Modelling and visualizing macroscopic movements

by

Ananya Chowdhury
Matriculation Number TODO

Piotr Mrówczyński
Matriculation Number 387521

Gabriel Vilén
Matriculation Number TODO

A project documentation submitted to

Technische Universität Berlin
School IV - Electrical Engineering and Computer Science
Department of Telecommunication Systems
Service-centric Networking

June 20, 2017

Supervised by:
Bianca Lüders
Friedhelm Victor
Prof. Dr. Axel Küpper

Service-centric NETWORKING

Technische Universität Berlin

1 Introduction

1.1 Macroscopic City Movements - problem definition

Here we should describe what problem project is about to solve. In the 1st term presentation I mentioned that we want to answer two questions:

Given a location, where and in what proportion people move to another location - Movement Graph?
Where and how long do people stay at certain location? - but we need to be more precise here.

1.2 Human Mobility within the city

According to research [5][6][**HumanMobility3**], the average walking speed in the city for man aged 40 is 1.4 m/s (5 km/h), and slowest one is for elderly (with walking impairment) below 1 m/s (3.6 km/h).

1.3 Methodology of obtaining location data

Here we should describe how and from where do we get the data, and emphasise the fact that location updates are based on MOVEMENTS, so each next point is triggered by some mobile phone movement. Point updates are not triggered while being stationary.
TODO: we need here more details about what our Sudanian guy were supposed to research

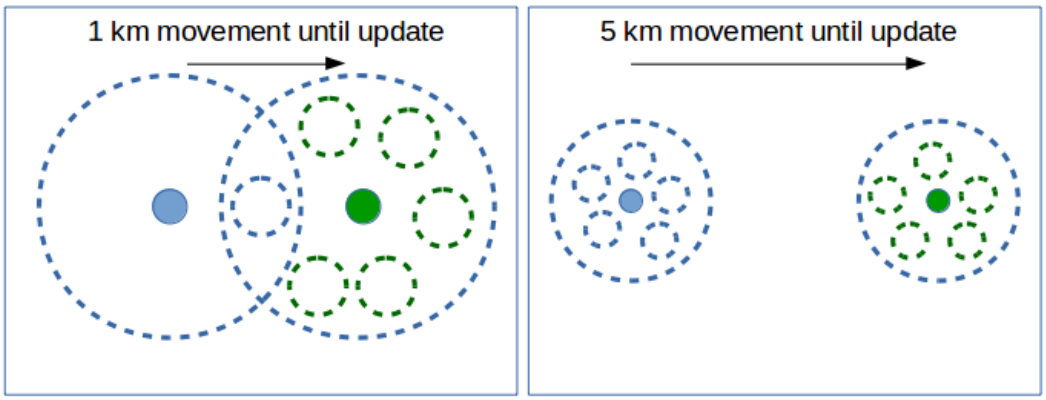

Figure 1.1: Continuous and Discontinuous location update after movement to section lacking area details

1.4 Location data characteristics for Berlin area

1.4.1 General overview

Movement triggered data from mobile devices are spanned across the Germany as shown on Figure 1.2.

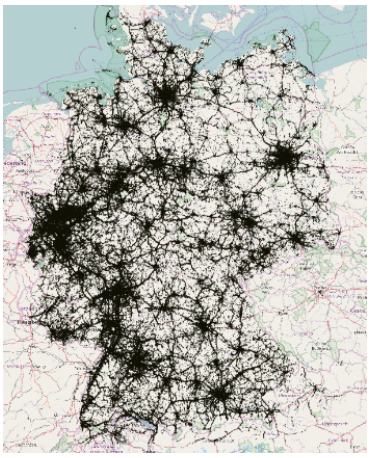

Figure 1.2: Visualisation of data points location in the geographical map of Germany

It points out, that most of the detected movements from one point to another are mostly gathered on the highways and inside the cities.

The given data batch has been gathered in the interval of 24h starting from 3am in the morning German time. It consists of over 675000 data points, belonging to 47300 unique mobile devices. Figure 1.3 shows, that for unique mobile devices, the durations between consecutive points in time are most frequent within 30 minutes and most of the data is in the interval 1-10h. Regarding distances, consecutive points are most frequent below 2 km and most data is in the interval

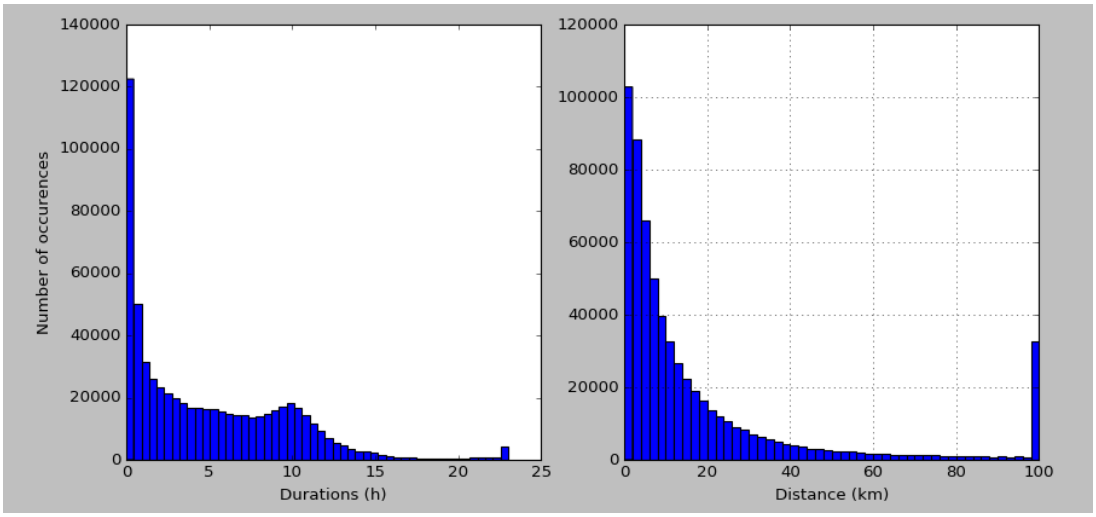

Figure 1.3: Histograms of durations and distances between consecutive points under and cumulatively over 100 km for the area of Germany

0-10km. Big chunk of distances is also over 100km.

Figure 1.4 shows the heatmap of data points for Berlin area. We can see that most of the movements detected are on major train/tram/metro stations, highways and major living areas.

1.4.2 Data analysis

Statistics performed using tool written in Python (https://github.com/mrow4a/macroscopic-movements-algorithm-prototypes) shows, that for users at least once visiting Berlin, had in average 17 points gathered because of their movements in the period of 24h, harmonic mean of 4 points and maximum 100 of points.

Figure 1.5 shows that there is significant fraction of points which timestamp or distance did not change, thus duplicated point has been recorded.
TODO: This needs to be investigated

Furthermore, majority of durations between points is within interval of 30 minutes. For distances between points, there is significant "jump" at 500-1000m and 1100-1500m, which might mean that at these intervals, continuous points been gathered, and distances over that values might be discontinuous (ref. Figure 1.1). It is due to the fact that these distance intervals are most frequent and that might suggest that this is an average "update" distance for the moving mobile devices, as described in section 1.3. Less frequent values might suggest that updates were obtained with lower accuracy (e.g. by presence inside the building, metro line or simply mobile device lag in obtaining its location) and are result of discontinuity between consecutive updates.
Figure 1.6 shows that considering the registered distances within range of 1500m, the wast majority of points are between 0-2 km/h, and also significant interval at 2-6 km/h. The rest of the

Figure 1.4: Heatmap of data point densities within the area of Berlin

values is sparse distributed in interval 6-100+ km/h. In case of registered point above range of 1500m, we observe that indeed most frequent occurrence is at interval of 0-4 km/h, however most are sparse distributed above 4 km/h, with most points being in interval of 4-20 km/h.

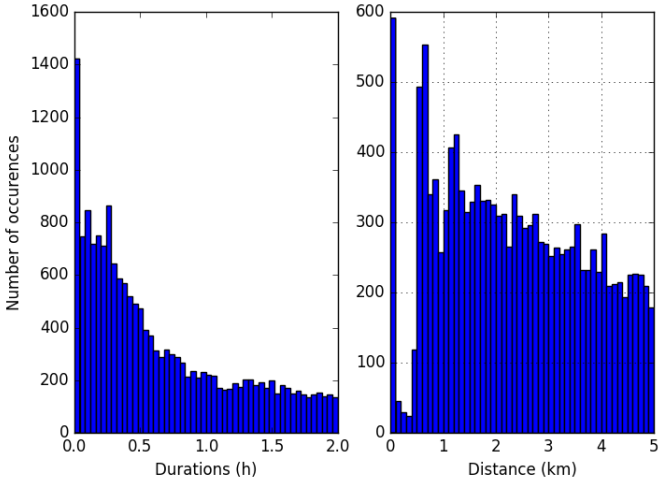

Figure 1.5: Histograms of durations and distances between consecutive points under 5 km and under durations of 2h for the area of Berlin

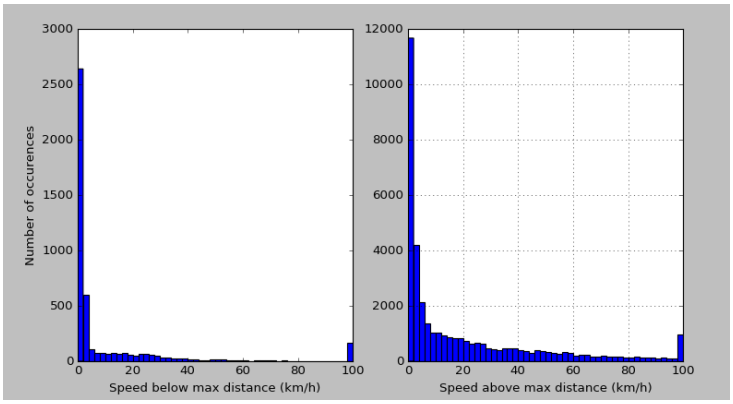

Figure 1.6: Histogram of speed between points with corresponding distance above or below 1.5 km

1.5 Approaches to stop detection for localization services

Here we should explain different approaches

1.5.1 Stop Detection analyzing repetitive appearance at location

1.5.2 Stop Detection based on continuous localization

1.5.3 Stop Detection based on mobility index

In the paper [4], movement of mobile devices between cells (handovers) is considered. To detect the periods of slow movements of stops at the specific location, parameter called Mobility Index is considered.
In a populated area, with many GSM cells, the mobile terminal can change from one cell to another within seconds or after several minutes. Given a set of consecutive records, the mobil-

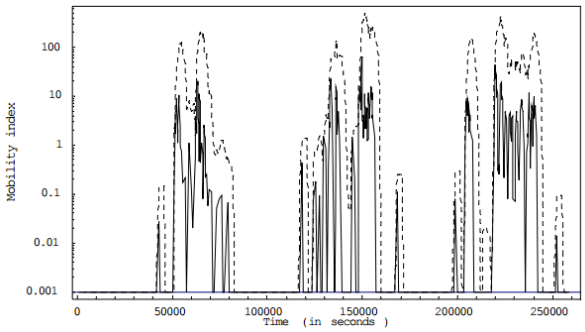

Figure 1.7: Mobility Index. Source: [4]

ity of a user can be estimated by calculating the Mobility Index over a pre-defined time period (sliding window). The Mobility Index is the sum of the distance between each record and the previous ones, where the distance is the inverse of the time spent on each cell. If the value of MI is below certain threshold, it is assumed that the object stopped or moved slowly.

In the publication, sliding window of 10 minutes and a mobility index threshold value of 6 has been used for certain area. The obtained results were in agreement with actual movements during more than 90% of the time.

High mobility index tells that user have been moving from checkpoint to checkpoint within very small time distance. Rapid drop in mobility index represents very long duration at the last point, and slight decrease in mobility index means that user changed position, but within longer duration.

2 Concept and Design

We need to describe here what do we want to achieve, what do we mean by modelling and visualizing macroscopic movements etc - thus general architecture that we have 3 components, Stop Detection from telekom data based on movements and updates of "areas" in your mobile phone, Clustering of Stops to find most popular "stops" in the city, Graph Generation to find where do the people move and in what amount

2.1 Stop Detection

Analyzing different approaches for stop detection based on localization data (ref. section 1.5), we decided to base our solution on standard human behavior regarding mobility within the cities (ref. section 1.2) and reusing the concept of Mobility Index (ref. subsection 1.5.3)).

Stop detection is divided into two phases:

Identifying stop candidates according to stop detection algorithm

Reconciliation of stop candidates taking into account mobility indexes of points.

2.1.1 Identifying stop candidates

Stop algorithm, shown on Figure 2.1, is considering 3 parameters - MinWalkSpeed, MinTransportSpeed, ThresholdDistance.

So called ThresholdDistance is distance mentioned in subsection 1.4.2 (which in case of Berlin been in range 800-1500m) as distances between points, where there is significant "jump" in frequency of occurences, which might mean that at these intervals, continuous points been gathered, and distances over that values might be discontinuous. Furthermore, these distances are relatively small compared to other observed distances, allowing more accurate decision process. We assume, that at these distances, no significant "interuption" in gatharing the points has been introduced in the form of buildings, metro or other factors.

Thus, for these higher accuracy points one could assume, that if mobile device have been moving within the minimum walking speed MinWalkSpeed (ref. section 1.2), this point is identified as stop candidate.

Thus, for lower accuracy points one could assume, that if mobile device have been moving within the minimum transport speed MinTransportSpeed (ref. section 1.2), this point is as well identified as stop candidate. In this example, it could mean, that over longer distance, one

could move with transportation with significant speed, stop for longer period and then use fast transportation again, resulting in higher then minimum walking speed, but with speed which is in average lower then other psychological value as minimum transport speed or maximum walking speed.

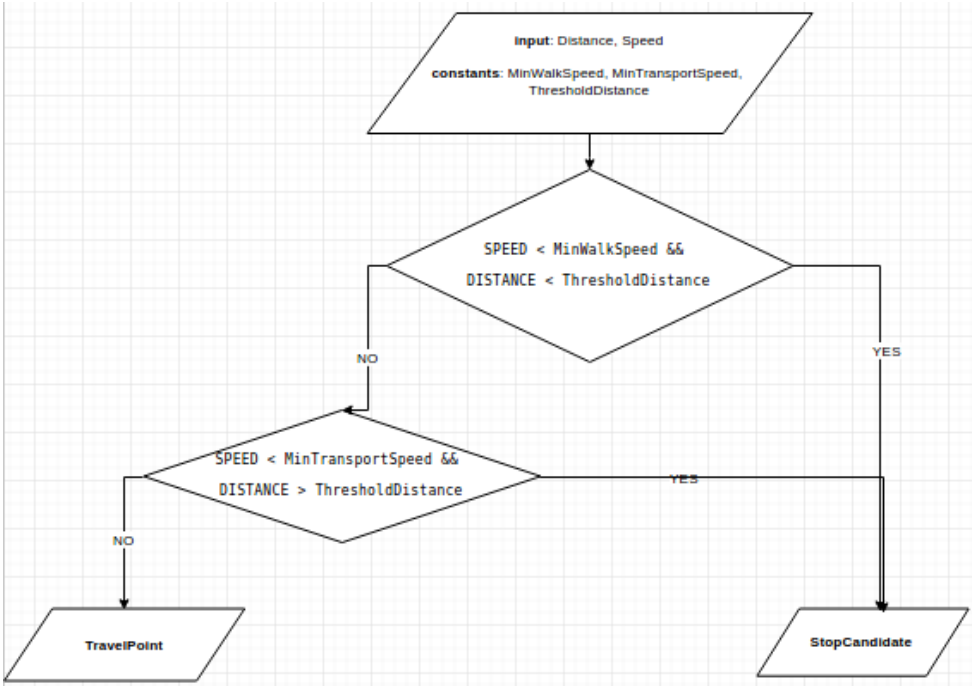

Figure 2.1: First phase of stop detection algorithm

2.1.2 Mobility Index Analysis

Figure 2.2 shows that there are many scenarios in which stop candidate has been detected, taking into account movement speed between points, distance and mobility index. From the statistics about collection of points per user - subsection 1.4.2) - the following movement profiles can be derived:

Standard daily movements with few relocations across the city - In average mobile devices collected 17 points. The profile of this person could be matching e.g. - going to work in the morning, going for lunch and back, going to the shop/entertainment after work and home. This could result in points collected during transportation to those places (5-40).

Relocations over small distances - harmonic mean has been 4 points, which could mean that person during that day was indeed moving, but frequently requiring "area updates", thus distance covered daily was small.

Many relocations over larger distances - number of points collected in range above 60, might mean that user is moving a lot during the day (e.g. delivery, professional driver) or moves over long distance to another city.

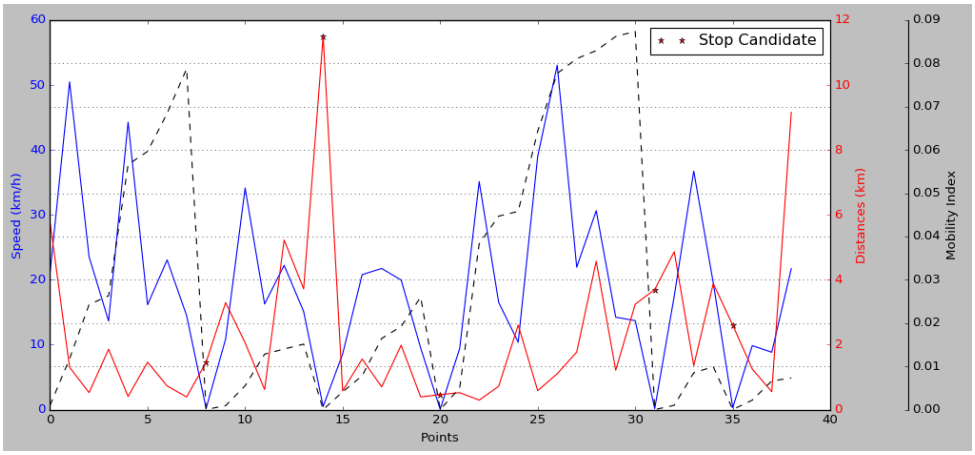

Figure 2.2: Result of identifying stop candidates for single user with their corresponding mobile index, distance and speed between the detected and previous point

In the example for a single user over 24h shown on Figure 2.2, stop detection algorithm identified 5 stops. The dashed line represents the mobility index which was considering past 60 minutes from each of the points, representing how movable was that person in the last 60 minutes till that point. Analyzing mobility index in the above example (ref. subsection 1.5.3), we identify "rising mobility periods" as periods of mobility/transportation, while rapid drop, as very low mobility/stay indicator.Worth notifying is that mobility index in that example very accurately correlates with the identified stop candidates.

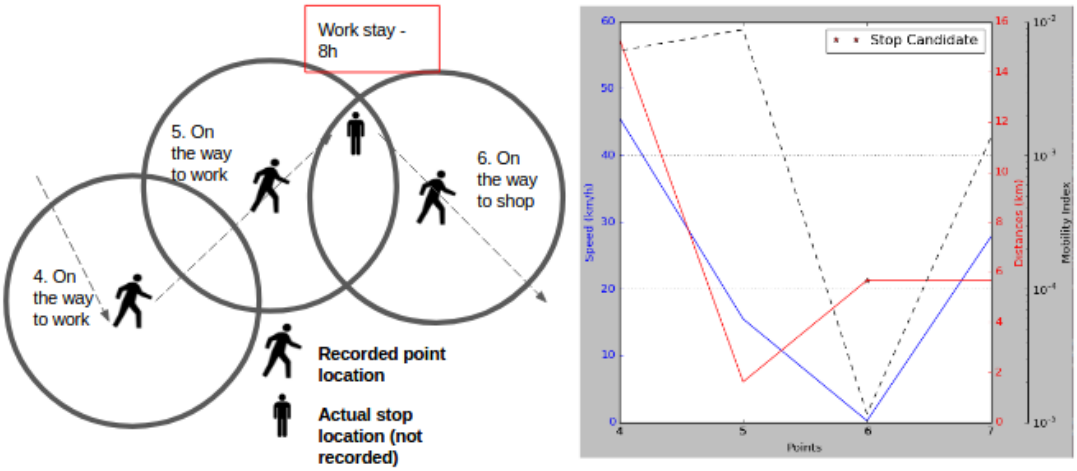

Figure 2.3: Correlation of the real life example (left) to the graph of distance, speed and mobility index between consequtive points (right)

Due to the method of obtaining the data - Figure 2.3 - one cannot precisely identify the location of the stop, it can be closer to Point 5, Point 6 or somewhere in the middle the two. Furthermore, statistically, most stops identified have mobility index below certain threshold **(10e-3)** - which can be taken as a point of reference for other considerations - meaning that user continued movement, but with low speed.

2.1.3 Reconciliation of stop candidates

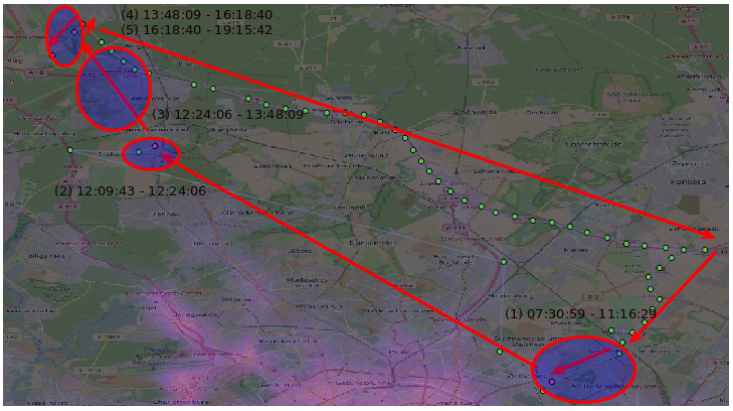

Figure 2.4: Movement example for one user in the period of 24h, with recorded locations due to the movement and annotated timestamps for detected stop candidates

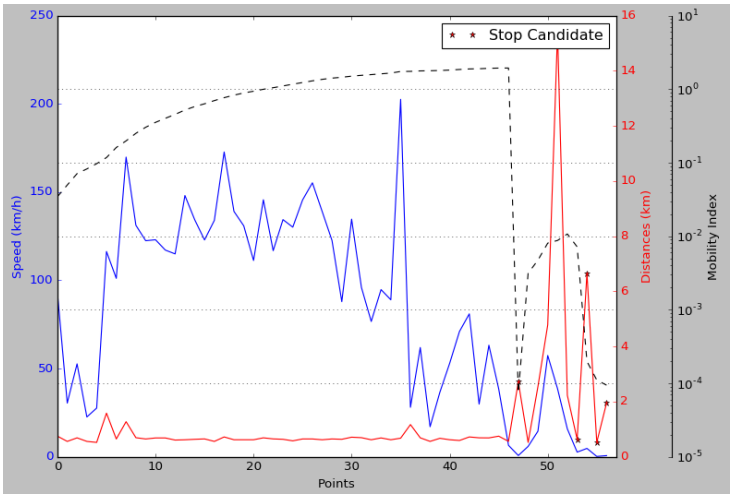

Figure 2.5: Visualization of distances, speed and mobility indexes for example from Figure 2.4 with identified stop candidates

Analyzing traces from Figure 2.4 and Figure 2.5, we identify that user has been moving from Point 0 to Point 47, in the time period 06:59 - 07:30. In between Point 47 and Point 48, the stop candidate has been identified. We notice, that from Points 0-47, measurement has been continuously gathered over a distance. This could mean, that user stopped somewhere around Point 47, stayed there some time, and when he left that location, after around 3 km from that point, at 11:16 user has been recorded in location Point 48, starting his journey to Point 54. Thus, Stop Candidate at Point 48 is meaning in that case Stop might be close to Point 47 on that way (previous to stop candidate).

At Point 54, we recorded Stop Candidate, however, considering that its mobility index being high (above 10e-3, also validated by small distance and walking speed covered between 12:09-12:24) and next mobility index being low, we can assume that this Stop Candidate is not

a stop and have to be ignored, it is most likely approaching to the stop location.

In between Points 54-55, we record another Stop Candidate, in period from 12:24-13:48 and over significant distance. Recorded mobility index is low, and because it is over long distance, we assume that Stop is close to Point 54 on the way to Point 55.

Due to the small distance covered in between Points 55-56 and Points 56-57, we expect that the Stop Points in that cases are somewhere in between or around these Points.

Figure 2.6: Visualization of distances, speed and mobility indexes for example from Figure 2.7 with identified stop candidates

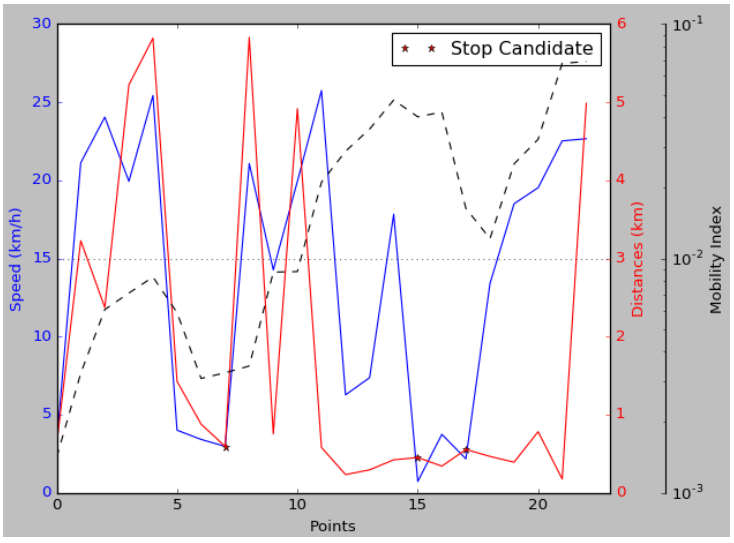

Figure 2.7: Movement example for one user in the period of 24h, with recorded locations due to the movement and annotated timestamps for detected stop candidates

Analyzing traces from Figure 2.7 and Figure 2.6 we identify example in which person is constantly moving between 8:28-12:04, and having 3 short stops in range 15-40 minutes, in Points 7, 15, 17. Mobility index in both examples is above the mobility index threshold, however, in the next point that person is again moving (above mobility index threshold), which might identify short stay at that location.

Further examples, showing the above mentioned events can be found in chapter 4.1.1, chapter 4.1.1

TODO: Now should decide what should be algorithm based on above mentioned values, most probably for:
-> mobility index higher then the threshold, and next point mobility index being lower then the threshold, there is no stop, but slow movement to possible stop location.
-> mobility index higher then the threshold, and next point mobility index being also higher then the threshold might mean that there is short stop at the location
-> short distance covered, stop is either in both current and previous or in between
-> for long distance the stop is close to previous point

2.2 Clustering of Stops

Clustering can be described as "*the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters)*" **[clustering]**.

After our stop detection algorithm has detected the proper stops we want to perform some analysis on these. Our goal is to get the overall movement patterns of the population of a certain area, which means we are interested in **macroscopic** movements and not microscopic (individual) movement. If we would consider every individual our analysis algorithms would get fed with a lot of noise (outliers) which would make our data mining and prediction difficult and inaccurate. Meaning, *Macroscopic Movements* are not interesting in tracking individuals but rather seeing the big picture.

Once our clustering algorithm has been applied, and we have the overall picture, we can see which stops are most popular, where people are moving from them, and how long they stay at a certain location. One could then perform some interesting graph analysis on these locations, answering questions such as "what is the probability that a person moves from point A to B?", or "what is the average stop time at location A?".

To achieve these clusters, which represents our interesting stop locations, we need to apply some clustering algorithm. There are different types of clustering (such as centroid, distribution, or density-based), and different algorithms with parameters that can be applied (such as K-means, DBScan, or OPTICS). However, they all work to achieve the same goal of structuring (clustering, grouping) similar objects and neglecting outliers from these groups. The algorithm and parameters are application sensitive and our choice will be handled in the *evaluation* section.

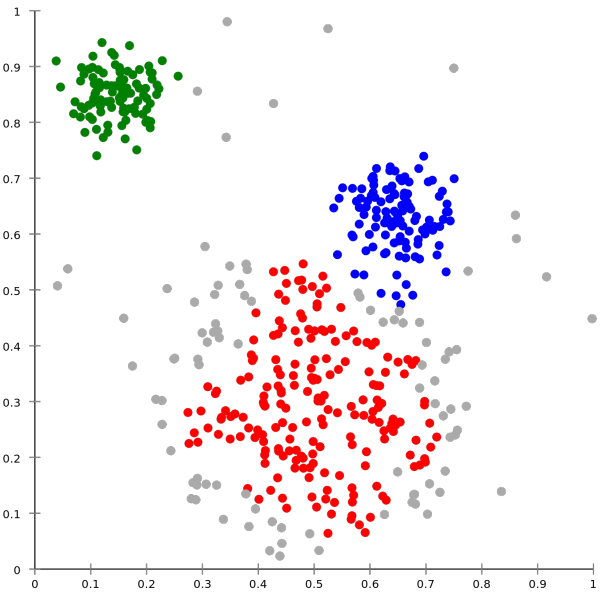

Figure 2.8: Clustering applied to a dataset.

2.3 City Movements Graph

3 Implementation

We need to describe here some details about the implementation, project structure, configuration of the environment, how to use it, important implementation aspects like why did we choose these parameters not the others etc and how did we implement Graphical User Interface.

3.1 Clustering algorithm

For clustering we started off with DBScan, described in **??**. The difficulty with DBScan is figuring out the two parameters, the minimum points per cluster (*minPts*) and the radius of the clusters (*eps*).

3.1.1 DBSCAN on Spark

We wanted a scalable implementation of DBSCAN that could handle large data sets, preferable in parallel. We decided to use Apache Spark, a scalable engine for large-scale data processing [1]. For this we used an implementation by Irving Cordova, based upon an algorithm called MR-DBSCAN built for the MapReduce framework [2]. The implementation of DBScan runs in parallel by splitting the data space into boxes, using the number of boxes as a cost estimator for the algorithm. Each box then grows to include one *eps* in it. After each box and its points has been determined the traditional DBScan algorithm is run on the points in each box. Finally it examines the intersection points betweeen boxes and merges the result together [3]. The figures below visualises the execution steps.

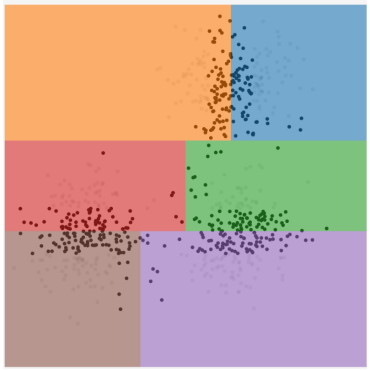

Figure 3.1: Step 1. DBSCAN on Spark assign the data space into boxes.

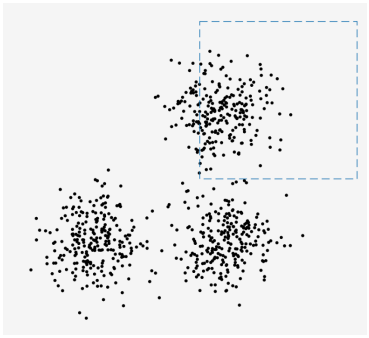

Figure 3.2: Step 2. Each box grows to include the points that are within one *eps* of it.

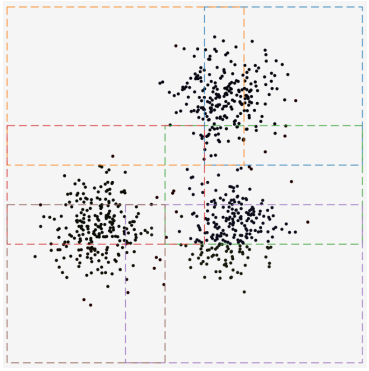

Figure 3.3: Step 3. Traditional DBScan algorithm is applied in parallel for each box. Each different color represents a different cluster.

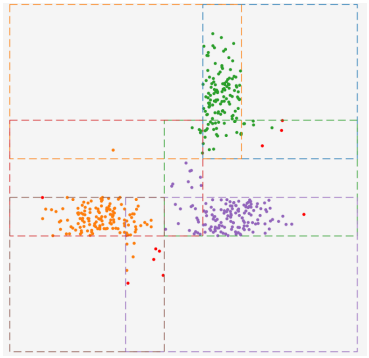

Figure 3.4: Step 4. After DBSCAN is done, all points witihin the borders of two clusters are examnied. If they are part of a cluster within two boxes they are merged to one cluster.

Figure 3.5: Step 5. Finally all remaning points are labeled with the global identified cluster and we are done. (red points are noise points).

4 Evaluation

We need to describe here some details about how did we evaluate that our graph is correct e.g. show-casing screens from user interface or QGIS etc. Then we need to show how did we test our code, if it scales, and maybe some performance measurements

4.1 Clustering algorithm

In this section we will describe how we evaluated our dbscan algorithm and the choice of the *epsilon* and *minimum points per cluster* parameter.

4.1.1 DBSCAN

We visualised our results in QGIS, a powerful graph visualisation tool. Running the dbscan with different parameters resulted in different cluster sizes and number of clusters. We started off with the target area of Berlin. The *epsilon* parameter (radius of the cluster) was chosen by looking at the graininess (accuracy) of our given data set. The accuracy between points was 110 meters, this distance is refered to as *accuracy distance*. We took Alexanderplatz in Berlin as an example cluster (a popular train stop area in Berlin). We interpret one cluster as being one stop point of interest, and for this application want Alexanderplatz to be represented as one cluster. Setting the *epsilon* parameter to the *accuracy distance*, 110 meters, gave us good looking clusters whilst higher value of the parameter resulted in clusters being unnaturally large and distance below the *accuracy distance* resulted in only single-point clusters (points are stacked at the same location). Note that the *minPts*, minimum points per cluster, parameter is not taken into *consideration* here (it is set arbitary and only determins the number of clusters, while we here are interested in the *size* of the clusters). See figures below.

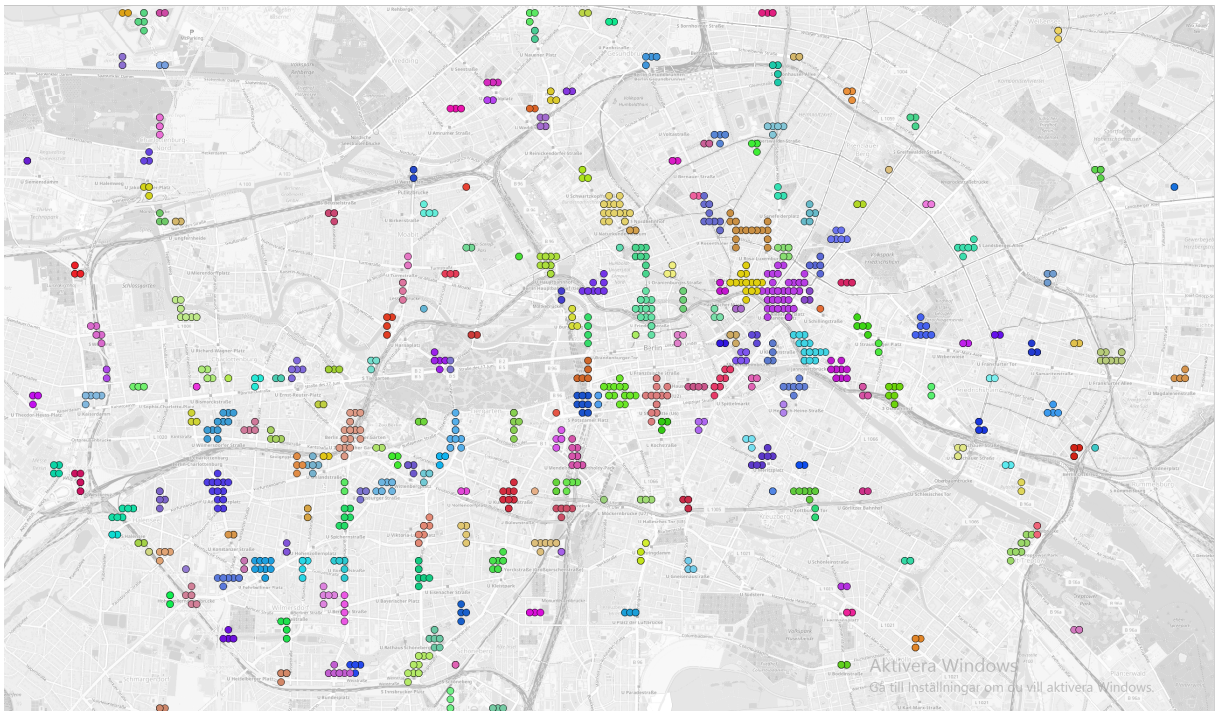

Figure 4.1: Clusters with good parameters *epsilon* = 110 meter, *minPts* = 5. Alexanderplatz (pink) has 85 data points in it.

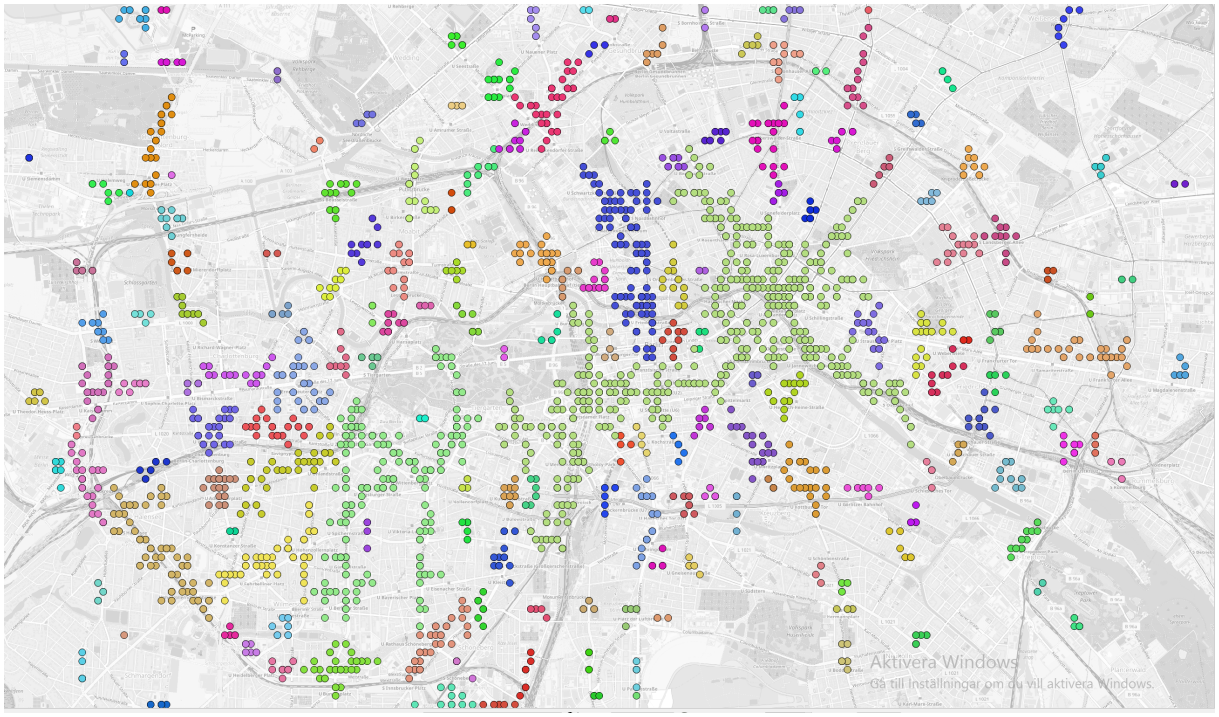

Figure 4.2: Example of a clusters with bad parameters, here *epsilon* = 220 meter is too large, *minPts* = 5. "Alexanderplatz" (light green) has 748 data points in it and stretches over the whole centre (Mitte) of Berlin.

Bibliography

[1] *Apache Spark*. URL: https://spark.apache.org/ (visited on 06/18/2017).

[2] Irving Cordova. *DBSCAN on Spark*. URL: https://github.com/irvingc/dbscan-on-spark (visited on 06/18/2017).

[3] Irving Cordova. *Visualizing DBSCAN on Spark*. URL: http://www.irvingc.com/visualizing-dbscan (visited on 06/18/2017).

[4] Adriano Moreira Filipe Meneses. "Using GSM CellID Positioning for Place Discovering". In: *Pervasive Health Conference and Workshops* (2006). DOI: 10.1109/PCTHEALTH.2006.361692. URL: http://ieeexplore.ieee.org/abstract/document/4205183/.

[5] Minsu Shin Injong Rhee. "On the Levy-walk Nature of Human Mobility". In: *IEEE/ACM Transactions on Networking (TON)* (2011). DOI: 10.1109/TNET.2011.2120618. URL: http://dl.acm.org/citation.cfm?id=2042974.

[6] Yong Yang. "Walking Distance by Trip Purpose and Population Subgroups". In: *PMC* (2013). DOI: 10.1016/j.amepre.2012.03.015. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3377942/pdf/nihms369465.pdf.

Appendices

Appendix 1

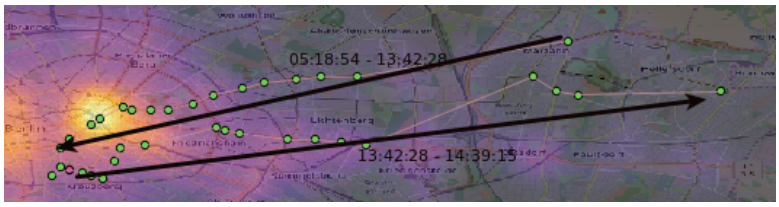

Figure 1: Visualization of distances, speed and mobility indexes for example from Figure 2 with identified stop candidates

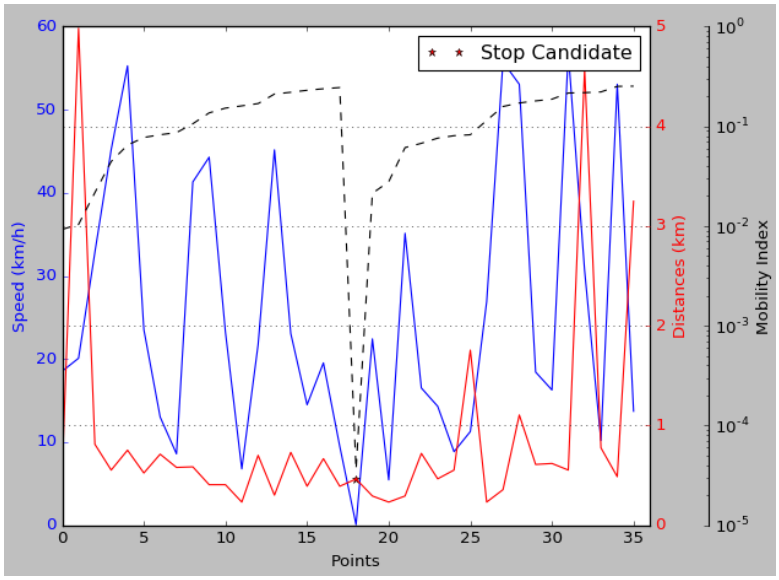

Figure 2: Movement example for one user in the period of 24h, with recorded locations due to the movement and annotated timestamps for detected stop candidates

Appendix 2

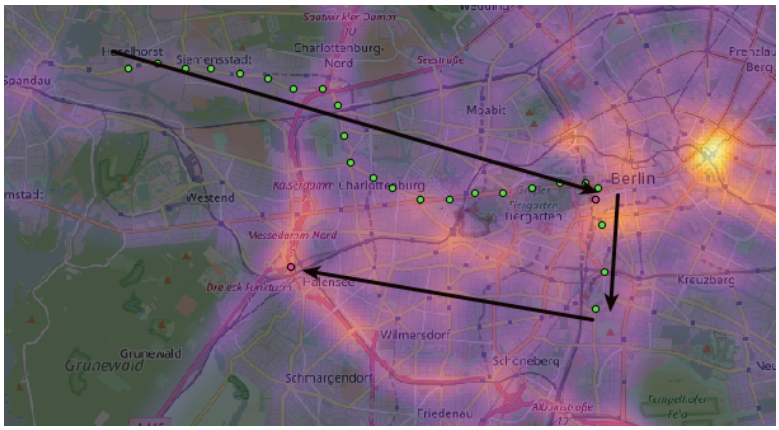

Figure 1: Visualization of distances, speed and mobility indexes for example from Figure 2 with identified stop candidates

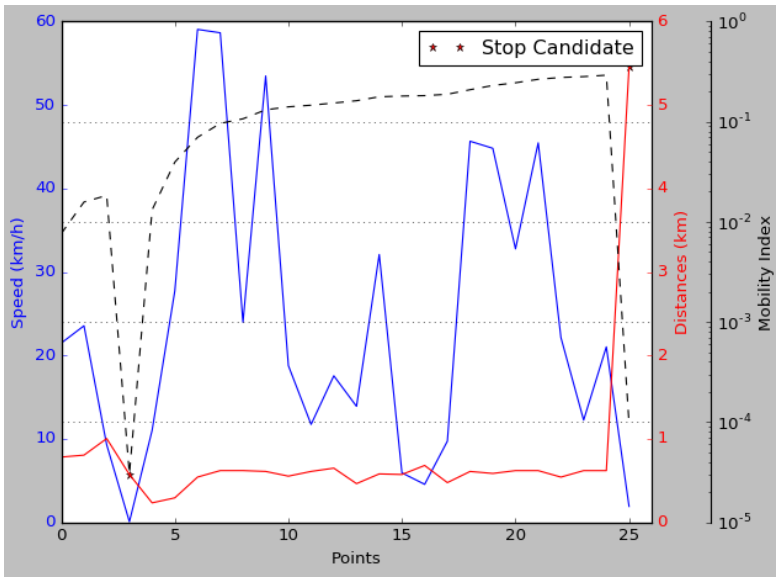

Figure 2: Movement example for one user in the period of 24h, with recorded locations due to the movement and annotated timestamps for detected stop candidates