Streaming computation algorithms for spatiotemporal micromobility service availability

William Barbour*, Michael Wilbur, Ricardo Sandoval
Abhishek Dubey, Daniel B. Work

Vanderbilt University

* william.w.barbour@vanderbilt.edu

1

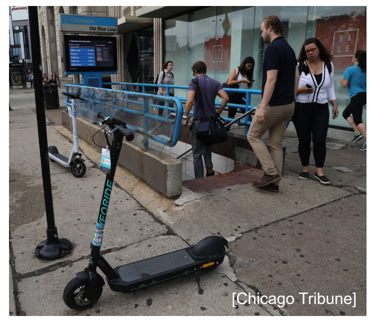
Outline

- Introduction: micromobility and transportation equity considerations
- Problem statement
- Methodology and algorithms
- Case study and results
- Conclusions and future work

2

Motivation: micromobility equity

- Transportation equity, generally, is one of the largest barriers to socioeconomic mobility.
  - Impacts ability of people to maintain and change jobs.
  - Major economic burden on personal finances.
- Micromobility designed to serve short trips (1-3 miles) without a car and can supplement public transit (first/last mile).
  - Docked bike systems (e.g., Citi bike, Divvy).
  - Dockless devices (e.g., Bird, Lime scooters).



[Chicago Tribune]

3

Motivation: micromobility equity

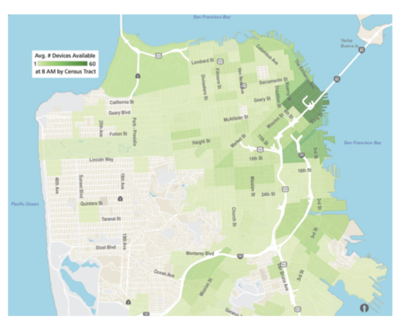
In a study of mid-sized cities with scooter share systems, 8 of 9 cities had equity requirements for operators.

- Device distribution requirements were the most common.
- These typically take the form of a minimum number of devices that must be distributed to pre-determined areas.

[Janssen et al., 2020]

Equity is difficult to measure.

- Distribution requirements do not fully capture availability to riders.
- Aggregated quantities can be biased in space and in time.



San Francisco MTA
Mid-Pilot Evaluation report, 2019

4

Problem statement

VANDERBILT

Compute exact micromobility equity for all points in a city, at all points in time.

- Eliminate spatial bias by disaggregating all locations across a city.
  - As opposed to aggregating at neighborhood level where availability can be highly non-uniform.
- Eliminate temporal bias by computing availability across continuous time.
  - As opposed to looking only at morning device distribution or computing periodically.

5

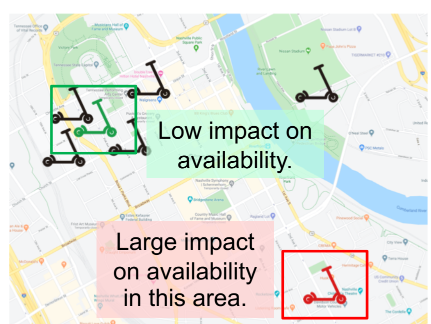
Methodology: streaming updates

VANDERBILT

Define equity: use “availability distance” as a proxy – distance from a given location to the nearest available device.

Develop algorithms: process each streaming availability update on network graph quickly in real time.

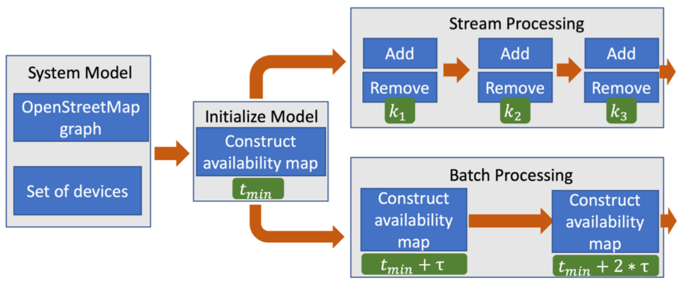
- Updates consist of **addition** or **removal** of a device from the list of available devices.
- Each update changes the availability distance of points only in a small area around it.



6

Methodology: computation paradigms

1. Traditional: run full computation every τ minutes, disregard changes in between. (a.k.a., batch, periodic)
2. Streaming: process each model update in real time; model is always up-to-date.

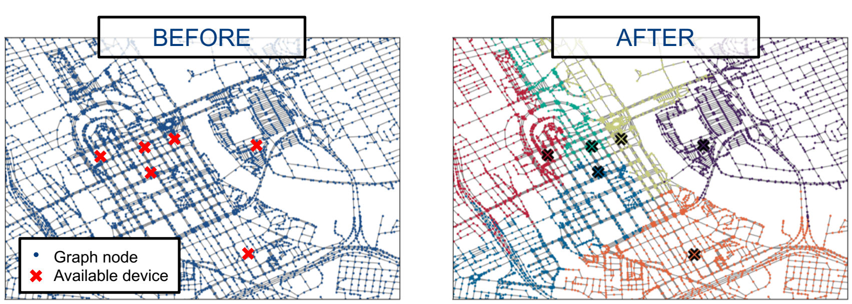


7

Methodology: algorithms

Full mapping of device availability

- Input: locations and IDs of available devices at time *t*.
- Map each available device to nearest node using Eucl. distance.
- Shortest path Dijkstra's from device nodes to all other nodes.
- Output: for each node, distance to available device and its ID.



Methodology: algorithms

Add
available device

Input: device ID to add to availability.

- Determine nearest node by Eucl. distance.
- If 2nd available device at its node, put in availability queue at node and STOP.
- Otherwise, begin shortest path Dijkstra from nearest node; terminate active route explorations when new cost exceeds existing cost (availability distance to another node).

Output: updates to availability values for affected nodes.

BEFORE

AFTER

Methodology: algorithms

Remove
available device

Input: device ID to remove from availability.

- If 2nd available device at its node, change nearest ID refs. on nodes and STOP.
- Otherwise, determine nodes with ref. to removed device = unassigned.
- Determine set of *assigned* nodes that border unassigned nodes = boundary.
- Shortest path Dijkstra from boundary to all unassigned; begin route with cost of availability distance at boundary node.

Output: updates to availability values for affected nodes.

BEFORE

AFTER

Methodology: benchmark comparison

VANDERBILT

Streaming availability updates

- Initialize availability map with full construction once at the start of the day.
- Process each availability update (add or remove device) when it occurs.
- Localized availability changes.

Periodic availability calculation

- Run a new availability map construction every 5 minutes.
  - Can simulate larger period (e.g., 15 minutes) by subselect.
- Does not consider any changes in availability between constructions.

Compare execution time of streaming update procedure to periodic full construction.

In resource-constrained computing, what is the equivalent periodic computation interval to streaming updates?

11

Case study

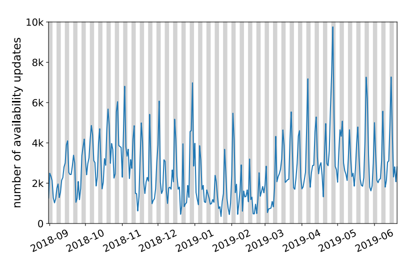
VANDERBILT

Nashville dockless electric scooter operations.

- 14 weeks between March 1 and June 7, 2019.
- 10 hour operating days between 9:00am and 7:00pm.
- Average of 4,000 availability updates per day.

Road network as 3 graphs of increasing resolution.

- Generated from OpenStreetMap data.
- Edges segmented to max 300-meter, 100m, and 25m lengths.
- Graphs have 81,000, 172k, and 293k nodes, respectively.



12

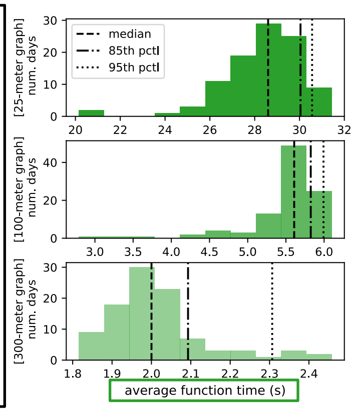
Results: algorithm timing

VANDERBILT

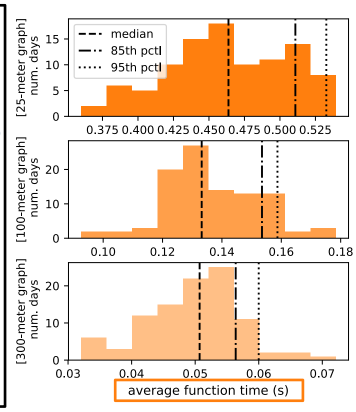
Compare execution time of streaming updates to full construction.

- Availability updates computed in 2% the time of full map compute.
- Process 50 updates in equivalent time as one full map compute.
- Advantage slightly greater with higher-resolution graph.

Full availability map construction



Add/Remove streaming updates



13

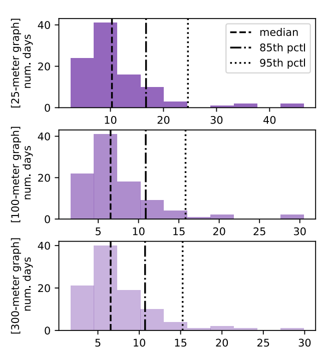
Results: periodic computation interval

VANDERBILT

How often could the full construction algorithm be run in the equivalent computation time of streaming updates?

- Based on average 10-hour day with 4,000 availability updates.
- For 300m and 100m graphs, full construction every 6.5 minutes is equivalent computation time to all updates for the day.
- For 25m graph, full construction could run every 10.3 minutes.
- 95th percentile interval values are over 2x higher.

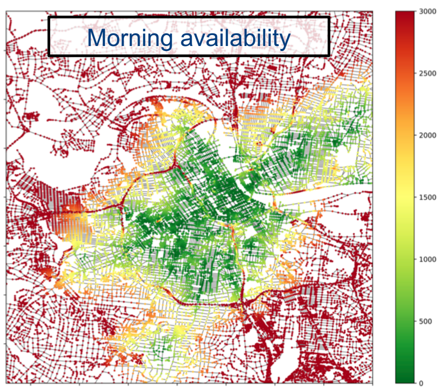
	Compute budget (s)	Function equivalence interval (minutes)		
Graph	**Median**	**Median**	**85th pctl.**	**95th pctl.**
25-m.	1658	10.3	16.7	24.6
100-m.	506	6.5	10.9	15.8
300-m.	183	6.5	10.7	15.2

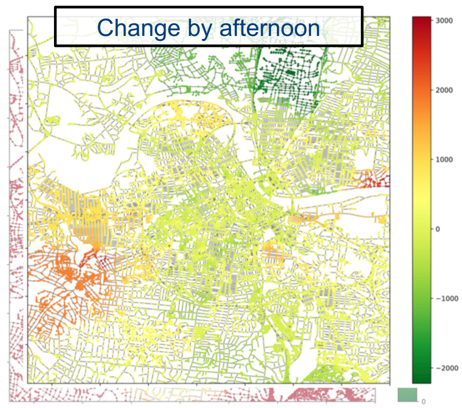


14

Results: mapping

- Streaming updates inherently computes availability for every time of the day.
- Can see how availability evolves and the underlying cause.

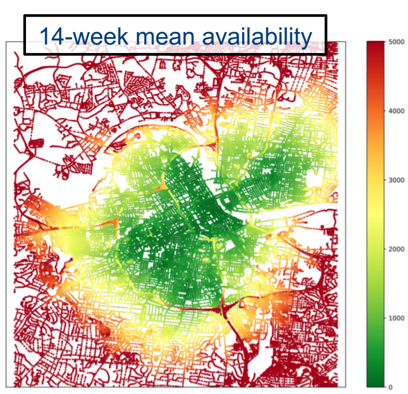


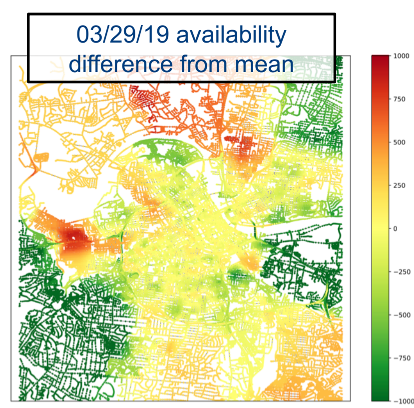


15

Results: mapping

- 14-week mean availability map considering every update computes in 13 hours.
  - Lossless computation using temporal-regularized average.
- Daily difference from mean highlights atypical availability.





16

Conclusions

- Localized streaming updates compute 50x faster than full map computation.
- In equivalent execution time, full construction could run only every 15-25 minutes (95th percentile).
- Streaming updates result in zero temporal error for availability values because of real-time processing.
- Up-to-date and responsive availability calculation allows operators and/or cities to respond quickly to device distribution equity considerations.
- *Future work: analyze accuracy tradeoff between streaming and periodic computation.*
- *Future work: variance and sensitivity of availability.*

Special thanks to Metro Nashville for support, advice, and data sharing.

17

Streaming computation algorithms for spatiotemporal micromobility service availability

William Barbour*, Michael Wilbur, Ricardo Sandoval
Abhishek Dubey, Daniel B. Work

Vanderbilt University

* william.w.barbour@vanderbilt.edu

18