Location Prediction Algorithm

Shashank Sharma

August 2018

1 Introduction

This algorithm is designed to predict human locations in a real world scenario. The GPS data is taken as input and the processed using the below algorithm.

The Algorithm has several steps:

- Detect stay-points (also detect start or end of the trajectory)
- Group stay-points to form states
- Calculate hourly weights for the states
- Apply Markov chain for the data available

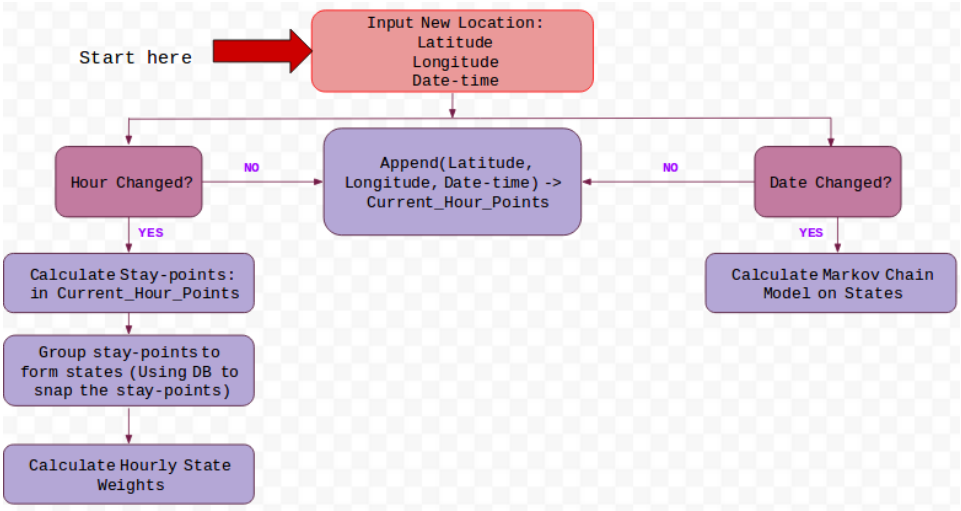

Figure 1: Algorithm Flow-chart

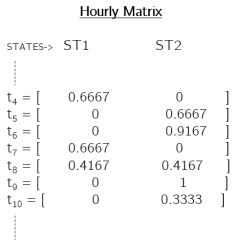

Figure 2: Algorithm Flow-chart

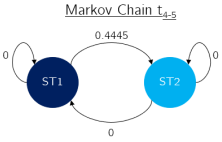

Figure 3: Algorithm Flow-chart

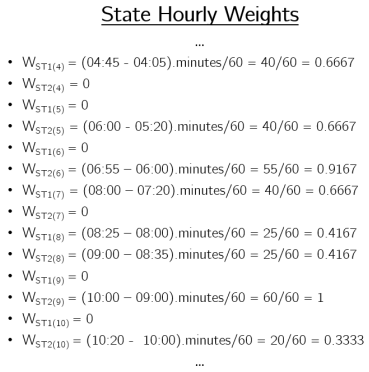

Figure 4: Algorithm Flow-chart

2 Definitions

- **Stay-points:** Stay-points are any points which are stayed by the user in user trajectories or it is the start or the end of the trajectory. For example, if user start at his home, the home itself is a stay-point. Now he move towards work, but he visit a cafe in between for breakfast. The cafe is also a stay-point and then he finishes his trajectory at work, where work is again a stay-point. The places like cafe in this case is identified using distance and time based clustering. For example, a set of points within 200m with total duration of stay greater than 20 minutes can be regarded as a stay-point within the trajectory.

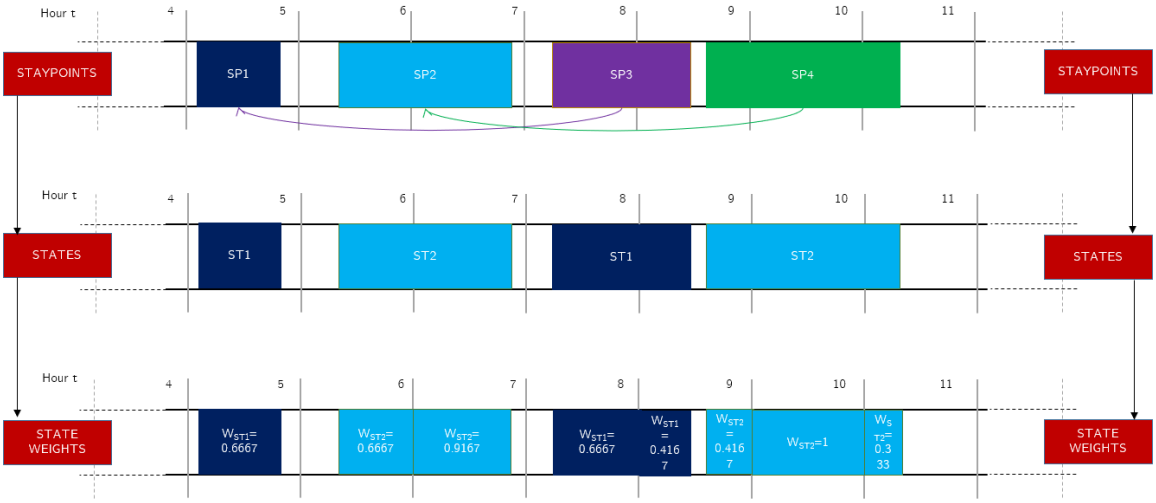

Figure 5: Algorithm Flow-chart

- **State:** A state is formed using a group of stay-points. This is done using a distance threshold for states. All the stay-points within this threshold distance are grouped together as a single state. This is called snapping stay-points to the states. The mean of all location latitudes and longitudes from stay-points within a state are stored per state. Finally Markov Chain model is applied to the states. *Note: A new stay-point is only added to the state if after calculating the mean of the new state, all the existing stay-points still stay within the distance threshold from this mean. This is done to avoid drifting problem while aggregating the stay-points into states.*

3 Algorithms

Algorithm 1 Read new location and process

Input: GPS Points *P[x, y, d]*
Output: Markov Chain *mc*

for each *P[x, y, d]* **do**
  $newHour \leftarrow d_h$
  $newDate \leftarrow d_d$
  if $newHour! = prevHour$ **then**
    $prevHour \leftarrow newHour$
    extractStayPoints()
    formStates()
    calculateStateWeights()
    if $newDate! = prevDate$ **then**
      $prevDate \leftarrow newDate$
      updateMarkovModel()
    end if
    clear *lst_hr_pts[x, y, d]*
  else
    $lst\_hr\_pts[x, y, d] \leftarrow P[x, y, z]$
  end if
end for

Algorithm 2 extractStayPoints() : Calculate stay-points

Input: Last hour GPS points *lst_hr_pts[x, y, d]*
Output: Stay-points *sp*

for each $lst\_hr\_pts[x, y, d]_i$ in *lst_hr_pts[x, y, d]* **do**
  if $d_{\mathrm{i}} - d_{\mathrm{i}+1} > th\_tck$ **then**
    $sp \leftarrow lst\_hr\_pts_{\mathrm{i}}, lst\_hr\_pts_{\mathrm{i}+1}$
    adjustStartEndStaypoint()
    continue
  end if
  if $distanceBtw(lst\_hr\_pts_{\mathrm{i}}, cluster) <= th\_d$ **then**
    $cluster \leftarrow lst\_hr\_pts_{\mathrm{i}}$
    calculate New Cluster Mean
  else
    if $(cluster! = empty)$ And $duration(cluster) >= th\_t$ **then**
      $sp \leftarrow cluster$
      adjustStartEndStaypoint()
    end if
  end if
end for

Algorithm 3 adjustStartEndStaypoint() : Adjust start-end time of stay-points

Input: Stay-points sp
Output: Stay-points with adjusted start and end points sp

for each sp_i in sp **do**
    $distance \leftarrow distanceBtw(sp_{\mathrm{i}}, sp_{\mathrm{i+1}})$
    $time \leftarrow timeDifference(sp_{\mathrm{i}}, sp_{\mathrm{i+1}})$
    $avgSpeed \leftarrow distance/time$
    $delta_t \leftarrow min(distance, thresholdDistance)/AvgSpeed$
    Update endTime d_e+*delta_t/2* for sp_i
    Update startTime d_s-*delta_t/2* for sp_{i+1}
end for

Algorithm 4 formStates() : Form states from stay-points

Input: Stay-points sp
Output: States st

for each sp_i in sp **do**
    for each sp_j in sp **do**
        if $distanceBtw(sp_{\mathrm{i}}, sp_{\mathrm{j}}) <= th\_d$ **then**
            Calculate new mean latitude and longitude for st with sp_i
            $newMean \leftarrow mean(st)$
            if $distanceBtw(st, newMean) <= th\_d$ **then**
                Combine sp_i and sp_j in st
            else
                Form new st from sp_j
            end if
        end if
    end for
end for

Algorithm 5 calculateStateWeights() : Calculate Hourly Weights of States

Input: States st
Output: States Weights w

This creates a weights of all states from 0 Hrs to 24 Hrs for each date
for each st_i in st **do**
    $w_{\mathrm{i}} \leftarrow minutesContribution(st_{\mathrm{i}})/60$
end for

Algorithm 6 updateMarkovModel() : Recalculate the Markov Model(This algorithm creates the transition probabilities from state i to i+1 from hour j to j+1)

Input: State Weights w
Output: Markov Chain Model mc

1: **for** each $h - hour$ from $0 to 24$ **do**
2: **for** each w_i in w **do**
3: **for** each w_j in w **do**
4: $mc \leftarrow w_{\mathrm{i}}[h] * w_{\mathrm{j}}[h+1]$
5: **end for**
6: **end for**
7: **end for**