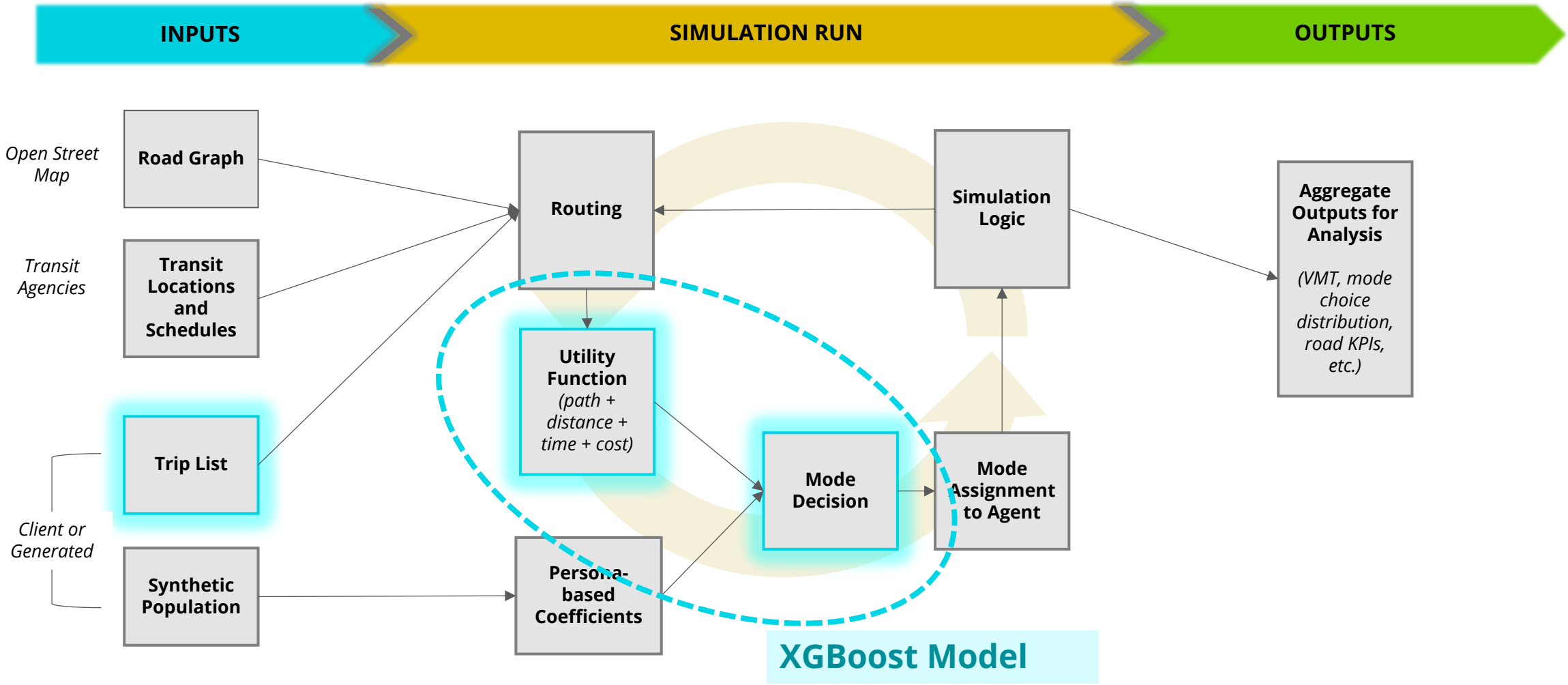
Travel Simulation Process Flow



How do we model mode choice?

Let's say we have a person, and **that person wants to go from Point A to Point B.**

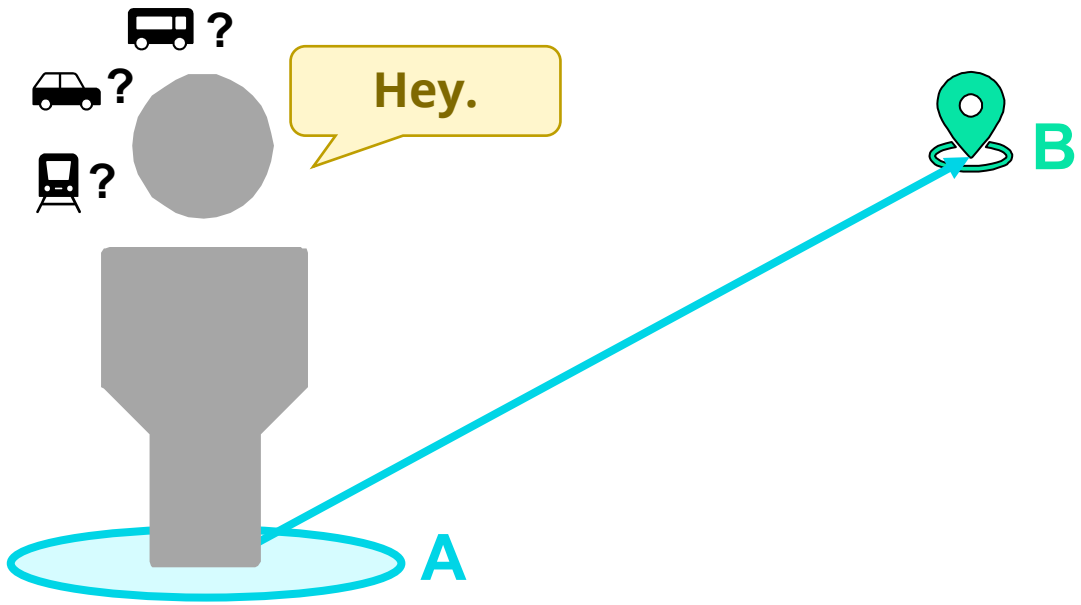
Our guy has a set of options for *how* he gets from A to B.

For any mode i we can express the utility of the mode for this trip as:

$$U_i = V_i + \varepsilon$$

Where V_i can be broken down into:

$$\begin{aligned}\text{Utility}_{\text{Transit}} &= a * \text{in-vehicle time} \\ &+ b * \text{fare} \\ &+ c * (\text{access time} + \text{egress time}) \\ &+ d * \text{wait time} \\ &+ \text{mode-specific constant}\end{aligned}$$

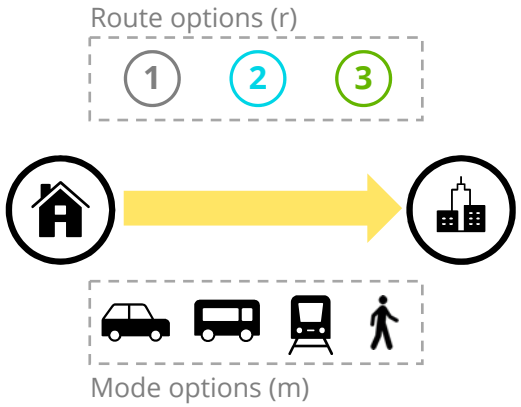


The complete process

Person X wants to go from Home to Work. What route and mode do they choose?

Male
50
$$

Value of time (VOT): **15**



ROUTE DETAILS

$$(a(time) + b(distance) + c(cost) + d(value\ of\ time)) * mode\ specific\ term$$

PERSONA-BASED COEFFICIENTS

$$(VOT_X(time_{r,m}) + VOT_X(distance_{r,m}) + PrSen_X(cost_{r,m}) + Pref_X(route_{r,m})) * mode\ term$$

Cost of time	Cost of distance (effort)	Cost of trip (parking, tolls, etc.)	Cost related to socioeconomic factors	Constant to represent intangibles

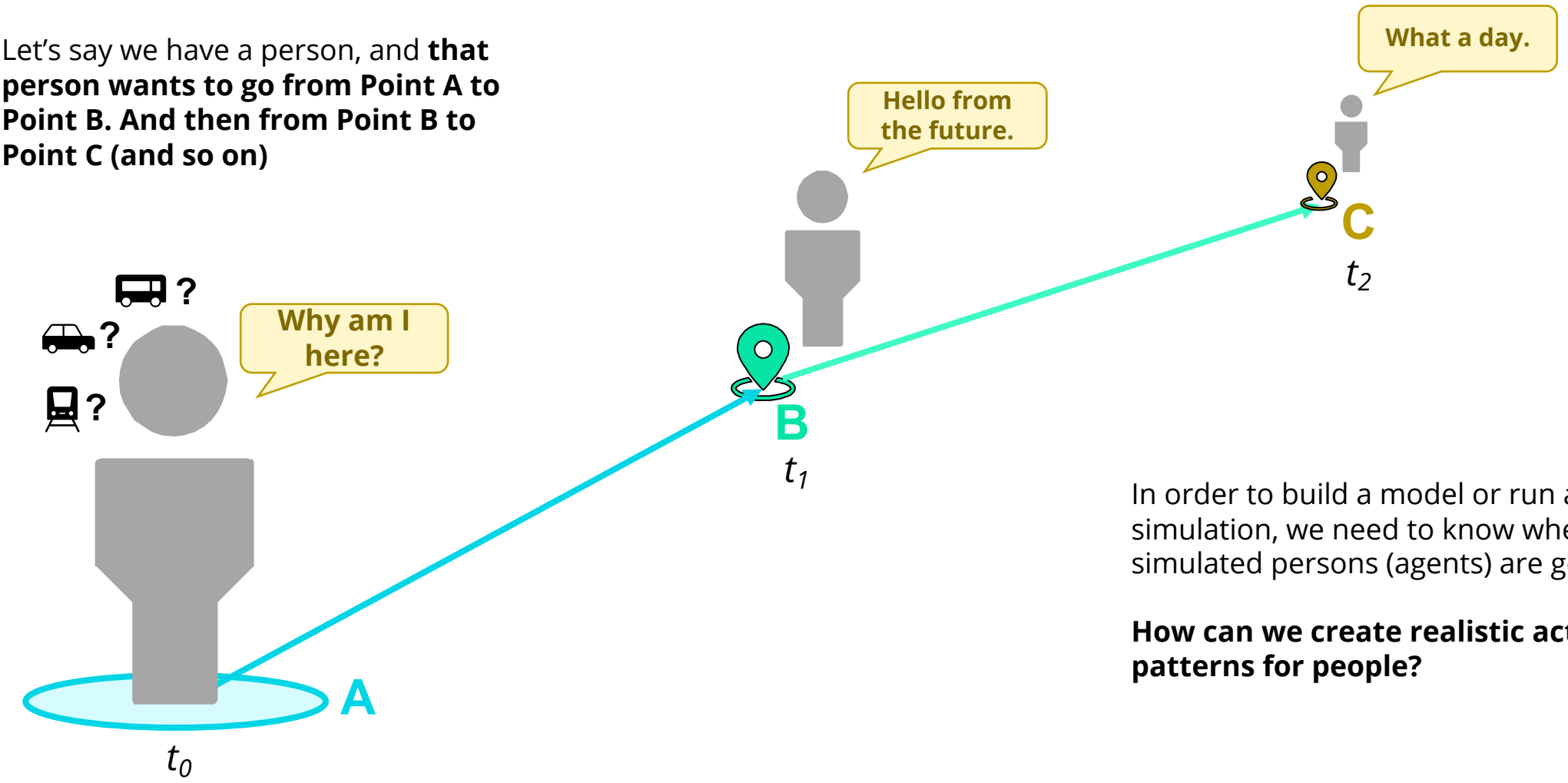
Monte Carlo to make final choice

$$\{ U_{1,car},\ U_{1,bus},\ U_{1,train},\ \ldots,\ U_{3,train},\ U_{3,walk} \}$$

Final choice of mode for trip

How do we model activity patterns?

Let's say we have a person, and **that person wants to go from Point A to Point B. And then from Point B to Point C (and so on)**



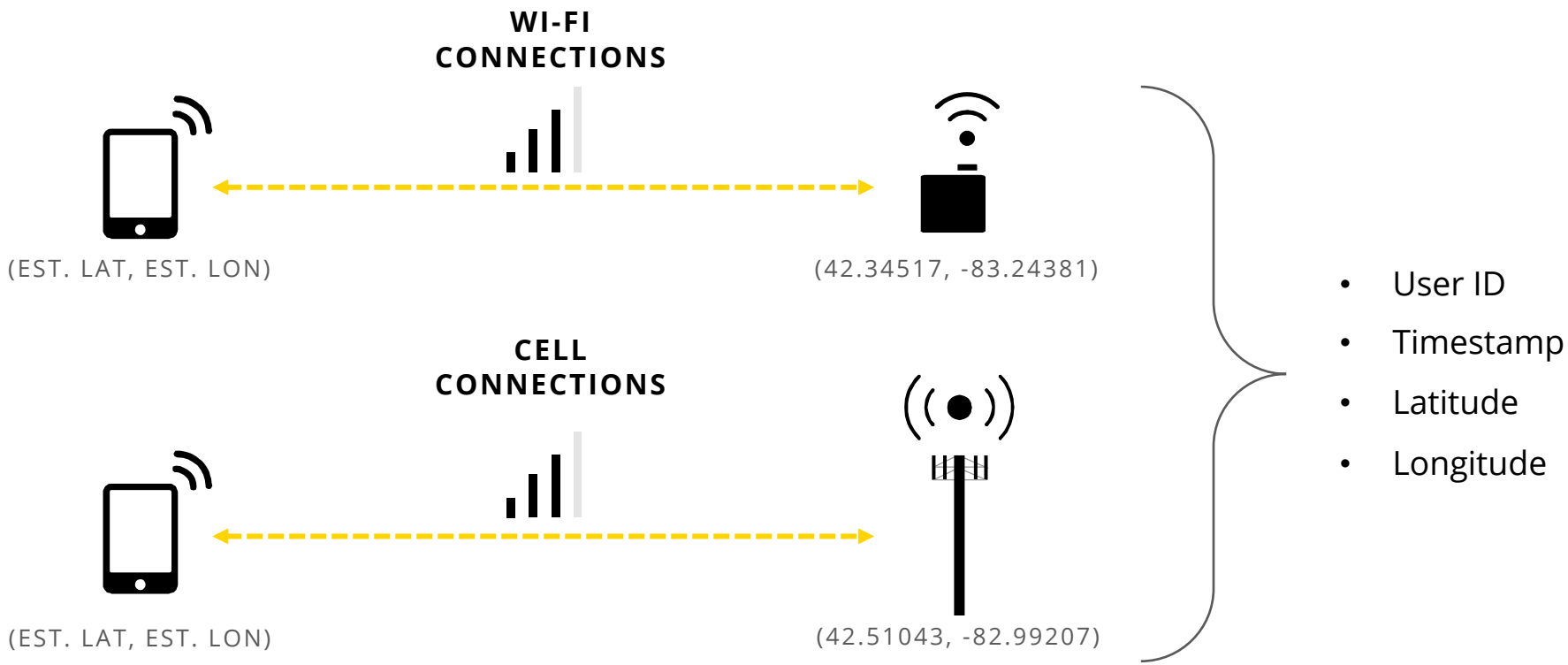
In order to build a model or run a simulation, we need to know where the simulated persons (agents) are going.

How can we create realistic activity patterns for people?

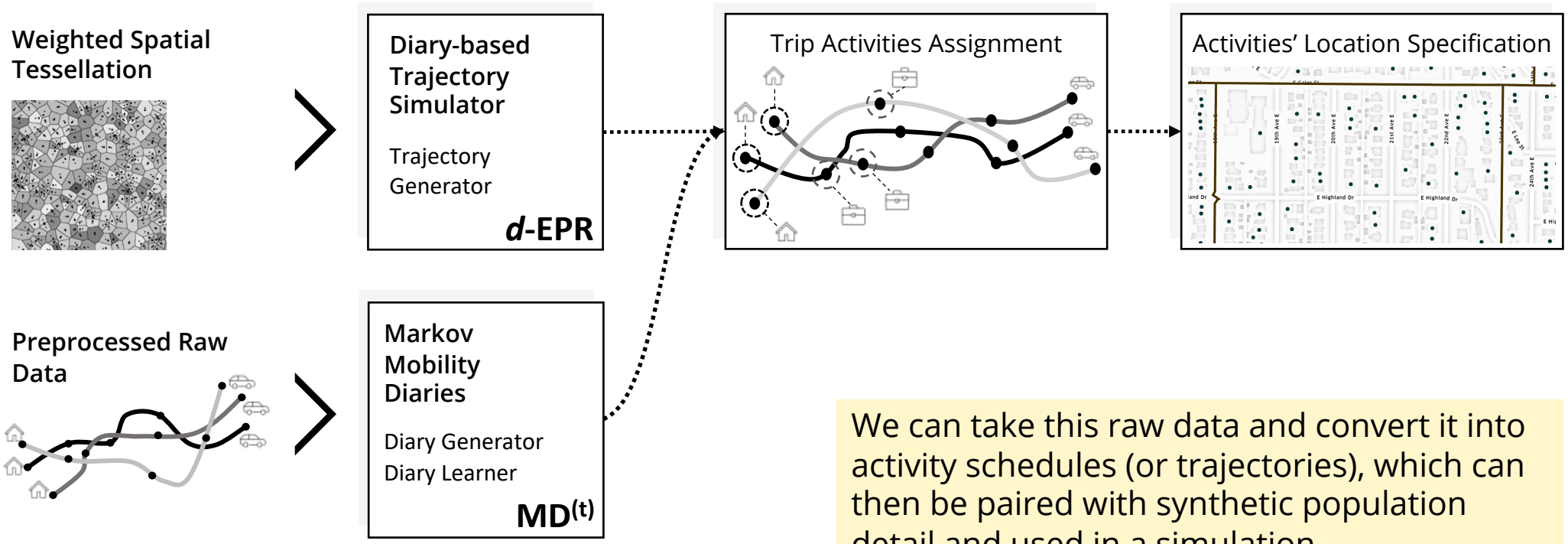
Using data from mobile devices for trajectory modeling

By default, mobile devices capture information about users including apps used, links clicked, locations, etc.

This data can be captured by a number of third parties and then packaged up and resold for advertising or location intelligence purposes.



Using data from mobile devices for trajectory modeling

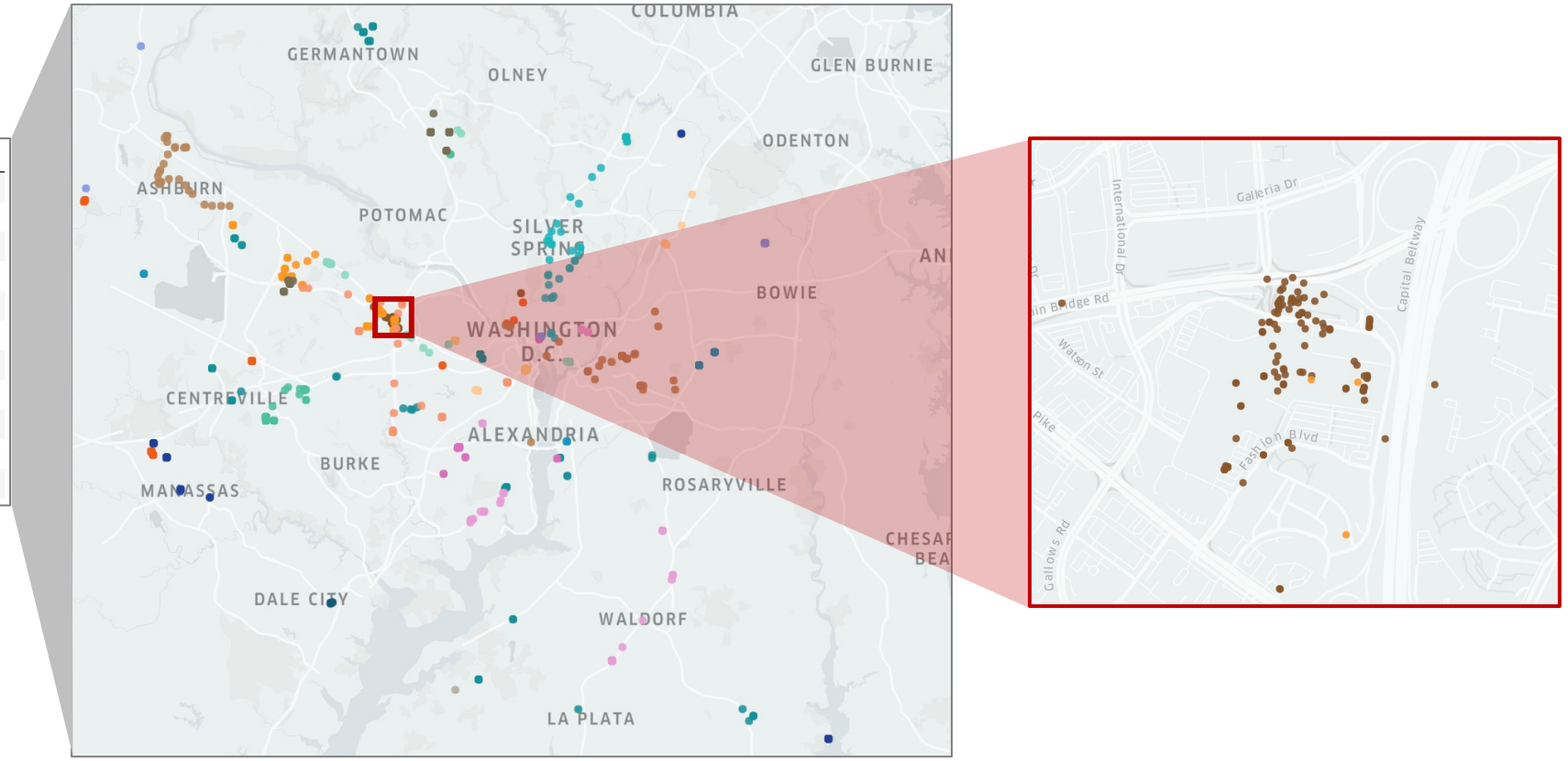


We can take this raw data and convert it into activity schedules (or trajectories), which can then be paired with synthetic population detail and used in a simulation.

Scikit Mobility: https://scikit-mobility.github.io/scikit-mobility/index.html

Using data from mobile devices for trajectory modeling

	device_id_value	latitude	longitude	month	day_of_month	hour	minute
0	0	38.884300	-77.112790	1	11	10	40
1	0	38.884307	-77.112792	1	11	10	41
2	0	38.884307	-77.112790	1	11	10	42
3	0	38.884300	-77.112799	1	11	10	43
4	0	38.884301	-77.112801	1	11	10	44
...	...	...	...	...	...	...	...
5433618	393024	38.846846	-77.039092	1	29	3	39
5433619	393025	39.427959	-77.527010	1	3	13	13
5433620	393025	39.427959	-77.527010	1	3	13	14
5433621	393025	39.427959	-77.527010	1	3	13	15
5433622	393025	39.427959	-77.527010	1	3	13	16



Identifying movement and non-movement

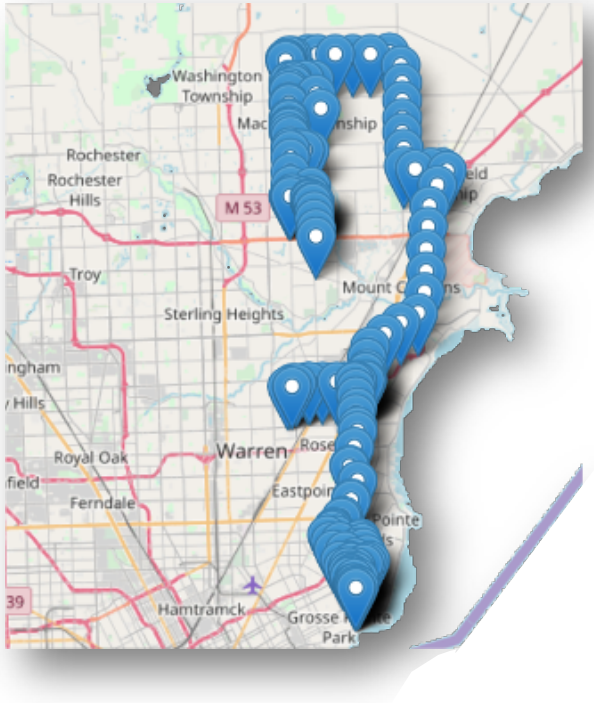
Challenge:

How can we identify locations and times when users were stationary?

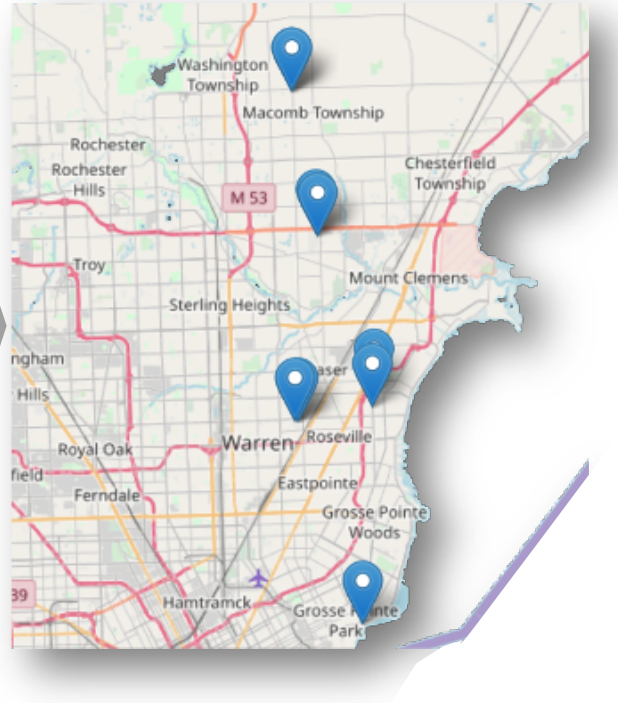
Solution Methodology:

- Focus on isolating periods of non-movement, known as stay locations
- Control for contingencies in movement data, such as short stops or slow movement over a period of check-ins
- Combine periods of non-movement to reduce data size by 85-90% (since many data points can exist during a single non-movement period)

RAW DATA SHOWING A USER'S TRAVEL PATH



STAY LOCATIONS DERIVED FROM THE SAME DATA



Identifying key user stay locations

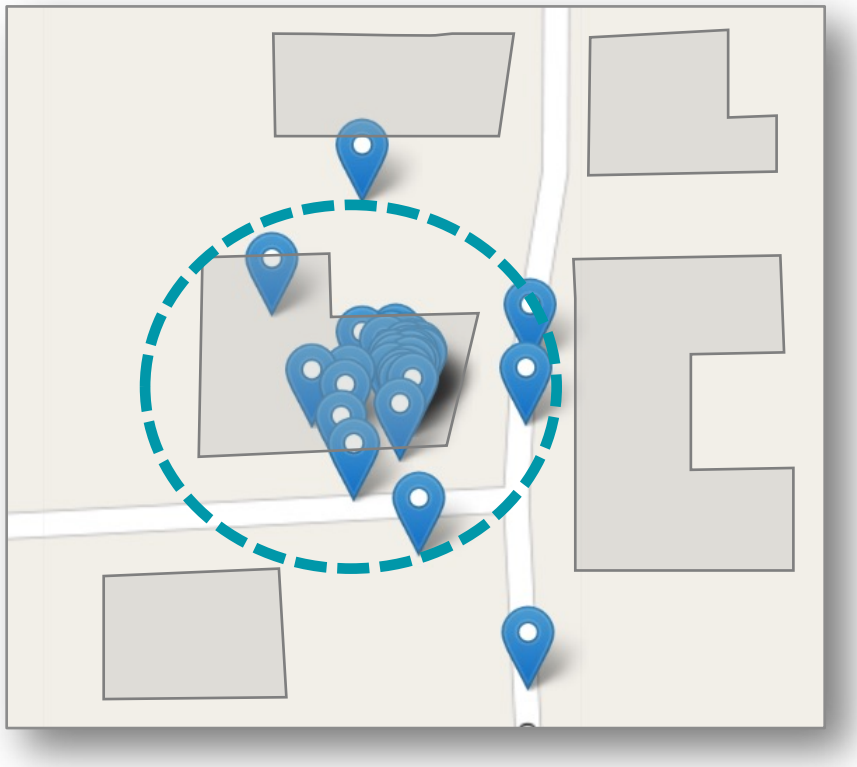
Challenge:

How can we group check-in points that appear separate but relate to the same location?

Solution Methodology:

- Use a **density-based clustering algorithm (e.g., DBSCAN)** to cluster multiple data points that reflect a single user location.
- Determine two classes of stationary behavior: clustered locations where user data points appear repeatedly and irregular travel locations where users spent time, but not with enough frequency to be captured as a cluster.
- Link key user locations to time periods and generate behavioral profiles of users based on the percentage of each hour of the day that a user spends in each of their stationary locations.

DENSITY-BASED CLUSTERING TO GROUP POINTS



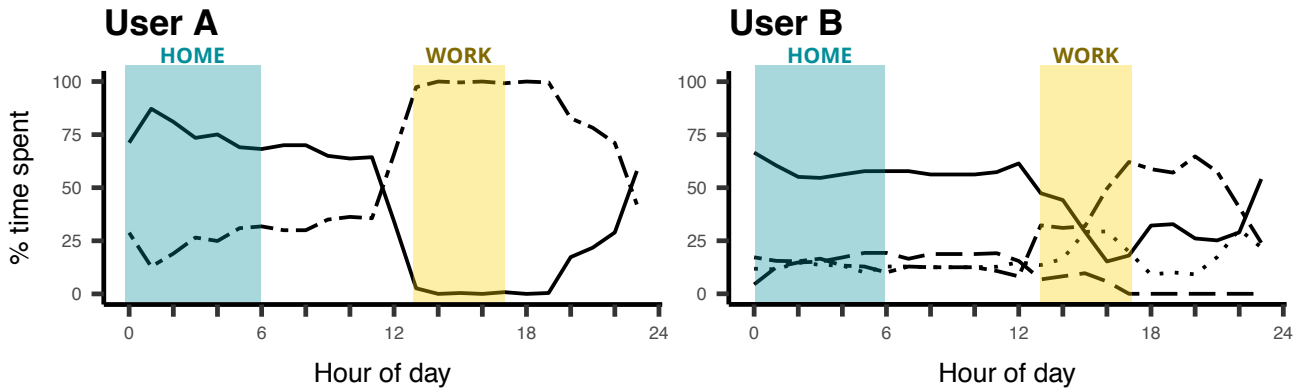
Inferring activity types

Challenge:

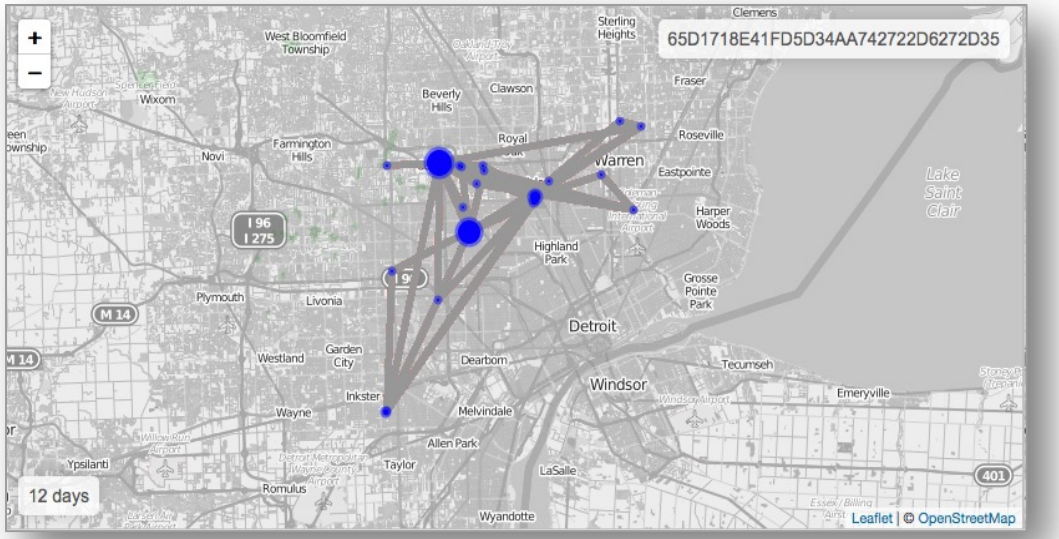
How can we best label user stay locations and output final schedules from the data?

Solution Methodology:

- Use a rules-based approach to assign labels for home and work locations
    - If a user spent more than 50% of their total time between the hours of midnight and 6am at a location, that location was labeled home, and if a user spent more than 50% of their total time between the hours of 1pm and 5pm at a location, that location was labeled work
- Condense any duplicative stay locations and eliminate short stops that were missed in the earlier data cleaning process



USER SCHEDULE SHOWING HOME AND WORK LOCATIONS



 |

Abstracted Mobility Patterns

Challenge:

How can we capture patterns of mobility in a way that is more easily generalizable?

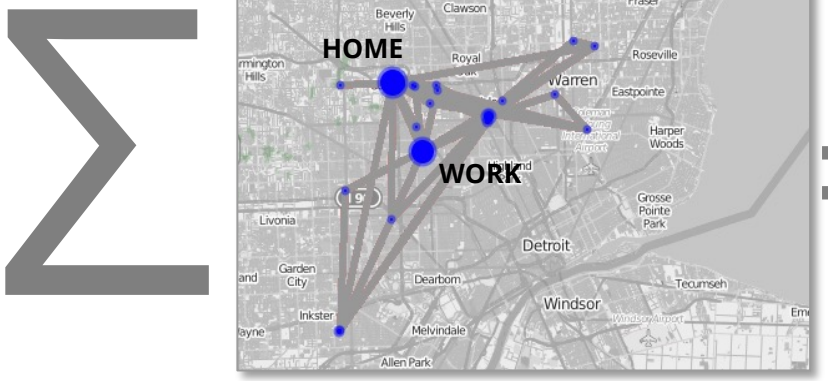
Solution Methodology:

- Create a mobility diary to represent each individual's routine. Here, we only care about whether the individual was at a primary (i.e., home or work) or not–primary location at each given time step, along with the transmission probability of going to a primary or non-primary location at each possible time-step.
    - Primary == 1
    - Not-primary == 0
    - Timesteps are in hours in the example, but are typically set at smaller intervals (e.g., 10 minutes)

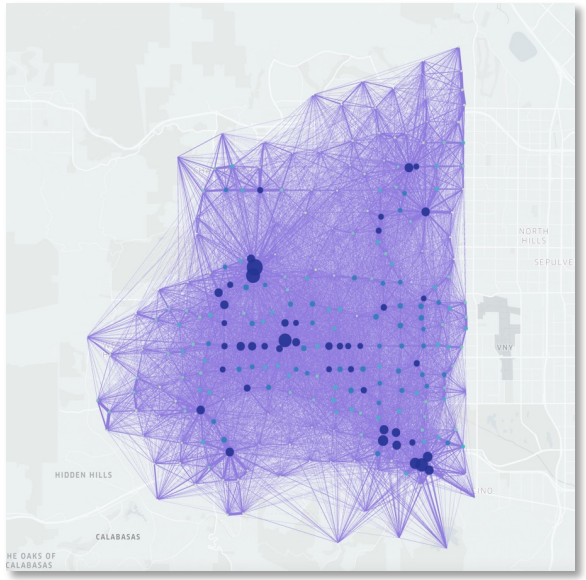
```
(6, 0): defaultdict(float,
            {(0, 0): 0.0,
             (0, 1): 0.0,
             (1, 0): 0.0,
             (1, 1): 0.0,
             (2, 0): 0.0,
             (2, 1): 0.0,
             (3, 0): 0.0,
             (3, 1): 0.0,
             (4, 0): 0.0,
             (4, 1): 0.0,
             (5, 0): 0.0,
             (5, 1): 0.0,
             (6, 0): 0.0,
             (6, 1): 0.0,
             (7, 0): 0.0,
             (7, 1): 0.5,
             (8, 0): 0.0,
             (8, 1): 0.0,
             (9, 0): 0.0,
             (9, 1): 0.0,
             (10, 0): 0.5,
             (10, 1): 0.0,
             (11, 0): 0.0,
             (11, 1): 0.0,
             (12, 0): 0.0,
             (12, 1): 0.0,
             (13, 0): 0.0,
             (13, 1): 0.0,
             (14, 0): 0.0,
             (14, 1): 0.0,
             (15, 0): 0.0,
             (15, 1): 0.0,
             (16, 0): 0.0,
             (16, 1): 0.0,
             (17, 0): 0.0,
             (17, 1): 0.0,
             (18, 0): 0.0,
             (18, 1): 0.0,
             (19, 0): 0.0,
             (19, 1): 0.0,
             (20, 0): 0.0,
             (20, 1): 0.0,
```

```
(6, 1): defaultdict(float,
            {(0, 0): 0.0,
             (0, 1): 0.0,
             (1, 0): 0.0,
             (1, 1): 0.0,
             (2, 0): 0.0,
             (2, 1): 0.0,
             (3, 0): 0.0,
             (3, 1): 0.0,
             (4, 0): 0.0,
             (4, 1): 0.0,
             (5, 0): 0.0,
             (5, 1): 0.0,
             (6, 0): 0.0,
             (6, 1): 0.0,
             (7, 0): 0.061224489795918366,
             (7, 1): 0.8163265306122449,
             (8, 0): 0.02040816326530612,
             (8, 1): 0.0,
             (9, 0): 0.02040816326530612,
             (9, 1): 0.0,
             (10, 0): 0.0,
             (10, 1): 0.0,
             (11, 0): 0.0,
             (11, 1): 0.0,
             (12, 0): 0.0,
             (12, 1): 0.0,
             (13, 0): 0.04081632653061224,
             (13, 1): 0.0,
             (14, 0): 0.0,
             (14, 1): 0.0,
             (15, 0): 0.0,
             (15, 1): 0.0,
             (16, 0): 0.02040816326530612,
             (16, 1): 0.0,
             (17, 0): 0.0,
             (17, 1): 0.0,
             (18, 0): 0.0,
             (18, 1): 0.0,
             (19, 0): 0.0,
             (19, 1): 0.0,
             (20, 0): 0.0,
             (20, 1): 0.0,
```

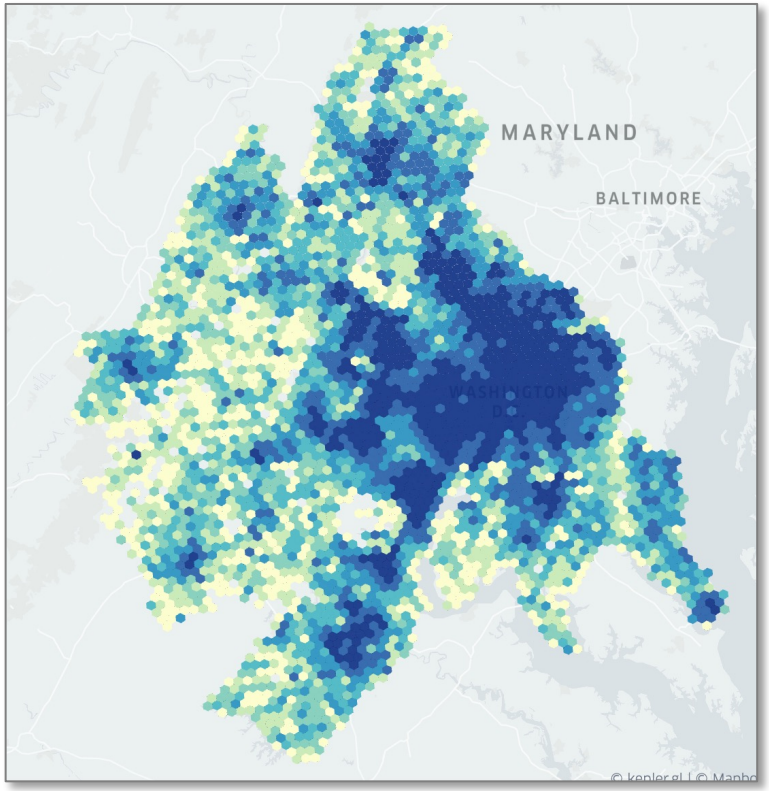

Generate a Spatial Tessellation



Activity pattern for a single user (device)



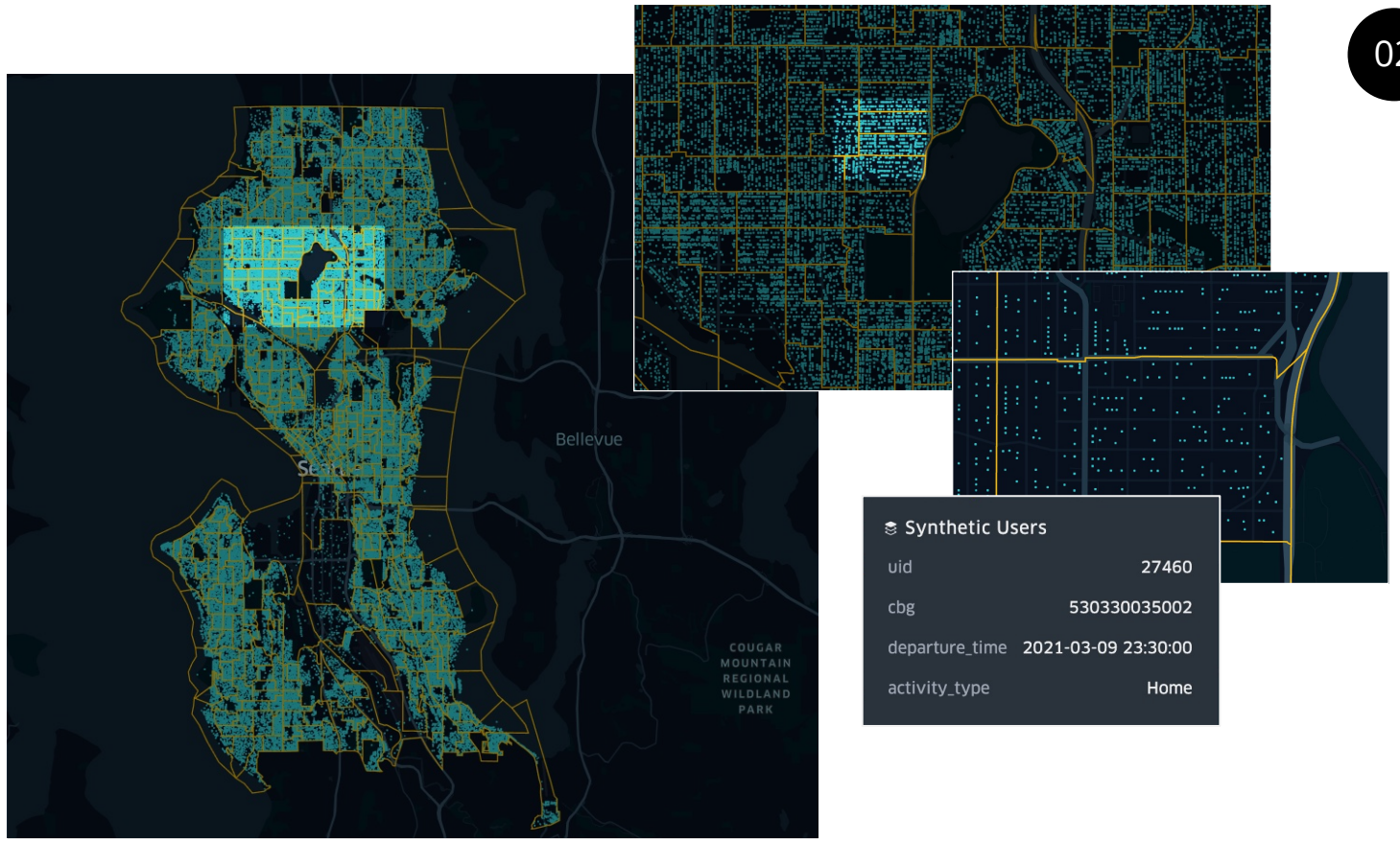
Origin-destination pairs for a specified area and the level of activity between them



Weights on the tessellated areal units signify ‘attractiveness’ for a particular use or activity type

Methodology in action

We built a synthetic population of 100,000 people using abstracted activity patterns from pre-existing trip lists. The resulting activity schedule made it possible to simulate a day in a city without any travel pattern details.



We trained our trajectory generation model using data from multiple pre-existing trip lists. By assigning synthetic user locations to parcels, we achieve a high degree of spatial specificity.

BUT NO TRAVEL MODE YET!

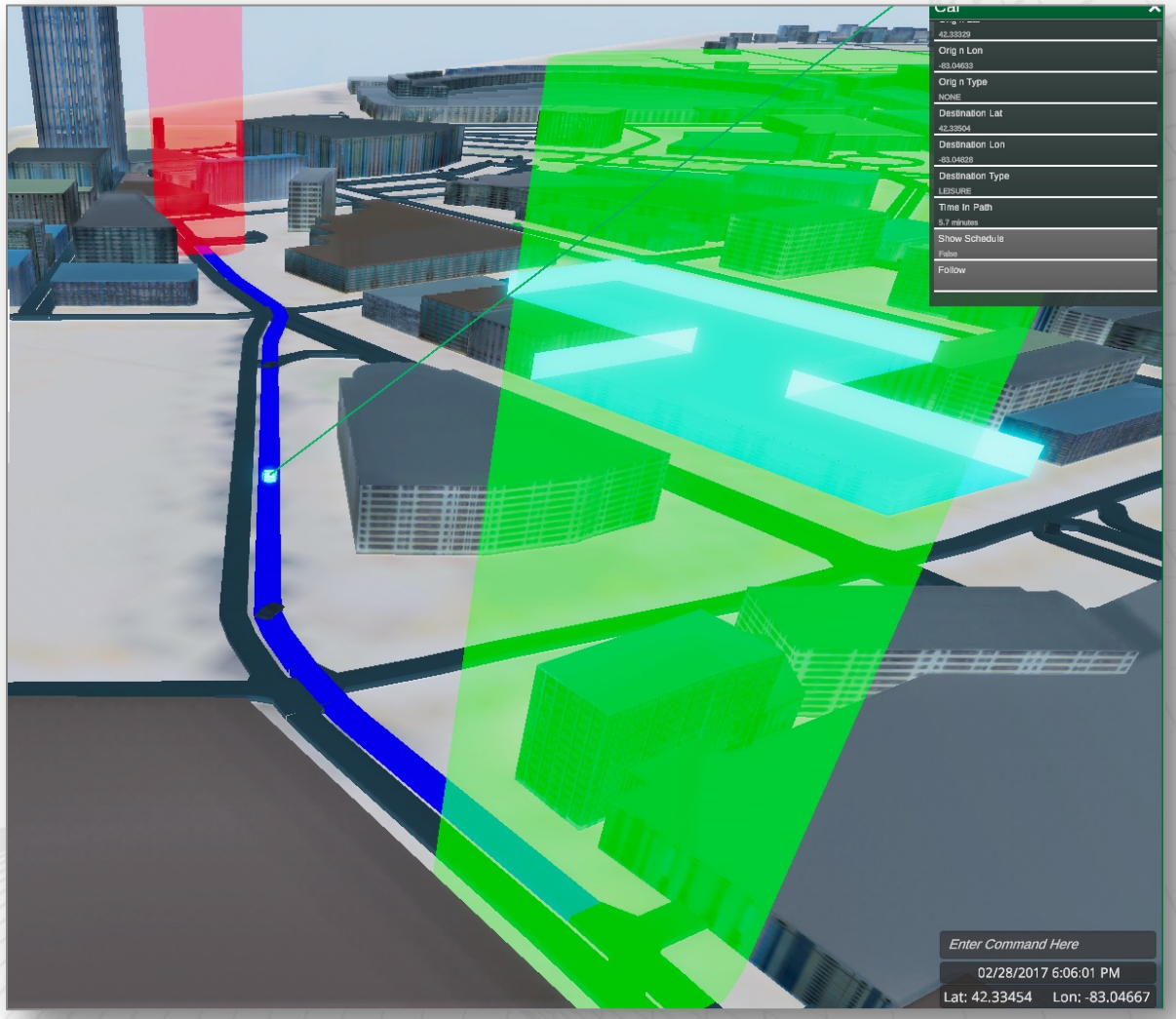
Agent ID	Departure Time	Activity Type	Lat	Lon	Age	Gender	Household Income	...	Work Status	Education Status
1	Datetime Stamp 1	Home	Y1	X1	25 - 44	M	60k - 100k	...	Full Time	No School
1	Datetime Stamp 2	Other	Y2	X2	25 - 44	M	60k - 100k	...	Full Time	No School
1	Datetime Stamp 3	Work	Y3	X3	25 - 44	M	60k - 100k	...	Full Time	No School
1	Datetime Stamp 4	Home	Y4	X4	25 - 44	M	60k - 100k	...	Full Time	No School

Ultimate Goal

WE TURN RAW GEOLOCATION DATA...

User ID	Location Timestamp	Latitude	Longitude
903924FEEC5FD7D39CB41798946F1586	3/19/2017 1:17:09 AM	42.678528	-83.149698
B10D6B8569128EA349E000C3E84091EA	3/19/2017 10:47:18 PM	42.352421	-83.288686
B10D6B8569128EA349E000C3E84091EA	3/19/2017 8:37:40 AM	42.352514	-83.28831
B10D6B8569128EA349E000C3E84091EA	3/19/2017 6:49:09 AM	42.35251	-83.288282
B560EFBEAF764CAAD3AB77824269210D	3/19/2017 6:44:05 AM	42.352442	-83.28858
39F87831B9517FF417C8B0E8D5C0263E	3/19/2017 8:55:28 PM	42.352514	-83.28831
5C0F612DF8D0C5ECEB19F09E1E664A2A	3/19/2017 4:29:10 PM	42.352394	-83.288555
5C0F612DF8D0C5ECEB19F09E1E664A2A	3/19/2017 5:21:13 PM	42.352531	-83.288377

...INTO TRAVEL SCHEDULES THAT WE CAN SIMULATE



Building an Input-Output Hidden Markov Model

Challenge:

The Skyhook data includes some demographic and geographic bias, and there are privacy concerns if we use the schedules in the data directly in the simulation. **How can we generate user schedules from the data in light of these issues?**

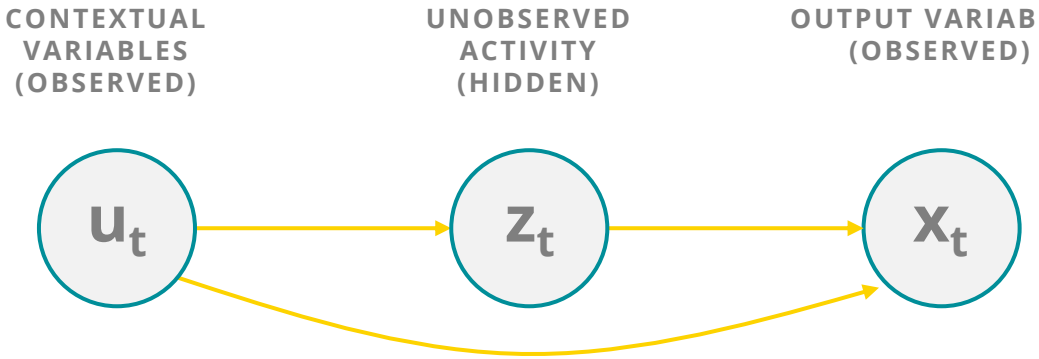
Solution Methodology:

Following the ITS paper, we constructed an Input-Output Hidden Markov Model (IOHMM):

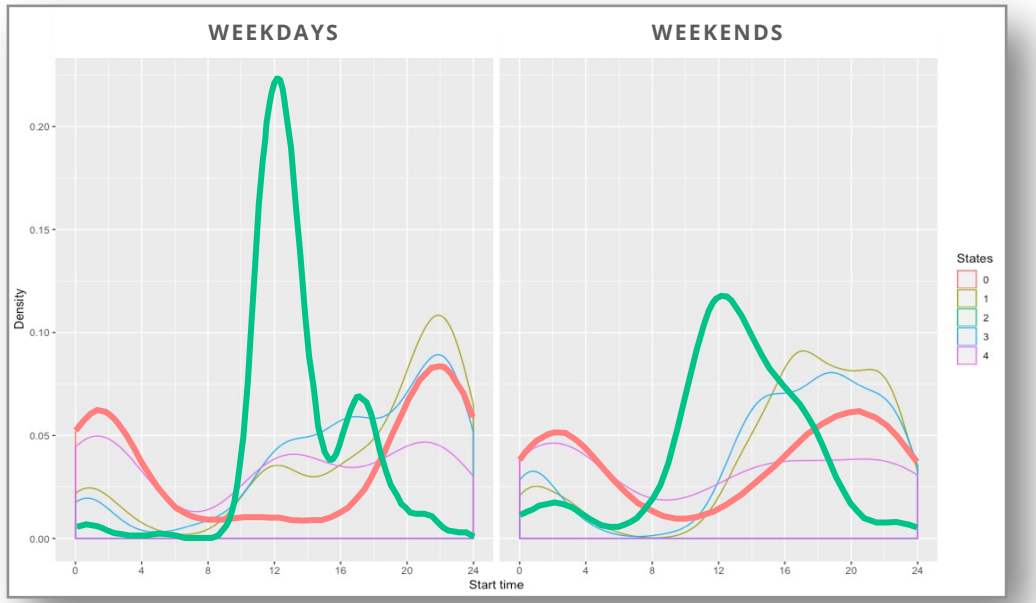
Model inputs:
- Binary variable for whether the day is a weekday or weekend
- Binary variables for different periods of time during each day
- Numeric variable for the number of hours worked

Hidden states:
- Five (home, work, and three other states)

Model outputs:
- Numeric value for distance to home
- Numeric value for distance to work
- Numeric value for time spent at the location
- Binary indicator for whether the location was visited before



ACTIVITY PROBABILITIES BY START TIME: WE SEE THE HIDDEN STATES FOR HOME (RED) AND WORK (GREEN)



 |

Building an Input-Output Hidden Markov Model

Results:

We can link the output from the IOHMM to geospatial positions, thus creating activity paths for synthetic users. We assign locations in the following way for each user:

Home: The home location is based on probability of any TAZ in the area on which the IOHMM was trained containing a residential location, based on the Census population of the TAZ

Work: The work location is based on distribution of block groups containing work locations in the training data, which are then sampled from with uniform probability

Other: All non-home and non-work locations are determined based on the distances to home and work output by the IOHMM

State	Start Time	End Time	Home Distance	Work Distance	Duration
1	00:00	11:07	0.00 mi	8.55 mi	11.1 hrs
0	11:07	11:27	4.67	7.51	0.3
2	11:27	14:39	8.54	0.00	3.2
2	14:39	16:31	8.54	0.00	1.9
0	16:31	17:06	4.67	7.51	0.6
1	17:06	00:36 +1d	0.00	8.55	7.5

WE CAN TAKE THESE ACTIVITY SCHEDULES AND IMPORT THEM DIRECTLY INTO THE SIMULATION

Summary of Markov model varieties

Markov Model

Observed States:

- Home
- Work
- Other

Hidden States: (none)

Contextual Information: (none)

Since we know all of the states, and transition probabilities are fixed, we can estimate the probability of any event at time t like this:

$$\Pr(home_t) = \Pr(home_t \mid state_{t-1})$$

Hidden Markov Model

Observed States:

- Distance to home
- Distance to work
- Duration of activity

Hidden States:

- Home
- Work
- Other

Contextual Information: (none)

$$\Pr(home_t) = \Pr(home_t \mid state_{t-1} \,\&\, distance\ from\ home_t \,\&\, distance\ from\ work_t \,\&\, duration_t)$$

Input-Output Hidden Markov Model

Observed States:

- Distance to home
- Distance to work
- Duration of activity

Hidden States:

- Home
- Work
- Other

Contextual Information (observed):

- Weekday or weekend
- Time of day that activity occurs
- Number of hours worked thus far

Hidden Markov Model vs. Input-Output Hidden Markov Model

Each model really contains a set of models that estimate three different probability distributions:

	HMM	IO-HMM
INITIAL STATE PROBABILITY	**$Pr(z_1 = i)$** What's the probability that I'm at home now?	**$Pr(z_1 = i \mid u_1)$** What's the probability that I'm at home now given some contextual information about my current state?
TRANSITION PROBABILITY	**$Pr(z_t = j \mid z_{t-1} = i)$** What's the probability that I go to any other activity (state) next given that I'm at home now?	**$Pr(z_t = j \mid z_{t-1} = i, u_t)$** What's the probability that I go to any other activity (state) next given that I'm at home now and given some contextual information about my next state?
EMISSION (OBSERVATION) PROBABILITY	**$Pr(x_t \mid z_t = i)$** What's the probability that I observe certain characteristics about my next activity given all possible activities I could do?	**$Pr(x_t \mid z_t = i, u_t)$** What's the probability that I observe certain characteristics about my next activity given all possible activities I could do and given some contextual information about that state?

