Comparison of User Trajectories with the Needleman-Wunsch Algorithm

Maroš Čavojský(✉) and Martin Drozda

Faculty of Electrical Engineering and Information Technology,
Slovak University of Technology, Bratislava, Slovakia
{maros.cavojsky,martin.drozda}@stuba.sk

Abstract. We show that the Needleman-Wunsch algorithm for sequence alignment can be efficiently applied to comparing user trajectories, where user locations are provided by Global positioning system (GPS). We compare our approach based on this algorithm with other approaches such as the pairwise method and the proximity method. We describe all steps necessary to apply the Needleman-Wunsch algorithm when comparing user trajectories. In our experiments we use two different data sets: a data set that we collected with 455 mobile devices distributed among our students and the Geolife data set (Microsoft Research Asia). We conclude that our approach based on the Needleman-Wunsch algorithm performs better than other approaches, especially, in terms of true negatives, false positives and false negatives, while still offering improvement in terms of true positives.

Keywords: GPS · Needleman-Wunsch algorithm · Sequence alignment · User movement patterns · Experimental evaluation

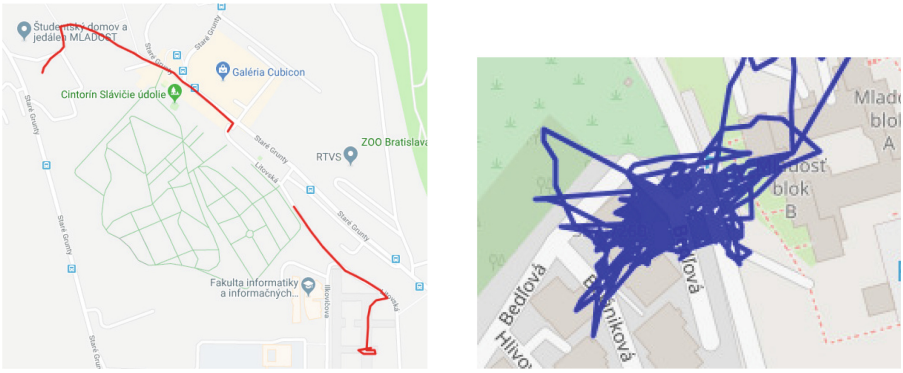
1 Introduction

We investigate whether the Needleman-Wunsch algorithm for sequence alignment can perform well for comparison of user trajectories. Comparing user trajectories is relevant when there is a need to group users by visited locations with sequential dependence. Such a grouping has a wide range of applications such as navigation, location recommendation, friend recommendation etc.

The Needleman-Wunsch algorithm was published in 1970 [11]. It is one of the first applications of dynamic programming, where a large problem is divided into a series of smaller problems to reconstruct a solution to the larger problem. The Needleman–Wunsch algorithm is still widely used for optimal global alignment, particularly when the quality of the global alignment is of the utmost importance.

The algorithm takes as input two sequences, score matrix and gap penalty. The objective is to align these two sequences by matching letters and introducing gaps, where score with respect to matched letters and penalty for introducing gaps are computed. Matching only identical letters can be desirable, however,

Y. Yin et al. (Eds.): MobiCASE 2019, LNICST 290, pp. 141–154, 2019.
https://doi.org/10.1007/978-3-030-28468-8_11



(a) Gap - missing location data (b) Nest - false movement

Fig. 1. Gap and nest.

in bio-informatics score is often set according to mutation probabilities, where matching certain letters can have a higher score than matching other letters. It is also often desirable that alignment has few small gaps, therefore penalty for starting a gap can also be introduced. Let us consider the following sequences:

Sequence 1: G A T T A C A
Sequence 2: G T C G A C G

An alignment for these two sequences, when only identical letters are allowed to match, can be as follows:

Sequence 1: G - T - - C G A C G
Sequence 2: G A T T A C - A - -

We argue that the Needleman-Wunsch algorithm can be efficiently applied for aligning user trajectories defined by coordinates received with GPS (Global Positioning System). When comparing user trajectories, one has to cope with the following phenomena:

- Gaps arise when acquisition of GPS coordinates is interrupted; such gaps naturally map to gaps in the sequence alignment problem.
- Nests arise when signal reflection and multi-path propagation negatively impact GPS signal reception. This phenomenon can be perceived as a random walk around the user's real location.

Examples of these two phenomena are shown in Figs. 1(a) and (b). Our ambition is to show that gaps and nests can be efficiently addressed with algorithms for sequence alignment, in our case with the Needleman-Wunsch algorithm.

The rest of this document is organized as follow. In Sect. 2 we introduce the Needleman-Wunsch algorithm, Sect. 3 has relevant definitions necessary for discussing and introducing our approach and results, in Sect. 4 we review the

related work, in Sect. 5 we introduce our approach to user trajectory comparison, in Sect. 6 we present the applied experimental setup, Sect. 7 contains the obtained results, and finally, in Sect. 8 we conclude and give suggestions for possible future work.

2 Needleman-Wunsch Algorithm (NWA)

NWA is a variant of string-editing algorithm, where the objective is to maximize alignment scores along the entire length of two sequences. Let $x_1, x_2, ..., x_m$ and $y_1, y_2, ..., y_n$ be two sequences, one having m letters and the other n letters. The scoring schema S defines scores when letters match or mismatch, for example, $S(G, G) = 1$ or $S(G, T) = -1$. In its simplest form, S returns score 1 for identical letters and -1 for different letters including letter to gap mismatch (gap penalty). More complex schemes were proposed in order to capture different mutation probabilities; see e.g. BLOSUM scoring [3].

Having a scoring scheme, we can build a matrix M of size $(m+1) \times (n+1)$, where each entry $M(i, j)$ represents the score for optimal alignment of partial sequences $x_1, ..., x_i$ and $y_1, ..., y_j$. The matrix M needs to be initialized as follows: $M(0, 0) = 0$, $M(i, 0) = i * gp$ and $M(j, 0) = j * gp$, where gp is gap penalty. The remaining entries of M are filled recursively:

$$M(i,j) = max \begin{cases} M(i-1, j-1) + S(i,j), \\ M(i-1, j) + S(i, \_), \\ M(i, j-1) + S(\_, j), \end{cases} \tag{1}$$

where $S(i, \_)$ and $S(\_, j)$ is gap penalty and $S(i, j)$ is the score of matching (or mismatching) at i-th and j-th position of sequences. In order to compute optimal alignment, we also need to record which of the three considered cases was applied, i.e. which resulted to maximum value. When computing the optimum alignment we need to backtrack from $M(m, n)$ in the direction of recorded choices, where moving up or left means introducing a gap and moving on diagonal means no gap. NWA complexity is $O(mn)$, therefore this algorithm is also suitable for computing sequence alignments of considerable length.

3 Definitions

To define a movement of device (user) it is necessary to determine its location at any point. Therefore, we define *position* as a location of user using geo-location coordinates as follows.

Definition 1. *Position $\mathcal{P}$ is a couple (lat, lon), where*

- *lat is latitude in decimal degrees (e.g. 48.1518568),*
- *lon is longitude in decimal degrees (e.g. 17.0711559).*

Alongside with device position, we also record other relevant information such as time, accuracy or process of obtaining the position.

Definition 2. *Location $\mathcal{L}$ is a tuple $(\mathcal{P}, \mathcal{T}, \mathcal{A})$, where*

- *$\mathcal{P}$ is position,*
- *$\mathcal{T}$ is time in seconds from 1.1.1970 (UTC),*
- *$\mathcal{A}$ is horizontal accuracy in meters.*

According to the Android documentation [2], we define horizontal accuracy $\mathcal{A}$ as a radius with 68% reliability. In other words, if we draw a circle with radius $\mathcal{A}$ and center $\mathcal{P}$, there is 68% probability (one standard deviation) that the actual position is inside of this circle. The value of 0.0 means that the accuracy of the actual position is not defined.

Definition 3. *Path $\mathcal{S}$ is sequence of locations $(\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_n)$, where $(\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_n)$ is a non-decreasing sequence of time values at which location was recorded.*

Next we define two quantitative indicators for paths, in particular *length* and *size*.

Definition 4. *The length of path $\mathcal{S} = (\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_n)$ is the sum of distances between subsequent pairs of locations:*

$$length(\mathcal{S}) = \sum_{i=1}^{n-1} distance(\mathcal{L}_i, \mathcal{L}_{i+1}),$$

where $distance(\mathcal{L}_i, \mathcal{L}_{i+1})$ is the distance based on either Euclidean distance or Vincenty's formulae [5, 13], where the latter calculates the distance of two points on a spheroid.

Herein we apply Vincenty's formulae as they are available in a number of libraries.

Definition 5. *The size of path $\mathcal{S} = (\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_n)$ is the number of locations in path:*

$$size(\mathcal{S}) = n.$$

In general, as each path contains temporal $\mathcal{T}$ and spatial element $\mathcal{P}$, one can consider three types of path similarities: spatial, temporal and spatial-temporal [19].

- For *spatial* approach, only sequence of positions $(P_i)_n^{i=1}$, regardless of temporal aspect, gets considered. To compare two or several paths, one can calculate distance between individual path positions or compare them with respect to a fixed position Q.
- For *temporal* approach, only sequence of time $(\mathcal{T})_n^{i=1}$, where locations get obtained regardless of their actual positions, gets considered. To compare paths, one can search for appearance of common time intervals in individual paths or appearance of paths in certain interval.

– For *spatial-temporal* approach, both $(P_i)_n^{i=1}$ and $(\mathcal{T})_n^{i=1}$ get considered, we use path $(\mathcal{S})_n^{i=1}$. To compare paths, one can calculate distance between individual path positions in relation to time of the position or given path in given time.

An appropriate approach for modeling, representation and comparison of paths is chosen based on task requirements. In this work, we are focusing on spatial and spatial-temporal representation of paths.

Next, we define clusters and tracks that we need when applying NWA. The interpolated path is referred to as *track*, which is defined with new interpolated locations called *clusters*.

Definition 6. *Cluster* $\mathcal{K} = (P_0, \delta)$ *includes a set of locations* $\{\mathcal{L}_1, ..., \mathcal{L}_n\}$ *such that:*

$$\forall \mathcal{L}_i, \mathcal{L}_j : distance(\mathcal{L}_i, \mathcal{L}_j) < \delta,$$

where $i = 1...n$, $j = 1...n$, δ *is the cluster diameter and* $\mathcal{P}_0$ *is the cluster center such that it holds* $\forall \mathcal{L}_i : distance(\mathcal{L}_i, \mathcal{P}_0) \leq \frac{\delta}{2}$.

Definition 7. *Size of cluster* $|\mathcal{K}|$ *is the number of locations in cluster.*

Definition 8. *Distance between clusters* $\mathcal{K}_i = (\mathcal{P}_i, \delta)$ *and* $\mathcal{K}_j = (\mathcal{P}_j, \delta)$ *is defined as:*

$$distance(\mathcal{K}_i, \mathcal{K}_j) = distance(\mathcal{P}_i, \mathcal{P}_j),$$

for arbitrary i *and* j, $i \neq j$.

Definition 9. *Track* $\mathcal{C}$ *is a sorted sequence of clusters* $(\mathcal{K}_1, ..., \mathcal{K}_n)$ *such that*

$$\forall \mathcal{K}_i, \mathcal{K}_{i+1} : distance(\mathcal{P}_i, \mathcal{P}_{i+1}) = \delta,$$

where $i = 1...(n-1)$, $\mathcal{K}_i = (\mathcal{P}_i, \delta)$ and $\mathcal{K}_{i+1} = (\mathcal{P}_{i+1}, \delta)$.

Definition 10. *Size of track* $|\mathcal{C}|$ *is the number of clusters it contains.*

4 Related Work

A number of approaches such as those described in [4,8–10] apply various extraction and prediction approaches based on position information provided by GPS. In other approaches, instead of using locations from a collection of GPS positions, the authors constructed a semantic trajectory. Ying et al. [15] proposed a framework by exploring semantic trajectories of mobile users, in order to model the next location of a mobile user in support of various location-based services.

Mavoa et al. [7] attempted to link GPS positions and a manually inserted travel diary using sequence alignment in study of children's independent mobility. They concluded that sequence alignment is a promising method. This approach provided motivation and a starting point for our research.

Herein we focus on approaches that take advantage of comparing user trajectories. As our literature review revealed, such approaches are common when comparing user trajectory with possible trajectories derived from a map. In the following sub-sections we review the most relevant methods for comparing user trajectories and map trajectories. These approaches constitute a consistent set of options that we took into consideration, when proposing our approaches.

4.1 Locational Proximity

The Locational proximity method for comparison of multiple paths has been described by Yang et. al. [14]. In this method distances are calculated between each pair of positions in recorded path and map segment. Comparison of paths is sequential, where each location $(\mathcal{L})_n^{i=1}$ in the path is compared with the map segment in order to find the closest segment. The approach used for assigning locations $(\mathcal{L})_n^{i=1}$ to map segments is based on map-matching. The method calculates distance d from the closest position in assigned map segment for each assigned location in the recorded path. Subsequently, the total deviation is calculated for all locations from the recorded path as:

$$\overline{d} = \frac{\int_0^{l_M} d(l)dl}{l_M},$$

where l_M is a total size of compared path.

4.2 Shape Similarity

Yang et. al. also presented another approach for comparison of paths called the Shape similarity method [14]. To compare a recorded path with matched map segment, the method uses a shape based similarity in geometrical context. A level of similarity is expressed by the differential deviation:

$$\delta\overline{d} = \frac{\int_0^{l_M} |\delta d(l)|dl}{l_M},$$

where $\delta d = d - \overline{d}$. The derived differential deviation is computed as an average of deviations retrieved by the Location proximity method for a given recorded path. Such differential deviation can be then shown in a graph for an easy and quick comparison of both compared paths. The lower the deviation, the more similar is the shape of the compared paths.

4.3 Directional Consistency

The movement direction is another approach for effective comparison of paths [14]. The directional consistency method enables to determine consistency between direction of recorded path and selected map-matched path. To apply this approach, it is necessary to calculate the difference between selected

path α_O and direction of map-matched path α_M, for each pair of locations as $\Delta\overline{\alpha} = \alpha_M - \alpha_O$. The overall similarity of the movement direction is calculated as follows:

$$\Delta\overline{\alpha} = \frac{\int_0^{l_M} |\alpha(l)|dl}{l_M}$$

4.4 Behavioral Consistency

The analysis of user behavior is one of the fields focusing on comparison of paths. The behavioral consistency method requires information about the change of speed (e.g. speeding, slowing down) and change of direction. As shown in [14], the similarity of two paths is directly related to the user behavior. This method allows identification of various habits and patterns of the movement when used for comparison of various spacious paths.

4.5 Grid Sequencing

The Grid sequencing method described by Thiagarajan et. al. in [12] uses Hidden Markov Model containing a set of hidden states and observables. Individual states emit an observable whose probability is defined by an emission score $E(F, G)$. This emission score captures the probability of finding a user location F in a cell G (grid cell). The higher the emission score, the higher the probability of user location being matched with the map segment. The location with highest emission score is considered for the one being truly visited by user. The transition score is calculated as a distance between neighboring cells, and it represents the probability of further transition from one cell to another.

5 Our Approaches

5.1 Pairwise Method

This approach is a naive approach aimed at straightforward interpretation of results. We included it as a useful base case. Let $S_a = (L_{a_1}, ..., L_{a_n})$ and $S_b = (L_{b_1}, ..., L_{b_m})$ be paths. In pairwise method we compare pairs of locations (L_{a_i}, L_{b_i}) as follows:

$$d = distance(L_{a_i}, L_{b_i}),$$

where $i = 1...x$ and $x = min(m, n)$. We say that locations (L_{a_i}, L_{b_i}) are similar, if $distance(L_{a_i}, L_{b_i}) < \delta$. In order to compute similarity score for path S_a and S_b, we count the number of similar location pairs. This approach requires that paths get manually synchronized, i.e. it must be decided which initial pair of positions from either path will be used in computation.

5.2 Proximity Method

This approach is derived from the locational proximity approach introduced in the previous section. In proximity method we compare paths by computing:

$$d = distance(L_{a_i}, L_{b_x}), \text{ and}$$
$$\forall L_{a_i} \exists L_{b_x} : distance(L_{a_i}, L_{b_x}) = min(distance(L_{a_i}, L_{b_j}),$$

where $i = 1...n$ and $j = 1...m$. As for pairwise method, we count the number of similar (L_{a_i}, L_{b_j}) with respect to δ. The proximity method applies pairs of positions with least distance, therefore unlike the pairwise method, this method can be applied without any initial synchronization.

5.3 Upward Proximity Method

Upward proximity method is similar to proximity method with the difference that subsequent pairs for comparison are chosen in increasing order. In upward proximity method we compare paths by computing:

$$d = distance(L_{a_i}, L_{b_x}),$$
$$\forall L_{a_i} \exists L_{b_x} : distance(L_{a_i}, L_{b_x}) = min(distance(L_{a_i}, L_{b_j})),$$
$$b_j > k,$$

where $i = 1...n$, $j = 1...m$ and L_k is the location used in previous iteration. As for previous methods, we count the number of similar (L_{a_i}, L_{b_j}) with respect to δ. We have proposed Upward proximity method as an iterative improvement over Proximity method. The rationale is that when comparing pairs of locations, back-tracking should not be possible.

5.4 NWA Approach

Our proposed approach is based on NWA alignment. In order to apply NWA, it is necessary to convert paths S_a and S_b to tracks $\mathcal{C}_a = (\mathcal{K}_{a_1}, ..., \mathcal{K}_{a_r})$ and $\mathcal{C}_b = (\mathcal{K}_{a_1}, ..., \mathcal{K}_{a_s})$. Then we compute alignment of $\mathcal{C}_a$ and $\mathcal{C}_b$ with NWA. As for previous methods, we count the number of aligned clusters with respect to fixed distance δ:

$$d = distance(\mathcal{K}_{a_i}, \mathcal{K}_{b_i}),$$

where the pair $(\mathcal{K}_{a_i}, \mathcal{K}_{b_i})$ is aligned applying NWA.

6 Experimental Setup

6.1 Data Sets

The evaluation of our approach to sequence alignment is based on data sets that record people movement. We considered two data sets:

– A data set that we collected using 455 mobile devices distributed among our students. The results presented herein were collected by our students during a 10-month period starting from September 2016 to July 2017. Over 20 million location records provide insights into our students' behavior patterns (bars, restaurants, clubs etc.). Recording of locations was done using our implemented mobile application for energy efficient trajectory recording of mobile devices using WiFi scanning, described in more details in [1,6].
– The Geolife data set (Microsoft Research Asia) was collected by 182 users in a period of over three years (from April 2007 to August 2012). This data set contains 17,621 trajectories with a total distance of about 1.2 million kilometers and a total duration of 48,000+ hours. These trajectories were recorded by different GPS loggers and GPS capable phones, and have a variety of sampling rates. This data set recorded a broad range of users' outdoor movements, including not only life routines like go home and go to work but also some entertainments and sports activities, such as shopping, sightseeing, dining, hiking, and cycling [16–18].

6.2 Pre-processing

In pre-processing phase it is necessary to identify errors or incomplete paths, which were excluded from evaluation. Therefore, in the first step we eliminated paths shorter than $size_{min}$; we have set this parameter to 500 m.

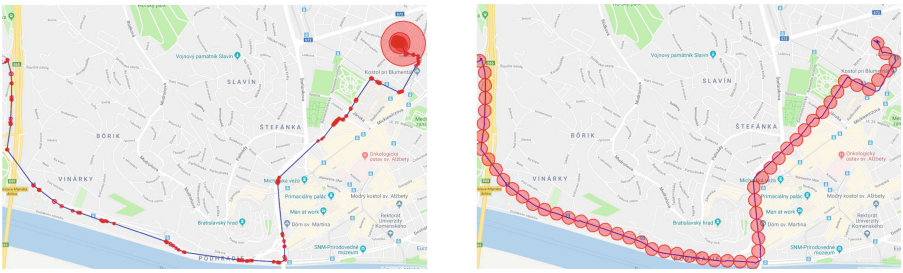
During the recording of paths we encountered situations where for some reason devices stopped collecting information about the location. In those cases the location data includes gaps; see Fig. 1(a). Identified gaps can have a length of up to few kilometers, therefore it is crucial to exclude them by dividing the path into separate sub-paths.

In a situation when mobile device does not move, for example it lies on a table, a nest can arise; see Fig. 1(b). Nests are often formed by a large number of positions. For this reason we implemented the process of clustering individual positions that are close to each other.

Initially, we implemented a clustering method based on a fixed-size square with side a, where the center of a square represented the starting location. However, as this approach did not result in an effective clustering of locations, we proposed and implemented a clustering method, where square center gets moved in order to include a new location. In other words, location belongs to the existing square if a center of that square can be adjusted to contain this location with all locations that belonged to this square before.

The proposed clustering with dynamic center can lead to the situation where individual squares overlap each other while each location belongs to only one square. An advantage of the proposed clustering is its ability to minimize the number of required squares.

An example of recorded locations is shown in Fig. 2(a), where locations are marked by circles with diameter representing location accuracy $\mathcal{A}$. The figure represents a usual nature of recorded path with a large number of gaps. The interpolated movement of user is represented by line.



(a) Recorded path locations (b) Interpolated path locations

Fig. 2. Recorded and interpolated path.

Comparison of paths with many gaps can lead to inaccurate results. In order to overcome this problem, any recorded path needs to be interpolated with evenly distributed locations along the entire path. Therefore we implemented backward interpolation of path. With this process we ensure constant distribution of locations along the path as shown in Fig. 2(b).

The final step was to remove all short paths, that can also be a result of previous steps. As in the first step, we removed all paths shorter than $size_{min}$.

6.3 NWA Parameters

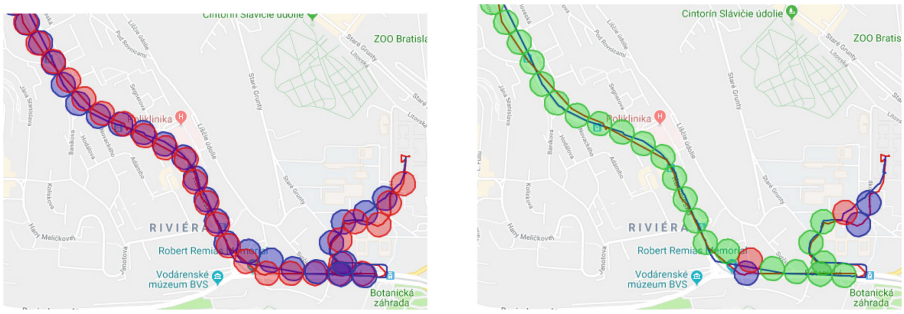
NWA requires scoring schema that defines score for match, score for mismatch and gap penalty. In our case, we set these parameters to 1, −1 and 0, respectively. The given scoring schema was chosen by multiple tests, to ensure the highest accuracy for aligning clusters of given tracks. Arguably, choosing parameters for a scoring schema deserves a separate study with more detail and experimentation. We remind that scoring schemes such as BLOSUM were also derived experimentally with expert knowledge about a specific problem.

7 Experimental Results

7.1 Evaluation Details

When applying NWA, tracks need to get computed; see Definition 9. Figure 3(a) shows an example for computed tracks, more specifically, it shows two tracks that need to get compared.

The comparison of tracks has been based on two criteria: the number of subsequent mismatches and the total number of matches. As accuracy of user movement tracking based on GPS data captured by mobile device varies, we consider for evaluation the maximum number of subsequent mismatches instead of the total number of mismatches. This means, a path can contain any number of mismatches as far as the maximum number of subsequent mismatches does not exceed a certain threshold, in our case we set this threshold to 3.



(a) Compared tracks (b) Alignment using algorithm

Fig. 3. Comparison and alignment applying NWA

When a mismatch in one path is directly followed by a mismatch in other path, this is counted as a single mismatch due to the fact that both mismatches happened in two different clusters. Mismatches are counted as two only if they follow each other. For this reason it is necessary to check also an overall similarity of tracks based on a total number of matches.

Definition 11. *Tracks $\mathcal{C}_a$ and $\mathcal{C}_b$ are similar if these tracks have at least α matches and no more than β subsequent mismatches.*

Table 1. Alignment computed by NWA

3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 2, 1, 3, 3, 3, 3, 3, 3, 3, 2, 2, 1
(1)=mismatch of Track 1, (2)=mismatch of Track 2, (3)=match

The parameters α and β have been experimentally set to $\alpha = 75\%$ a $\beta = 3$. An example of track alignment is shown in Table 1, representing the same result as depicted in Fig. 3(b).

In this particular case, Table 1 shows 23 matches in interpolated locations and 3 mismatches. In this example, one of the tracks is longer, consisting of 26 clusters, thus having 1 cluster more than the other track. As Track 1 contains 23 matches from 26 and the maximum occurrence of subsequent mismatches is 2, the compared tracks are considered similar.

Table 2 shows the distances of locations for similar or dissimilar pairs of tracks with respect to some threshold ϵ, which was set to $\sqrt{a^2 + a^2} = 70,71$ m, i.e. to the diagonal length of a square with sides equal to 50 m. We remind that such a fixed-size square was used for clustering in the pre-processing phase.

Table 2 shows that shortest distances between pairs of locations can be expected for the proximity approach and the NWA approach. This indicates that these two approaches might be the best candidates for comparing user

Table 2. Results for total number of 286, 431 pairs of locations or cluster centers for NWA approach.

Method	$d \leq \epsilon$	$d > \epsilon$	$d \leq \epsilon$	$d > \epsilon$
Pairwise	99 874	186 557	35%	65%
Proximity	268 024	18 407	94%	6%
Upward proximity	92 672	193 759	32%	68%
NWA	271 078	15 353	95%	5%

tracks. Notice that in the case of the NWA approach, we compute distances of cluster centers, not individual locations.

With respect to Definition 11 we filtered the two considered data sets so that we have 1, 500 pairs of similar tracks and 500 pairs of tracks of dissimilar tracks.

7.2 Results

When evaluating the two data sets, we compared the performance of following approaches:

- NWA approach,
- Pairwise method,
- Proximity method,
- Upward proximity method.

We consider the pairwise method and proximity method to be useful base cases, i.e. approaches with straightforward implementation and arguably leading to results that are simpler to interpret.

Table 3 shows the results for the four considered approaches, where true positive is when a pair of tracks that was correctly identified as similar, true negative is when a pair of tracks that was correctly identified as dissimilar, false positive when a pair of tracks was dissimilar but identified to be similar, and finally, true negative is when a pair of tracks was similar but identified to be dissimilar.

Table 3. Performance of different methods.

Method	True positive	False positive	False negative	True negative
Pairwise	87	159	913	341
Proximity	862	465	138	35
Upward proximity	284	307	716	193
NWA	918	74	82	426

Table 3 thus shows that NWA approach is the winning approach in all four categories, especially, this approach offers very low false positive and false negative rates.

8 Conclusion and Future Work

Our goal was to apply NWA, better known from biology and bio-informatics rather than other research areas, to user trajectory comparison. We proposed several adjustments to this algorithm, so that it can be applied to data sets with different properties than data sets arising in the above mentioned areas.

NWA requires a scoring scheme as an input, and arguably, more research effort might need to be invested in finding optimal scheme. Our results show that NWA can be successfully applied to user trajectory comparison. When comparing NWA to the proximity method, we show that NWA offers 5.6% improvement for true positives. When comparing NWA to the pairwise method, we show that NWA offers 19% improvement for true negatives. NWA dominates other considered approaches in terms of false positives and false negatives. Given the high false positives and high negatives rates of other methods, these can be considered useless, at least in the context of the applied data sets.

We plan to do further tests with other existing data sets, which could lead to improved knowledge about the performance of our NWA approach in situations insufficiently covered by the data sets considered herein.

Acknowledgment. The authors were supported by the project "STU ako líder Digitálnej koalície", project no. 002STU-2-1/2018, financed by Ministry of Education, Science, Research and Sport of the Slovak Republic. Maroš Čavojský also thankfully acknowledges a conference grant received from MAIND, s.r.o.

References

1. Čavojský, M., Drozda, M.: Energy efficient trajectory recording of mobile devices using wifi scanning. In: Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 International IEEE Conferences, pp. 1079–1085 (2016)
2. Google: Location—Android Developers. https://developer.android.com/reference/android/location/package-summary.html
3. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. Proc. Natl. Acad. Sci. **89**(22), 10915–10919 (1992)
4. Hung, C.C., Chang, C.W., Peng, W.C.: Mining trajectory profiles for discovering user communities. In: Proceedings of the 2009 International Workshop on Location Based Social Networks - LBSN 2009. pp. 1–8. ACM (2009). https://doi.org/10.1145/1629890.1629892
5. Karney, C., Deakin, R.E.: FW bessel (1825): The calculation of longitude and latitude from geodesic measurements. Astron. Nachr. **331**(8), 852–861 (2010)
6. Čavojský, M., Uhlar, M., Ivanis, M., Molnar, M., Drozda, M.: User trajectory extraction based on wifi scanning. In: FiCloud 2018, The IEEE 6th International Conference on Future Internet of Things and Cloud, pp. 115–120 (2018)
7. Mavoa, S., Oliver, M., Witten, K., Badland, H.M.: Linking GPS and travel diary data using sequence alignment in a study of children's independent mobility. Int. J. Health Geogr. **10**(1), 64 (2011)

8. Michael, K., McNamee, A., Michael, M., Tootell, H.: Location-based intelligence – modeling behavior in humans using GPS location-based intelligence – modeling behavior in humans using GPS location-based intelligence – modeling behavior in humans using GPS. In: 2006 IEEE International Symposium on Technology and Society (ISTAS 2006), pp. 1–8 (2006)
9. Monreale, A., Pinelli, F., Trasarti, R., Giannotti, F.: WhereNext: a location predictor on trajectory pattern mining. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 637–646. ACM (2009)
10. Montoliu, R., Blom, J., Gatica-Perez, D.: Discovering places of interest in everyday life from smartphone data. Multimed. Tools Appl. **62**(1), 179–207 (2013)
11. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. **48**(3), 443–453 (1970)
12. Thiagarajan, A., Ravindranath, L., Balakrishnan, H., Madden, S., Girod, L.: Accurate, low-energy trajectory mapping for mobile devices. In: Proceedings of USENIX Association (2011)
13. Van Brummelen, G.: Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry. Princeton University Press, Princeton (2012)
14. Yang, D., Zhang, T., Li, J., Lian, X.: Synthetic fuzzy evaluation method of trajectory similarity in map-matching. J. Intell. Transp. Syst. **15**(4), 193–204 (2011)
15. Ying, J.J.C., Lee, W.C., Weng, T.C., Tseng, V.S.: Semantic trajectory mining for location prediction. In: Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 34–43. ACM (2011). https://doi.org/10.1145/2093973.2093980
16. Zheng, Y., Li, Q., Chen, Y., Xie, X., Ma, W.Y.: Understanding mobility based on GPS data. In: Proceedings of the 10th International Conference on Ubiquitous Computing, pp. 312–321. ACM (2008)
17. Zheng, Y., Xie, X., Ma, W.Y.: Geolife: a collaborative social networking service among user, location and trajectory. IEEE Data Eng. Bull. **33**(2), 32–39 (2010)
18. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining interesting locations and travel sequences from GPS trajectories. In: Proceedings of the 18th International Conference on World Wide Web, pp. 791–800. ACM (2009)
19. Zheng, Y., Zhou, X.: Computing with Spatial Trajectories. Springer, New York (2011). https://doi.org/10.1007/978-1-4614-1629-6