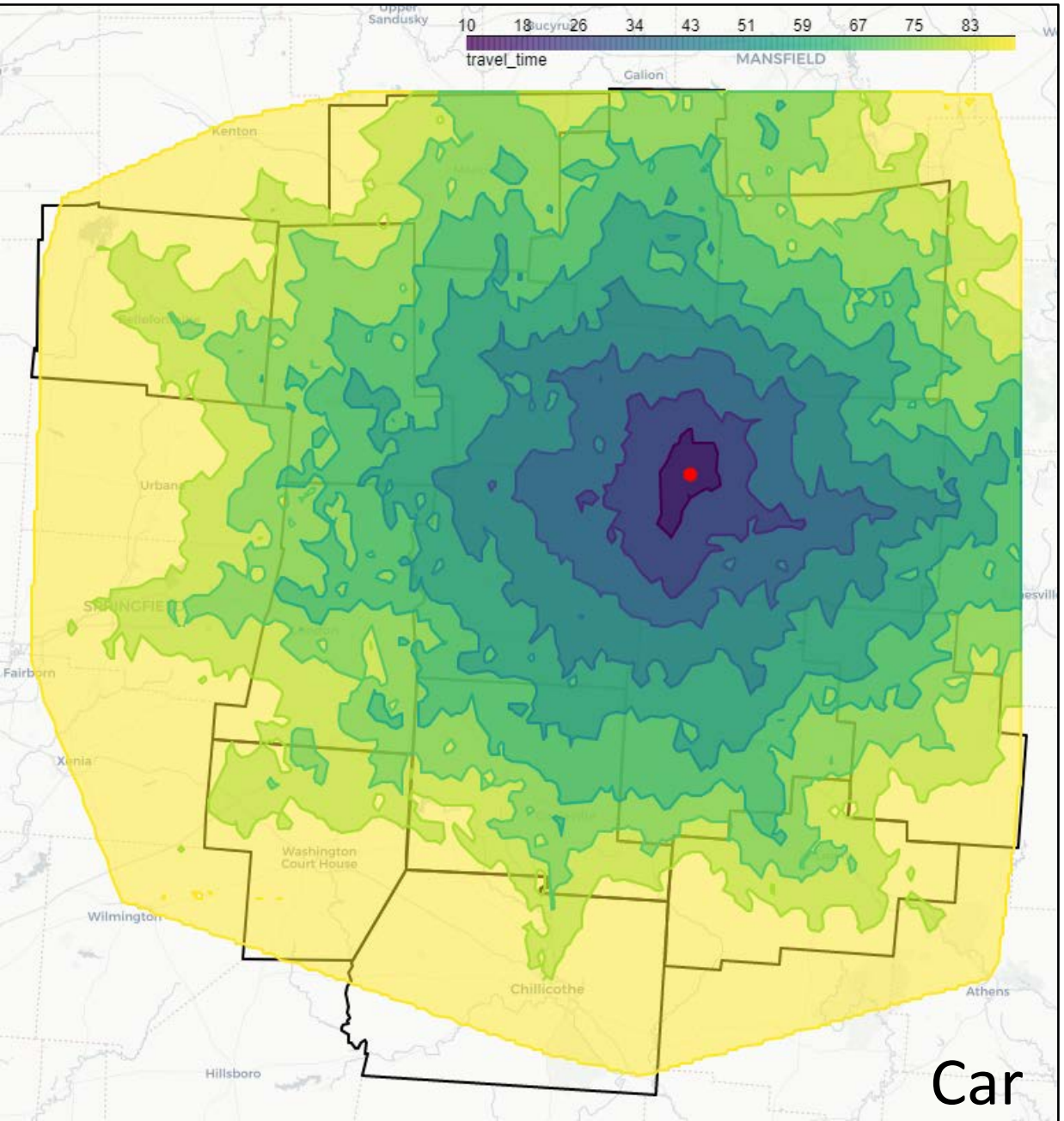
Accessibility analysis

Example - Travel time comparison by mode (from this conference)

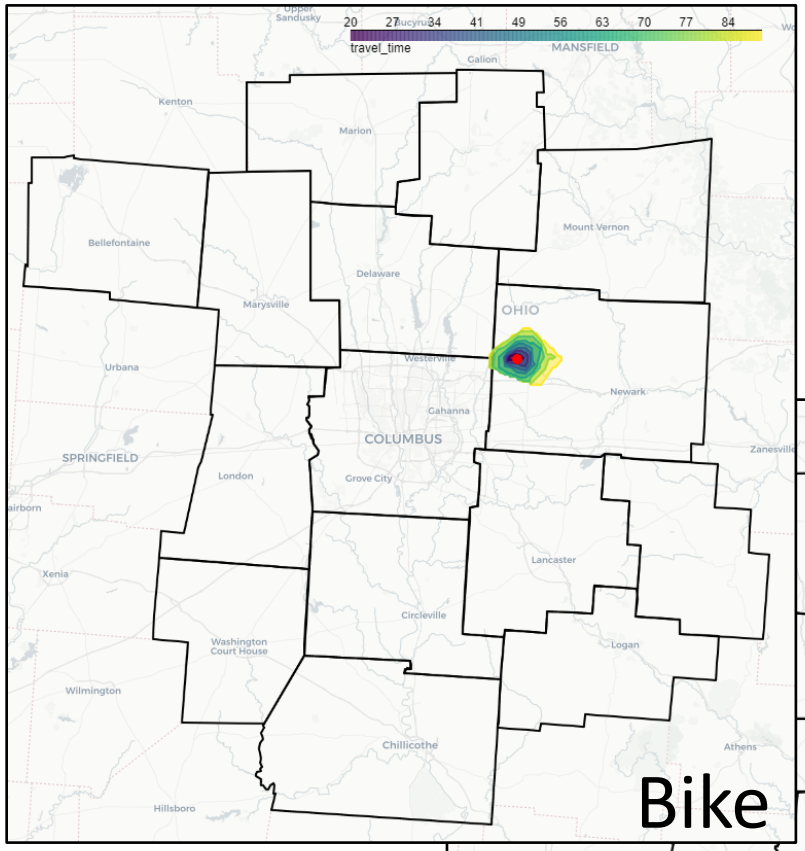




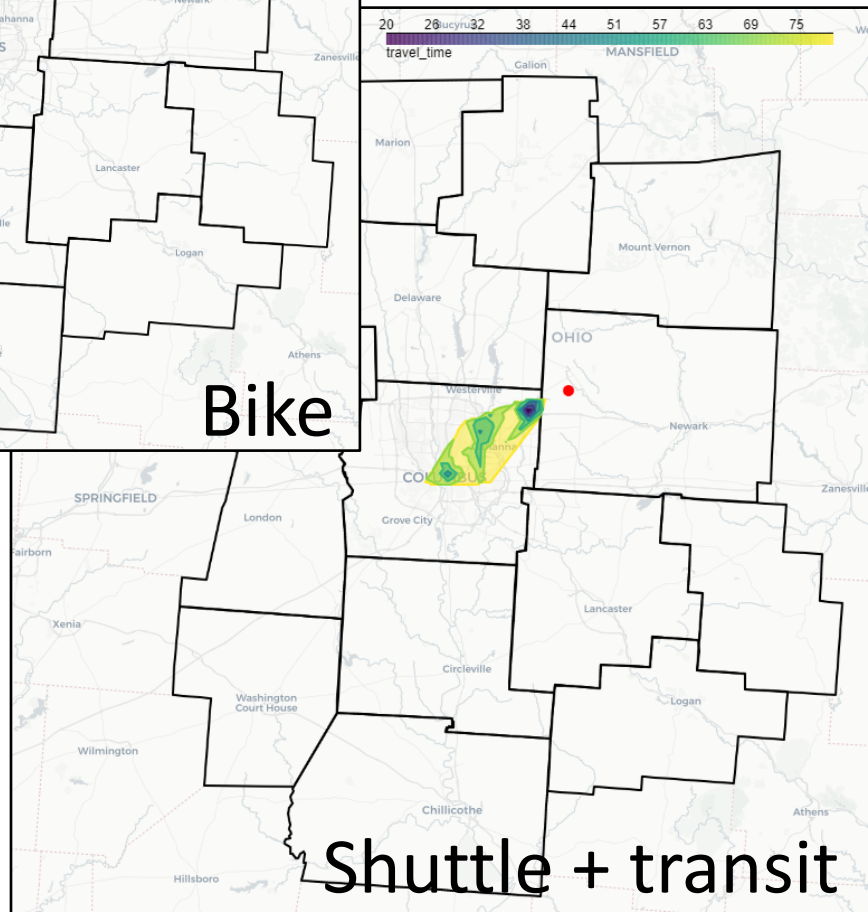
Example - Travel time rings (from Intel site)



Car

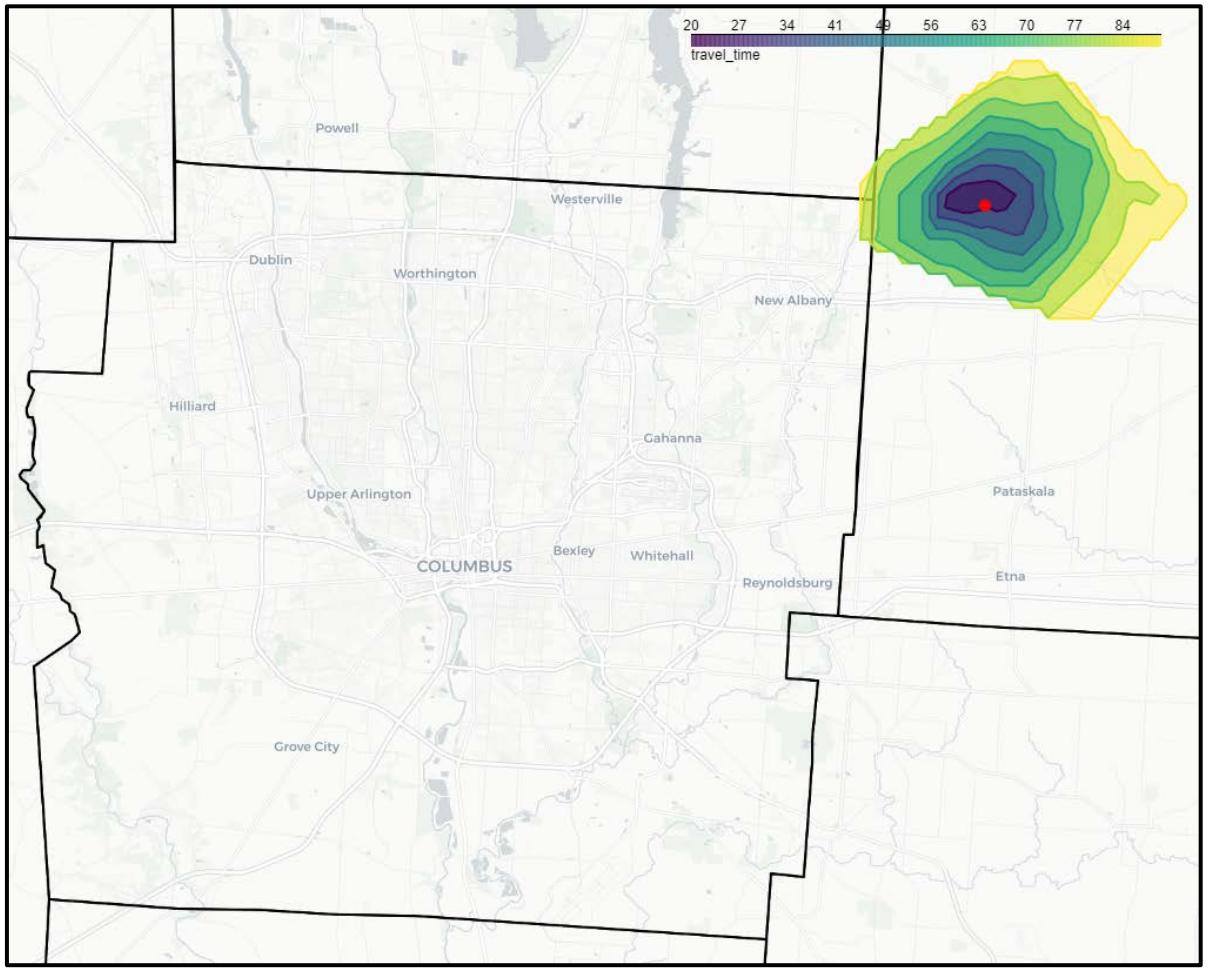


Bike

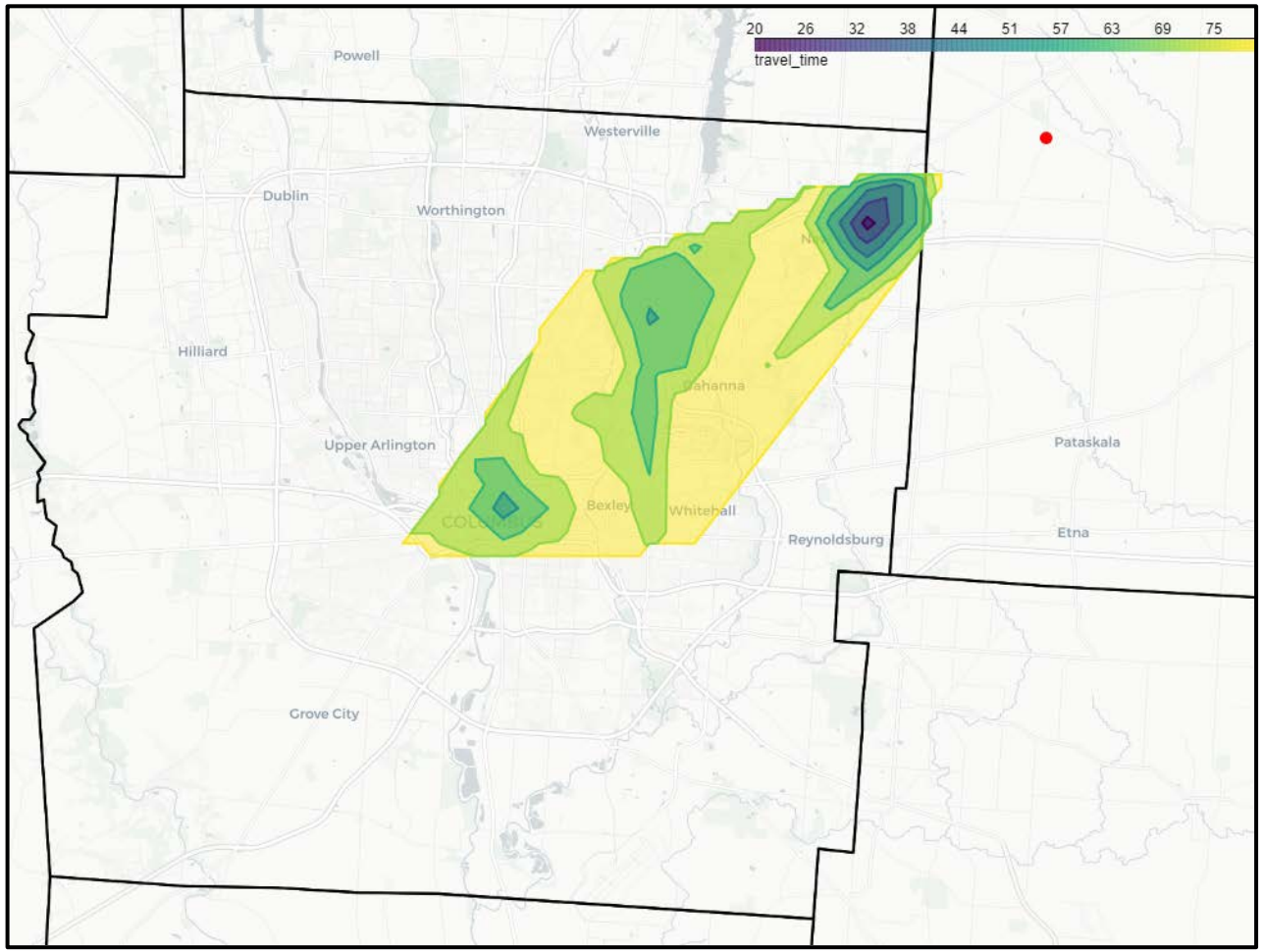


Shuttle + transit

Example - Travel time rings (from Intel site)

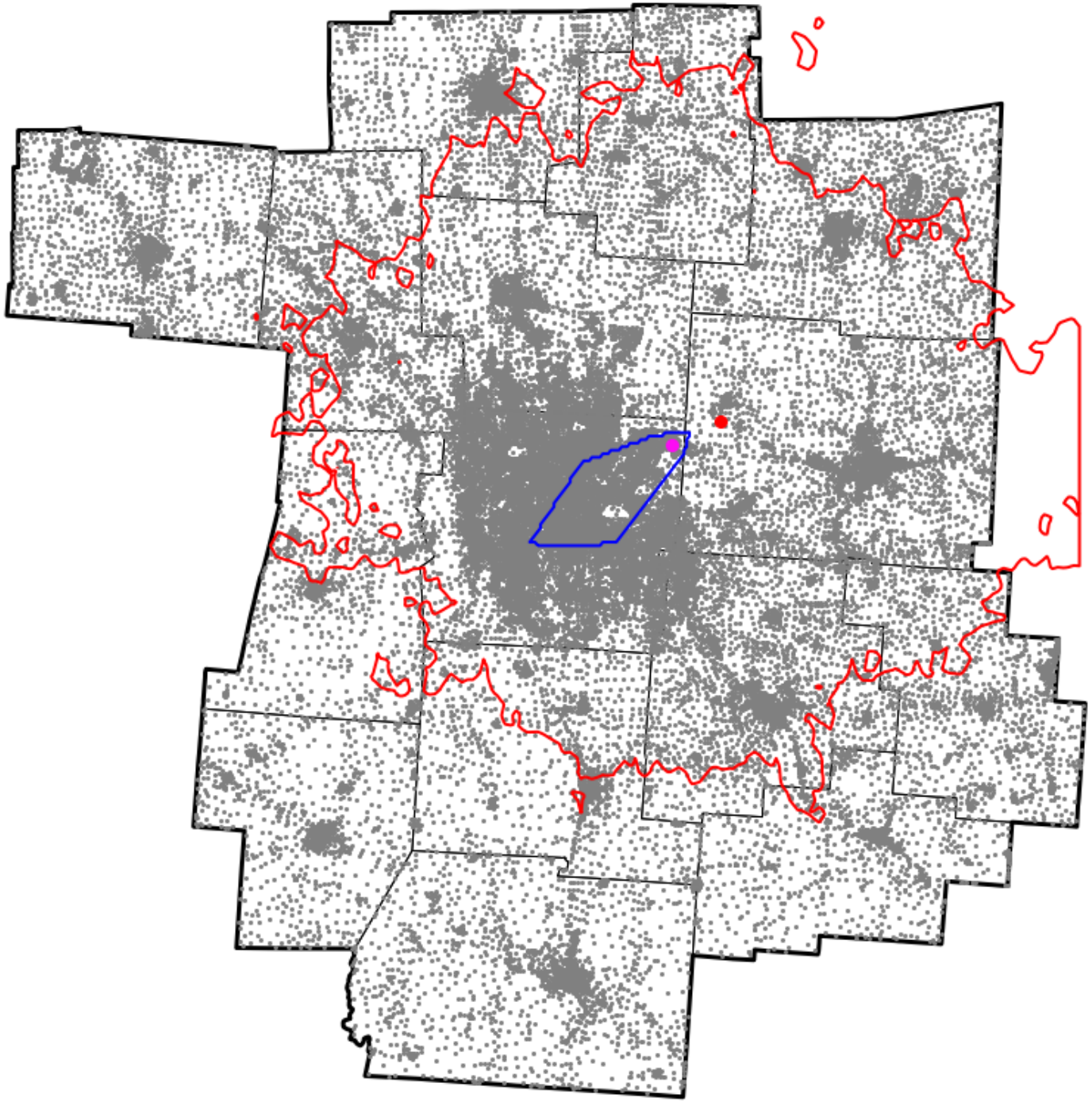


Bike

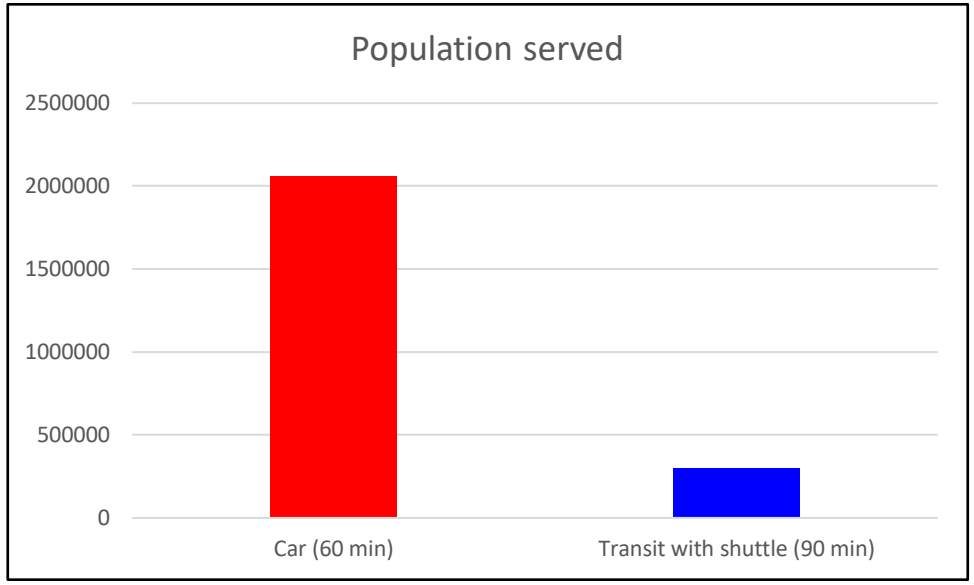


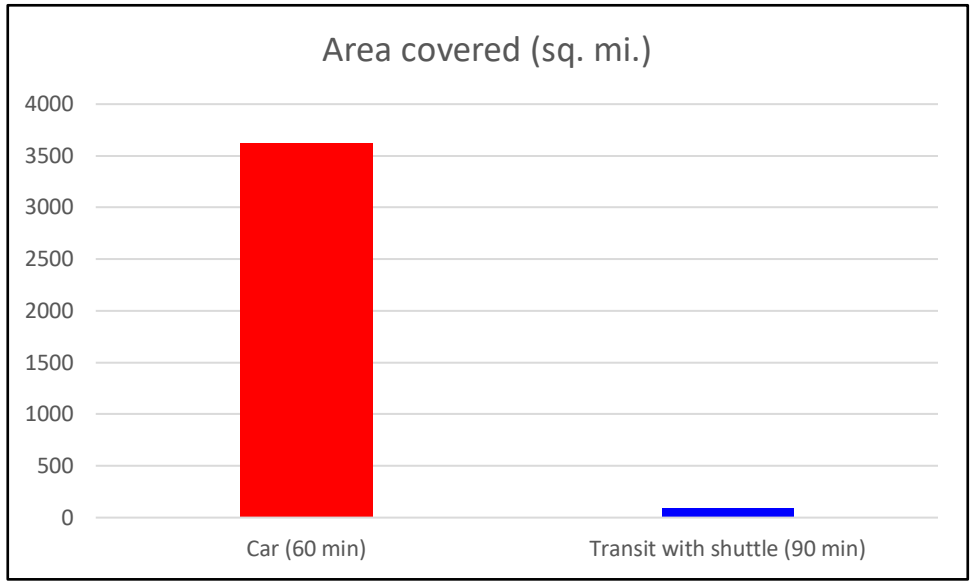
Shuttle + transit

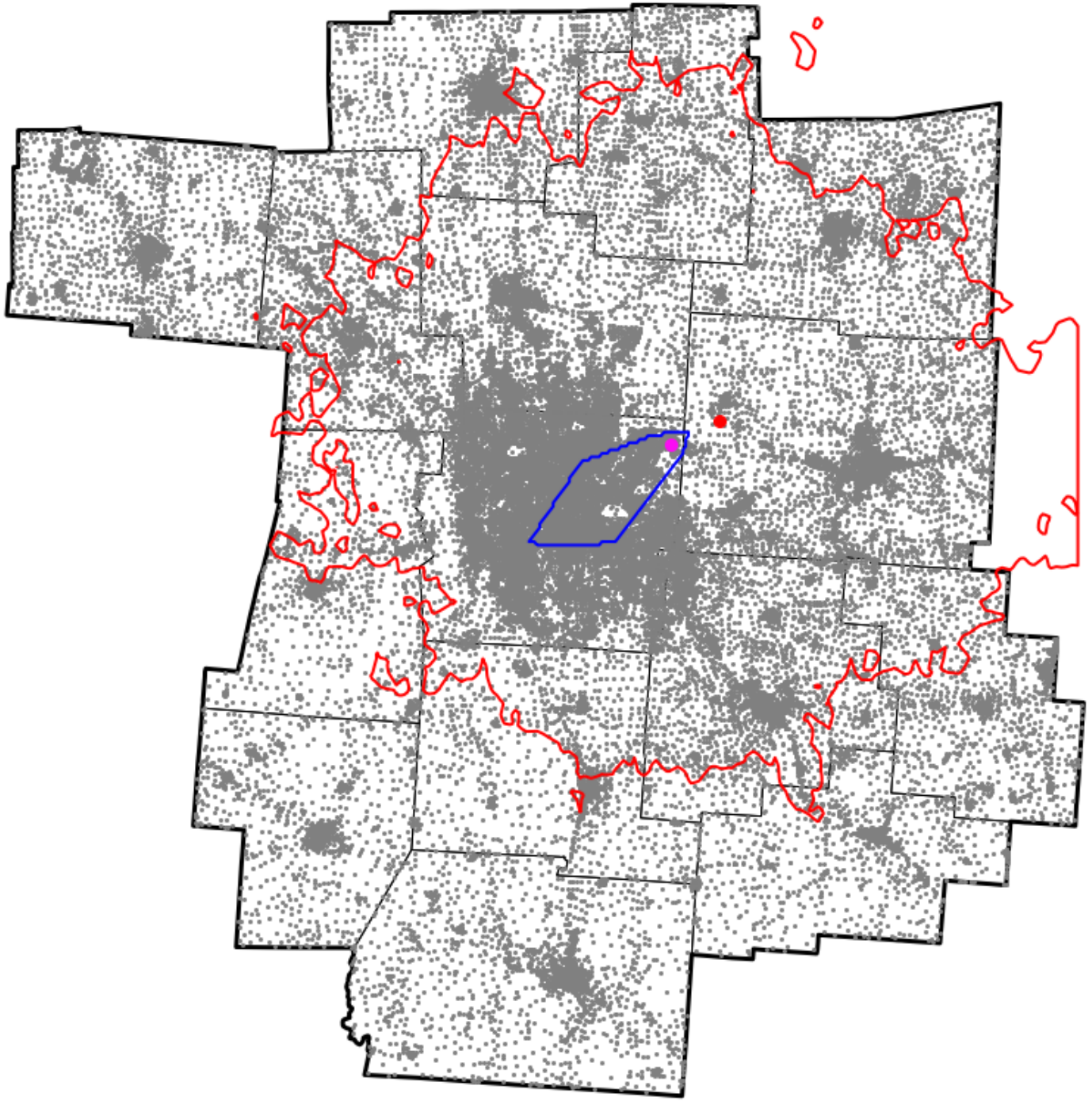
Example – Accessibility analysis



Example – Accessibility analysis

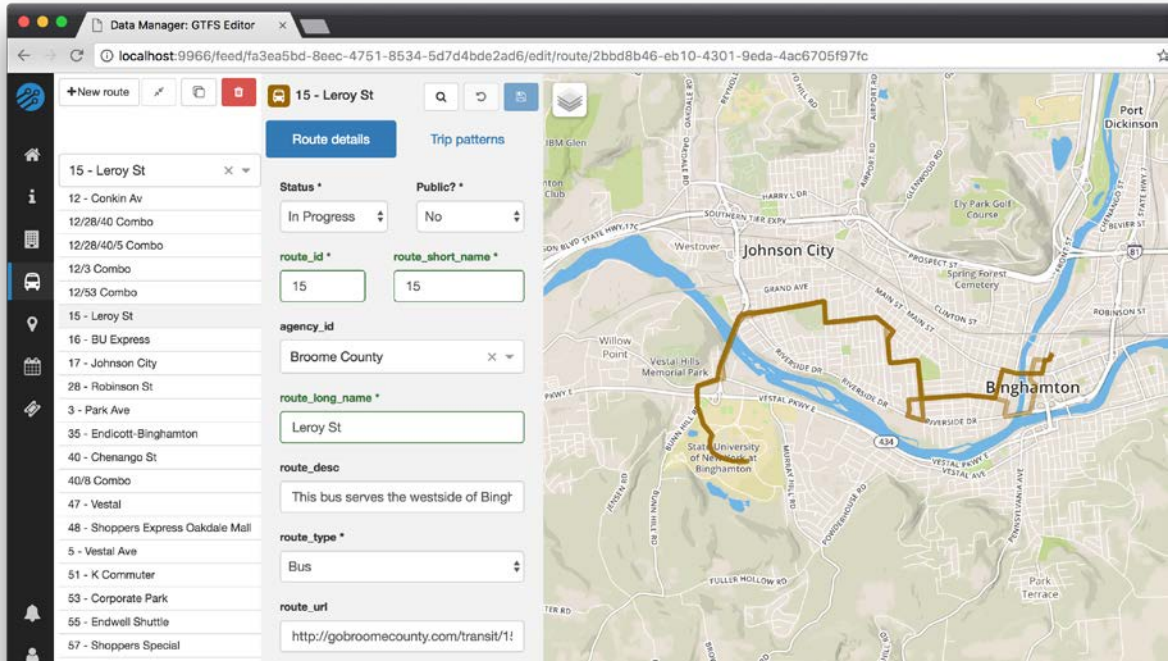






What if we were to…

- …model a new COTA stop at the Intel site?
- …model a new fixed-route service to serve Licking County?
- …model a new bike path between the site and the New Albany park and ride and/or Johnstown?
- **…improve the OSM data?**
  - Feature completeness (esp. bike/ped)
  - Attribute completeness (Level of Traffic Stress)

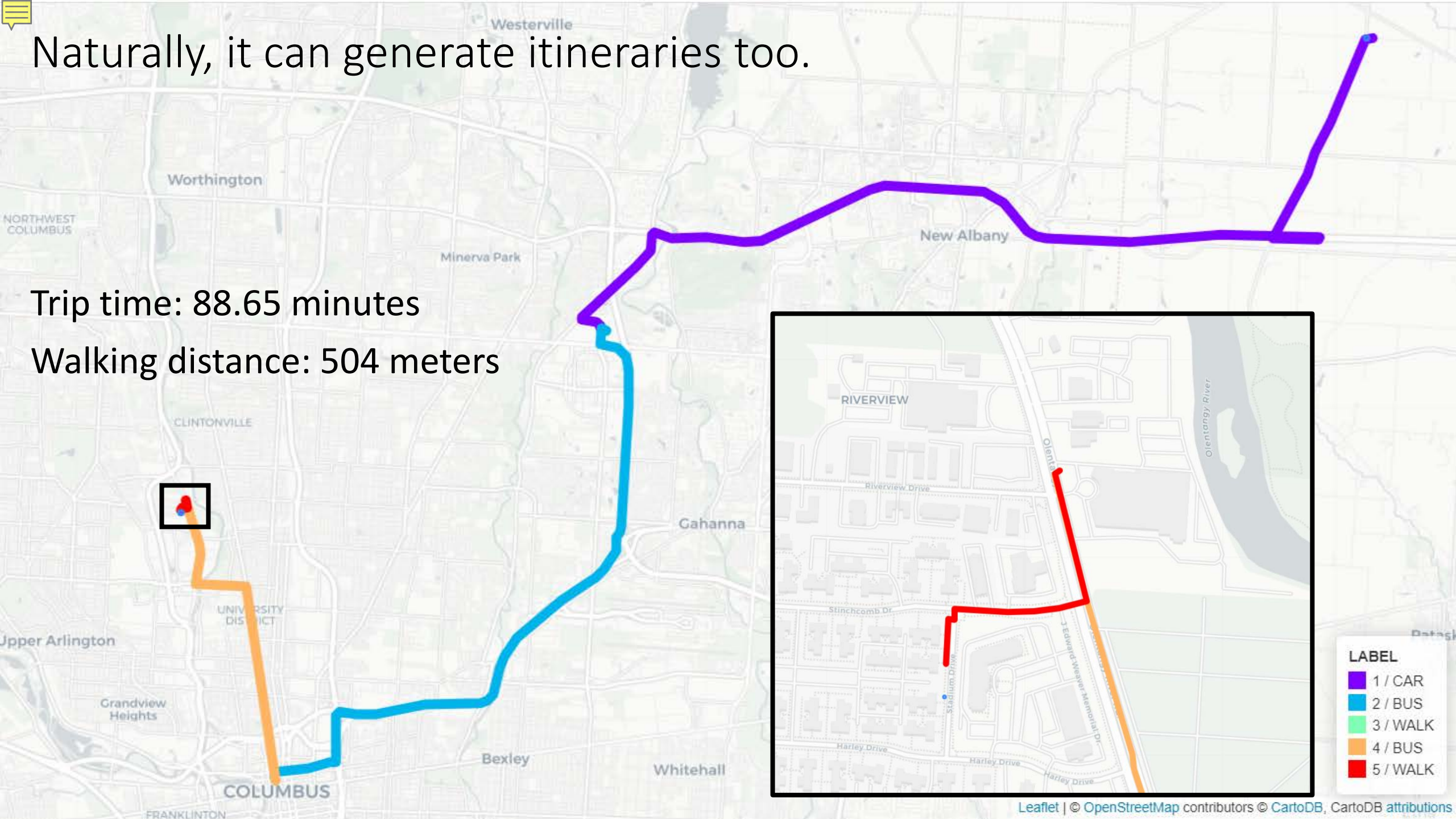


Source: https://data-tools-docs.ibi-transit.com

Naturally, it can generate itineraries too.

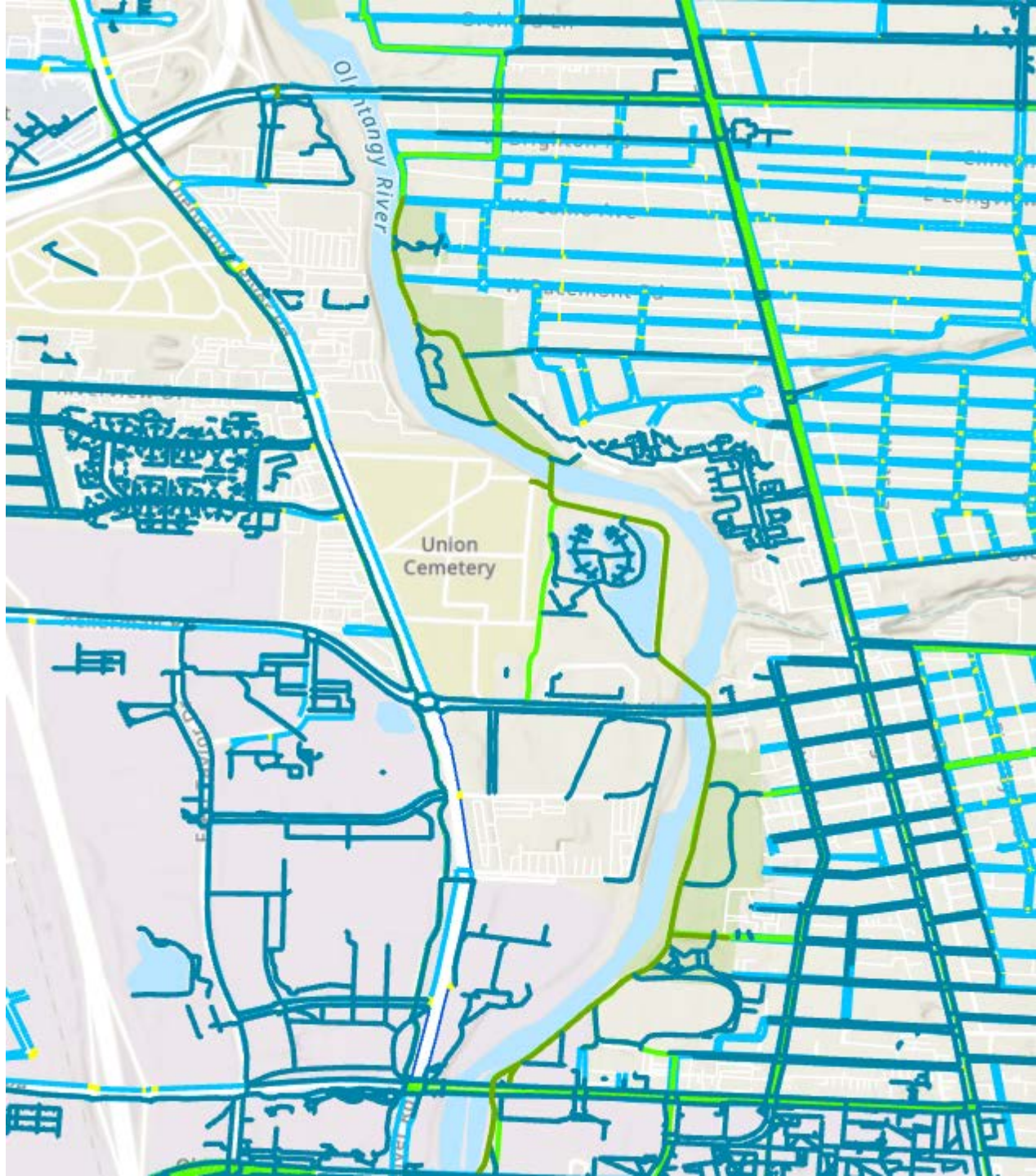
Trip time: 88.65 minutes

Walking distance: 504 meters

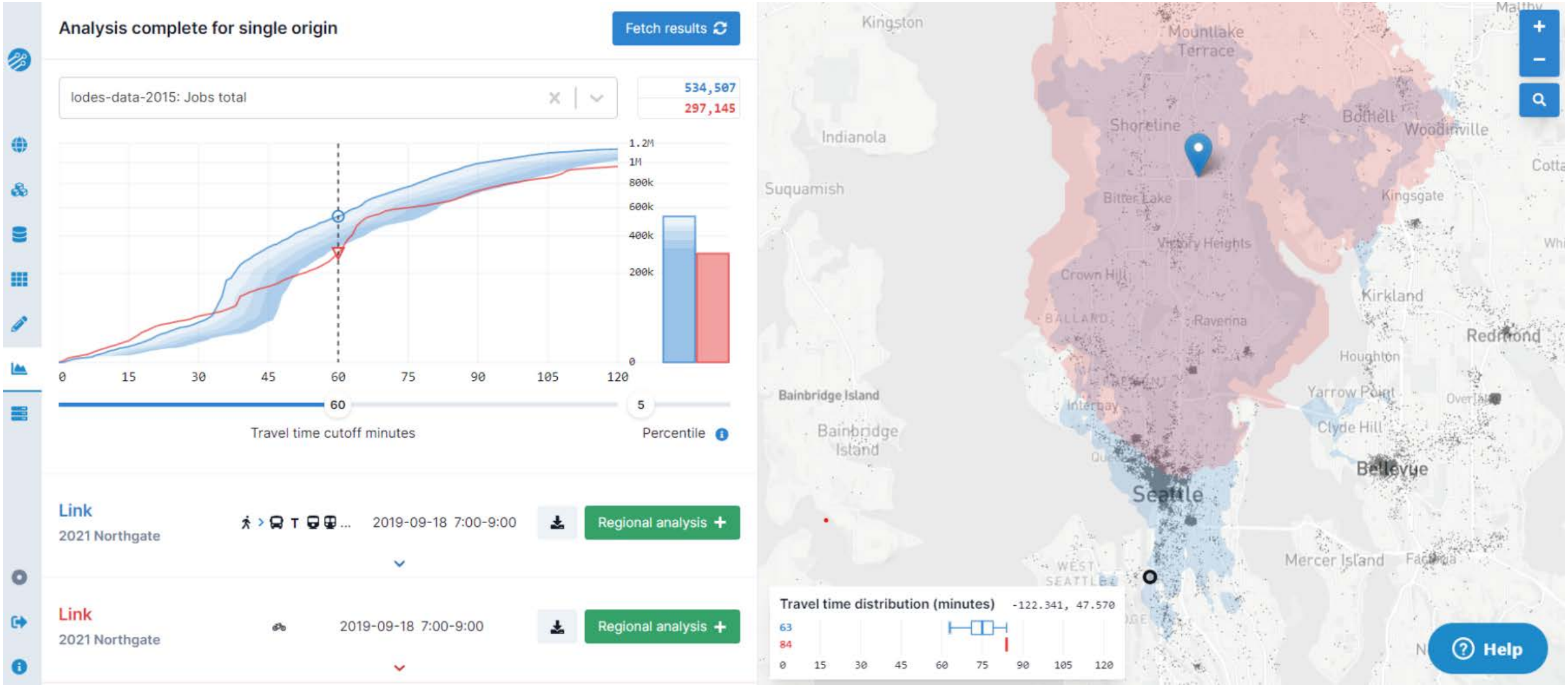


Lessons learned and caveats

- R5R doesn’t (yet) provide access to all features of R5 (e.g. custom ingress modes)
- Computation of inbound routes is much slower than outbound routes?
- Couldn’t get router to allow boarding at New Albany Park and Ride
- OpenStreetMap network isn’t perfect in Central Ohio…
  - Incomplete sidewalks
  - Incomplete greenways
  - No Level of Traffic Stress
- … and yet, many companies are selling or relying on OSM-based products!
- Need to fully understand or fine tune network behavior? Stick with ArcGIS (for now)
- There is a Python equivalent (r5py), but it doesn’t seem as mature or functional as r5r (yet).

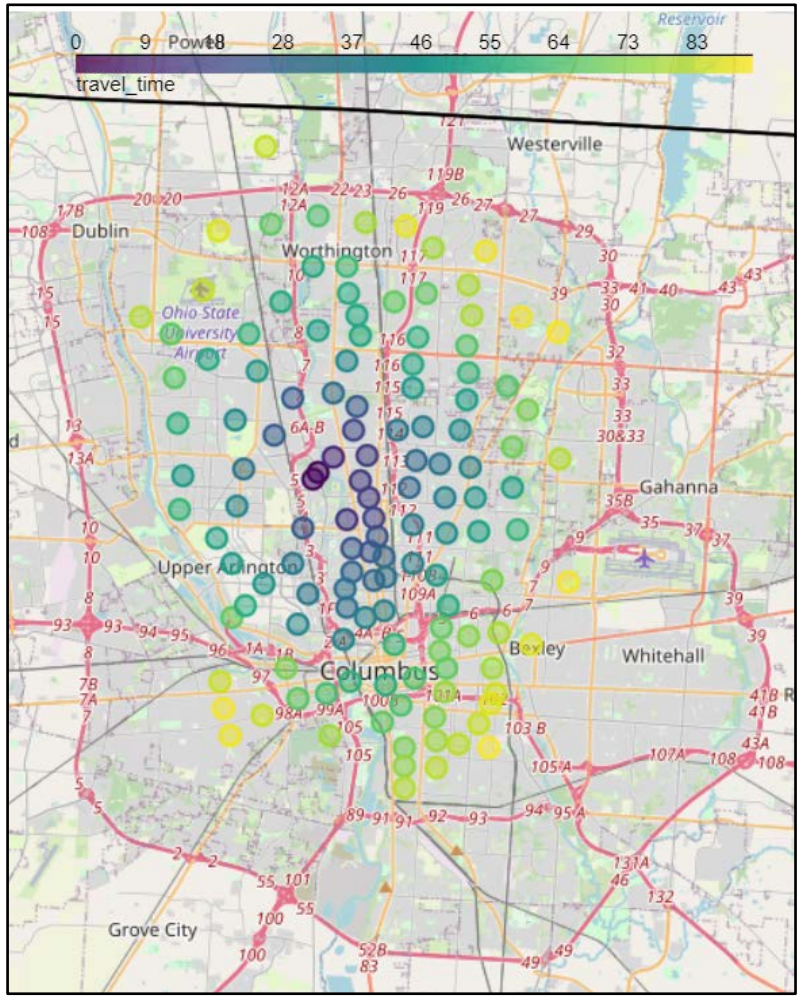


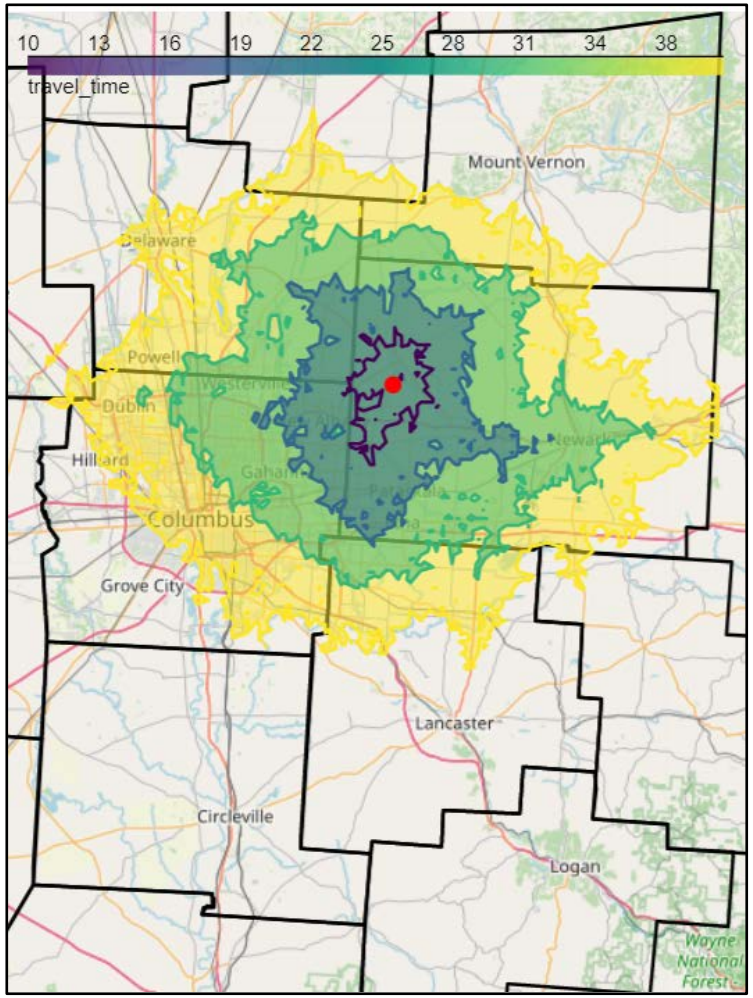
Does this sound like too much work? Try Conveyal!

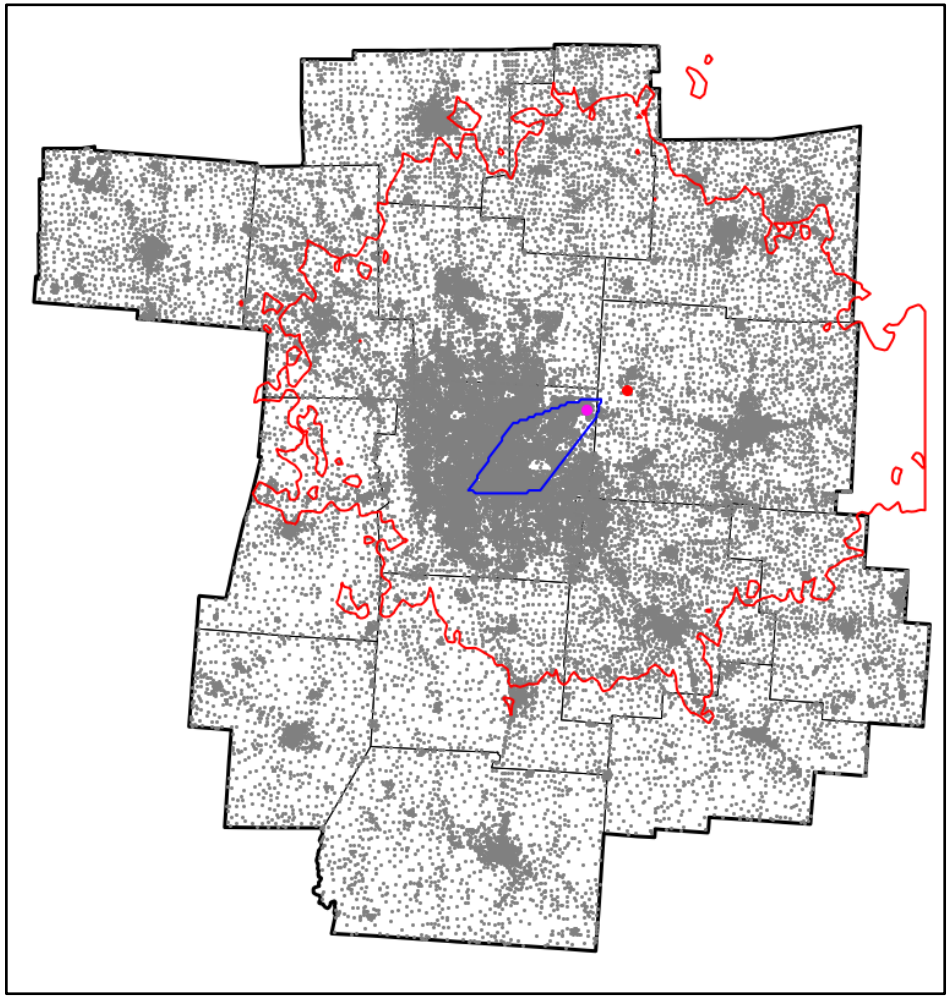


https://conveyal.com/

Thank you! Questions?







Source code: https://github.com/aporr/r5r-demo