Algorithms

Dan Champagne

April 2022

1 Introduction

Algorithm 1 Seed Algorithm: User Profile Setup

$$\mathcal{O}(n * q) \approx \mathcal{O}(n)$$

Require: $users \leftarrow$ List of user names
Require: $questions \leftarrow$ All possible survey questions
Ensure: $users > 0$
Ensure: $questions > 0$

```
for all users do
    for all questions do
        if QuestionType = MultipleChoice then
            for all possible answers to question do
                if random number is even then
                    selectedAnswers ← selectedAnswers + answer
                end if
                if selectedAnswersisempty then
                    selectedAnswers ← firstpossibleanswer
                end if
            end for
        end if
        if QuestionType = SingleChoice then
            selectedAnswer ← random answer
        end if
        if QuestionType = YesNo then
            if random number is even then
                selectedAnswers ← "Yes"
            else
                selectedAnswers ← "No"
            end if
        end if
    end for

    execute sp_RecalculateProfileMatch(user)
end for
```

Algorithm 2 Seed Algorithm: Location Data

$$\mathcal{O}(u * L + 2L) \approx \mathcal{O}(n^2)$$

Require: $users \leftarrow$ List of user without any location records

$[Min, Max]Latitude \leftarrow [41.383920730000604, 42.0420540561665]$
$[Min, Max]Longitude \leftarrow [-71.90395649891572, -73.48324099819017]$
$[Min, Max]PositionVariability = [0.000001, 0.01]$
$MaxLocations \leftarrow 100000$
$TimeVariabilityMinutes \leftarrow 1$

$startTime \leftarrow currenttime$

for all $users$ **do**
  $buffer \leftarrow \{\}$
  $t \leftarrow startTime$
  $position \leftarrow (0, 0)$
  for $i \leftarrow 1$ to $MaxLocations$ **do**
    if $position = (0, 0)$ **then**
      $position \leftarrow$ random values between min/max latitude/longitude
    else
      $r \leftarrow$ random number
      if r is even **then**
        $lat \leftarrow$ random double between min/max PositionVariability
        $lon \leftarrow$ random double between min/max PositionVariability
        if $random$ is even **then**
          $position.Latitude \leftarrow position.Latitude + lat$
        else
          $position.Latitude \leftarrow position.Latitude - lat$
        end if
        if $random$ is even **then**
          $position.Longitude \leftarrow position.Longitude + lon$
        else
          $position.Longitude \leftarrow position.Longitude - lon$
        end if
      end if
      $buffer \leftarrow$ generated location
      $t \leftarrow t + TimeVariabilityMinutes$
    end if
  end for
  for all $locations$ in buffer **do**
    $location_i \leftarrow$ calculated speed from $location_{i-1}$
    $location_i \leftarrow$ calculated rolling average speed from $location_{i-1}$ to $location_{i-10}$
  end for
  execute sp_ProcessLocationRecord($buffer$)
end for

Algorithm 3 GPS Coordinate pre-calculation

$$\mathcal{O}(n+p) \approx \mathcal{O}(n)$$

Require: $C \leftarrow$ decimal places to round to

for each inserted location record **do**

lastRollupRecord ← last roll-up record for current user
lastLocationRecord ← last location record for current user
lastTenLocationRecord ← last 10 location record for current user

currentRecord.Latitide ← *currentRecord.Latitide* rounded to C decimal places
currentRecord.Longitude ← *currentRecord.Longitude* rounded to C decimal places
speedFromLast ← MPH between *currentRecord* and *lastLocationRecord*
rollingAverage ← average MPH from *lastTenLocationRecord*

if *lastRollupRecord* is null **then**
Precalc_Locations table ← *currentRecord*
else if *speedFromLast* **between** (0.5 ∗ *rollingAverage*) **and** (1.5 ∗ *rollingAverage*) **then**
if *lastRollupRecord.Latitude* = *currentRecord.Latitude* **and** *lastRollupRecord.Longitude* = *currentRecord.Longitude* **then**
lastRollupRecord.EndTime ← *currentRecord.Timestamp*
else
Precalc_Locations table ← *currentRecord*
end if
end if
end for

Algorithm 4 Profile Match pre-calculation

$$\mathcal{O}(n + n * q) \approx \mathcal{O}(n)$$

Require: $L \leftarrow$ Survey responses for current user
Require: $R \leftarrow$ Survey responses for all other user

$Rows \leftarrow L$ joined to R on $QuestionId$

for each *row* **in** *Rows* **do**
  if both users' responses match **then**
    $row.ResponsesMatch \leftarrow True$
    if both users' response weight matches **then**
      $row.Scale \leftarrow 1.0$
    else
      $row.Scale \leftarrow \frac{max\ weight - (L.weight - R.weight)}{max\ weight}$
    end if
  end if
end for

Summarize Results and store in Precalc_ProfileMatches:

for each *user pair* **in** *Rows* **do**
  $Q \leftarrow$ Total number of questions
  $M \leftarrow$ Total number of matched questions
  $RawMatchPercentage \leftarrow \frac{M}{Q}$
  $WeightedMatchPercentage \leftarrow \frac{user.Scale * M}{Q}$

  $Precalc\_ProfileMatches \leftarrow currentRecord$
end for

Algorithm 5 Final User Match Algorithm

$$\approx \mathcal{O}(n^2)$$

Require parameter: $UserId$
Require parameter: $StateDate$
Require parameter: $EndDate$
Require parameter: $MinDistance$
Require parameter: $MinTime$

$L \leftarrow$ rows from$Precalc_Locations$ for $UserId$ between $StartDate$ and $EndDate$
$R \leftarrow$ rows from$Precalc_Locations$ for all other users between $StartDate$ and $EndDate$
$X \leftarrow L$ joined to R on *overlapping times*

for each row **in** X **do**
    $row.Distance \leftarrow$ arc distance between L and R
    $row.OverlapTime \leftarrow$ time between L and R
end for

$Results \leftarrow X$ filtered by:
    $Distance \leq MinDistance$
    $OverlapTime \geq MinTime$
    Genders are mutually attracted