Mobile Big Data Query Based on Double R-tree and Double Indexing

Liang Ye
Department of Computer Science
Beijing Foreign Studies University
Beijing, China
E-mail: liang_ye@sohu.com

***Abstract*—The amount of data in our industry and the world is exploding. Data is being collected and stored at unprecedented rates. The challenge is not only to store and manage the vast volume of data, which is also called big data, but also to analyze and query from it. In order to put forward the universal method to response mobile big data query, queries are separated and grouped according to kinds of query for massive mobile objects in the space. The indexing method for grouping the mobile objects with Grid (GG TPR-tree) has great efficiency to manage a massive capacity of mobile objects within a limited area, but it only could meet a part of requirements for mobile big data query if the GG TPR-tree was used solely. This thesis offers solutions to simple immediate query, simple continuous query, active window query, and continuous window query, dynamic condition query and other query requests by employing DTDI index structure. The experiments prove that with the support of DTDI index structure, query of massive mobile objects has higher precision and better query performance.**

Keywords-mobile query, big data, double R-tree and double indexing

I. INTRODUCTION

The mobile objects query usually deals with the massive mobile objects' data sets that their locations change all the time and the query itself is also dynamic. [1] In order to meet the forecast query request that facing the mobile objects, in recent years, researchers propose a variety of mobile objects index structure and its query algorithms. [2, 3, 4, 5] The representatives are the grid dividing method and the query method based on PRA-tree index structure.

Giving that the users concerns more about the query's response speed in some cases and allow certain error in the query results, Jimeng Sun[6] et al proposed a grid based on the two-dimensional space dividing. The mobile objects update their positions in the way of data stream and the grid saves the mobile objects' information collection that fall in it. They use statistical and sampling method to do the approximate query. This method can respond to the real-time query in a moment of the history, current and future and can not respond to the continuous query and the window query.

In order to meet all kinds of forecast query quest that facing the mobile objects, establishing an efficient mobile objects index structure is a necessity. In 2000, Simonas Saltenis[7] et al proposed TPR-tree index structure to meet the forecast query request; however, it contains the problem that the index performance declined rapidly with time in the process of indexing. Therefore, many domestic scholars have proposed a variety of improved methods. Among those methods, PRA-tree (predictive range aggregate) index structure and its query method is the most effective one which is proposed by Liao Wei et al in 2007, National Defense Technology University. [8] Its idea is that while indexing the mobile objects, firstly, it divides the velocity field into many velocity barrels with same speed taking the maximum and minimum values of the mobile objects' speed as the extreme points in the indexing group. And then map the objects into different velocity barrels according to the size of the velocity vector in this velocity field. At last, indexing the mobile objects in every velocity could barrel with the existing TPR-tree.

II. QUERY THOUGHTS BASED ON THE DIDI STRUCTURE

In most cases, the quantity of the massive mobile objects is much larger than that of the roads, and the establishment of GG TPR-tree makes it possible to manage the massive mobile objects overall. To complete the high efficient query to the various requirements under the guidance of the space partitioning tree, we preserve the original establishment thoughts of the GG TPR-tree and adjust the definition of GG TPR-tree's leaf node as follows:

We propose Rate Accumulation Model: the massive mobile objects go into the grid when t=0, the mobile object's mobile rate accumulation scalar value in the road of the grid is:

$$\tilde{v} = \sum_{t=1}^{T}(t \times v_t) / \sum_{t=1}^{T}(t) \qquad (1)$$

The Quintuples records of GG TPR-tree leaf nodes are changed into <grid, $ptr_{objects}$, MBR, $\tilde{v}_{Groop}$, ptr_{parent}>, respectively representing the grid that this group of massive mobile objects belong to, the pointer pointed to the massive mobile objects queue, the bounding rectangle of the nodes, the node's velocity stable value, the pointer pointed to parent nodes, the record form of GG TPR-tree non-leaf node does not change.

Estimating equation (1) shows that supposing there are n members in a group of the massive mobile objects, the MBR stable value is achieved by the equation as follows:

$$\tilde{v}_{Groop} = \sum_{t=1}^{n}((T - t_i) \times \tilde{v}_i) / \sum_{t=1}^{n}(T - t_i) \qquad (2)$$

Under the help of GG TPR-tree and spatial partitioning tree, we can predict the approximate location mobile direction according to $\tilde{v}_{Groop}$ and road network structure, achieving the exact answer of the statistical query.

Meanwhile, in order to update the dynamic information of the massive mobile objects efficiently, we establish Hash index list pointed to every mobile object in the memory. This list may also achieve the exact answer of the deterministic query in collaboration with the spatial partitioning tree. Therefore, we establish the massive mobile objects' complete query basis based on DTDI structure.

III. THE QUERY METHOD BASED ON DTDI STRUCTURE

The massive mobile objects query method based on DTDI does not only answer efficiently the deterministic query request like simple real-time query and simple continuous query, but also answer the statistic query request like current window query, continuous window query and dynamic condition query. Next we will discuss how to use DTDI structure to achieve the query to the massive mobile objects.

A. Simple real-time query

The simple real-time query is a common deterministic query request which concerns about the location attribute information of a specific object at some point. The index tree mentioned above stored a triple information record for every mobile object < Loc_{ref} , $\tilde{v}$, t_{ref} >. Only when the discrepancy between actual position and theoretical position calculated by the triples is greater than the distance error allowed by the system, the triple group is updated. So we can precisely calculate the massive mobile objects' current location according to the triple group and road curve-fitting function.

$$S_t = \int_{t_{ref}}^{t} \tilde{v} dt + S_{ref} \tag{3}$$

In the formula, S_{ref} is the distance between the starting point and (x_0 , y_0) in the reference time Loc_{ref} ; is the distance between the starting point and the massive mobile objects in the query moment. We can obtain reversely the massive mobile objects' location coordinate (x, y) at the moment according to the road curve-fitting function.

B. Simple continuous query

The simple continuous query is the expansion of the simple real time query which queries the motional track of the massive mobile objects with a period of time in the future. In common condition, we can obtain the massive mobile objects' motional track function with the use of triples and the road fitting curve. But if the massive mobile objects need to pass the crossing point with the query duration, the prediction of the motional track will become more complex.

Suppose the location function of the simple continuous query is S(t), we can obtain the distance S_0 between the location of Loc:(x_0 , y_0) and the starting point and the distance S_{terminal} between the location of Loc:(x_0 , y_0) and next crossing point, and then the time to the crossing point is:

$$\begin{aligned} &\int_{t_{ref}}^{t_{\text{terminal}}} \tilde{v} dt = S_{\text{terminal}} - S_{ref} \\ &\Rightarrow t_{\text{terminal}} = \left(S_{\text{terminal}} - S_{ref}\right)/\tilde{v} + t_{ref} \end{aligned} \tag{4}$$

If the continuous time $t \le t_{\text{terminal}}$, it shows by (3):

$$S(t) = \int_{t_{ref}}^{t} \tilde{v} dt + S_{ref} \tag{5}$$

If the continuous time $t > t_{\text{terminal}}$, supposing the massive mobile objects go through the crossing points related to G_1 , G_2 , G_4 , G_5 , as shown in picture one, the possibility are 10%, 5%, 75%,10%, remaining in grid G_1 and in grid G_2 , G_4 , G_5 . After the massive mobile objects pass over the crossing point, the speed may change due to their move to other massive mobile objects group. Assuming the previous measured accumulated speed accumulation scalar values are $\tilde{v}_{G_1}$, $\tilde{v}_{G_2}$, $\tilde{v}_{G_4}$, $\tilde{v}_{G_5}$, and then:

$$\begin{cases} S(t) = \int_{t_{\text{terminal}}}^{t} \tilde{v}_{G_1} dt + S_{\text{terminal}} & (p = 10\%) \\ S(t) = \int_{t_{\text{terminal}}}^{t} \tilde{v}_{G_2} dt + S_{\text{terminal}} & (p = 5\%) \\ S(t) = \int_{t_{\text{terminal}}}^{t} \tilde{v}_{G_4} dt + S_{\text{terminal}} & (p = 75\%) \\ S(t) = \int_{t_{\text{terminal}}}^{t} \tilde{v}_{G_5} dt + S_{\text{terminal}} & (p = 10\%) \end{cases} \tag{6}$$

C. Current window query

The current window query is one of the common statistical queries, whose request content generally designates a piece of an area. It queries all the massive mobile objects within the area in the current time.

According to the massive mobile objects' query feature based on DTDI structure, we can know that when a group of massive mobile objects pass through the crossing point, the corresponding leaf-node record's grid pointer is bound to change, which index the massive mobile objects' group. Therefore, the massive mobile objects that go into the query window are those having the intersection with the query window on the road. To this end, we can take use of the spatial partitioning tree based on the grid to index the related roads and calculate the minimum MBR that related to these roads. Then we index the massive mobile objects group on these roads and estimate whether there is intersection

between MBR and the query window according to the velocity stable value of this group's node $\tilde{v}_{Groop}$. If there exists the intersection between the two, we say that part of the massive mobile objects belong to the window query result category. We need to use the formula (3) to calculate each $\tilde{v}$ and decide how many massive mobile objects meet query condition.

D. Continuous window query

The continuous window query is the expansion of current window query, which needs to know how many massive mobile objects meet the query condition during the future time period [0, T]. Since the massive mobile objects can go into the query field within the time t through the way of changing the direction at some crossing points, the fields that need to be inspected are no longer limited to the minimum MBR which is composed of the roads intersected with the query window, but need to inspect all the roads intersecting with the roads within the query window. Thus, we need to carry on the traversal query from the root node of GG TPR-tree and do the following calculation with the use of MBR and VBR of every node:

- Judging if we can extend MBR line to intersect with the query window according to the current maximum value of VBR within t.
- If they intersect with each other, we can calculate the shortest path from the current MBR to the query window according to the spatial partitioning tree structure. And then judging if they can arrive at destination within the time of T. When there is no linking condition, the shortest path is $+\infty$.
- Repeat the previous process until we choose out all the leaf nodes that can reach the destination. According to the formula (6), we can calculate that how many massive mobile objects can reach the query field within T among the massive mobile objects group that the leaf nodes point to.
- Sum the results of the last step and achieve the query outcome.

E. Dynamic condition query

The dynamic condition query is the one that a kind of the massive mobile objects and the query condition itself change dynamically with the time. Tao et al gave out the basic concepts of the distance function with dynamic query in document [6].

Definition of the spatial distance function: in d dimensional European type space, supposing the reference time is t_{ref} and the query Q and the mobile object O move with the linear vector speed. While in any moment t of the future, the function of the distance between the location Loc_t of query Q and the mobile object O is:

$$\mathrm{dist}(Q, O, t) = At^2 + Bt + C$$

(A, B, C are all constants)

After joining into the grid information, the massive mobile objects have to move on the limited road network, thus the spatial distance between the two is limited by the grid route. In addition, road condition changes dynamically among grids. Supposing the connecting way between two positions in the spatial roads is more than one, there may be the situation that the massive mobile objects consumes more time among the absolute distance that is shorter. Considering these factors above, we define the function of grid distance between the two points.

Definition of the grid distance function: in the grid space, supposing in the reference time t_{ref} , the query Q and the mobile object O move on the road of the edge with the linear speed, and there are k-1 crossing points between the next crossing point of Q and that of O, that is to say they will pass through k roads L_1, L_2, ⋯, L_k, and the blocking rate of every road are C_1, C_2, ⋯, C_k and $\mathrm{C}_n = 1/\tilde{v}_n$. In any moment of the future t, the function of the distance that Q leaves the mobile object O is:

$$\overrightarrow{\mathrm{dist}(Q, O, t)} = \min\left(\sum_k \mathrm{L}_k \times C_k\right) + \tilde{v}_Q \times t - \tilde{v}_o \times t \quad (7)$$

The function of the distance that the mobile object O leaves the query Q is:

$$\overrightarrow{\mathrm{dist}(O, Q, t)} = \min\left(\sum_k \mathrm{L}_k \times C_k\right) + \tilde{v}_O \times t - \tilde{v}_Q \times t \quad (8)$$

We know the grid distance function between the query Q and the mobile object O by (7), (8):

$$\mathrm{dist}(Q, O, t) = \min\left(\overrightarrow{\mathrm{dist}(Q, O, t)}, \overrightarrow{\mathrm{dist}(O, Q, t)}\right) \quad (9)$$

When the request of dynamic condition query is raised, firstly, we start to search from the root node according to GG TPR-tree. If the node MBR's four vertices have $dist_1 = 0$, $dist_2 = 0$, $dist_3 = 0$, $dist_4 = 0$ in the time t_1, t_2, t_3, t_4 ($t_1, t_2, t_3, t_4 \in [0, T]$), and $t_1 \neq t_2 \neq t_3 \neq t_4$ separately, within the range of [0, T]. Therefore, all the child nodes of this node do not meet the query condition. If t_1, t_2, t_3, t_4 do not exit, all the child nodes of this node do not meet the query condition. And then we cut the branches. If met $dist_i = 0, \exists i \in 1,2,3,4$, then repeat the process to calculate the grid distance among its child nodes. At last, summarizing the leaf nodes that satisfy the conditions and we achieve the query results set.

Algorithm 1: dynamic condition query algorithm

Input: GG TPR-tree, space dividing tree, query region, query continuance (T)

Output: query results nodestack

BEGIN

/* Put the root node of GG TPR-tree into the nodestack.*/

1. nodestack $\leftarrow$ root of GG TPR-tree

2. For all node nodestack

/*using the distance with space dividing tree, query region, T to calculate */

3. Calculate $dist_1$, $dist_2$, $dist_3$, $dist_4$

4. If not $dist_i = 0, \forall i \in 1,2,3,4$ then

/* Delete the node whose minimum distance is greater than that within the query request. */

5. Delete node from nodestack

6. else

7. if $dist_i = 0, \exists i \in 1,2,3,4$ then

/* Delete the node whose maximun distance is greater than that within the query request. */

8. Delete node from nodestack

/* Insert all child nodes of this node into the result stack. */

9. nodestack $\leftarrow$ all child nodes of node

10. End If

11. End If

12. End For

END

IV. EXPERIMENT

In order to evaluate the query performance of massive mobile objects based on DTDI structure, we conduct the following experiments: the space region is a rectangular frame consisting of 800*600 pixels. There are "four horizontal, five vertical"simulate road in it. Each mobile object on the road is represented by a circle with a 40-pixel radius. The objects are distributed randomly on all roads. We adopt the proportion of 100 steps versus 100 steps to allocate the time of red lights and green lights at the crossing roads. For simplicity, we assume that all the roads are two-way and one-lane, that is to say, the massive mobile objects are incapable of surpassing the objects ahead. In addition, all the turnings (include the right turnings) are controlled by signals and the turning probability of each mobile object is known already.

as the VBR of PRA tree differs little from the $\tilde{v}$ of DTDI and both approximate the actual velocity of massive mobile objects, therefore, the predicted data of massive mobile objects within a recent time range are relatively accurate. With the increase of the predicted time range, under the guidance of network structure, turning probability of crossing point, the accumulated EXP velocity value $\tilde{v}$ recorded in history record, DTDI is capable of ensuring that the predicted direction and velocity approximate the possibly actual direction and velocity of the massive mobile objects in the future, therefore, the predicted precision is relatively high. However, as $\tilde{v}$ is adopted as EXP after passing the crossing point, and there exist certain errors compared with actual velocity value in the future, as a result, the precision becomes gradually low as the predicted time range increases. To PRA tree, there is no guidance of network structure in it, so it is incapable of considering turning, therefore, its predicted success rapidly decrease. Meanwhile, because the mass data can compensate for the error, accordingly, the ratio between the predicted number and the actual number will not be extremely low.

V. CONCLUSION

This thesis offers solutions to simple immediate query, simple continuous query, active window query, and continuous window query, dynamic condition query and other query requests by employing DTDI index structure. The experiments prove that compared with other existing index structures of massive mobile objects, the results of predicted queries of massive mobile objects within a limited range. With the support of DTDI index structure, query method of massive mobile objects has higher precision and better query performance.

ACKNOWLEDGMENT

This paper is supported by the Fundamental Research Funds for the Central Universities (No.2012XJ031). Without this help, this work would never have been completed.

REFERENCES

[1] K. Tabassum, M. Hijab, and A. Damodaram, "Location Dependent Query Processing – Issues, Challenges and Applications", 2010 Second International Conference on Computer and Network Technology (ICCNT 2010), 2010, pp.239-243, doi: 10.1109/ICCNT.2010.39.

[2] Qing Zhu, and Zuoyan Qin, "HyDB: Access Optimization for Data-Intensive Service", 2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS 2012), 2012, pp.580-587, doi: 10.1109/HPCC.2012.84.

[3] Ye Liang, "Double R-tree and Double Indexing for Mobile Objects", Advances in Intelligent and Soft Computing, 2012, Volume 135, pp. 271-278.

[4] Ye Liang, "An Efficient Indexing Maintenance Method for Grouping Moving Objects with Grid", Procedia Environmental Sciences, 2011, 11(1), pp. 486-492.

[5] Ye Liang, "GG TPR-tree Indexing Method for Grouping Moving Objects", Proceedings of International Conference on Computers, Communications, Control and Automation, 2011, pp.375-378.

[6] Jimeng Sun, Dimitris Papadias, Yufei Tao, and Bin Liu, Querying about the past, the present, and the future in spatio-temporal database. 20th International Conference on Data Engineering, 2004, pp.202-213, doi: 10.1109/ICDE.2004.1319997

[7] Simonas Saltenis, Christian Jensen, and S Jensen, "Indexing the positions of continuously moving objects" SIGMOD 2000, 2000, pp. 331-342.

[8] Wei Liao, Ning Jing, and Zhinong Zhong, "An efficient prediction technique for range aggregation of moving objects", Journal of Computer Research and Development, 2007 Vol.44(6):1015-1021