The Dask at Hand

Using Dask to Speed Up the Production of CA Open Data

Tiffany Chu
Data Scientist @ Caltrans

What is Cal-ITP?

Managed by Caltrans for CalSTA, the California Integrated Travel Project (Cal-ITP) is a statewide initiative designed to standardize and organize transit in California.

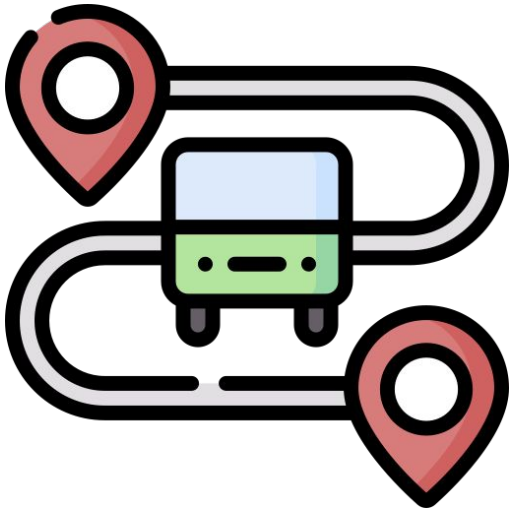
Standardizing Info for Trip Planning

Enabling Contactless Payments

Automating Customer Discounts

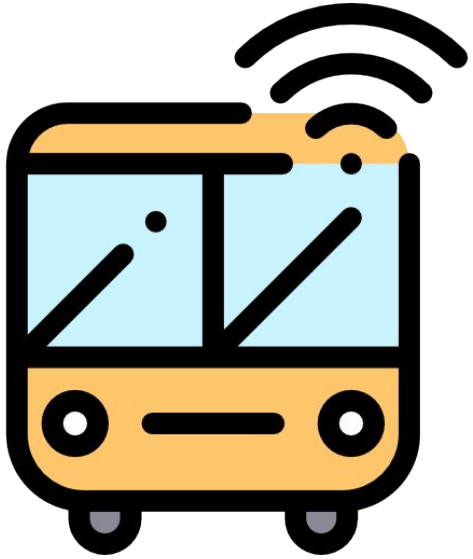
CAL ITP

General Transit Feed Specification (GTFS)



Schedule

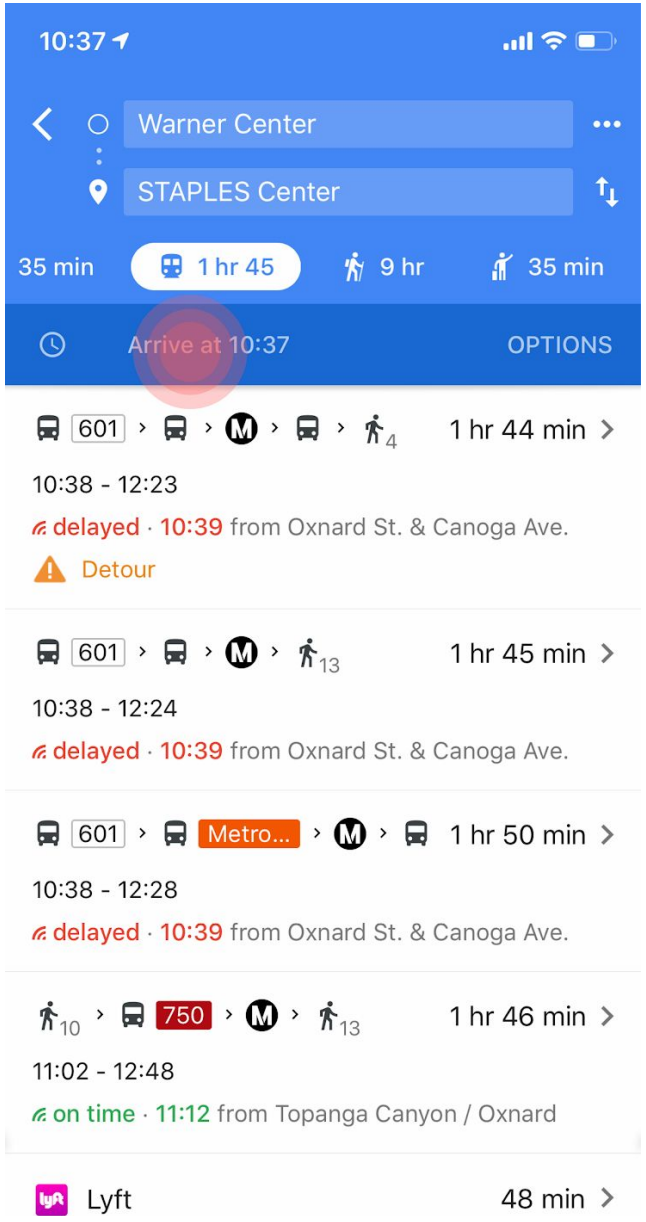
- service schedule
- fares
- geographic info

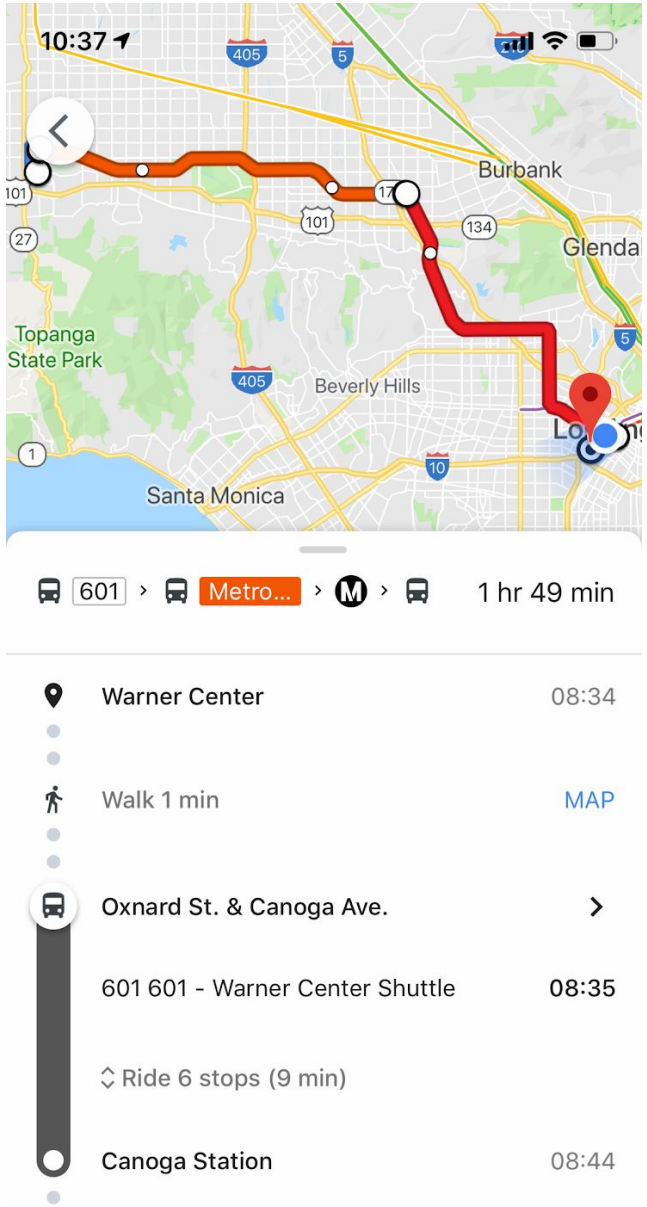


Real-Time

- arrival predictions
- vehicle positions
- service advisories

Public transport icons created by Freepik - Flaticon


10:37
Warner Center
STAPLES Center
35 min
1 hr 45
9 hr
35 min
Arrive at 10:37
OPTIONS
601
1 hr 44 min
10:38 - 12:23
delayed · 10:39 from Oxnard St. & Canoga Ave.
Detour
601
1 hr 45 min
10:38 - 12:24
delayed · 10:39 from Oxnard St. & Canoga Ave.
601
Metro...
1 hr 50 min
10:38 - 12:28
delayed · 10:39 from Oxnard St. & Canoga Ave.
750
1 hr 46 min
11:02 - 12:48
on time · 11:12 from Topanga Canyon / Oxnard
Lyft
48 min


10:37
Burbank
Glenda
Topanga State Park
Beverly Hills
Santa Monica
601
Metro...
1 hr 49 min
Warner Center
08:34
Walk 1 min
MAP
Oxnard St. & Canoga Ave.
601 601 - Warner Center Shuttle
08:35
Ride 6 stops (9 min)
Canoga Station
08:44

High Quality Transit Areas (HQTA)

Where are the high quality transit corridors & major transit stops in CA?

GTFS can help us identify these areas!

Why now, you dask?

Existing workflow

- Geospatial processing (Python)
- All CA transit operators (big!)
- Monthly open data portal updates (frequent)
- Sequential tasks

Dask solves this…

- Integrates with Python
- No memory issues
- Growing run times
  6 - 10 hrs / run / month
- Split sequential and parallelized

CAL ITP

Profiling Bottlenecks

- **Sequential vs parallelized**
- **Iterative rewrites - incorporate Dask's orchestration of tools**
  Loops sidestep memory issues, but imposes sequential framework

v1: 10 hrs		**v2: 5 hrs**		**v3: < 1.5 hrs**		**roadmap**
pandas dfs	→	dask ddfs	→	fewer loops	→	cloud cluster
make it work		*make it right*		*make it fast*		*make it faster*

Statute -> Code: high quality transit corridor

A **corridor** with fixed route bus	**segments**
service with **service intervals**	**stop with the max value**
no longer than 15 min	**4+ trips per hour**
during **peak commute hours**.	**AM: before 12 pm** **PM: 12 pm or after**

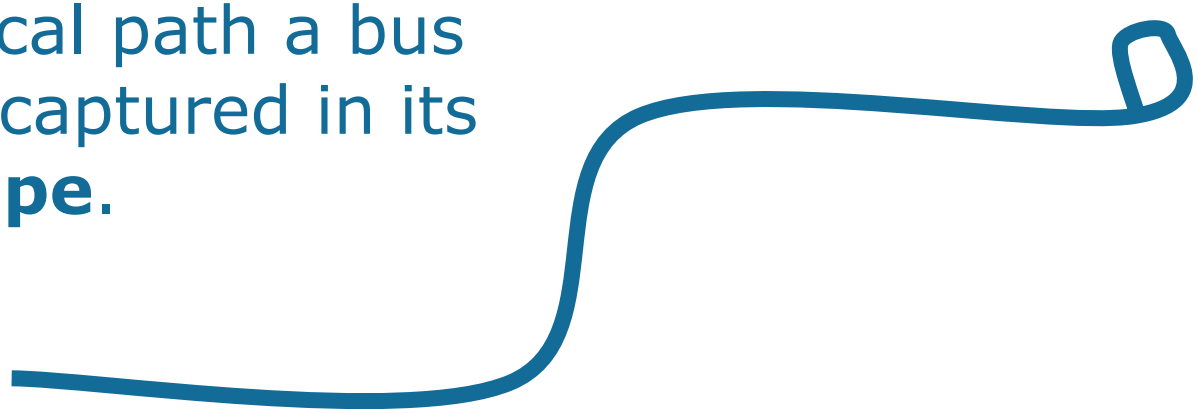
CAL ITP

1 - Full route network

The **route** has this physical coverage across all its **shapes.**

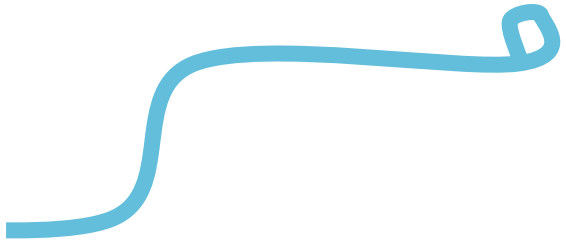
2 - Combine shapes to 1 route

The physical path a bus travels is captured in its GTFS **shape**.



eastbound

westbound



3 - Segment route into corridors

geopandas

- cut segments
- add segment's route direction

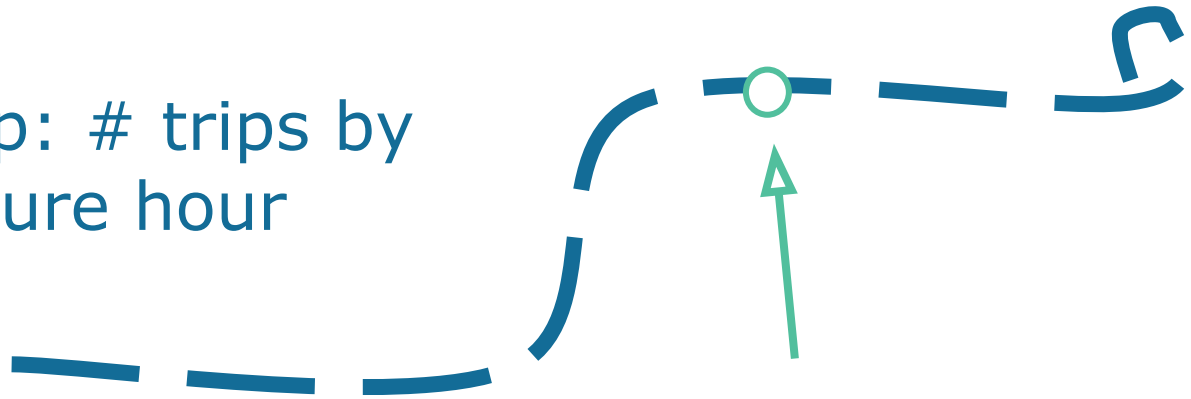
dask

- predominant route direction for the route

4 - Spatial join aggregated bus arrivals

dask

- by stop: # trips by departure hour

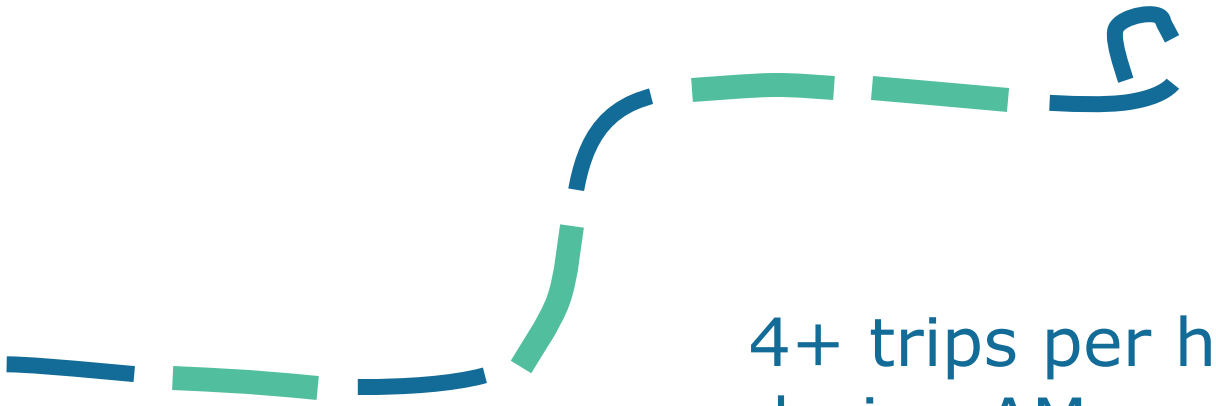


- by stop: max # trips in the AM / PM

dask_geopandas

- spatial join stops to segments
- highest value per segment

5 - Find high quality transit corridors



4+ trips per hour during AM and PM periods

Statute -> Code: major transit stop

The **intersection of 2+ major**

bus routes with…service interval

of **15 min or less** during the…

peak commute periods.

area of intersection between

2 high quality segments

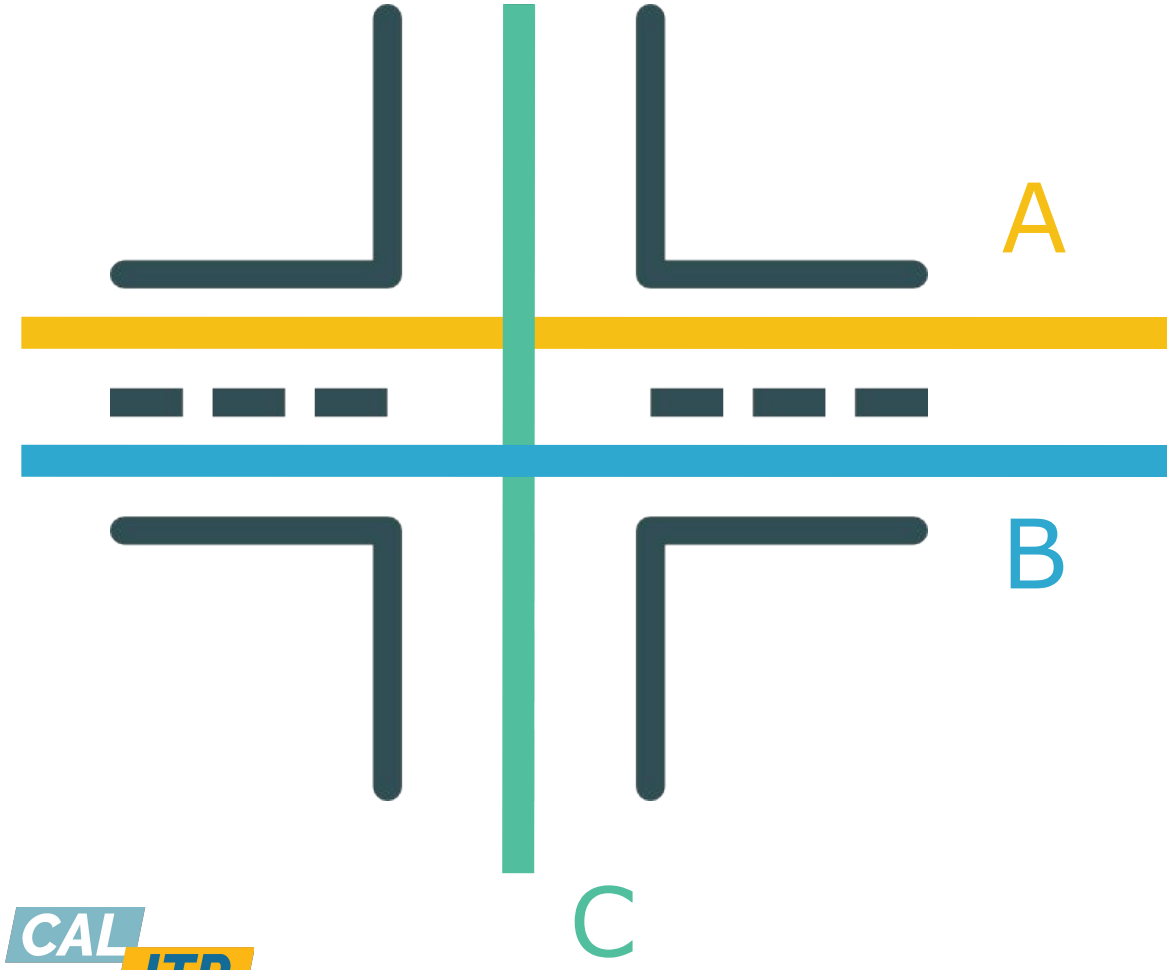
4+ trips per hour

AM: before 12 pm

PM: 12 pm or after

CAL ITP

What does *intersection* mean?



1 **Geography matters**

LA ≠ SF

2 **Direction matters**

A & **C**

B & **C**

Create a pairwise table

Route	Intersects…
Route A	Route C
Route B	Route C
Route X	Route Y
Route X	Route Z

dask_geopandas

- spatial join -> valid pairs (100k +)
- looping through operators to clip

smarter pre-processing:

- spatial join east-west to north-south
- pairwise table is set up for geopandas intersection

v1 - clipping each row (hours)

```
# Compare each row against all other rows to find possible
intersections, even if operators are in different geographic
areas

>> result = geopandas.clip(this_row, not_this_row)
```

v2 - add pairwise table (45 min)

```
# Loop by each operator (attempt to batch)

# across operators (factor in geography)
>> across_operators = dask_geopandas.sjoin(
                          this_operator, not_this_operator,
                          how = "inner",
                          predicate = "intersects"
                       )

# repeat within an operator (geography less of a factor)

>> result = dask_geopandas.clip(this_route, corresponding_pairs)
```

v3 - smarter use of pairwise table (1 min)

```
# Compare east-west segments to north-south segments

>> pairs_table = dask_geopandas.sjoin(
                    east_west, north_south,
                    how = "inner",
                    predicate = "intersects"
                  )

# repeat: compare north-south to east-west

>> results = pairs_table.geometry.intersection(
                 pairs_table.intersect_geometry,
                 align = True
              )
```

CA.GOV CALIFORNIA STATE GEOPORTAL

Filters

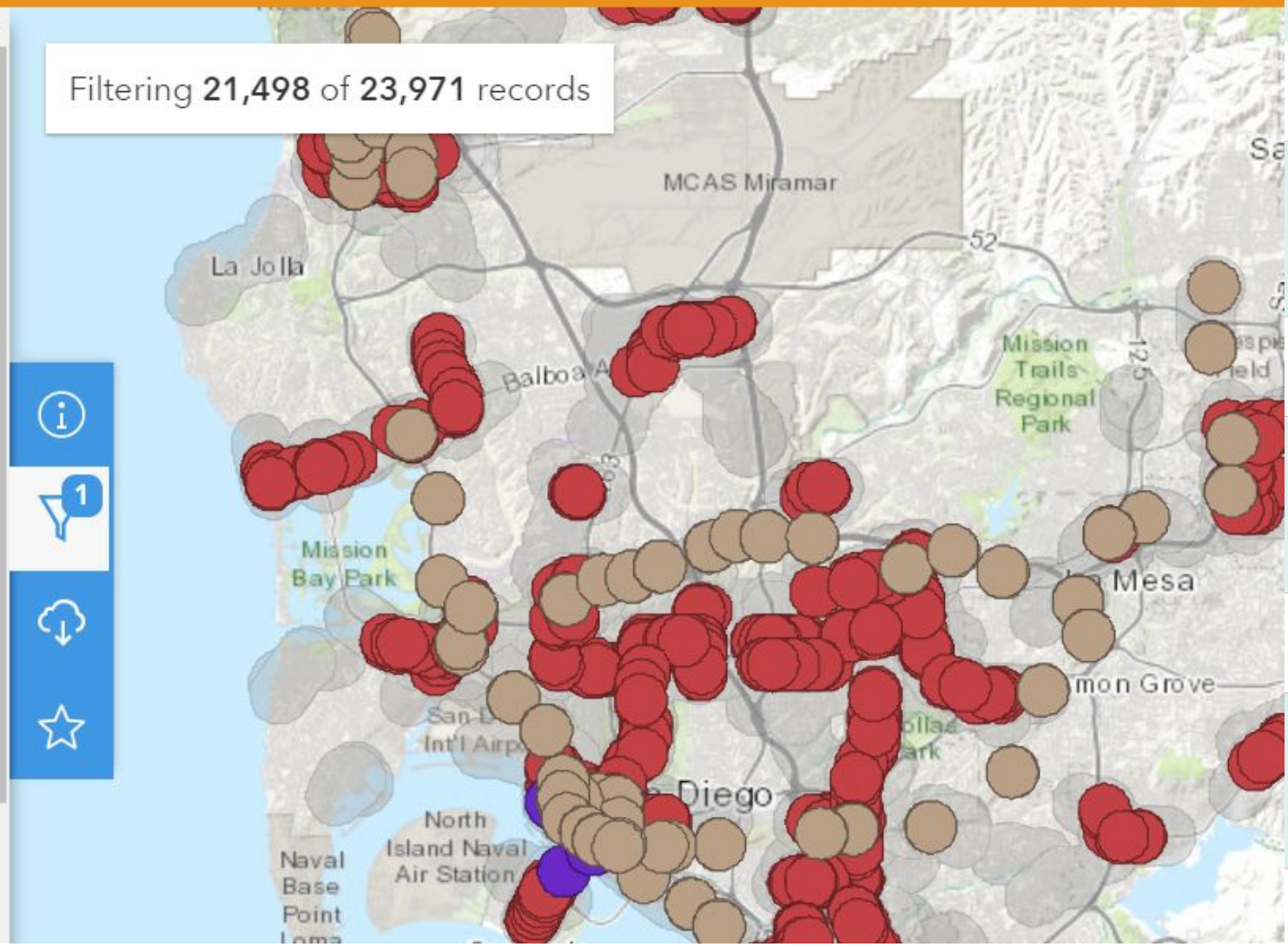
CA HQ Transit Areas

Filters | Styling

Filter as map moves

hqta_type

☑	major_stop_bus	83.86%
☐	hq_corridor_bus	9.45%
☑	major_stop_rail	5.74%
☐	major_stop_brt	0.86%
☑	major_stop_ferry	0.08%



CAL ITP

Lessons Learned

1. Use Dask to handle memory issues, not loops
2. Write…rewrite….to use more of Dask's tools
3. Get the most from pre-processing

Dask a wrap...
dask anyone have questions?

https://github.com/cal-itp/data-analyses

tiffany.chu@dot.ca.gov

hello@calitp.org

analysis.calitp.org

CAL ITP

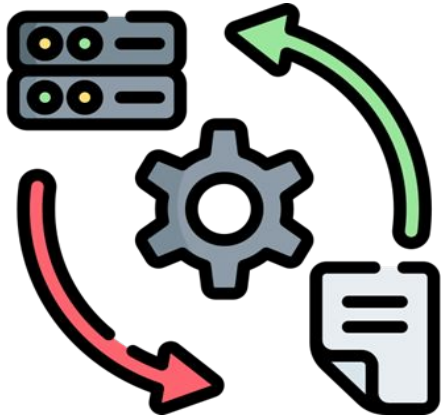
GTFS Analytics

Data Warehouse

Canonical
Reproducible Work

Analytics Pipeline

Long-term sustainability
Stable, reproducible data products



Insightful Analysis

Open data portal
Small team, big impact

Data warehouse, ETL, dashboard icons created by Flat Icons, Freepik, Eucalyptus - Flaticon