

Our Problem as a Busy and Lazy Traveller

Travelling can be fun and relaxing but the planning of the trip is not. The planning is a complex multi-scenario optimisation problem believe it or not!

- What is the first attraction you want to visit?
 - Depends on where you will start and where that attraction is and the transportation
- Which attraction should you visit next?
 - Depends on where you are, the time available and travel time
- Which attractions should I forgo if I do not have enough time?
 - Depends on your level of interest to the attractions, duration you wish / expect to stay in the attraction and the travel time
- etc...

Imagine you want to plan the trip with minimising travel time in mind.

You will have to find the locations of each of the attractions you wish to visit. Find out the transportation arrangements for each attraction. All the information required are readily available online. The problem is to go through the trouble to find them.

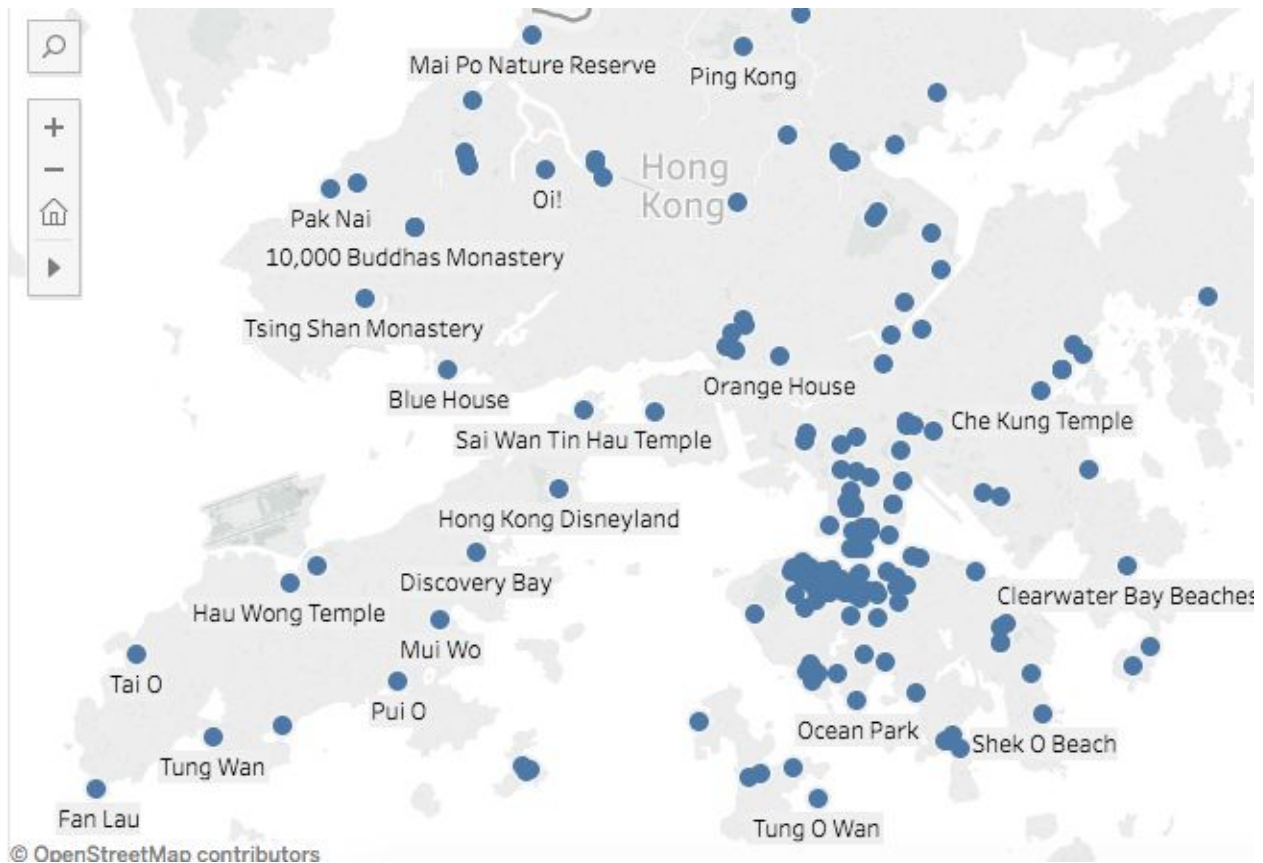
Why can't the computers do it automatically?

Yes, now they can!!

Using the Hong Kong Traveller's Planner

The travellers will simply need to choose their desired list of attractions, rank them according to their own preferences and decide how long they wish to visit each of the attractions. And then a customised itinerary will be available for the traveller.

1. Attractions in Hong Kong



<https://public.tableau.com/profile/bryant5992#!/vizhome/WheretogoMaps/HKDashboard>

There are more than 190 attractions in Hong Kong. It may take a long time to visit all of them.

Even if the traveller selected only a few attractions to visit, it may take forever to plan the itinerary just to minimise travel time and balancing different constraints.

2. Gest User Preference

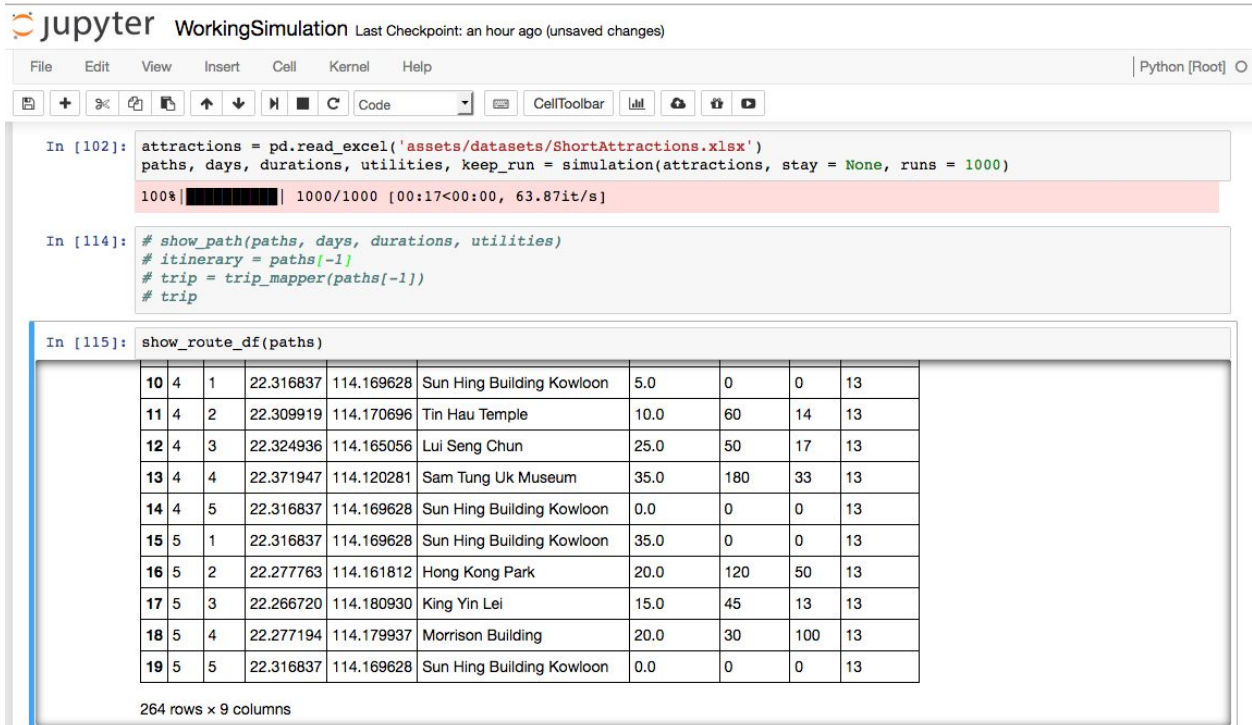
	A	B	C	D
1		Locations	Duration	Score
2	0	Hong Kong City Hall	40	20
3	1	Lui Seng Chun	50	17
4	2	Tin Hau Temple	60	14
5	3	King Yin Lei	45	13
6	4	Hong Kong Heritage Museum	180	11
7	5	Cattle Depot Artist Village	150	25
8	6	Morrison Building	30	100
9	7	Hong Kong Park	120	50
10	8	Police Museum	120	50
11	9	Sam Tung Uk Museum	180	33
12	10	Peak Tram	300	25
13	11	Sun Hing Building Kowloon	Hotel	Hotel

The traveller will choose the “attractions”, set “duration” and “score”.

Duration is the time which the traveller wish to spend in each attraction.

Score is the metric given by the traveller which measures the level of interest for the attractions. It will be a useful decision factor if the traveller happened to select too many attractions but without sufficient time to visit them all. Attractions with high travel time (high cost) but low score (low benefit) will be penalised.

3. Solve Optimisation



Jupyter WorkingSimulation Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Help Python [Root]

In [102]:

```
attractions = pd.read_excel('assets/datasets/ShortAttractions.xlsx')
paths, days, durations, utilities, keep_run = simulation(attractions, stay = None, runs = 1000)
```

100% |██████████| 1000/1000 [00:17<00:00, 63.87it/s]

In [114]:

```
# show_path(paths, days, durations, utilities)
# itinerary = paths[-1]
# trip = trip_mapper(paths[-1])
# trip
```

In [115]:

```
show_route_df(paths)
```

10	4	1	22.316837	114.169628	Sun Hing Building Kowloon	5.0	0	0	13
11	4	2	22.309919	114.170696	Tin Hau Temple	10.0	60	14	13
12	4	3	22.324936	114.165056	Lui Seng Chun	25.0	50	17	13
13	4	4	22.371947	114.120281	Sam Tung Uk Museum	35.0	180	33	13
14	4	5	22.316837	114.169628	Sun Hing Building Kowloon	0.0	0	0	13
15	5	1	22.316837	114.169628	Sun Hing Building Kowloon	35.0	0	0	13
16	5	2	22.277763	114.161812	Hong Kong Park	20.0	120	50	13
17	5	3	22.266720	114.180930	King Yin Lei	15.0	45	13	13
18	5	4	22.277194	114.179937	Morrison Building	20.0	30	100	13
19	5	5	22.316837	114.169628	Sun Hing Building Kowloon	0.0	0	0	13

264 rows x 9 columns

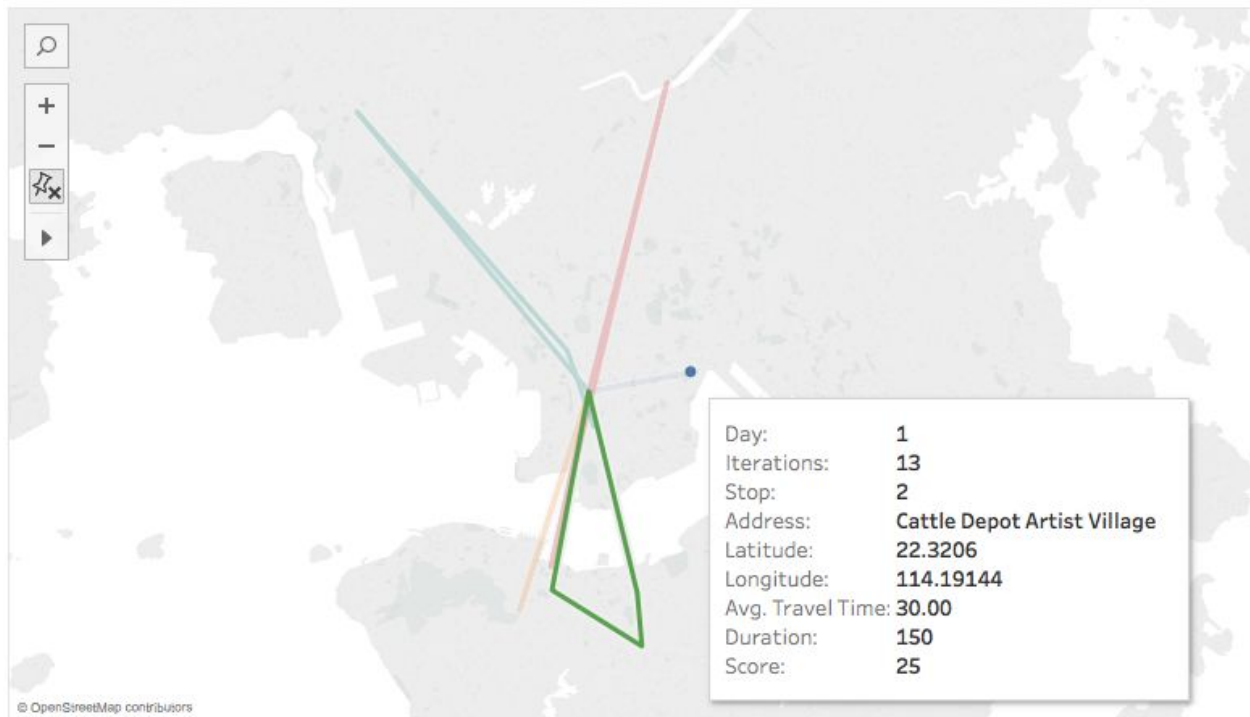
The itinerary is optimised using a simulation method which tries to take different routes.

The simulation is performed in many iterations which the result improves over time.

See the **Simulation** section for more details

4. See the Itinerary on Tableau

Trip Route 13, 5

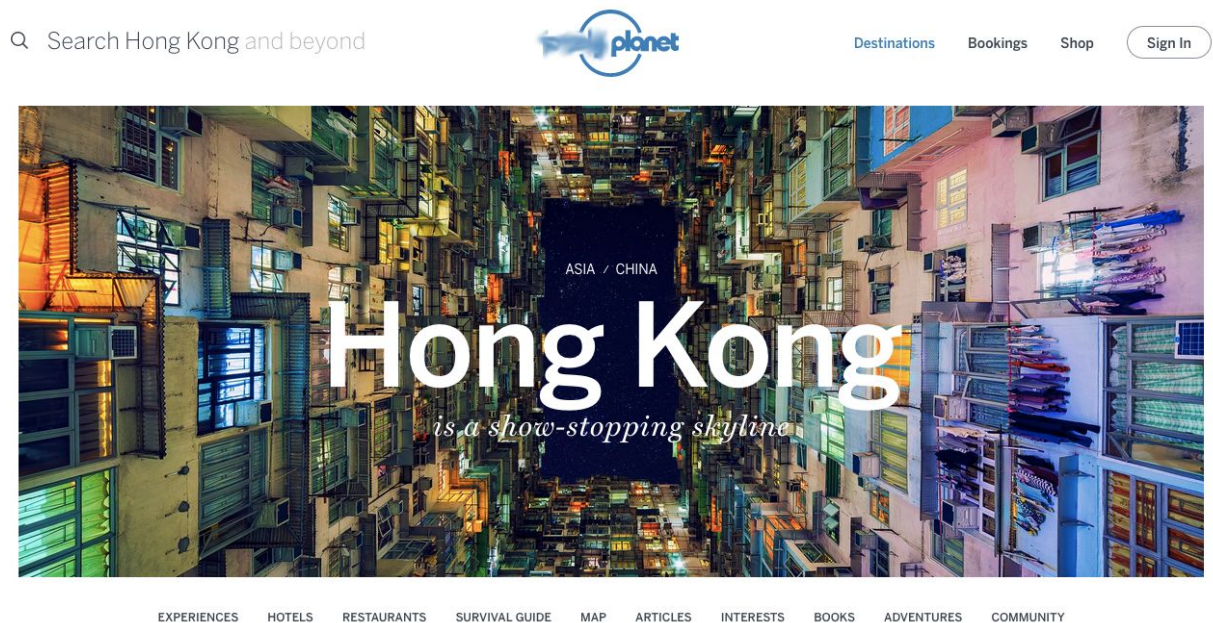


<https://public.tableau.com/profile/bryant5992#!/vizhome/MyRoute/TripRoute>

Getting Data to Support the Model

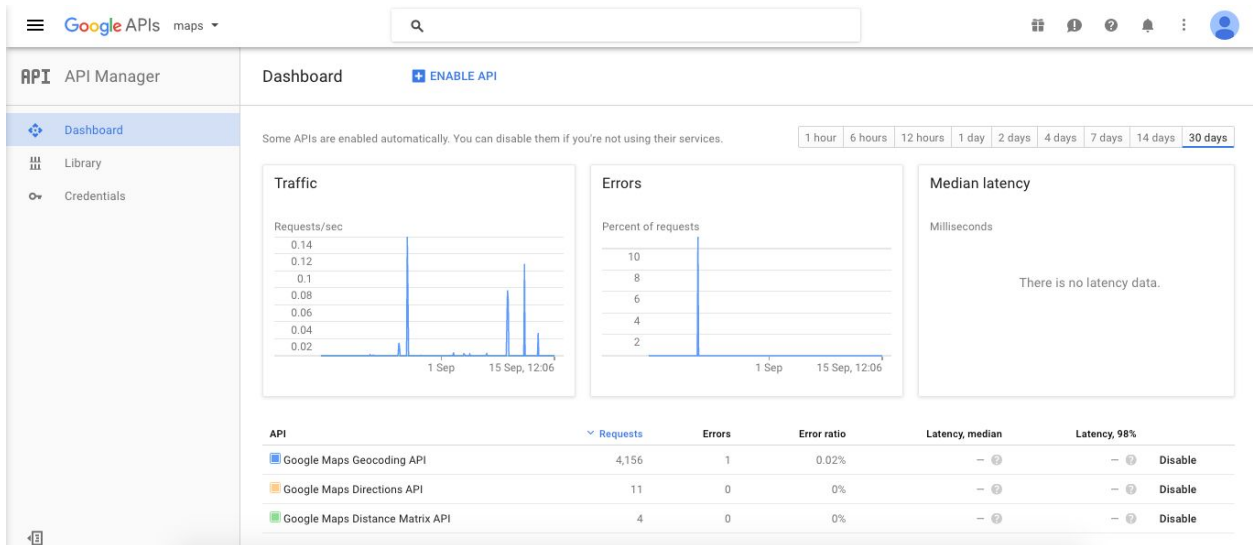
1. Build Attraction List

The list of attractions of Hong Kong was initially obtained from a well-known travel website (name not disclosed to protect the identity of the website).



A list of tourist attractions is obtained and saved in the model's database. But we only have the names of the attractions, we do not know where they are yet. We will need to obtain the latitude and longitude of the locations.

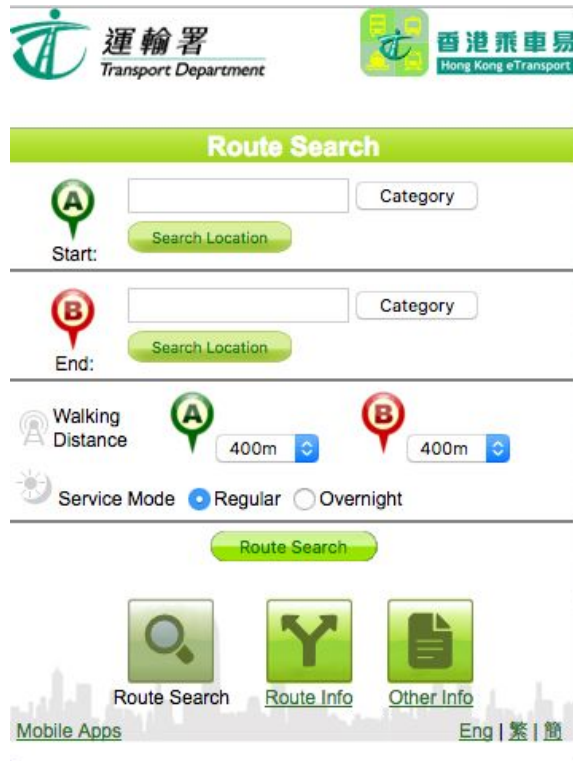
2. Obtain Latitude and Longitude for each locations - using Google Maps API



<https://developers.google.com/maps/>

The Google Maps API allows users to obtain the latitude and longitude of any locations given the address of the location (e.g. “Star Ferry, Hong Kong”).

3. Enquire the Public Transportation Database for instructions



The screenshot shows the 'Route Search' page of the Hong Kong eTransport website. At the top, there are logos for the 'Transport Department' and 'Hong Kong eTransport'. The main section is titled 'Route Search' in a green banner. Below this, there are two input fields for 'Start' and 'End' locations, each with a 'Search Location' button and a 'Category' dropdown menu. The 'Start' field is marked with a green 'A' icon, and the 'End' field is marked with a red 'B' icon. Below the location fields, there are two 'Walking Distance' input fields, both set to '400m', with a 'Walking Distance' label and a 'Service Mode' section with radio buttons for 'Regular' (selected) and 'Overnight'. A large green 'Route Search' button is positioned below these options. At the bottom, there are three icons: a magnifying glass for 'Route Search', a Y-junction for 'Route Info', and a document for 'Other Info'. The footer includes 'Mobile Apps' and language links for 'Eng | 繁 | 簡'.

<http://m.hketransport.gov.hk/routeSearch.php?lang=0&langchanged=true>

The “Hong Kong eTransport” app by the Hong Kong Transportation Department allows users to search for public transport information given the start and end location.

The pairwise connectivity database between attractions was acquired from enquiring the web app.

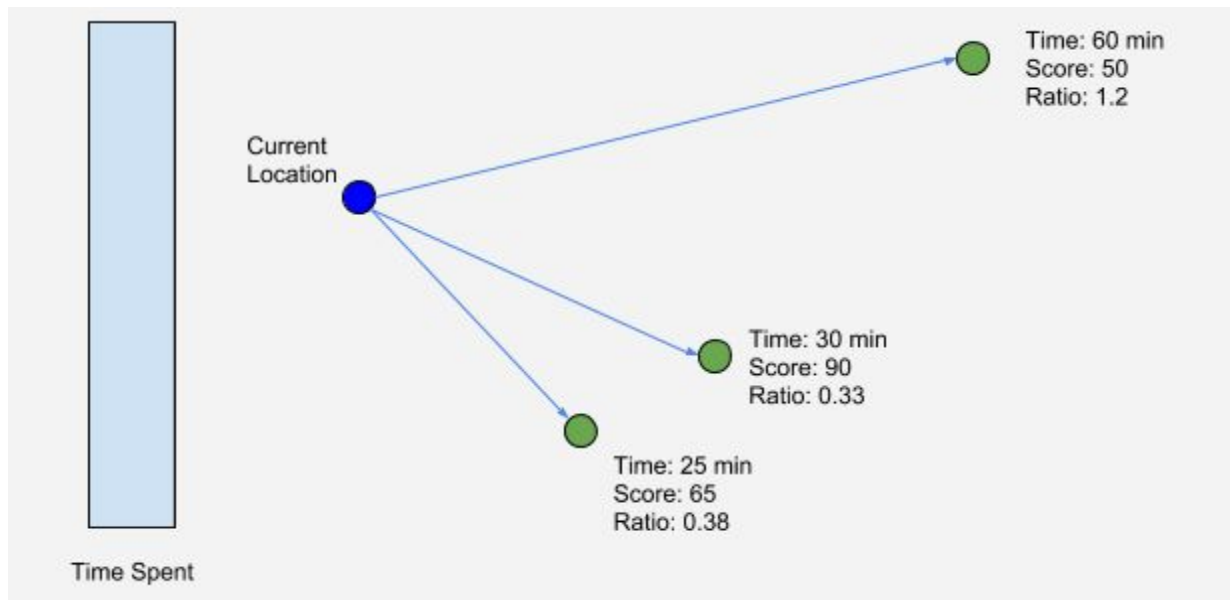
The information was stored in the form of a dictionary using the start and end location’s latitude & longitude for quick reference.

Simulation

The simulation takes the following approach:

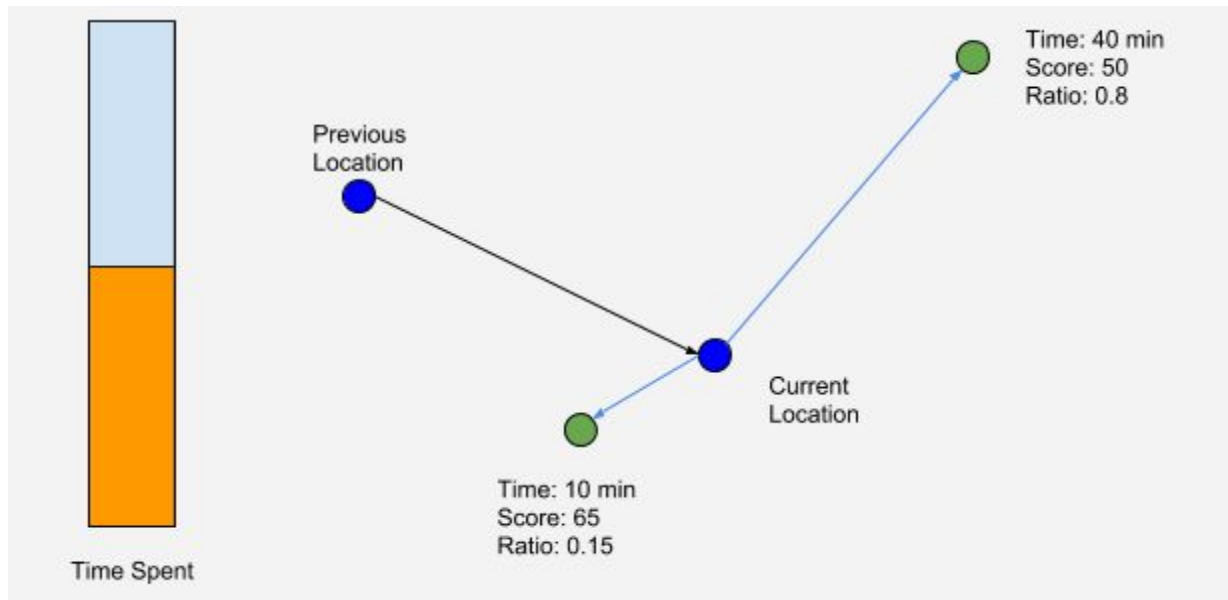
1. Fit attractions to a day-trip until day is full
 - a. Update remaining attractions
 - b. Update number of days travelled
2. Repeat 1 until no remaining attractions or number of days travelled exceeds limit

The following graphical illustration shows how a typical day-trip is filled.



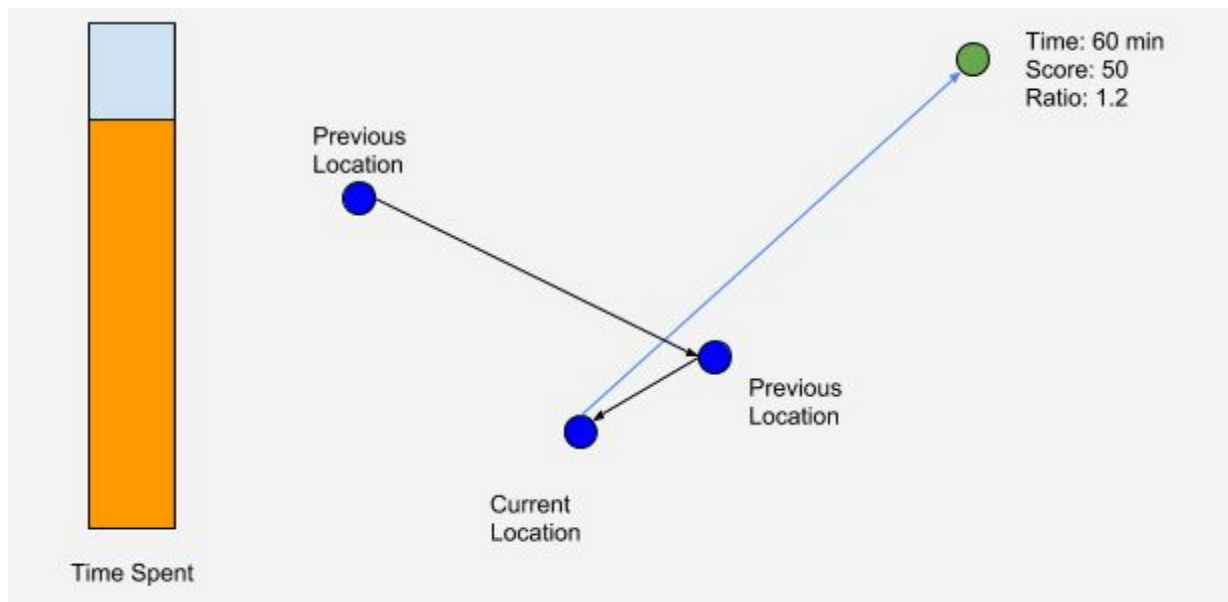
The simulation starts from the initial point (usually the hotel). The time and score ratio is calculated for each of the attractions. The lower the better

The next step is chosen randomly but points with better ratio are more likely to be selected.



The selection is repeated for the new point as the starting point. Also the time spent travelling to the location and duration spent at the attraction are added to the total “Time Spent” counter.

The next step is again chosen.



After 2 steps, there is only one remaining location to visit. But the algorithm will not add the location to the path because we will run out of time if it was added. The algorithm will force the trip to end and return to the hotel and arrange to visit the last remaining attraction the next day.

The process is repeated until all attractions were visited or total number of days travelled exceeds the limit set by the user.

The trip simulation is then repeated a number of times to search alternative routes which will give better results.

Progressive Improvement in Simulation

The optimum solution is “discovered” using the simulation method after a number of iterations. A solution better than the previous one is added while a solution worse than before is ignored.

The “next” step is selected randomly but the more promising attractions (takes less time but has high scores) will have higher probabilities of being selected. Helping the solutions to converge faster.

The ability to minimise the objective function given the criteria is the key success factor. The algorithm may not give the optimum solution.

The initial solution for a particular set of attraction parameters took 7 days while the optimal solution discovered took 5 only days. It is a significant improvement from the random walk and takes roughly 14 seconds to compute given a 11 node optimisation using 500 iterations.

Coding

Two notebooks were used, one for extracting the necessary data through mostly web scraping and API (obtain data) while the other will read the data and feed into the model for optimisation (simulation).

Link to “obtain data” notebook:

https://github.com/RandomEvent/RandomEvent.GitHub.io/blob/master/Projects/project-capstone-final/Obtain_Data.ipynb

Link to “simulation” notebook:

<https://github.com/RandomEvent/RandomEvent.GitHub.io/blob/master/Projects/project-capstone-final/WorkingSimulation.ipynb>

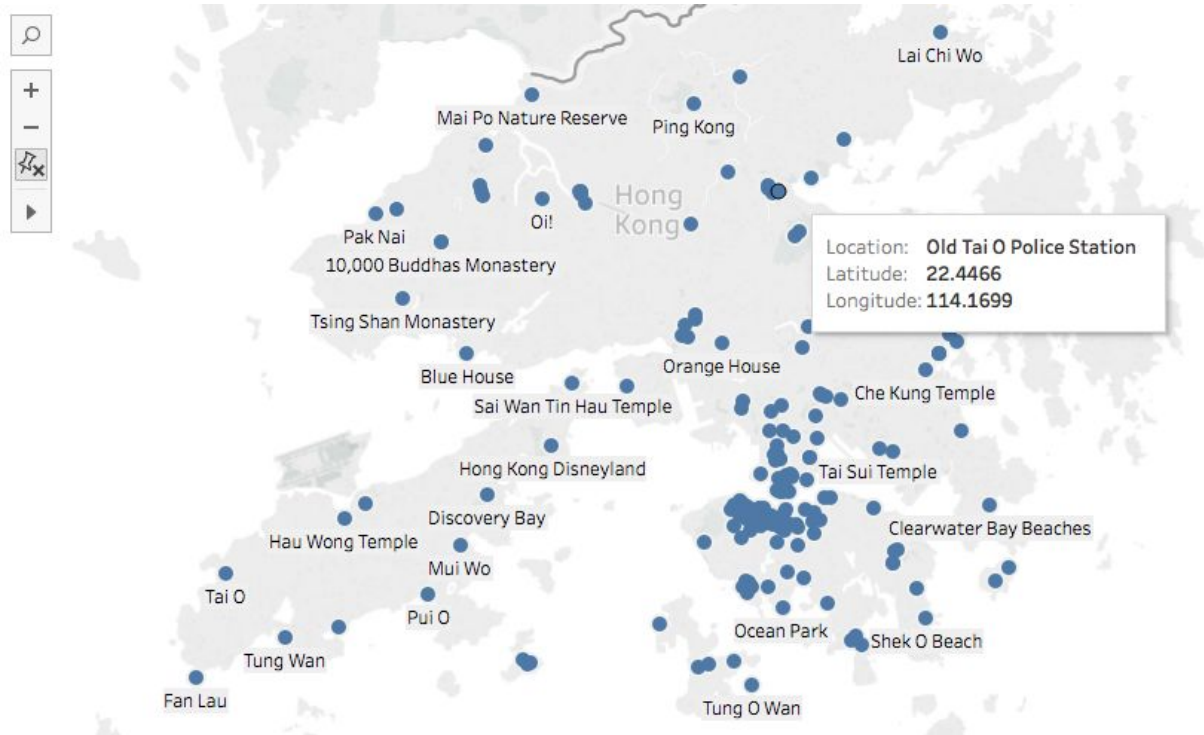
Link to full transport database:

<https://drive.google.com/file/d/0B6fqpdOwJNOSF9pcXhjU19iZjg/view?usp=sharing>

Further Investigations

The optimisation is currently only available in Hong Kong but is scalable given sufficient travel and geographic data in other countries.

Google Maps API does not always give the correction location information. In the example below, Google has mistakenly identified the Old Tai O Police Station to be in the Tai Po region (New Territories North) instead of in the Tai O region (Lantau West).



The optimisation does not take into account that the user can choose to book another hotel in order to make the trip shorter and more convenient. Given (real-time) hotel booking data, users may be able to book the most affordable hotel bookings while also making the trip shorter and a better experience.

Ranking of the attractions and how long the users intend to stay are currently given by user as they are mostly subjective as to how much they like to visit a place and how long they intend to spend at the attractions. However it may also be possible to build a recommender system to suggest the most relevant attractions to the users.

END