2020 IEEE International Conference on Big Data (Big Data) | 978-1-7281-6251-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/BigData50022.2020.9377839

Learning Minimum Bounding Rectangles for Efficient Trajectory Similarity Search

Hani Ramadhan
Big Data Department
Pusan National University, South Korea
Email: hani042@pusan.ac.kr

Joonho Kwon
School of Computer Science and Engineering
Pusan National University, South Korea
Email: jhkwon@pusan.ac.kr

***Abstract*—Early pruning of dissimilar trajectories is important in similar trajectory search on a big mobility data. R-trees can perform the pruning effectively, but the search and index size become inefficient due to numerous overlapping of minimum bounding regions in a dense and big dataset. Thus, we introduce the extended usage of learned index to learn the minimum bounding rectangles for trajectory similarity search. Our approach is designed to provide an effective pruning for trajectory similarity search with less storage size.**

***Index Terms*—big mobility data, similar trajectory search, learned index**

I. Introduction

Currently, trajectory recommendation as a spatial analysis on mobility data becomes more difficult due to tremendous growth of mobility data generation. Trajectory recommendation requires the trajectory similarity search query to identify the frequently used paths in the dataset as the best candidate route to use. To perform the trajectory similarity search effectively, tree-based indices are commonly used. Tree-based indices, such as R-tree, optimizes the trajectory similarity search by pruning dissimilar trajectories[1].

R-trees prune the dissimilar trajectories whose minimum bounding rectangles (MBR) do not intersect the MBR of the query trajectory. In large dataset, however, many trajectories may skewed to cover a common area, thus the R-tree creates numerous MBR in that area. If the MBR of the query trajectory intersects this common area, the R-tree leaves many trajectories unpruned. For example, in Figure 1, the similar trajectory candidate to the query trajectory Q might not exist in the dataset. However, due to the too many overlapping MBR in the lower right area, the R-tree fails to prune the other trajectories. Then, we need to scan all the trajectories in the dataset. Thus, the query time becomes longer and the index size also gets large.

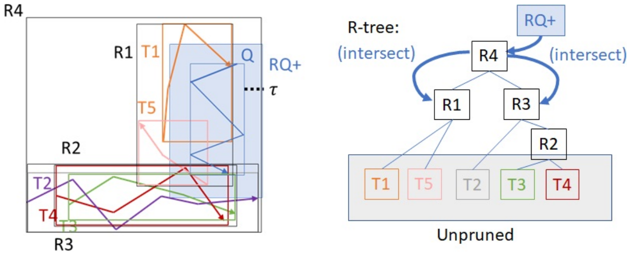


Fig. 1. The extended MBR RQ^+ of query trajectory Q within threshold τ intersects all the MBRs in an R-tree, thus leaving all trajectories in dataset unpruned.

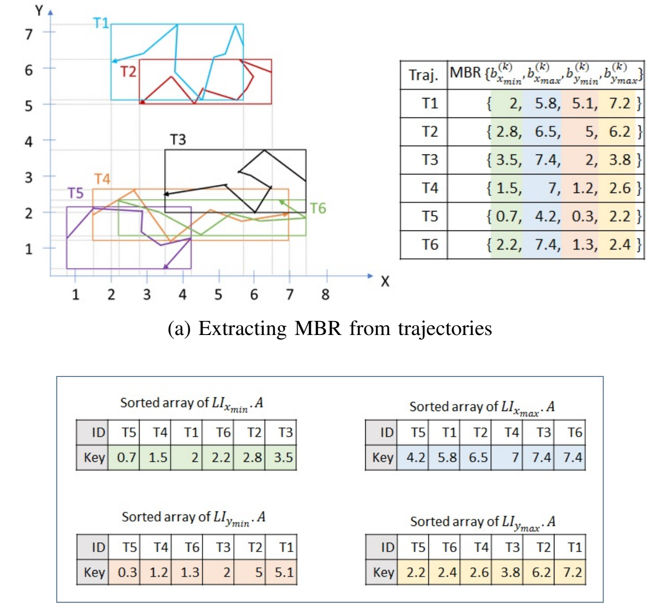
The recent learned index approach [2] showed faster query time on one attribute dataset with less index size than tree-based indices. However, it is hard to implement the learned index to more complicated structures, such as trajectory, polygon, and MBR, which have multiple points. Thus, our purpose is to extend the learned index for the MBR of trajectories for similar trajectory search. Any MBR in n-dimensional space can be viewed as $2n$ separate values. Therefore, we can perform $2n$ range queries given the threshold τ and the query trajectory MBR. With $2n$ learned index on each MBR values, the range query will only return the closest trajectories MBRs to the query trajectory. Thus, using the learned MBR index, we can prune dissimilar trajectories more effectively.

Our key contribution in this paper is the introduction of the learned index approach of MBR pruning. We present this paper as an initial study to perform an effective similar trajectory search on large trajectory datasets with the help of machine learning indexing.

II. Extending Learned Index to MBR for Pruning

First, we describe a trajectory and its MBR. A trajectory T is finite-length sequence of points in a n-dimensional space. The MBR of a trajectory as a set of minimum and maximum values of its points in the n-dimensional space. Thus, the MBR $M^{(k)}$ of $T^{(k)}$ consists of $2n$ values $M^{(k)} = \{b^{(k)}_{j_{min}}, b^{(k)}_{j_{max}}\}$, where $1 \leq j \leq n$, $b^{(k)}_{j_{min}} = \min_i p^{(k)}_{i,j} \in T^{(k)}$, $b^{(k)}_{j_{max}} = \max_i p^{(k)}_{i,j} \in T^{(k)}$, and $p^{(k)}_i = (p^{(k)}_{i,j} | 1 \leq j \leq n)$ is an n-D point member of trajectory $T^{(k)}$.

This work was partly supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2020R1I1A3072457), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-00121, Development of data improvement and dataset correction technology based on data quality assessment) and the MSIT(Ministry of Science and ICT) under the ITRC(Information Technology Research Center) support program (IITP-2020-0-01797) supervised by the IITP.



Traj.	MBR $\{b^{(k)}_{x_{min}}, b^{(k)}_{x_{max}}, b^{(k)}_{y_{min}}, b^{(k)}_{y_{max}}\}$
T1	{ 2, 5.8, 5.1, 7.2 }
T2	{ 2.8, 6.5, 5, 6.2 }
T3	{ 3.5, 7.4, 2, 3.8 }
T4	{ 1.5, 7, 1.2, 2.6 }
T5	{ 0.7, 4.2, 0.3, 2.2 }
T6	{ 2.2, 7.4, 1.3, 2.4 }

(a) Extracting MBR from trajectories

Sorted array of $LI_{x_{min}}.A$

ID	T5	T4	T1	T6	T2	T3
Key	0.7	1.5	2	2.2	2.8	3.5

Sorted array of $LI_{x_{max}}.A$

ID	T5	T1	T2	T4	T3	T6
Key	4.2	5.8	6.5	7	7.4	7.4

Sorted array of $LI_{y_{min}}.A$

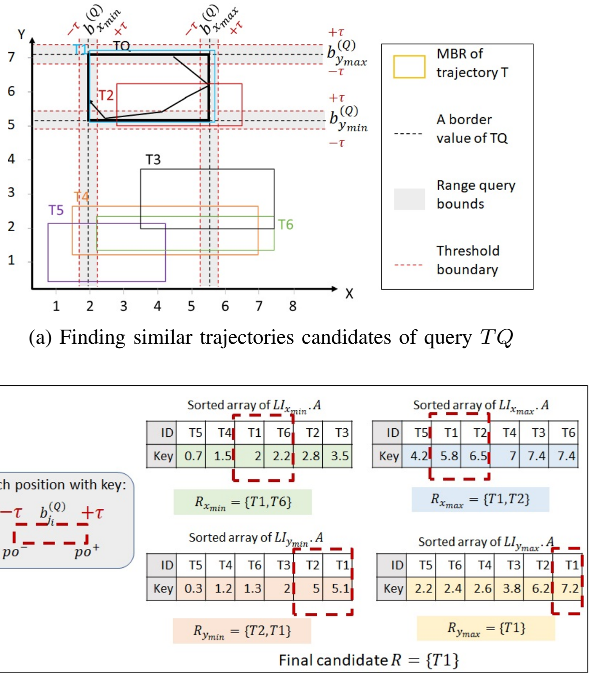
ID	T5	T4	T6	T3	T2	T1
Key	0.3	1.2	1.3	2	5	5.1

Sorted array of $LI_{y_{max}}.A$

ID	T5	T6	T4	T3	T2	T1
Key	2.2	2.4	2.6	3.8	6.2	7.2

(b) Sorted arrays of each border values of MBR of trajectories

Fig. 2. Constructing the sorted array of the learned MBR



(a) Finding similar trajectories candidates of query TQ

Sorted array of $LI_{x_{min}}.A$

ID	T5	T4	T1	T6	T2	T3
Key	0.7	1.5	2	2.2	2.8	3.5

$R_{x_{min}} = \{T1, T6\}$

Sorted array of $LI_{x_{max}}.A$

ID	T5	T1	T2	T4	T3	T6
Key	4.2	5.8	6.5	7	7.4	7.4

$R_{x_{max}} = \{T1, T2\}$

Sorted array of $LI_{y_{min}}.A$

ID	T5	T4	T6	T3	T2	T1
Key	0.3	1.2	1.3	2	5	5.1

$R_{y_{min}} = \{T2, T1\}$

Sorted array of $LI_{y_{max}}.A$

ID	T5	T6	T4	T3	T2	T1
Key	2.2	2.4	2.6	3.8	6.2	7.2

$R_{y_{max}} = \{T1\}$

Final candidate $R = \{T1\}$

(b) Pruning using learned MBR

Fig. 3. The similar trajectory candidates search using our learned MBR

Example 1. Suppose there are six trajectories in the dataset of 2D space as depicted in Figure 2(a). We extract the MBR of each trajectory by scanning all of its points and taking the minimum and maximum values of each dimension of their points. For example, trajectory $T3$ have a sequence of 7 points $\{(7.5, 3), (6.2, 3.8), (5.6, 3.2), (6, 3), (6.5, 2.8), (6, 2), (5.2, 2.8), (3.5, 2.5)\}$. Thus, the MBR $M^{(3)}$ of $T3$ is $\{3.5, 7.4, 2, 3.8\}$.

After we construct MBRs of all trajectories, we create the learned indices for those MBRs. For an n-D trajectory dataset D, we create $2n$ learned indices $\{LI_{j_i} | 1 \leq j \leq n, i \in \{min, max\}\}$ as the learned MBR. Each index L_{j_i} contains a sorted array $L_{j_i}.A$ of length $|D|$. The element of the sorted array is a key value pair $\langle b^{(k)}_{j_i}, k \rangle$ where $b^{(k)}_{j_i}$ is an element of $M^{(k)}$ and k is the id of $T^{(k)} \in D$. Then, each index L_{j_i} is trained by setting the existing keys of the sorted array as input and its position as the output. We use the piecewise linear model [3] to train the model and restrict its error bound to ϵ. Therefore, our prediction of the key position in the sorted array will never be more or less than ϵ to the actual position.

Example 2. Continuing from the previous example in Figure 2(a), we sort the MBR values of the trajectories according to each maximum and minimum value. Thus, in 2D trajectory dataset, we construct four sorted arrays for each values, either maximum or minimum, depicted in Figure 2(b). Then, for each sorted array, we train a set of machine learning model. For example, in $LI_{x_{min}}$, we train piecewise linear regression models with $\{(0.7, 0), (1.5, 1), (2, 2), (2.2, 3), (2.8, 4), (3.5, 5)\}$ as training set consisting pairs of input and output, and error bound $\epsilon = 1$. Thus, we will have a set of, at maximum, two linear models. With the query $key = 2$, we are guaranteed that the predicted position is in $[1, 3]$ inclusively.

Our MBR pruning identifies the similar trajectory candidates whose MBR are included within the similarity threshold, a positive real number, τ of query trajectory $T^{(Q)}$, as depicted in Figure 3. Each learned index L_{j_i} takes an input of $T^{(Q)}$ and extracts its MBR $M^{(Q)} = \{b^{(k)}_{j_{min}}, b^{(k)}_{j_{max}}\}$, where $1 \leq j \leq n$. Next, we perform $2n$ range queries according to $2n$ learned indices. The keys of each range query is composed by $\{b^{(Q)}_{j_i} - \tau, b^{(Q)}_{j_i} + \tau\}$ as lower bound and upper bound respectively. Then, L_{j_i} locates the positions $\{po^-, po^+\}$ of the search keys $\{b^{(Q)}_{j_i} - \tau, b^{(Q)}_{j_i} + \tau\}$ in the sorted array $L_{j_i}.A$, where po^- and po^+ correspond to the positions of the lower and upper bound range key, respectively. Next, each L_{j_i} reports the trajectory id in $R_{i_j} = \{k | k \in L_{j_i}.A[po^-, po^+].Ids()\}$. Finally, we acquire the id of similar trajectories candidates to $T^{(Q)}$ identified by MBR pruning as $R = \cap_{j \in [1,n], i \in \{min, max\}} R_{j_i}$.

Example 3. Suppose that we have a query trajectory TQ and $\tau = 0.25$ as depicted in Figure 3(a). We perform a range query for each learned index $\{LI_{x_{min}}, LI_{x_{max}}, LI_{y_{min}}, LI_{y_{max}}\}$ based on the border values of TQ. For instance, we execute range query with 1.8 as lower bound and 2.2 as upper bound on sorted array $LI_{x_{min}}.A$. Therefore, we acquire $R_{x_{min}} = \{T1, T6\}$ as the similar trajectory candidates. Next, we perform the similar range query with corresponding border values as key to $LI_{x_{max}}$, $LI_{y_{min}}$, and $LI_{y_{max}}$. Then, we intersect every results of the range search $\{R_{x_{min}}, R_{x_{max}}, R_{y_{min}}, R_{y_{max}}\}$ as depicted in Figure 3(b) to prune dissimilar trajectories. Finally, we return the final candidate of similar trajectory candidate $R = \{T1\}$.

III. Dataset

We plan to experiment on two different dense 2D trajectories datasets, which are Porto taxi drive[1] dataset (1.5 GB) [4] and Geolife mobility dataset [5], [6], [7] (648 MB). The number of trajectories in Porto and Geolife dataset are 1.6 million and 18,670, respectively.

IV. Conclusion

In this paper, we have presented an initial study of small-space consuming learned index for MBR in trajectory similarity search in a large dataset as an alternative to the tree-based indices. The future works include the extension of the learned MBR to multi-dimensional data, thus giving a constant size array regardless of the number of dimensions in a large trajectory dataset.

References

[1] Z. Shang, G. Li, and Z. Bao, "DITA: distributed in-memory trajectory analytics," in *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, G. Das, C. M. Jermaine, and P. A. Bernstein, Eds. ACM, 2018, pp. 725–740. [Online]. Available: https://doi.org/10.1145/3183713.3183743

[2] T. Kraska, A. Beutel, E. H. Chi, J. Dean, and N. Polyzotis, "The case for learned index structures," in *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, G. Das, C. M. Jermaine, and P. A. Bernstein, Eds. ACM, 2018, pp. 489–504. [Online]. Available: https://doi.org/10.1145/3183713.3196909

[3] V. Nathan, J. Ding, M. Alizadeh, and T. Kraska, "Learning multi-dimensional indexes," in *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo, Eds. ACM, 2020, pp. 985–1000. [Online]. Available: https://doi.org/10.1145/3318464.3380579

[4] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi-passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, 2013. [Online]. Available: https://doi.org/10.1109/TITS.2013.2262376

[5] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W. Ma, "Understanding mobility based on GPS data," in *UbiComp 2008: Ubiquitous Computing, 10th International Conference, UbiComp 2008, Seoul, Korea, September 21-24, 2008, Proceedings*, ser. ACM International Conference Proceeding Series, H. Y. Youn and W. Cho, Eds., vol. 344. ACM, 2008, pp. 312–321. [Online]. Available: https://doi.org/10.1145/1409635.1409677

[6] Y. Zheng, L. Zhang, X. Xie, and W. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, Eds. ACM, 2009, pp. 791–800. [Online]. Available: https://doi.org/10.1145/1526709.1526816

[7] Y. Zheng, X. Xie, and W. Ma, "Geolife: A collaborative social networking service among user, location and trajectory," *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010. [Online]. Available: http://sites.computer.org/debull/A10june/geolife.pdf

[1] http://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html