Location-allocation and public transit

An update on UCL student teacher placements

Nick Bearman

nick@nickbearman.com

Independent Scholar,
GIS Trainer + Consultant,
Associate Research Fellow, UCL Geography

James D. Gaboardi
Patrick J. Roddy
Qunshan Zhao
Huanfa Chen
Levi Wolf

April 23, 2025

The problem

- UCL IOE (Institute of Education) run a teacher placement programme.
- These students are learning how to become high school teachers and have two 60 day placements at schools.
- They have about 800 students who need to be placed across about 500 schools, across about 5 subject groups.
- Currently they use a combination of Excel spreadsheets and Google Maps (locations and public transport routing) to do this process manually.
- This is very time consuming and would take *at least* a week per subject group to complete.

We know there is a better way

- Location-Allocation could be a solution to this problem.

  → *Find locations for facilities from a set of demand locations.*

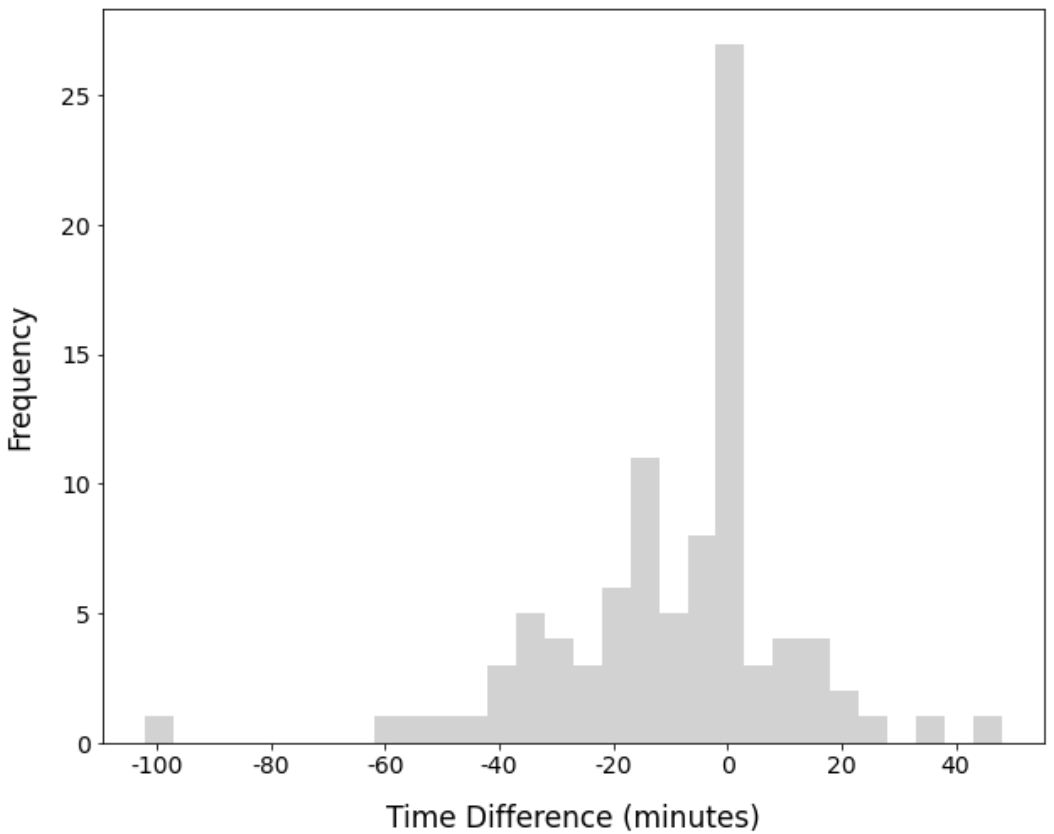
- Classic example:

  → *Where do we put a new hospital so as many residents as possible are within the shortest possible travel time?*

  → E.g. ~ run drive-time routes from every house to every possible hospital location. Select the shortest total time.

Initial Results

- Working with James Grindrod - IOE Placement Manager
- Developed an automated way of doing this



- Reduced calc time from ~a week to 2-3 days (> **60%**)
- Reduced travel time by **9.58** min per student

- Negative values = decrease in travel time
- Positive values = increase in travel time
- *(compared with the IOE allocation)*

What is Location Allocation?

- Has been around in GIS for a long time.
- Very commonly drive-time focused (or walk or cycle….)
- Relatively little in terms of public transport.
- In UCL IOE, public transport is key.

→

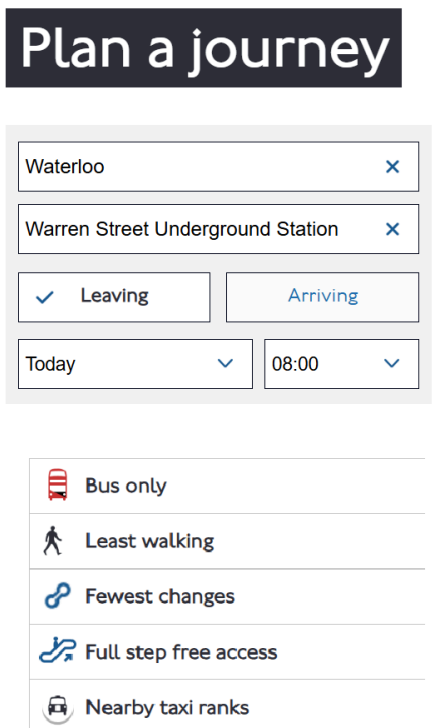
About 70% of students use public transport to get to their placement schools.

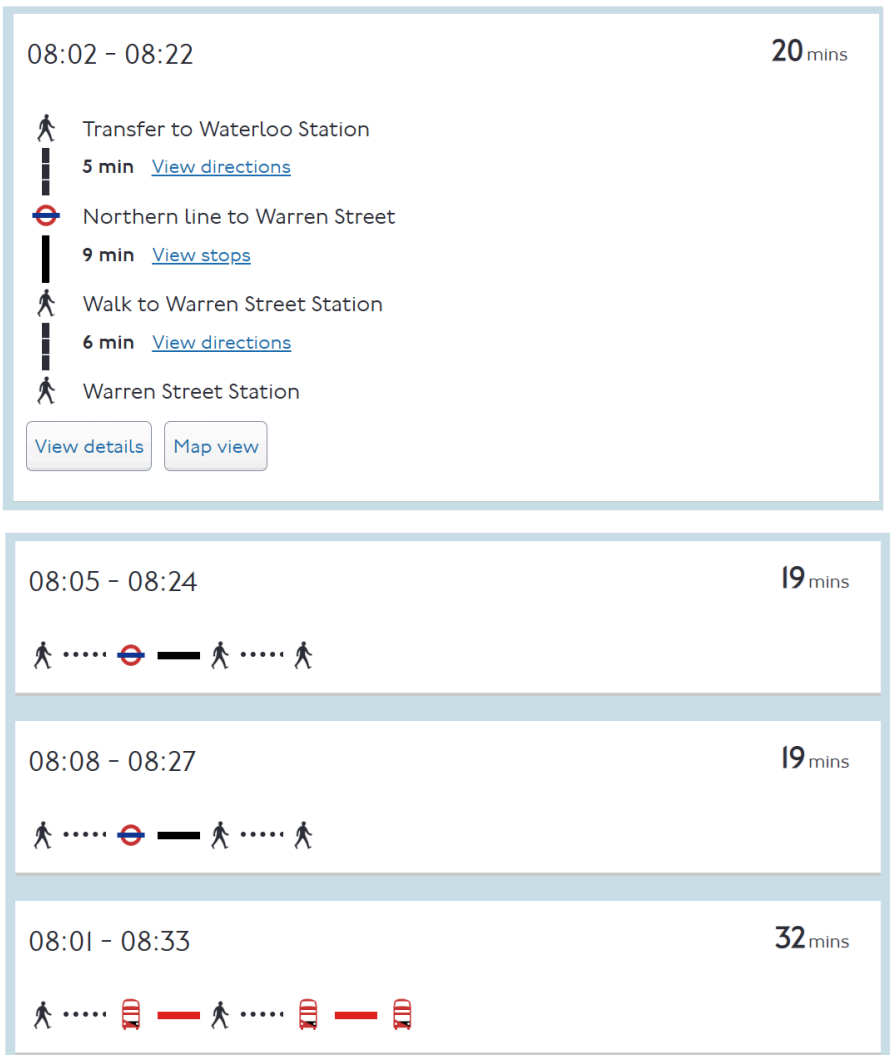
→

In general, ~8% of people use public transport to get to work Census 2021.

What is Location Allocation?

- So we need the TfL API.





How did we do this?

- I was contacted by UCL IOE (James Grindrod) and UCL ARC (Tim Machin) to see if there was a better way.
- Many ways of doing location-allocation

  → *I'm not an expert on the exact ins and outs of this*
- Need to integrate public transport data

  → *Can create an OD matrix from TfL API*
- Need to work with a large number of students (~800) and schools (~500)
- Considered ArcGIS Pro, FLP Spreadsheet Solver, PySpatialOpt, ..
- Reached out via Twitter

How did we do this? *phase 1*

- Qunshan Zhao suggested *spopt*, part of *PySAL, Python*
- *spopt* couldn't do what I wanted, but happy to develop
- Applied for AGILE funding to do this, successful (+ Glasgow):

  → Rongbo Xu added capacitated p-median location-allocation model to spopt
- UCL ISD supported first phase of development:

  → Patrick Roddy created code to calculate routes for student (postcode) to school (postcode) using TfL API

AGILE: Bearman, N., Xu, R., Roddy, P. J., Gaboardi, J. D., Zhao, Q., Chen, H., and Wolf, L. (2023) Developing capacitated p-median location-allocation model in the spopt library to allow UCL student teacher placements using public transport, AGILE GIScience Series., 4, 20, Paper https://doi.org/10.5194/agile-giss-4-20-2023, Details, Presentation & Notebook: https://github.com/UCL/ioe-student-school-allocation.

How did we do this? *phase 2*

UCL ISD developed a tool for IOE placements team to use

1. Upload
2. Input Data
3. Compute Journeys
4. Allocate
5. Review
6. Export

1. Upload

- **Upload** the data on students (placements) and schools as CSV.
- Student name, home postcode, transport modes, school restrictions, priority information related to caring, etc.
- School, location, priority, etc.

2. Input Data

- **Input Data** checks data and highlights any missing data.

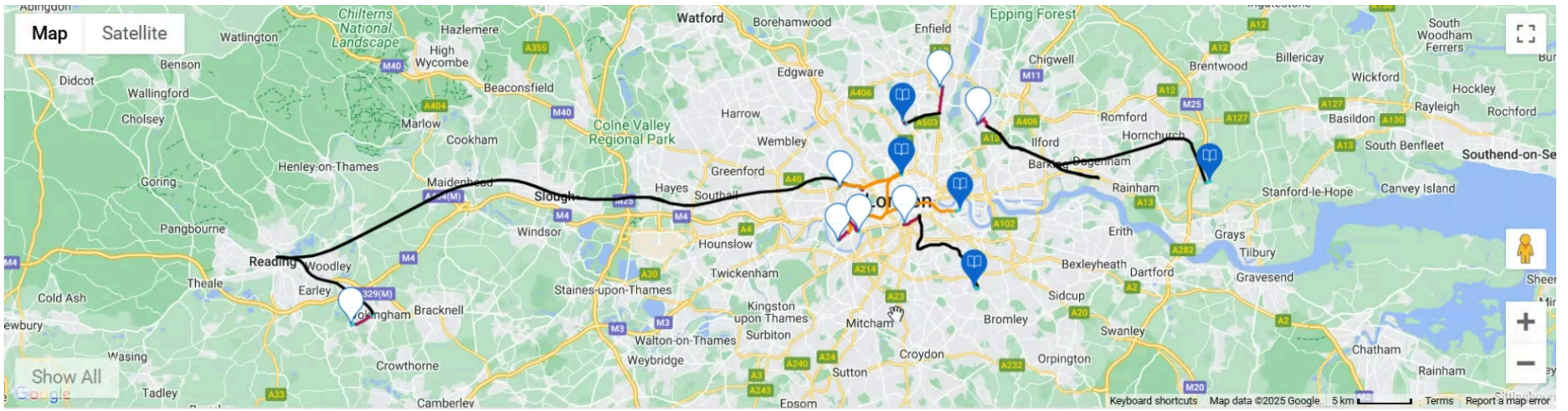
3. Compute Journeys

- Uses *TfL API* to calculate all possible public transport journeys from each student to each school
- Returns a big OD matrix of journey time (mins), number of legs, number of modes
- 800 students and 500 schools takes about 26 hours to process.
- This only needs to be done once.
- Currently being optimised.

4. Allocate

4. Allocate

5. Review



Terry Prachett **2** SW6 5SH

07:07 - 07:45 **38** mins

Post Office School

Henry Phillips **1** E10 7HP

07:10 - 07:55 **45** mins

Southwoods High

Check for **outliers** | Iterative process **Review -> Allocate -> Review**.

6. Export

Export data as CSV file - for human review and/or re-import into software.

Feedback - early stages

- Demo'd to the IOE users - James Grindrod & Placements team.
- A small subset of data worked well.
- Questions over "why did it allocate this student here?"
  - → the algorithm isn't very transparent.
  - → *complicated by the fact the data was a small subset of real data, and comparison was with the full data set done by hand*.
- Try to show in the software why certain allocations were done.
  - → Show range of options for each student.
  - → Impact of changing one student on other students travel time, *+ 3.2 min*.

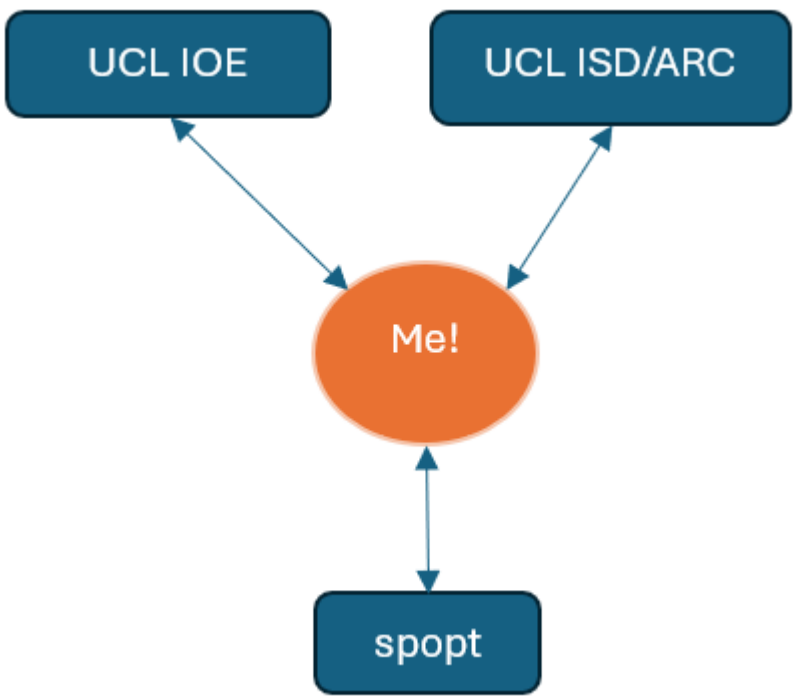
Feedback - early stages

It is intended that there will always be a element of manual allocation needed.

Most restrictions are included in the software, but a few are not present. Some are only stored in the heads of the placements team!

Reflection

- I worked as a bit of a project manager in the early stages



- Sometimes progress with these types of development can be very slow.
- Ideally needs to be someone to push it forward.
- Now they are leading it themselves.

Next Steps

More feedback from IOE - compare a full data set done “by software” and done “by hand” to compare the differences.

Hope to share a Juypter Python Notebook demonstrating the potential of this technique, with dummy data. Levi will do this when he has some time!

Ultimately software to be shared with other users at UCL - e.g. health placements, etc.

And possibly also shared with others.

Open source - not sure? TBC.

Links & Thanks

- GISRUK Article & Slides
- spopt: Spatial Optimization
- AGILE Paper, Bearman et al. 2023
- AGILE GitHub site for code, including Jupyter Notebook with reproducible example and sample student data
- Thanks to AGILE, University of Glasgow and UCL for funding
- Thanks to Rongbo Xu for her contributions to the development of spopt
- Thanks to UCL Colleagues: James Grindrod & team, Nick Mulliss (IOE), Naved Kadri, Arianna Franculacci and Tim Machin (ISD), Jonathan Cooper (ARC)

Dr Qunshan Zhao has received the ESRC's ongoing support for the Urban Big Data Centre [ES/L011921/1 and ES/S007105/1].

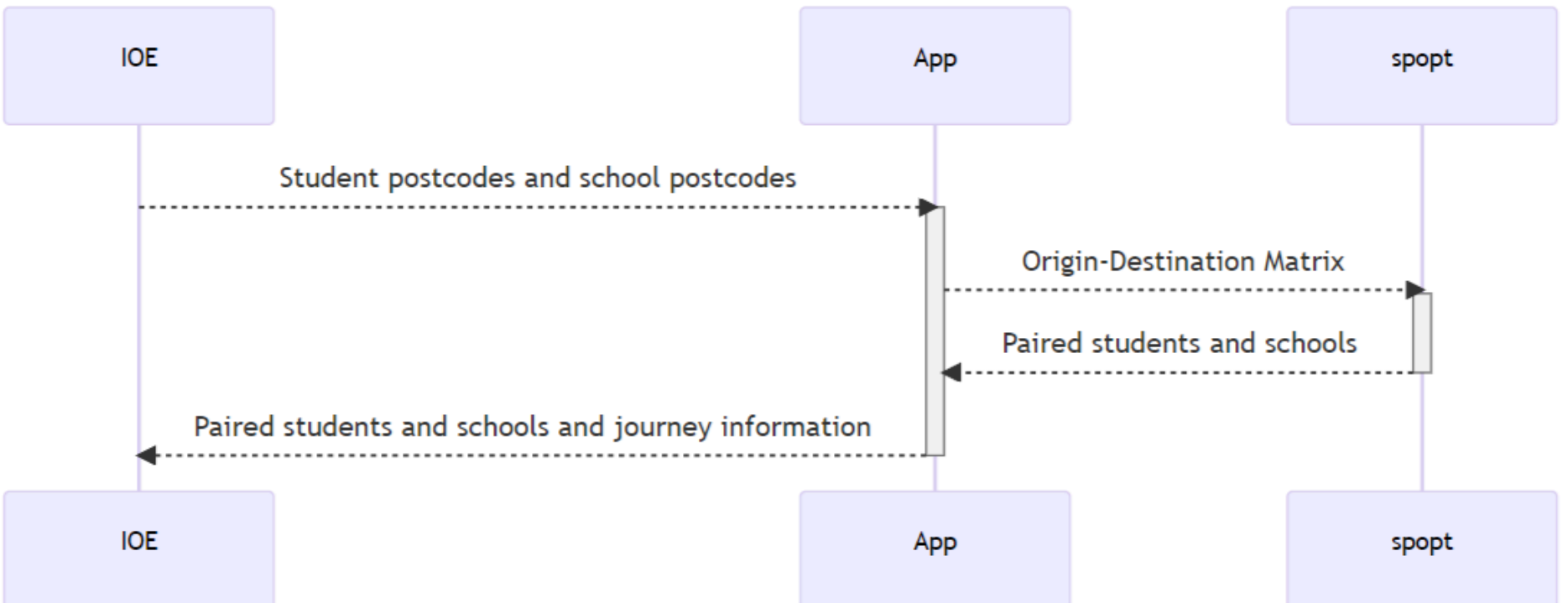
The following acknowledgement applies to James D. Gaboardi (affiliation 2) only: This manuscript has been authored in part by UT-Battelle LLC under contract DE-AC05-00OR22725 with the US Department of Energy (DOE).

Questions?

Image by Gerd Altmann from Pixabay

Extra Slides

Data Flow


IOE
App
spopt
Student postcodes and school postcodes
Origin-Destination Matrix
Paired students and schools
Paired students and schools and journey information
IOE
App
spopt

Model Formulation

$$\text{Minimise} \quad \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}X_{ij} \tag{1}$$

Subject to:

$$\sum_{j=1}^{m} X_{ij} = 1, \quad \forall i \in I \tag{2}$$

$$\sum_{i=1}^{n} X_{ij} \leq b_j, \quad \forall j \in J \tag{3}$$

$$\sum_{i=1}^{n} X_{ij} = b_j, \quad \text{when } p_j = 1 \tag{4}$$

$$X_{ij} \leq Y_j, \quad \forall j \in J \tag{5}$$

Where:

$$X_{ij} = \begin{cases} 1, & \text{if student } i \text{ is assigned to school } j; \\ 0, & \text{otherwise.} \end{cases}$$

$$Y_j = \begin{cases} 1, & \text{if school } j \text{ is selected;} \\ 0, & \text{otherwise.} \end{cases}$$

- i - student, I - all students.
- j - schools (facilities), J - all schools

2. ensures that each student is assigned to one slot
3. ensures that number of students assigned to each school for each subject does not exceed the available slots.
4. ensures that the slots of priority 1 schools are fulfilled.
5. ensures the number of students X is smaller than or equal to the number of slots/schools Y

Sample data: 19 students, 38 schools 0.2 seconds | 20 students, 70 schools 0.7 seconds