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

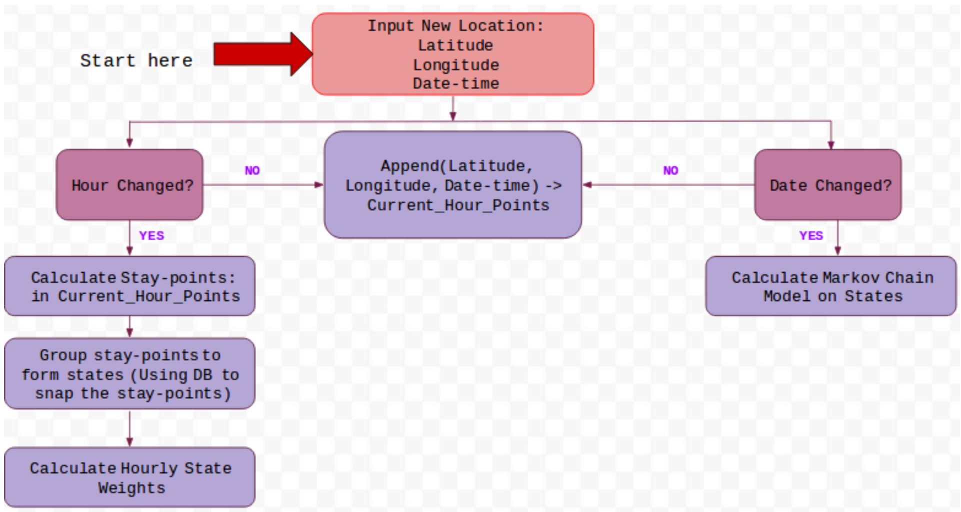

Figure 1: Algorithm Flow-chart

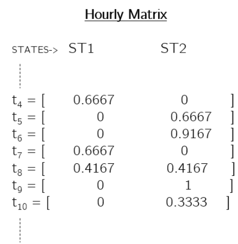

Figure 2: Algorithm Flow-chart

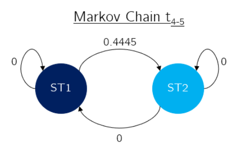

Figure 3: Algorithm Flow-chart

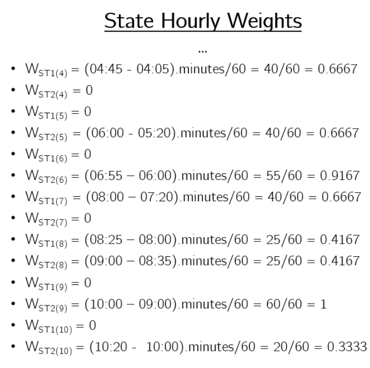

Figure 4: Algorithm Flow-chart

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.

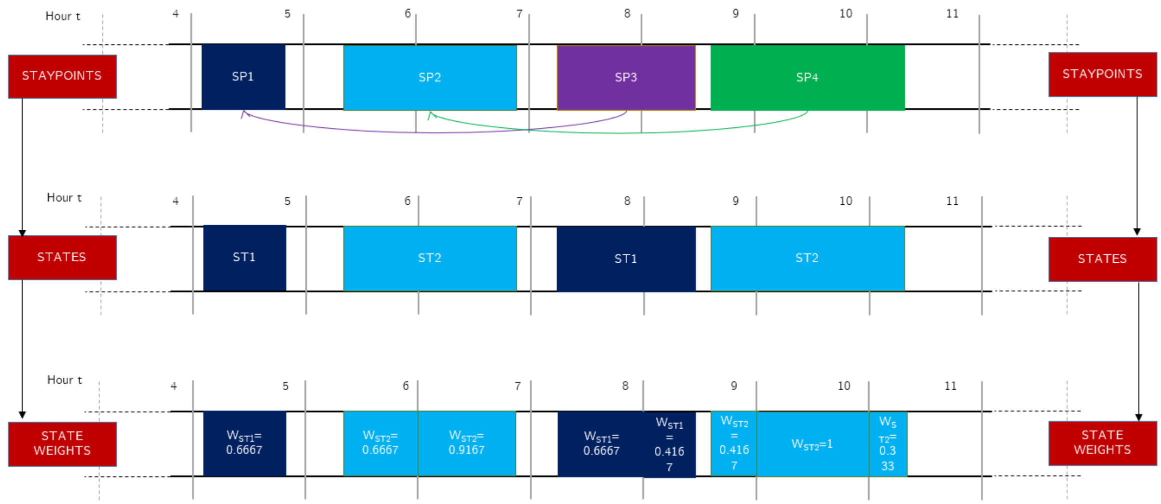

Figure 5: Algorithm Flow-chart

- **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

```
for each newpoint[x, y, d] do
    newHour ← d_h
    newDate ← d_d
    if newHour != prevHour then
        prevHour ← newHour
        calculateLastHourStayPoints()
        formStates()
        calculateStateLastHourWeights()
        if newDate != prevDate then
            prevDate ← newDate
            recalculateMarkovModel()
        end if
    end if
end for
```

Algorithm 2 calculateLastHourStayPoints() : Calculate last hour stay-points

```
for each lst_hr_pts[x, y, d] do
    if d(i) − d(i − 1) > th_tck then
        sp ← lst_hr_pts(i), lst_hr_pts(i − 1)
        recalculateStartEndStaypoint()
    end if
    if distance(lst_hr_pts(i), cluster) <= th_d then
        cluster ← addlst_hr_pts(i)
        calculate Cluster Means
    else
        if (cluster != empty) And duration(cluster) >= th_t then
            sp ← cluster
            recalculateStartEndStaypoint()
        end if
    end if
end for
```

Algorithm 3 recalculateStartEndStaypoint() : Calculte start-end of staypoints

```
for each sp do
    distance ← distance(sp(i), sp(i + 1))
    time ← timeDifference(sp(i), sp(i + 1))
    AvgSpeed ← distance/time
    AddTime ← min(distance, thresholdDistance)/AvgSpeed
    Update endTime for sp(i)
    Update startTime for sp(i+1)
end for
```

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

for each st **do**
  if $distance(st(i), st(i+1)) <= th\_d$ **then**
    calculate new state mean latitude, mean longitude
    if distance(All (orig lat, orig lon) forming st(i) , st(i+1)) <= th_d **then**
      combine st(i), st(i+1)
      calculate combined st Means
    end if
  end if
end for

Algorithm 5 calculateStateLastHourWeights() : Calculate Hourly Weights of States

This creates a weights of all states from 0 Hrs to 24 Hrs for each date
for each st **do**
  if ($HourChanges$) Or ($StateIDChanges$) **then**
    $st\_hr\_wt \leftarrow st(i), startTime, endTime$
  end if
end for

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

for each $jth - Hour$ from $0to24$ **do**
  for each ith-st **do**
    $mc \leftarrow st\_hr\_wt[i][j] * st\_hr\_wt[i+1][j]$
  end for
end for