International Journal on Semantic Web and Information Systems
Volume 18 • Issue 1

Semantic Trajectory Frequent Pattern Mining Model:
The Definitions and Theorems

Jun Li, Hunan International Economics University, China
Jie Su, Hunan International Economics University, China*

ABSTRACT

A method for mining frequent patterns of individual user trajectories is proposed based on location semantics. The semantic trajectory is obtained by inverse geocoding and preprocessed to obtain the top-k candidate frequent location item sets, and then the spatio-temporal sequence intersection and the divide and conquer merge methods are used to convert the frequent iterative calculation of long item sets into hierarchical sets' regular operations, and the superset and subset of frequent sequences are found. This kind of semantic trajectory frequent pattern mining can actively identify and discover potential carpooling needs and provide higher accuracy for location-based intelligent recommendations such as carpooling and HOV lane travel (high-occupancy vehicle lane). Carpool matching and recommendation based on semantic trajectory in this paper is suitable for single carpooling and relay-ride carpooling. The results of simulation carpooling experiments prove the applicability and efficiency of the method.

KEYWORDS

Carpooling, Frequent Pattern, Semantic Integration, Semantic Networks, Semantic Trajectory, Semantic Zoom

1. INTRODUCTION

Spatio-temporal data mining usually refers to mining or predicting the historical trajectory of moving objects (Radmanesh M,2021; Meng J J, 2016; Harwood A, 2018). Spatio-temporal trajectory data are many sample records from different sensing devices. The understanding and modeling of spatio-temporal trajectory data provides a new perspective for learning people's movement patterns, and it is also an auxiliary tool for urban planning and smart city management with great potential (Zhang M H, 2021; Pan X Y, 2019; Yang W L,2021). The semantic analysis model is widely used to analyze the various semantic relationships that may arise in the document. A new semantic analysis model is proposed to find possible time relationships between posts (Chen L C, 2019; Gao Q, et al., 2017) .

As more high occupancy toll (HOT) facilities are planned and under development, a comprehensive understanding of HOT operations is required for establishing appropriate HOT policies. To enhance the understanding, the factors affecting drivers' choices on HOT lane use and carpooling are investigated in the Atlanta I-85 HOT corridors (Guensler R.,2020). A coevolutionary algorithm is proposed for two solution sets, population and archive, the objective-wise local search and set-based simulated

DOI: 10.4018/IJSWIS.297031 *Corresponding Author

binary operation are used in order to address the MOCSPTW (Huang S. C.,2020). An ant path-oriented carpooling allocation approach is proposed to solve this problem in the time domain (Huang S. C.,2019). Agreedy approach is propose based on the iterative matching and merging (Duan Y. B.,2020). An intelligent carpool system is first presented based on the service-oriented architecture, a fuzzy-controlled genetic-based carpool algorithm is proposed by using the combined approach of the genetic algorithm and the fuzzy control system (Huang S. C.,2015),

Trajectory semantic enhancement refers to the association and related application scene data in the original coordinate trajectory. The trajectory data is generated by the behavior characteristics of the moving object, it is called as semantic trajectory data. The additional data is also called as semantic tags. The tag information has three levels: the track layer, the sub-track layer, and the track point layer. The analysis of the semantic track mainly focuses on the sub-track layer. In conversion, the spatio-temporal information in the original trajectory is fused with the text data, and a trajectory with semantic labels is obtained, it is called as a semantic trajectory (Alvares L O, et al., 2007).

Semantic trajectory analysis was proposed by Alvares et al. (Alvares L O, et al., 2007). The semantic trajectory is regarded as a sequence of positions marked with semantic labels. The label data represents the behaviors that have appeared on the trajectory. The trajectory data with the same semantic label sequence can be regarded as the same trajectory sequence. The semantic trajectory is easy to read and understand, because the operand is a character string, its disadvantage is that it is inconvenient to calculate, compress and decompress. A compression and calculation method was proposeed for semantic trajectories (Zuo K Z, et al., 2018), which combines road network data or POI(Point of Interest) data set and road network matching, trajectory sampling points are mapped to road network, a semantic trajectory sequence is formed. Road network matching plays an important role in assessing traffic flow, guiding vehicle navigation, predicting vehicle driving routes, and finding frequent travel routes from departure to destination. However, matching calculations between multi-layer networks, multiple compression and decompression both are relatively cumbersome and difficult to implement in batches.

In the study of frequent sequence mining of trajectories, classic algorithms include Apriori (Agrawal R & Srikant R,1994), AprioriAll (Srikant R & Agrawal R,1995), GSP (Srikant R & Agrawal R,1995), SPADE (Zaki M J, 2001) and Prefix Span (Pei J & Han J W,2004). Among them, the AprioriAll algorithm is a further improvement of the Apriori algorithm. The algorithm considers the order of elements in the generation of candidate item sets and frequent sequence patterns. GSP is based on Apriori's horizontal formatting method. SPADE uses a vertical format based on Apriori to mine all frequent sequence sets from many sequence data sets. The Prefixspan algorithm is an algorithm that generates frequent sequences based on the recursive growth of projection. In 2013, a Time Period-based Most Frequent Path (TPMFP) query was proposed (Luo W, et al., 2013), frequent pattern was used to mining, the most frequent path in a specific time period is found from a large amount of historical data. it guides the shortest path query. In 2015, a frequent path sequence mining algorithm PMWTI (Path Mining With Topology Information) was proposed, logistics network topology information is considered in PMWTI (Yang J Y, et al., 2015). This algorithm is an improvement of Apriori, and further prunes when generating candidate path sequences, the size of candidate path sequences is reduced. Experimental comparison results show that PMWTI has a higher efficiency of frequent path mining. In 2014, the fine-grained sequential pattern mining of spatio-temporal trajectories waspropposed (Zhang C, et al., 2014). Position points in continuous space are not suitable for sequential pattern mining, and items composed of positions with the same semantics are more suitable for this situation.

The study of spatial trajectory similarity is the main method to query similar trajectories in the trajectory set, that is to reduce the number of trajectories. Since many trajectories carry a large amount of text information, the study of spatial trajectory similarity considers the distance, and lacks text information. In response to this problem, the consideration of text similarity is added on the basis of spatial similarity (Gao Y Q, et al., 2020), and the entire space is gridded, and the range

query is performed by calculating the upper and lower limits of the spatial text similarity, the final result is obtained. A future location prediction model was proposed for mining semantic trajectory information (Zhang J L, et al., 2019), it was combined with travel mode. This model can first realize similar user mining based on semantic trajectory, and individual semantic trajectory is combined with similar user location trajectory, a frequent pattern set is obtained. Finally, the target trajectory is used to obtain the future location prediction candidate set. The identification of the future travel mode can be realized, the historical travel mode and location trajectory data are combined to establish a Markov model, the future location is predicted to obtain the candidate set, and the candidate set of the previous part is finally combined to obtain the final future location results. This model can not only combine semantic trajectories to mine similar user behaviors, but also external data of travel modes are integrated to overcome the limitations of location trajectories.

The above methods are clustering and mining common stay points from the trajectories of multiple users, discovering frequent patterns, and analyzing and using these information. These methods are conducive to extracting public attributes of similar populations. It can be used to generate location-related recommendations in collaborative recommendation. However, due to the group-based analysis, there is no semantic mining analysis of individual behavior patterns, so the semantic behavior characteristics of individual users cannot be identified. In order to analyze the frequent patterns of a single user's trajectory, and people is facilitated to understand the meaning of the frequent patterns, and the amount of calculation is reduced. In this paper, a semantic trajectory compression and clustering method is designed based on inverse geocoding.

2. SEMANTIC TRAJECTORY FREQUENT PATTERN MINING MODEL

2.1 Problem Description

A user's trajectory data set over a period of time is given, a geographic semantic label data set is generated by reverse coding, the candidate starting point sequence or end point sequence are merged based on semantics, and the most frequent geographic semantic sequence and its similarity are solved for the anterograde and retrograde direction. Combining the time slice attributes, it is possible to dig into the Top-k group personal schedule behavior pattern over a period of time. If this method is applied to a multi-user group, the schedule behavior patterns of multiple users can also be obtained, which can realize the mining of neighborhood and colleague relationships, and it can also provide candidate geographic tags, directions and candidate time periods for applications such as carpooling.

2.2 Formal Definition

Definition 1: The location record g at a certain time can be represented by a triplet: $g = (u,l,t)$, where u represents the user ID, l contains the longitude x and the latitude y, and t represents the timestamp.

Definition 2: Trajectory T_u. user u is given, its trajectory T_u is a sequence of triples, $<u_1,l_1,t_1>$,..., $<u_i,l_i,t_i>$,..., $<u_N,l_N,t_N>$, where N represents the length of the sequence. The tuples are arranged in the order of timestamp, and the user's path can be obtained by sorting timestamp, $T_u = g_1 \rightarrow g_2 \rightarrow \ldots \rightarrow g_i \rightarrow \ldots \rightarrow g_N$.

Definition 3: Staying point P. The time stamp of the trajectory can be used to calculate the time of the user stays at the location. If the time the user stays at a certain location exceeds the threshold, it means that such location information may generate more interesting events, and most locations with a short stay may be just passing sections. $P = <g, \Delta t>$ is the combination of location record and time window. Each time the user stays at a certain location, the stay point information will be left. However, due to accuracy and error, similar but different P may be different locations. It may represent a location with the same semantics in the real world, such as a store or a tourist attraction.

Definition 4: trajectory semantic label S_L, $S_L = (l,\varphi)$, where $\varphi = \{x | x \hat{I} A\}$, it represents the semantic label information which is attached to the point, x includes structured address, direction, nearby POI points, etc.. The inverse geocoding address based on the coordinate value is used as the semantic label in this article.

Definition 5: According to Definition 4, the semantic trajectory or semantic trajectory sequence $T_S = [S_{L1}, S_{L2}, \ldots, S_{Ln}]$ is a sequence, it is composed of a series of trajectory semantic tags in the original coordinate sequence, where $S_{Li} = (l_i, \varphi_i)$ contains the trajectory point l_i at time i and the semantic label information φ_i of the point. When the geocoding address is used as the semantic label, φ_i is the character string which is defined by the geocoding data structure, and the semantic trajectory is a chronological order of <location, Address string> table.

Definition 6: Frequent trajectory sequence. A sequence T_S containing k items is called an itemset T_S, and the length of T_S is k. T_f is a non-empty set consisting of one or more items. If T_f Í D, T_f ¹ Æ, then the itemset T_f is said to appear in transaction D. If the itemset T_f is frequent, if and only if T_f appears in D not less than $\theta|D|$, where $\theta(0 < \theta \pounds 1.0)$ is the minimum support which is given by the user, it is expressed by *minSup*, $|D|$ represents the number of transactions which is included in the data set D, and $\theta|D|$ is the minimum support count, which is represented by *minCount*.

According to the above series of definitions, it can be seen that semantic trajectory frequent pattern mining is actually the process of finding all frequent subsequences T_f that satisfy the minimum support, when the semantic trajectory database $\{T_{S1}, T_{S2}, \ldots, T_{Sn}\}$ and the minimum support *minSup* are given. In this article, the user's geographic semantic trajectory data set is used to mine frequent itemsets that meet the minimum support, so as to discover the user's frequent patterns and display them in the form of geographic semantic sequences.

Sequence database $S = \{T_{S1}, T_{S2}, \ldots, T_{Sn}\}$, α and β are frequent subsequences of S. They conform to one of the following two theorems.

Theorem 1: If for every T_{Si} $\hat{I}$ S, there is a prefix subsequence $\beta = \varphi_i$, $i = (1,2,\ldots,k)$, and there is α Í φ_i, and α is before the item set β belongs to.

Theorem 2: If only part of T_{Si} $\hat{I}$ S has a prefix subsequence β, α Í φ_i, and α is before the item set β belongs t, all other T_{Sj} $(j ^1 i)$ and β do not constitute a suffix relationship. Then in the sequence database S, the sequence pattern set is prefixed by α and also prefixed by β, it is the same as the sequence pattern set which is prefixed by β in the sequence database S.

According to the above theorem, the mining of long frequent sequence patterns can be converted into the continuous mining of two short sequence patterns. When constructing the projection database, by checking the prefix of the sequence prefix and cutting out the repeated projections of the same frequent prefix, the calculation efficiency can be improved.

3. SEMANTIC TRAJECTORY PREPROCESSING BASED ON INVERSE GEOCODING

3.1 Candidate Start and End Are Generated According To Segment Cropping And Sliding Filtering Of The Stay Time

The time stamp T_i in the historical coordinate trajectory T_u data of a single user is converted to a continuous time value format. If $T_{i+1} - T_i < T_{thresh}$, it means that T_{i+1} and T_i belong to the same sub-trajectory, otherwise they belong to two different sub-trajectories. Trajectory, T_{thresh} represents the time interval threshold, which can be similar to the usual parking time for work or residence. According to this method, each user's historical trajectory data can be divided into multiple continuous sub-

trajectories T^1_v, T^2_v,...,T^m_v. The starting point and ending point of each sub-trajectory are key points, which define the geographical boundaries of a movement behavior.

Then, the sub-trajectories T^i_v, $i = 1,2,...,m$ are filtered. Due to the influence of obstructions, weather, and electromagnetic interference, there may be burst noise in the trajectory data, which requires filtering. The median filter process is based on the ranking of the images contained in the image area protected by the filter, and then the value determined by the statistical ranking result is used to replace the value of the central area. The impulse noise (such as the salt and pepper noise) can be filtered very well in the median filter. In this paper, the median method is used to smooth the burst noise in the track points. It should be noted that the sub-trajectories should be divided before filtering, otherwise the two points before and after the interval exceeds the threshold may seriously reduce the filtering performance.

3.2 Semantic Track Conversion and Compression

Since trajectory coordinates are often accurated to 13 digits after the decimal point, the same geographic location may be correspond to multiple different coordinates, it would cause redundancy, and trajectory analysis based on pure coordinate values is difficult to give the actual physical meaning or explanation. The coordinate sequence is converted into a position label sequence containing actual semantics, and then various analyses are easier to understand and apply. Limited to the technical conditions at the time, the reverse geocoding conversion from coordinate values to semantic tags was carried out through the yellow pages *(Cao X, et al.,2010)*, which greatly limited the conversion efficiency and prevented it from being popularized and applied. With the development of geographic location service platform technology in recent years, cloud platform-based API services have become quite mature. For example, AutoNavi's WebAPI interface cloud service can provide batch reverse geocoding conversion services, analyze the administrative division of coordinates, and recall surrounding areas. POI data with higher weight is used for location description, and the daily free limit for a single keyID is 6 thousand. Inverse geocoding returns the address structure in JSON or XML form. The hierarchy rules are: province + city + districts and counties + development zone + town + village + street (commercial district or point of interest) + house number.

After the trajectory is converted to a geotag, the coordinate value data is replaced by an address. Some consecutive coordinate points are usually the same street or the same place name, and they may be inversely encoded by GIS to the same address, which provides the possibility of trajectory compression. A simple cleaning and compression strategy is to merge compression, if the address of the previous item is the same as the address of the next item, only the previous item is retained, the latter item is deleted, and then the next item is traversed. This strategy can compress and retain key information such as track names and directions, and it is easy to calculate. For example, in the test data set, there are a total of 7095 sets of data with user coordinate values. After inverse geocoding conversion, a total of 2026 items of address data are compressed according to this strategy, and the compression ratio is about (7095-2026)/7095=71.81%..

But even for the big data analysis platform, the result of the combined compression is still beyond the computing power. For example, when the test user sequence is calculated on the test machine, the total length of the itemsets constituting each sequence is 2026 items, the sparkMllibprefixspan algorithm is used to search for frequent patterns. If the output is restricted to only frequent 2 item sets, it takes 13s, and the output is restricted to 6 items. The collection time takes 33 min, and the output time of 7-item collection exceeds 5h. Longer frequent sequences mean deeper search depth and computational complexity. It cannot be completed until the calculation overflows. Theoretically, the total number of sequence combinations to be searched and aligned is $2026!-1=2.65\times e^{5821}$, so the itemset must be further compressed or this problem must be avoided.

3.3 Pattern Mining Algorithm of Reverse Geocoding Sequence

In the semantic trajectory, a frequent address sequence exceeding 10 different addresses is a high probability event, and the direct solution of a very long sequence often exceeds the computing power. For this reason, a sequence pattern mining algorithm is proposed in this paper based on reverse geocoding RGA (Reverse Geocode-based Apriori mining), frequent pattern mining of single-user multi-semantic trajectories is achieved: the Apriori algorithm is first used to mine the candidate Top-*k* most frequent 1-itemset and 2-itemset sequences, and then the continuity characteristics of the trajectory and frequent features are used to reasonable large-scale branching and pruning, and then the frequent sequence mining is disassembled into hierarchical frequent subset to be calculationed, and regular operations are set. The efficiency of this method for frequent pattern mining of semantic trajectory sequence data is higher than that of the traditional Apriori, FPGrowth, and PrefixSpan algorithms. The essence is to use the frequent nature of trajectories to decompose a deep search calculation into operation and merge of multi-layer subsets. The RGA algorithm performs mining according to the following steps:

Step 1: A frequent 1-item set of length 1 is obtained. Scan T_S once, all frequent items are found according to the support degree of frequent items, each frequent item is a sequence pattern of length 1, the frequent items are sorted according to their support degree from largest to smallest, and Top-*k* of the most frequent itemsets are kept as candidate item sets, and *k* can be set to 2~5 in this paper. But at this time, due to the differences in the link fields in the address structure, frequent 1 item sets may be repeated, such as [wuchang district 027 community] [wuchang district xiongchu avenue 28, 027 community] [wuchang district yitong preschool education 027 community]. These three address items actually point to the same cell, which belongs to the continuous track at the starting point. In order to reduce the repetitive item set, the difference of each address structure field of the frequent 1 item set can be distinguished. If the address above the village and the house address in the address structure are the same, but only the street or business district address is different, it can be merged into a frequent 1 item set to obtain a reduced Top-*k* candidate item set. The merge judgment process can also be performed before scanning get on. Through this step, the candidate starting point and candidate ending point of the most frequent pattern can be found, and all trajectory sequences T^{l}_{si} $(i\hat{I}[1\sim k])$ containing Top-*k* candidate item sets can be retained and other trajectory sequences can be clipped.

Step 2: The trajectory direction is found based on the starting point. For the most frequent items in the Top-*k* candidate item set, scan T^{k-1}_{si}, the corresponding projection database is built, and the frequent 2 item sets containing the most frequent items are output. If the most frequent item is a prefix, it means forward, and if it is a suffix, it means backward. According to the direction, a candidate trajectory sequence T^{k-2}_{sFi} or T^{k-2}_{sBj} $(i,j\ \hat{I}[1,k])$ containing frequent 2 item sets can be constructed. Continue to traverse the next item in the Top-*k* candidate item set until all mining is completed.

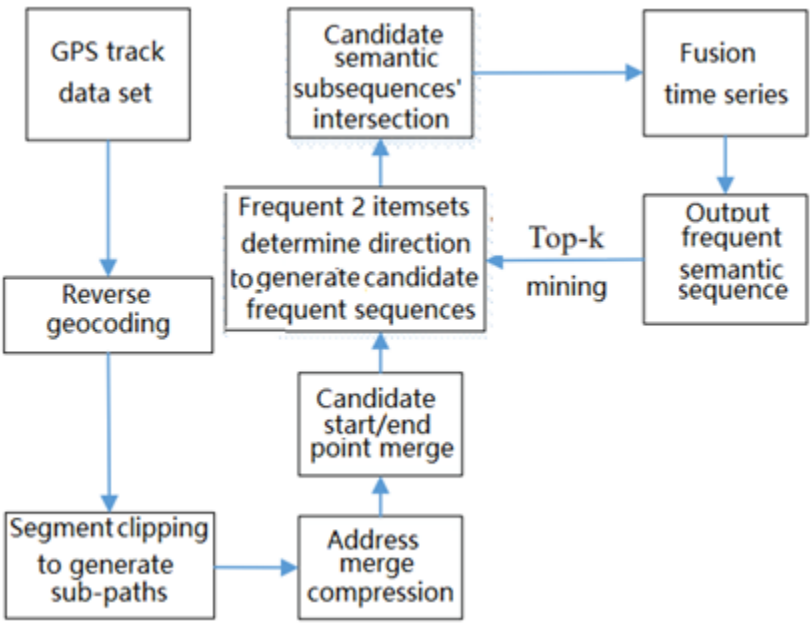
Step 3: The set intersection operation is performed on T^{k-2}_{sFi} or T^{k-2}_{sBj} respectively. The comparison and intersection of the address sequence can be solved by a hash table, that is, the sequence A is stored in the hash table, and then the sequence B is stored in the hash table. If a hash collision occurs, add 1 to the statistics to indicate the intersection. Finally, the intersection of the sequence can be obtained. Assuming that the length of the hash table is n, the time complexity of this calculation is $O(n)$. Through the intersection operation, the most frequent sequence sets in the forward and retrograde directions can be obtained respectively. If you need to solve in multiple directions, continue to scan and output frequent 3-item sets, and repeat the steps from steps 2 to 3.

The above mining framework is shown in Figure 1. Through this mining, users' most frequent locations and their address sequences, sub-frequent locations and their address sequences can be

obtained, from which potential carpooling needs can be discovered, and intelligent recommendations can be generated:

(1) For users with frequent addresses that overlap, they can directly recommend carpooling with each other, travel costs are reduced;
(2) When a user who does not have a historical address tag inserts a carpooling request, as long as these frequent addresses are inquired, the closest carpool recommendation can be given;
(3) It can effectively organize multiple carpooling demands along frequent routes.

Figure 1. Semantic frequent trajectory extraction framework



3.4 Inverse Geocoding and Frequent Sequence Mining Experiment

In this experiment, GPS data from multiple private cars are collected in Wuhan for three months. The basic data types in the original data include vehicle terminal ID identification number, latitude and longitude, time stamp, etc. The data format after privacy clearing is ABOXPhoneNum is 354071018570591, Latitude is 30.62255904, Longitude is 114.3160127, CreatTime is in 2019/11/13 -07:45:05.

The following only selects user data with the terminal ID of 354071018570591 to show the frequent trajectory extraction method. The total amount of data is generated by this single user during the interception period, it is 7100.

Because some consecutive coordinate points are inversely coded into the same address by GIS, resulting in data redundancy, the data is simply cleaned and compressed. The address compression algorithm used here is described as follows:

In the above algorithm, Geoseqs(gecode, time) represents set of all location information, and each piece of information contains an inversely encoded address and a time stamp. λ_{time} represents the threshold time, *k1* and *k2* represent the maximum length of the sequence to be sought, *minsup1*,

Algorithm: PatternGeneration //Algorithm name

Parameter: Geoseqs (geocode, time), λ_{time}, *k1, k2, minsup1, minsup2* //parameter
1. new seqset =$geoseq_i$ //Seqset is used to store the result of cleaning and compression
2. for each $geoseq_i$ in Geoseqs do
3. if ($geoseq_{i+1}$.time- $geoseq_i$.time>λ_{time}) then //select points of interest
4. new seq_{i+1}=$geoseq_{i+1}$
5. append seq_{i+1} to seqset //store in the result set
6. elseif ($geoseq_{i+1}$.addr==$geoseq_i$.addr) then //Compare consecutive addresses
7. delete $geoseq_{i+1}$ //If the address is the same, only keep one item
8. else Append $geoseq_{i+1}$ to seq_i //merge items
9. for each seq_i in seqset do //Save frequent items
10. sub1_prefixs=generate_1_size_freqitem(seqset, minsup1) //generate 1 item set
11. new_projs=generate_proj(seqset, sub1_prefixs)
12. if(sizeof(sub1_prefixs)>k1) then break;
13. for each seq_i in sub1_prefixs do // Traverse a set
14. if (seq_i.$geoseq_{i+1}$.subaddr==seq_i.$geoseq_i$.subaddr)
15 .then delete seq_i.$geoseq_i$ //Merge duplicate sets
16. for each sub1_prefixs[i] in seqset do
17. sub2_prefixs=generate_2_size_freqitem (*seqset, minsup2*) //generate 2 item set
18. final_projs=generate_proj (new_projs, sub2_prefixs)
19. if (sizeof(sub2_prefixs)>*k2*) then break;
20. for each $projection_i$ in final_projs do
21. $pattern_i$=$projection_i$.seq_{i+1}C$projection_i$.seq_i //Frequent sequence intersection

minsup2 represent the minimum support for screening sequences of length *k1* and *k2*, respectively. Then the position information set seqset is traversed, scan T^l_s, and the frequent items are projected in turn according to the support of frequent items, and 2 item set is output, it contains Top-*k* candidate item sets. According to the support of frequent items, the corresponding projection database is constructed, and each item is mined recursively. After further pattern mining, the frequent candidate address sequence of the user is obtained during the interception period. The support $k = 0.3$, the maximum frequent sequence length is 10, the longest frequent sequence is obtained by training, it is displayed as a dark path after data visualization on the map, and the light color is the second frequent sequence path.

4 SEMANTIC TRACK FREQUENT PATTERN CARPOOLING

4.1 Description and Formalization of Carpool Problems

Carpooling demand can be divided into two types, single carpooling and relay-based carpooling. Carpooling demand can come from the active request of passengers or the active discovery based on trajectory. Based on the frequent paths of users obtained by mining, demand matching and recommendation can be performed based on frequent itemsets.

$TR_i = (L_1, L_2, \ldots, L_k)$ represents a frequent sequence of addresses of vehicle C_i with length k in a certain period of time. If there are multiple carpooling demands in the same area in the same period of time, $N_j = \{s_j, d_j, t_j\}, j$ represents the demand number, t_j is the agreed time of boarding, s_j, d_j represent the source and destination points of the boarding respectively. The longest waiting time T for carpoolers can be set to 5 min. $T_{sim}(v_s, TR_i)$ represents the waiting time for carpoolers to board the car. If the waiting time for carpoolers is shorter than T, and the carpooling source is away from the boarding point, If it is less than S, it is considered that the carpooler and the vehicle are in the same temporal and spatial neighborhood. From a system perspective, the carpooling problem can be transformed into carpooling parties, they do not change the driving trajectory of the private car as much as possible and do not limit the carpooling mode, the carpool success rate and time and economic benefits are optimized. Since there is a positive correlation between the carpooling success rate and the path matching degree, the solution of this problem can be transformed into the following two objective functions:

$$\max \sum_{i \in C} \sum_{x,y \in V} X^i_{x,y} \tag{1}$$

$$\min \left(\pm \times \sum T_{cost} + ^2 \times \sum M_{cost} + \right) \tag{2}$$

The meaning of formula (1) is to select the carpool route with the largest matching rate corresponding to a certain carpool demand. $X^i_{x,y}$ represents the probability of the vehicle from point x to point y, $X^i_{x,y} = 1$ means that vehicle i can travel from point x to point y, otherwise $X^i_{x,y} = 0$. V represents the set of ride points for carpooling requirements during the vehicle driving process, and C is the collection of all vehicles that complete the carpooling requirements. The meaning of formula (2) is the optimization of time cost and economic cost of passenger carpooling. α and β are the coefficients of time cost and economic cost. The larger α indicates that the demand has strict time requirements, and the larger β indicates the demand hope keep fares as low as possible. The coefficient can be adjusted according to the actual situation. Multi-objective problems can take the form of equation (2) with leverage weighted optimization.

4.2 Carpool Recommendation Algorithm Based on Frequent Sequences Of Semantic Trajectories

Carpooling demand is often matched with the help of complex calculations, and the Hausdorff distance algorithm is improved to match the carpooling demand (Cao Y Y, et al., 2012; Liu C, et al., 2017; Zhang C D & Bie Z N, 2019). The distance is calculated between location points, it is used as the basis for demand matching (Guo H & Wang L X, 2017). The above methods are all complicated in terms of calculation degree. Based on this, in this paper, a carpool demand matching algorithm is proposed based on frequent sequences of semantic trajectories.

The Levenshtein distance is often used to calculate the difference between two sets of data, and it is used for the similarity between string sequences (Jiang H, et al., 2014). The Levenshtein distance was proposed in 1965 by the Russian mathematician Vladimir Levenshtein. Assuming there are sequences a and b, the Lewinstein distance calculation is formula:

$$lev(i,j) = \begin{cases} \max(i,j) & if \min(i,j) = 0 \\ \min \begin{cases} lev_{a,b}(i-1,j-1)+1 \\ lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \end{cases} & a_i \neq b_j, otherwise \end{cases} \quad (3)$$

According to the formula definition, the greater the Levinstein distance, the lower the matching degree, that is, the two show an inverse relationship. Multiple groups of user raw data is selected for frequent pattern extraction, a multi-user trajectory frequent sequence set TR_{set} is established, $TR_{set} = \{TR_1, TR_2, \ldots, TR_i\}$ $(0 \pounds i \pounds K)$, suppose there are K frequent sequences in TR_{set}, each frequent sequence is marked as $TR_i = <l_1, l_2, \ldots, l_n>$, where l represents the frequent item of the semantic track in the frequent sequence.

Levenshtein distance calculation process

Function: Get the distance of the two given strings

Step Description
1 Set n to be the length of s.
Set m to be the length of t.
If n = 0, return m and exit.
If m = 0, return n and exit.
Construct a matrix containing 0..m rows and 0..n columns.
2 Initialize the first row to 0..n.
Initialize the first column to 0..m.
3 Examine each character of s (i from 1 to n).
4 Examine each character of t (j from 1 to m).
5 If s[i] equals t[j], the cost is 0.
If s[i] doesn't equal t[j], the cost is 1.
6 Set cell d[i,j] of the matrix equal to the minimum of:
a. The cell immediately above plus 1: d[i-1,j] + 1.
b. The cell immediately to the left plus 1: d[i,j-1] + 1.
c. The cell diagonally above and to the left plus the cost: d[i-1,j-1] + cost.
7 After the iteration steps (3, 4, 5, 6) are complete, the distance is found in cell d[n,m].

In the same time period, there are J carpooling demands, $N_j = (s_j, d_j)(0 \pounds j \pounds J)$, s_j, d_j are respectively represented as carpooling source point and carpooling destination corresponding to carpooling demand Nj. According to the definition of route matching degree, the matching degree range [0, 1] is defined, and the improved route matching degree calculation method is formula (4):

$$M(N_j,TR)=\begin{cases}1 & lev(N_j,TR)==0\\ \dfrac{1}{lev(N_j,TR)} & lev(N_j,TR)\neq 0\end{cases} \tag{4}$$

M(N$_j$,TR) is calculated according to Levinstein distance, it is suitable for single carpool recommendation, because a single carpool means that user *A*'s frequent sequence set includes or is included in user *B*'s frequent sequence set. If the geocoding of the start point and the end point of the two are the same, the matching degree between the two is considered to be 1. If the frequent set T^m_{sA} *É* T^n_{sB}, it indicates that the trajectory of user *B* falls on the interval of user *A*, the matching degree is calculated according to the sequence similarity. For relay carpooling, the improved algorithm based on Levinstein distance is applicable. Suppose that the carpooling source point s falls on the frequent sequence TR_i, that is, *sÎ* TR_i, and the carpooling destination point falls on the frequent sequence $TR_{i+k,}$ that is, *dÎ*TR_{i+k}, and TR_i *Ç* TR_{i+1} *Ç…* $ÇTR_{i+k}{}^1$ *Æ* . Then, the multiple sequences are spliced into a new frequent sequence TR with the intersection point as the connection point, and the Levinstein distance is reapplied to obtain the similarity between the demand and the frequent sequence. Finally, formula (4) is used to calculate the route matching degree.

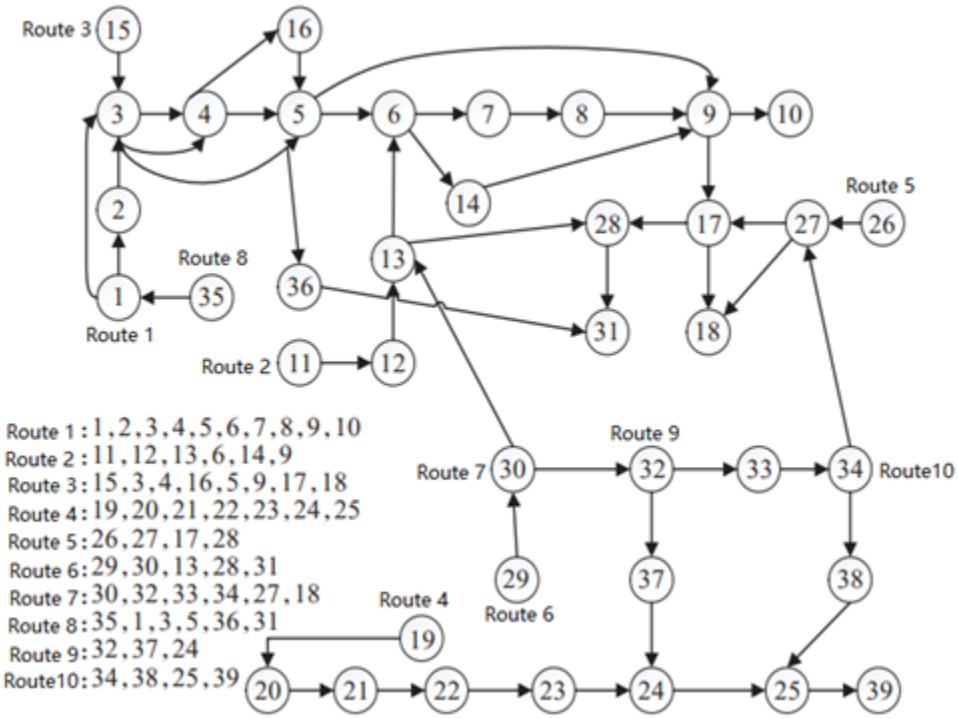
All the results are clustered, an appropriate threshold is selected, and carpooling recommendations are filtered with high matching degrees. After the carpool demand matching algorithm of frequent sequences of semantic trajectories, carpool demand is allocated to different vehicles. When private cars face multiple carpooling demands, in order to make the vehicles better provide carpooling services, it is also necessary to compare and rank the carpooling service demands which are provided by the vehicles, and the most suitable demand is selected from them to provide the corresponding vehicles. Whether the carpool demand can be met is mainly affected by time and route factors. The route matching degree has been defined above, so the time matching degree definition is added, and the two jointly restrict the demand matching. The time match is as follows:

Time matching degree: the time difference between the agreed ride time T_{vs} of the carpooler and the actual carpooling time T_{Rj} of the passenger is recorded as Δt, $\Delta t = |T_{Rj}-T_{vs}|$, the longer the Δt, the longer the carpooler or private car's waiting time, the lower the time matching degree, that is, the time matching degree is inversely proportional to Δt, so $M(t) = \alpha * 1/\Delta t$ is taken, $0 £ \Delta t £ 5$, and α is the coefficient. In summary, the comprehensive matching degree *M(P$_j$,C$_i$)* is as formula (5):

$$M(P_j,C_i)=\begin{cases}\alpha M(t)+\beta M(N_j,TR) & N_j\subseteq TR,\Delta t\leq 5\\ 0 \quad other\end{cases} \tag{5}$$

M(P$_j$,C$_i$) is a comprehensive measure of time matching degree and route matching degree. The shorter the time difference between the agreed boarding time and the actual boarding time, the higher the time matching degree. Among the frequent candidate addresses of vehicles, the shorter the interval between the starting point and the ending point of the vehicle, the higher the degree of route matching, and the greater the probability of successful matching. In the carpooling method of relay transfer, *TR* represents set of frequent candidate addresses for multiple private car routes, and there is an intersection between frequent addresses of private cars, and P_j *Í TR* represents that the carpool demand is split into multiple private carpoolers. These private car relay carpooling can meet the carpooling demand. In addition, the fewer frequent address items, the higher the matching degree, which also meets the needs of customers to complete carpooling with as few transfers as possible. In summary, this comprehensive measurement indicator is applicable to both single-trip carpooling and relay-ride carpooling.

Figure 2. Multi-user frequent roadmap



4.3 Experimental Design

10 sets of user data are selected to respectively perform inverse geocoding semantic trajectory preprocessing and frequent pattern extraction, the minimum support $k = 0.3$ and the maximum length of frequent sequences $Maxlen = 10$ *are* taken by the test, frequent candidate address sequences are obtained during the interception period. Integrating all frequent candidate address sequences can draw a multi-user frequent route map, it is shown in Figure 2.

The frequent sequence addresses are obtained from the 10 groups of user data, they contain multiple frequent items. For ease of understanding, the sequence elements of each group of frequent addresses are numbered in turn. For example, (1 Xiongchu Avenue Viaduct, 7:35) means that the address item number is 1, the address is Xiongchu Avenue Viaduct, and the time stamp is 7:35. After numbering, Matlab is used to randomly generate the start and end position numbers and time arrays from the frequent address item numbers. The two are combined into a carpool demand list, the list is shown in Table 1. To ensure sufficient experimental data, if the generated demand completely overlaps, it will be merged into one. The frequent address sequence is built into the database, the frequent item of candidate address is used as the *key* by the Hash algorithm, and the route mark ID where the frequent item is located is used as the *value*. The application of the Hash algorithm in the later matching will greatly improve the efficiency of search and comparison.

When the demand is matched to the vehicle, the route matching constraint and the time constraint need to be considered at the same time. The data format in the address compression algorithm is *(s, d, t)*, which respectively represent the starting point, end point of the carpool demand and the agreed-upon time. For users whose starting point and ending point are exactly the same or whose frequent sets are included, priority recommendations can be generated. According to the carpool demand matching algorithm of the frequent sequence of semantic trajectories, the starting point and end point

of carpool demand are first searched in the frequent sequence database, the Levinstein distance is calculated, and then the route matching degree is obtained.

Carpool recommendations are selected with a matching degree higher than 0.75. The results of the first demand allocation are shown in Table 2. The "Single" column contains all the requirements that can be directly matched based on the frequent sequence library. The "Transfer" column lists the requirements that passengers can meet by multiple private cars under the condition of accepting the relay-based ride-sharing. The data column in Table 3 shows the matching process between the

Table 1. Carpool demand list

Number	Carpool demand list	Number	Carpool demand list
1	(12, 7) 7:51	11	(32, 39) 7:14
2	(27, 18) 7:31	12	(17, 5) 7:56
3	(30, 28) 7;38	13	(13, 10) 7:49
4	(4, 28) 7:16	14	(30, 9) 7:39
5	(3,5) 7:49	15	(32, 18) 7:11
6	(29, 34) 7:30	16	(35, 36) 7:30
7	(15, 5) 7:01	17	(32, 24) 7:19
8	(2, 31) 7:39	18	(34, 39) 7:28
9	(32, 25) 7:14	19	(37, 14) 7:23
10	(16, 17) 7:22	20	(20, 23) 8:00

Table 2. First demand matching

Vehicle	Demand ID (pick-up point, drop-off point)	Single	Transfer
One	5 (35^{-}), 8 (23^{-}), 8 (25^{-}),	5	8, 13,
Two	13 (910^{-}), 13 (610^{-}),		14, 16
Three	14 (69^{-}), 16 (15^{-})	10	14
Four	13 (139^{-}), 13 (136^{-})	20	2, 4
Fives	14 (139^{-}), 14 (136^{-})		2, 4
Six	2 (1718^{-}), 4 (417^{-}),	3	2, 14
Seven	10 (1617^{-})	15	2, 9, 11
Eight	20 (2023^{-})	5, 16	8, 16
Nine	2 (2717^{-}), 4 (1728^{-})	17	
Ten	3 (3028^{-}), 14 (3013^{-})	18	9, 11

simulated demand and the vehicle. For example, (2818^{-}) means getting on the car at address number 28 and getting off at address number 18.

It can be seen from the first matching table that carpooling needs are divided into several categories: carpooling requirements are not met, single carpooling is completed, transfer-type carpooling is completed, and either single or transfer-type carpooling is completed. Certain vehicles

can match multiple carpooling needs. In the optimal route mining in this article, it is stipulated that the single completion of the same requirement has priority over the completion based on relay transfer, that is, the requirements that can be completed in a single carpool can be matched to the vehicle first, and for carpools with multiple completion methods, the demand needs to calculate the comprehensive matching degree to achieve the optimal demand distribution. Carpool demand 5, 8, 13, 14 still have multiple route matching problems, so the comprehensive matching degree is further calculated.

According to the comprehensive matching degree, the comprehensive matching degree of requirements 5, 8, 13, and 14 is obtained, they are shown in Table 3. For the allocation of each demand, the option with high matching degree is selected, so requirements 5, 8, 13, and 14 are selected to match for high-degree carpool routes, Table 4 is the final result matching table.

Table 3. Partial demand comprehensive matching

Demand ID	Carpool Route 1	Carpool Route 2	Comprehensive matching degree	Match result
5	Eight	One	11/25, 7/15	Route 2
8	One, Eight	One, Eight	17/105, 31/210	Route 1
13	Two, One	Two, Seven	1/6, 31/210	Route 1
14	Six, Two	Six, Two, One	1/6, 13/110	Route 1

Table 4. Match result

Vehicle	Demand ID	Vehicle	Demand ID
One	5, 8, 13, 16	Six	3, 14
Two	13, 14	Seven	15, 2, 9, 11
Three	10, 4	Eight	16, 8
Four	20	Nine	17
Fives	4	Ten	18, 9, 11

4.4 Result Analysis

The experimental results verify that the carpooling method can basically meet people's requirements for carpooling. Based on the frequent sequence of semantic trajectories, the carpooling demand matching algorithm first allocates the carpooling demand in the area to each vehicle. Then for private cars that may be allocated multiple needs, the optimal demand matching is based on the comprehensive matching degree. Because the strict time constraints are adopted by this carpooling method (in this experiment, the agreed waiting time difference and the transfer waiting time difference are not more than 5 min), resulting in unsuccessful matching of some requirements. In actual applications, it can be adjusted appropriately according to the situation, which will greatly improve match rate of overall demand data. In addition, time matching and route matching coefficients can be flexibly adjusted according to people's travel conditions, overall carpooling efficiency is improved.

In terms of algorithm complexity, the worst-case calculation complexity of this algorithm is $O(mn + k)$, and the calculation complexity based on the improved *Hausdorff* algorithm is $O(kmn)$

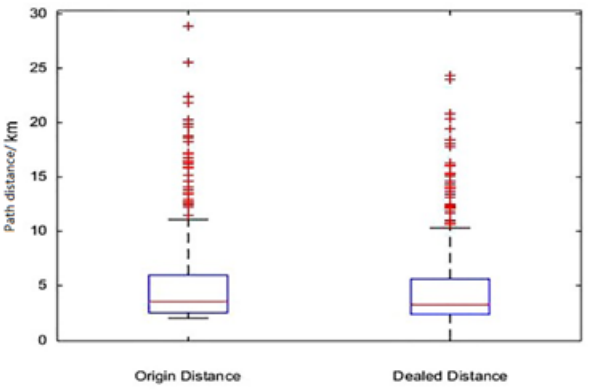
(carpooling demand length is m, frequent sequence length is n, there is frequent sequence of k trajectories). Therefore, it has a good effect in reducing the computational complexity. Based on the Apriori algorithm in association rule mining, under certain conditions, local improvements are made to improve the time efficiency of mining. This improvement itself does not require much system overhead, but it can make the algorithm run faster in some cases some. FP-Growth algorithm is used in this paper. The Apriori algorithm is the least efficient, because it needs to scan the database many times; secondly, the FP-Growth algorithm performs poorly on long transaction data, because when the transaction is very long, the depth of the tree is also very large, and the sub-problems that need to be solved change Because the Eclat algorithm has the highest efficiency, but because we have used the recursive idea in advance, when the amount of data is large, it will bring a huge burden to the system, so it is not suitable for large amounts of data. Situation; of course there is a technique called diffset that can solve the shortcomings of the Eclat algorithm. For data sets with dense patterns, the efficiency of the three algorithms is reduced due to the generation of particularly many and relatively long patterns. Among them, the FP tree level in the FP-Growth algorithm is deeper, which will generate more sub-problems. The Eclat algorithm requires a lot of intersection operations and consumes a lot of storage space, while the Apriori algorithm requires more scans of the database, so the efficiency is the lowest.

4.5 Contrast Test With Minimum Completion Time (MCT) Algorithm

670 driving routes are randomly selected in Minzu Avenue, Guanshan Avenue, Nanhu Avenue, and Xiongchu Avenue in the road map of Wuhan, Hubei Province, China, and the ride-sharing demand of passengers is simulated in a certain period of time. Through trajectory processing, the best excellent carpooling program is obtained the path similarity matching.

The city road map of Wuhan is crawled from Google Maps, and more than 7,000 roads among them are initially screened. The driving distance of urban taxis is generally 2 km to 30 km, the route between 2 km and 30 km is extracted. As a result of preliminary screening, a total of more than 1,000 routes meet the requirements. Next, the turning point is extracted. For sequences with more than 10 turning points, they are discarded. After final processing, more than 670 turning point sequences with 2 to 10 points are obtained, of which 60 are carpool routes for users who have boarded the car, and 610 are carpool routes for users who are waiting to get on the car, they are as the output result of trajectory preprocessing. In order to verify the rationality and necessity of the turning point selection, the path distance before and after the turning point extraction is compared, as shown in Figure 3.

Figure 3. Original path distance and processed path distance box plot



After the route composed of point sets is processed as a turning point sequence, the average distance of each path is shortened by 0.348km, and the standard deviation of the average distance difference between the two is 0.641km. This shows that after the route is processed as a turning point sequence, The reduction in distance should not be too drastic. After the turning point is extracted, although the calculation accuracy is reduced, the overall running time is greatly shortened, which ensures the on-demand matching of carpool requests and reduces the overhead and burden of server computing.

After preliminary screening of trajectory matching, trajectory matching is performed on the carpool route. The turning point sequence of the trajectory of the passengers that need carpooling is set B, and the turning point sequence of the trajectory of the passengers who have boarded the car is set A. Match each route in B with each route in A. The detailed experimental results are shown in Table 5. The Levenshtein distance algorithm and the MCT carpool algorithm are compared in detail in terms of the average user waiting time and the matching success rate. The average waiting time of ride-sharing users is *Total_ prT* (unit h), and the average waiting time can be calculated by the formula *Total_ prT =sum(timeDifference)/sum(Direction_sequence)*.

Table 5. Comparison of algorithm performance under different matching degree M values

Matching degree	Evaluation index	Ours	MCT	Matching degree	Evaluation index	Ours	MCT
M=0.9	Total_prT/h	0.04	0.12	M=0.5	Total_prT/h	0.07	0.13
	Success_rate/%	24.54	21.22		Success_rate/%	63.17	62.89
M=0.8	Total_prT/h	0.05	0.12	M=0.4	Total_prT/h	0.08	0.14
	Success_rate/%	36.55	30.24		Success_rate/%	69.00	69.35
M=0.7	Total_prT/h	0.06	0.13	M=0.3	Total_prT/h	0.10	0.14
	Success_rate/%	47.28	42.34		Success_rate/%	84.06	69.77
M=0.6	Total_prT/h	0.07	0.13				
	Success_rate/%	56.39	54.43				

Wherein, *timeDifference* is the time that each carpool user needs to wait; *sum(Direction_ sequence)* is the total number of carpool users. *Success_rate* is the matching success rate, that is, the ratio of the number of carpool routes that meet the critical threshold of the detour similarity measurement to the total number of routes, which can be calculated by the formula *Success_rate =sum(number)/(sum(Direction_sequenceB)×sum(Direction_sequenceA))*, *Sum(number)* is the number of carpool routes that meet the critical threshold of the detour similarity measure, *sum(Direction_ sequeceB)×sum(Direction_sequeceA)* is the total number of routes.

As shown in Table 1, *M* is the critical threshold of matching degree, which is used to measure the quality of the matching route. The larger the *M*, the better the matching degree of the matched route, and the shorter the waiting time of the user. Under different *M*, the pros and cons of different algorithms can be compared. Under the condition that the critical threshold of matching degree *M = 0.9* and B and A have been successfully matched, the average waiting time of users in the carpool scheme is reduced by 66.67% compared with MCT. As the critical threshold of matching degree *M* becomes smaller, the matching success rate of Ours method is getting higher and higher. When *M = 0.3*, the overall success rate reaches 84.06%; although the critical threshold of matching degree is reduced, the average waiting time increases. But it is still 71.43% of the MCT waiting time. This shows that the average waiting time of Ours carpool users has been greatly reduced, and the matching

success rate is still maintained at a high level. At the same time, when M=0.3, the success rate of Ours carpool matching algorithm is increased by 14.29% compared with MCT, and the time is shortened by 0.04h, which is better than MCT carpooling algorithm.

When evaluating the performance of carpooling algorithms, the average waiting time cost of passengers and the success rate of route matching are two important indicators. Compared with the MCT algorithm and the carpooling algorithm without time constraints, the newly proposed carpooling algorithm saves more time. The matching success rate of the MCT algorithm is significantly lower than the proposed carpool algorithm. The algorithm complexity is not much different, and the MCT carpool algorithm requires more waiting time, so the effectiveness of the newly proposed algorithm is better.

5. CONCLUSION

Carpooling is a means of vehicle sharing by which drivers share their cars with one or more riders whose travel itineraries are similar to their own. As such, carpooling can be an effective way to ease traffic congestion. With the rapid development of the mobile Internet and the widespread popularity of smart mobile devices, the amount of trajectory data carrying text information has greatly increased, and the subsequent trajectory similarity query research is also in full swing. The existing research on location prediction methods mostly focuses on the mining and analysis of trajectory data, but the research on how to improve the accuracy of location prediction through the information content contained in trajectory data and external data is not in-depth, and there is a lot of research.

Association rule mining is a very important research direction in data mining, and it is also a long-standing topic. Its main task is to find the inner connection between things. Frequent pattern mining, correlation mining, association rule learning, Apriori algorithm, etc. These seemingly different but essentially the same concepts have been used to describe the relevant content of data mining. The so-called data mining refers to the use of statistical methods to discover valuable and unknown laws from a certain data set. Classification or clustering methods are used, just to explore the relationship between the various subsets in the data set, to find which things often appear at the same time, which things are dependent on each other, and which things have connections. In this paper, based on inverse geocoding, the frequent patterns of the semantic trajectory sequence of a single user are mined, and the obtained frequent patterns are applied to carpooling scenes. Experiments found that this semantic trajectory-based carpool matching and recommendation is suitable for single carpooling. It is also suitable for relay transfer and carpooling. Its computational complexity is low, so that users can get a high matching carpool recommendation without changing their driving trajectory as much as possible. It can be used for location-based smart recommendations such as carpooling, HOV lane travel, etc. The service proactively explores needs and provides higher accuracy and timeliness guarantees. HOV lanes (High-Occupancy Vehicle Lane), also known as shared lanes or multi-occupant lanes, are lanes that can only be used by vehicles with at least a certain number of passengers as high-capacity lanes in traffic management. Vehicles in the lane include buses, cars or trucks with more than 2 people. It is a traffic management measure adopted by the United States, Canada and other countries to improve the efficiency of road use, alleviate traffic congestion, and promote transportation energy conservation and emission reduction. The track records the user's activities in the real world, and these activities reflect personal intentions, preferences and behavior patterns to a certain extent. The knowledge can be mined from the trajectory in this method: (1) user behavior, intention, experience and lifestyle are mined from personal data; (2) multi-person data are gathered to discover hotspots and classic routes; (3) understand people relevance between people, and the mode of activity between people and regions.

According to the trajectory dimension characteristics, the mining algorithm is divided into location-based trajectory frequent pattern mining, activity cycle-based trajectory frequent pattern mining and semantic-based trajectory frequent pattern mining. Location-based trajectory frequent pattern mining uses different search strategies to mine frequently repeated location sequences in

spatiotemporal databases. The algorithm features are generalized first search, depth first search and graph mining. It does not consider objective information of objects such as semantics and habits. Activity cycle-based trajectory frequent pattern mining is to convert the user trajectory sequence into a collection of points of interest, and perform periodic mining for the points of interest. The algorithm supports fine time granularity, the algorithm is complex, multiple intersections, and its data is sparse and incomplete. The algorithm is conducive to data compression, and due to the limitation of data sparseness, there is obvious uncertainty in mining information. Semantic-based trajectory frequent pattern mining is to give semantic information to spatial trajectories, and to mine frequent semantic behaviors from the set of semantic trajectories. Algorithm features are the similarity measurement of semantic trajectory, dwell time, and semantic behavior. It combines semantic information into frequent trajectory mining, which can extract motion laws more efficiently. The character sequence alone cannot fully express the semantic trajectory information. Semantic-based trajectory frequent pattern mining has gradually become the mainstream, and it will promote intelligent and personalized information services.

ACKNOWLEDGMENT

This research is supported by the Scientific research project in Hunan education department (18B528) and the double first-class application characteristic discipline (Applied Economics) of Hunan Province, China.

REFERENCES

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. *Proceedings of the 20th International Conference on Very Large Data Bases*, 487-499.

Alvares, L. O., Bogorny, V., & Kuijpers, B. (2007). A model for enriching trajectories with semantic geographical information. *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems*. doi:10.1145/1341012.1341041

Cao, X., Cong, G., & Jensen, C. S. (2010). Mining significant semantic locations from GPS data. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, *3*(1-2), 1009–1020. doi:10.14778/1920841.1920968

Cao, Y. Y., Cui, Z. M., & Wu, J. (2012). An Improved Hausdorff Distance And Spectral Clustering Vehicle Trajectory Pattern Learning Approach. *Computer Applications and Software*, *29*(5), 38–40. doi:10.3969/j.issn.1000-386X.2012.05.011

Chen, L. C. (2019). Based on The Document-Link and Time-Clue Relationships Between Blog Posts to Improve the Performance of Google Blog Search. *International Journal on Semantic Web and Information Systems*, *15*(1), 52–75. doi:10.4018/IJSWIS.2019010103

Duan, Y. B., Wu, J., & Zheng, H. Y. (2020). A greedy approach for carpool scheduling optimisation in smart cities. *International Journal of Parallel. Emergent and Distributed Systems*, *35*(5), 535–549. doi:10.1080/17445760.2018.1539718

Gao, Q., Zhang, F. L., & Wang, R. J. (2017). Trajectory Big Data: A Review of Key Technologies in Data Processing. *Journal of Software*, *28*(4), 959–992. doi:10.13328/j.cnki.jos.005143

Gao, Y. Q., Pan, X., & Wu, L. (2020). Similarity Join Query Algorithm Based On Semantic Trajectories. *Computer Applications and Software*, *37*(7), 14–21. doi:10.3969/ j.issn.1000-386x.2020.07.003

Guensler,, R., Ko, J., Kim, D., Khoeini, S., Sheikh, A., & Xu, Y. (2020). Factors affecting Atlanta commuters' high occupancy toll lane and carpool choices. *International Journal of Sustainable Transportation*, *14*(12), 932–943. doi:10.1080/15568318.2019.1663961

Guo, H., & Wang, L. X. (2017). Research on Carpool Path Matching Algorithm Based on Personalized Needs. *Computer Technology and Development*, *27*(1), 57–60. doi:10.3969/j.issn.1673-629X.2017.01.013

Harwood, A., & Karunasekera, S. (2018). Enhancing the quality of geometries of interest (GOIs) extracted from GPS trajectory data using spatio-temporal data aggregation and outlier detection. *Journal of Ambient Intelligence and Humanized Computing*, *9*(1), 173–186. doi:10.1007/s12652-016-0426-8

Huang, S. C., Jiau, M. K., & Lin, C. H. (2015). Optimization of the Carpool Service Problem via a Fuzzy-Controlled Genetic Algorithm. *IEEE Transactions on Fuzzy Systems*, *23*(5), 1698-1712. doi:10.1109/TFUZZ.2014.2374194

Huang, S. C., Jiau, M. K., & Liu, Y. P. (2019). An Ant Path-Oriented Carpooling Allocation Approach to Optimize the Carpool Service Problem With Time Windows. *IEEE Systems Journal*, *13*(1), 994-1005. doi:.279525510.1109/JSYST.2018

Huang, S. C., Lin, J. J., & Jiau, M. K. (2020). Global and Local Pareto Optimality in Coevolution for Solving Carpool Service Problem With Time Windows. *IEEE Transactions on Intelligent Transportation Systems*, *21*(3), 934–946. doi:10.1109/TITS.2019.2899160

Jiang, H., Han, A. Q., & Wang, M. J. (2014). Solution Algorithm of String Similarity Based on Improved Levenshtein Distance. *Computer Engineering*, *40*(1), 222–227. doi:10.3969/j.issn.1000-3428.2014.01.047

Liu, C., Tan, M. Q., & Shao, X. K. (2017). Carpooling algorithm research based on location data mining. *Computer Engineering and Applications*, *53*(8), 76–80. doi:10.3778/j.issn.1002-8331.1510-0229

Luo, W., Tan, H., & Chen, L. (2013). Finding time period-based most frequent path in big trajectory data. *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, 713-724. doi:10.1145/2463676.2465287

Meng, J. J., Yuan, J. S., Yang, J., Wang, G., & Tan, Y.-P. (2016). Object Instance Search in Videos via Spatio-Temporal Trajectory Discovery. *IEEE Transactions on Multimedia*, *18*(1), 116–127. doi:10.1109/TMM.2015.2500734

Pan, X. Y., Zhao, Q., & Zhao, P. (2019). Frequent Trajectory of Pattern Mining with Spatio-Temporal Attribute and Relationship Label. *Computer Engineering and Applications*, *55*(10), 83–89. doi:10.3778/j.issn.1002-8331.1801-0471

Pei, J., & Han, J. W. (2004). Mining sequential patterns by pattergrowth: The prefixspan approach. *IEEE Transactions on Knowledge and Data Engineering*, *6*(10), 1–17.

Radmanesh,, M., Sharma, B., Kumar, M., & French, D. (2021). PDE solution to UAV/UGV trajectory planning problem by spatio-temporal estimation during wildfires. *Chinese Journal of Aeronautics*, *34*(5), 601–616. doi:10.1016/j.cja.2020.11.002

Srikant, R., & Agrawal, R. (1995). Mining sequential patterns:generalizations and performance improvements. In *Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology*. Springer-Verlag.

Yang, J. Y., Meng, Z. Q., & Jiang, L. (2015). An algorithm for mining frequent routes based on topology information. *Computer Science*, *42*(4), 258–262.

Yang, W. L., & Feng, H. F. (2021). Analysis of Spatio-temporal Interaction Characteristics of Urban Area Based on Taxi GPS Trajectory. *Computer and Modernization*, (1), 87-93. doi:.1006-2475.2021.01.01710.3969/j.issn

Zaki, M. J. (2001). SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning Journal. Special Issue on Unsupervised Learning*, *41*(1), 31–60.

Zhang, C., Han, J., Shou, L., Lu, J., & La Porta, T. (2014). Splitter: Mining finegrained sequential patterns in semantic trajectories. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, *7*(9), 769–780. doi:10.14778/2732939.2732949

Zhang, C. D., & Bie, Z. N. (2019). Research on Improved Carpool Algorithm Based on 3D Space-Time Trajectory. *Computer Engineering and Applications*, *55*(13), 239–245. doi:10.3778/j.issn.1002-8331.1804-0095

Zhang, J. L., Shi, H. L., & Cui, L. (2019). Location Prediction Model Based on Transportation Mode and Semantic Trajectory. *Journal of Computer Research and Development*, *56*(7), 1357–1369. doi:10.7544/issn1000-1239.2019.20170662

Zhang, M. H., Du, D. H., & Zhang, M. Z. (2021). Spatio-temporal Trajectory Data-driven Autonomous Driving Scenario Meta-modeling Approach. *Journal of Software*, *32*(4), 973–987. doi:10.13328 /j.cnki.jos.006226

Zuo, K. Z., Tao, J., & Zeng, H. Y. (2018). Algorithm for Trajectory Movement Pattern Mining Based on Semantic Space Anonymity. *Netinfo Security*, (8), 34–42. doi:10.3969/j.issn. 1671-1122.2018.08.005