

Hot Paths – traversed_paths.py

The program has 1 required argument:

- --ini
 - The path to the configuration file

Alternatively, the program takes 7 optional arguments that can be used instead of the configuration file.

1. -n [name]
 - Database name
2. -i [ip]
 - Database IP
3. -po [port]
 - 1. Database port
4. -u [username]
 - Username
5. -pa [password]
 - Password
6. -fc [count]
 - Minimum trip frequency count
7. -mc [cardinality]
 - Max/stop cardinality

The output format is as follows:

path;count;{trips}\n

This pattern is replicated for every trip that was traversed enough times and have a number of edges equal or greater to the cardinality.

path → the edges that was traversed. An edge is represented as a start and end location

count → how many times this path was traversed

trips → trip identifiers(trip_id in the database) of trips that contains the path

The edges and trip identifiers are comma separated.

Here is an example where the cardinality is set to 6:

1-3, 3-2, 2-11, 11-7, 7-2, 2-9;4;{1,3,110,200}

Notice that the count(4) equals the number of elements in trips.

The output files that contains the trips are named as this 'frequent_paths_<cardinality>' where <cardinality> is replaced with the cardinality that was used for that given set of trips.

When the program is executed it will create a file for every cardinality in the interval [2, mc].

There are some errors in the trips database table, and the program tries to correct them. Examine the trip below:

14-4, 4-7, 9-11, 11-2, 3-5

This trip is erroneous as the vehicle either skips a step or teleports when it goes from 7 → 9 and 2 → 3

This is solved in the program by splitting the trip into multiple smaller trips. The splits/trips will be as follows for this example:

trip 1: 14-4, 4-7

trip 2: 9-11, 11-2

trip 3: 3-5

The new trips will use the same trip_id as the original trip.