JOURNAL OF KOREA MULTIMEDIA SOCIETY, VOL. 9, NO. 12, DECEMBER 2006(pp. 1529-1541)

Index based on Constraint Network for Spatio-Temporal Aggregation of Trajectory in Spatial Data Warehouse

JingJing Li[†], Dong-Wook Lee[††], Byeong-Seob You[†††], Young-Hwan Oh[††††], Hae-Young Bae[†††††]

ABSTRACT

Moving objects have been widely employed in traffic and logistic applications. Spatio-temporal aggregations mainly describe the moving object's behavior in the spatial data warehouse. The previous works usually express the object moving in some certain region, but ignore the object often moving along as the trajectory. Other researches focus on aggregation and comparison of trajectories. They divide the spatial region into units which records how many times the trajectories passed in the unit time. It not only makes the storage space quite ineffective, but also can not maintain spatial data property. In this paper, a spatio-temporal aggregation index structure for moving object trajectory in constrained network is proposed. An extended B-tree node contains the information of timestamp and the aggregation values of trajectories with two directions. The network is divided into segments and then the spatial index structure is constructed. There are the leaf node and the non leaf node. The leaf node contains the aggregation values of moving object's trajectory and the pointer to the extended B-tree. And the non leaf node contains the MBR (Minimum Bounding Rectangle), MSAV (Max Segment Aggregation Value) and its segment ID. The proposed technique overcomes previous problems efficiently and makes it practicable finding moving object trajectory in the time interval. It improves the shortcoming of R-tree, and makes some improvement to the spatio-temporal data in query processing and storage.

Keywords: Trajectory Aggregation, Constraint Network, Spatio-temporal Data Warehouse

※ Corresponding Author : Hae-Young Bae, Address : (402-715) 253 Yonghyun-dong Nam-gu Incheon, Korea, TEL : +82-32-860-8719, FAX : +82-32-862-9845, E-mail : ljj916@gmail.com

Receipt date : Aug. 29, 2006, Approval date : Oct. 23, 2006

[†] Dept. of Computer and Information Engineering, Inha University
(E-mail : longjing916@gmail.com)
[††] Dept. of Computer and Information Engineering, Inha University
(E-mail : dwlee@dblab.inha.ac.kr)
[†††] Dept. of Computer and Information Engineering, Inha University
(E-mail : subi@dblab.inha.ac.kr)
[††††] School of Information Science, Korea Nazarene University
(E-mail : yhoh@kornu.ac.kr)
[†††††] Dept. of Computer and Information Engineering, Inha University

※ This research was supported by the MIC (Ministry of Information), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

1. INTRODUCTION

During the more and more applications and GIS data accumulation for a long time, the aggregation of moving objects becomes an important role in businesses and economic geography in a local area. As estimation, more than 80 percent business and economic data are related the spatial time identity and location. If we explore and apply these data efficiently, it will optimize resource allocation, decrease the business exertion cost and improve the economic environment. On the other hand, spatio-temporal data warehouse, is the technology that not only for geography data's capture, storage, transform, analysis and presentation, but also supplying the way to know the spatial phenomenon and to solve temporal problem. It can be used in defining, analyzing and presenting the complex

spatio-temporal economic phenomenon. So it is available for decision support and makes interrelated policy meliorate the competition[1].

In the real world, the car usually moves in the road and it becomes the network when there are so many roads. That means that the trajectory of moving object is a subject about constraints, and the network is the constrained network.

The behavior of moving objects is described by spatio-temporal aggregations as indexing techniques in spatial data warehouse. In this case, how to find a suitable index structure, which can make the balance between spatial and temporal, becomes a big challenge for query processing in spatial data warehouse. This challenge is used in calculating spatio-temporal aggregation and query efficiently. For the problems of historical data retrieval, moving object presentation and the future position's estimation, the research is based on modeling, indexing and the querying.

There are many proposed methods in the previous spatio-temporal research, such as Q+R-tree, IMORS, which combine R-tree and other index structure[2,3]. These methods just concern the spatial relationship, but the temporal relationship is ignored. So aRB-tree is presented[4]. It combines R-tree and B-tree, meanwhile, R-tree presents spatial and B-tree presents temporal[5,6]. However, it has a limitation according to the characteristic of moving object. That is, moving object often moves along as the trajectory. R-tree performs well only about a certain region, but not the trajectory. Other researches focused on aggregation and comparison of trajectories. They divide the spatial region into units which records how many times the trajectories passed in unit time[7]. Spatial unit aggregation makes some statistics for aggregation which uses the grid. It not only makes the storage space quite ineffective, but also can not maintain spatial data property. However, in the case of the long life span objects queries, it can also generate a lot of dead space[8].

An index structure, toward above problems, is proposed according to the characteristics of spatial data warehouse and traffic analysis. Like FNR-tree, this structure is a kind of spatial index based on the network instead of the region[9]. Firstly, the network is divided into segments on the point of intersection. After each partition, it is connected by cross points. The network index is constructed with defined arrangement. The aggregation value is stored with spatial object entry.

And then the spatial index structure is constructed. There are two kinds of node in spatial parts (i.e. leaf node and non leaf node). The leaf node contains the aggregation values of moving object's trajectory and the pointers to the extended B-tree. An extended B-tree node contains the information of timestamp and the aggregation values of trajectories with two directions. And the non leaf node contains the MBR (Minimum Bounding Rectangle), MSAV (Max Segment Aggregation Value) and its segment ID.

Trajectories counter aggregation, timestamp, route sample are proposed toward for each segment describing the trace of moving object during a period. The segment ID is the identification about the route for the description of the trajectory. Extended B-tree is established with a timestamp according to the specific movement trajectory.

Query search work can be operated using this index. The aggregation at a certain location in the network can be easily found and the aggregation or comparison about the similar object moving track in the segment can be managed. Better granularity is available according to the timestamp and spatial index. And better performance can be achieved for storage management.

The spatial data is located based on the trajectory segment index for new data insertion. Data is inserted with appropriate granularity. The suitable location will be chosen and an extended B-tree to fit the right timestamp is updated. If the deletion of the data for a moving object is necessary, the

segment where the object locates should be found first. Thus the timestamp will be able to get by using the extended B-tree. And then it can be deleted or updated.

The proposed technique overcomes previous problems efficiently and makes it practicable finding moving object trajectory in the time interval. It improves the shortcoming of R-tree in constraint network, and gets the improvement of spatio-temporal data in query processing and storage.

In the remainder of this paper is organized as follows. We begin with a brief introduction of the related works in section 2. In section 3, aCN-RB Tree construction is presented. And performance evaluation is explained in following section. Our conclusion and future work are given at last.

2. RELATED WORKS

In previous work, R-tree based index, with adding the timestamp, was proposed. This kind of structure just overcomes the object's spatial relationship. But it is difficult to describe region aggregation, so Q+Rtree is proposed[2]. This kind of tree combines the quad tree and R-tree structure. It connects the aggregation values and spatial data, but ignores the temporal aspect. You can store the aggregation values in quad tree. But if there are so many aggregation values in different times, it needs build many quad trees for every timestamp.

aRB-tree is presented[4]. It combines the R-tree and B-tree, the timestamp and the aggregation are B tree's nodes[5,6]. This work's motivation requires querying summarized spatio-temporal data, not the exact id of every data. For example, we can query the quantity of visitors, and needn't know other characters, such as name or age. These studies can find how many moving objects in a unique time or an interval time on some exact ranges. It is a mission impossible if we want to get data with more accurate granularity in data warehouse. This tree is for the range query, if the query object moves in the network, it can't get a high efficiency because of drawback of R-tree. And the other drawback is just describing the sum count aggregation, but it can't present the direction which is an important attributes of moving objects.

The R-tree based index shows its limitation according to the characteristics of the moving object. That is, the R-tree index particularly shows the characteristics of a certain region well, but not for the network. If this region is tilted, the MBR (minimum bounding rectangle) cannot describe its characteristics very well. As a result the MON-tree has to be introduced[10]. MON-tree divides the spatial network into several sections by using the spatial identities of the network, stores these spatial sections by a hash-function and connects the sections with R-tree structure. In each section, the motion of every object will be stored though the limitation of R-tree also exists for this situation. And it will be quite bothered for searching time due to there is no orderliness for the stored time. It is difficult to deal with the time query.

In IMORS, the neglect about time also exists[3]. The network is divided into several segments, and the R-tree is used for road sector. Each road sector entry on nodes points to the information of the moving objects. The flaw of this structure is that the position of the moving object can only be recorded at that split second without taking the time into account. Thus it disagrees with the spatial data warehouse.

The aggregation of trajectories of the similar moving object happened is also illustrated in[7]. The map is mainly divided into spatial units. These units are defined as "minimum spatial extent (cell size) of interest in aggregating the trajectories" to record how many times the trace passed in unit during unit time. The disadvantage of this method is that many empty space have to be established to store data and the spatial relationship integrality is destroyed.

3DR-tree simply add the time as the another dimension to 2D-Rtree and transform it to 3DR-tree[2,11,12]. We can query the interval efficiently with 3DR-tree. But if we want to query the long life span objects, it will generate a lot of "dead" space and make the timestamp query inefficient.

3. aCN-RB TREE CONSTRUCTION

The index structure proposed in this paper issues the trajectories' aggregations of objects moving along the routes and networks in the time interval. It consists of the three blocks: network, trajectory's routes and timestamp.

For the network part, it makes us aware of finding an index that we can search the router aggregation recording on spatial data. For example, which road is the busiest now or in the 2005?

The spatial aggregation of the moving object is a trajectory in a certain time. the previous works divides the map into the spatial units equivalently, and then summarizes the count of trajectories passing by every spatial unit. This structure can't adapt trajectory and destroys the poly line's relationship, so we make some improvements. We scan some roads and separate each road into segments when meet the point of intersection. At last, we make some statistics about the count of every trajectory in current.

For the routes, as Fig. 1.shows, the direction exists in the object's movement. So the aggregation is not unique. There are two kinds of aggregations. The reason is that there are usually two directions in a closed road in accordance with the traffic rule.

In this section we present the index structure to retrieve and store aggregations efficiently.

3.1 The aCN-RB-Tree Structure

The network is divided into several segments (or edges) according on roads' intersecting points. Fig. 2. (a) is an example of the network which includes 4 roads <R1, R2, R3, R4>. Fig.2. (b) is the situation that the roads are divided into many segments.

The aCN-RB-tree we propose includes the main and sub index. The main part is the index for network, which uses the fixed network R-tree. And the sub index is the extended B-tree for timestamp.

Fig. 3. is the figure separated from the integer network map. We use the aCN-RB-Tree construct

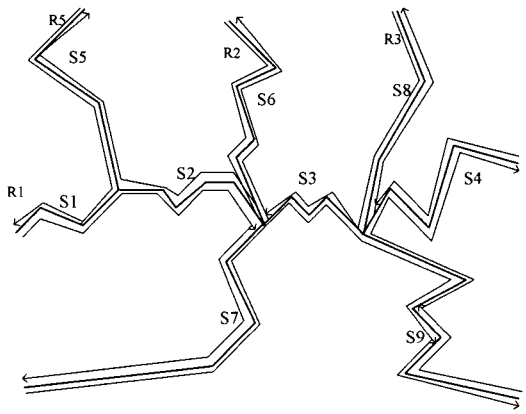


Fig. 1. Routes of Network.

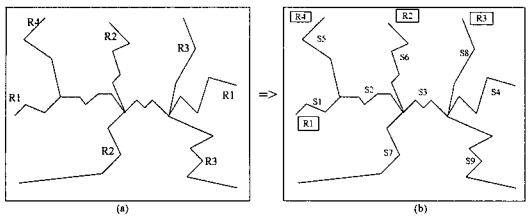


Fig. 2. Roads' Separating

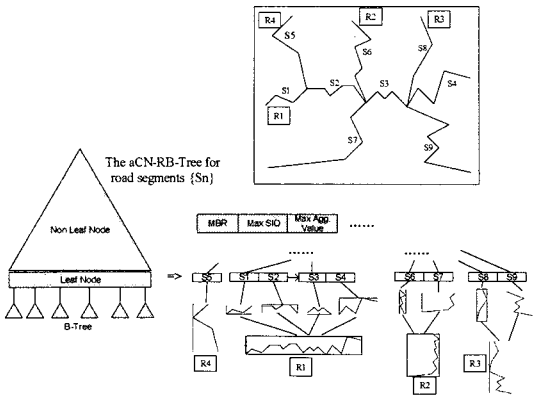


Fig. 3. Structure of aCN-RB-Tree.

the index about these roads. At first, we separate the network into unit segments. And then, we assume the node size can store two segments at most. We use the pointer to connect with next segment till combine a integrate roads. The aCN-RB-Tree also consists of two parts: leaf node and non-leaf node. Each leaf nodes of the aCN-RB-Tree for road segments include the identification of the segment, the aggregation values and the pointer to the extended B-tree.

As Fig. 4. showing, we present as <Segment ID, Aggregation Value, Pointer to extended B-tree>, The aggregation value is the number of the trajectories at the latest time, it is the sum aggregation of two directions . The leaf node is used as follows: we can query the aggregation value in the certain area from the nodes. The segment unit Sn connects each other with pointer. The non leaf nods is shown the form as<MBR, Max.SID, MaxAgg.Value>. We make the MBR corresponding to the main liner in order to keep the data's integrity. This situation happens in the beginning of building the index. If the trajectories appear continually between different roads, we can make some fixes. The MaxSID means the segment ID which has the biggest count of the trajectory passing this segment, and the MaxAgg.Value is the average value of that segment.

Because the time is unique in a road segment, we can make the extended B-tree index for timestamp. The extended B-tree's entries form is presented as <TID, L.agg[], R.agg[]>(Fig. 5.). L and R aggregations mean the aggregation values

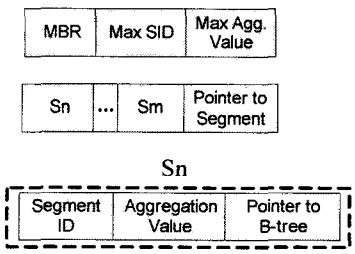


Fig. 4. Segments' Node Structure.

with two directions. In the leaf node of segment, the pointer link to the extended B-tree. We can query the some aggregations in the time interval using the extended B-tree.

Fig. 6. is an example of the extended B-tree. There are 3 kinds of value in each node: timestamp, two aggregation values with two directions. In the middle level, the node can store two timestamps and three aggregation groups at most; the left aggregation group is the average value which the timestamp is earlier than next timestamp. For example, the aggregation value before T3 is the average values of T1 and T2. The middle value means the average between the timestamps. For instance, the value after T3 means the average of the timestamp between T2 and T5, viz. T3 and T4. And the right value is the average with the later timestamp, such as T5 and T6.

Multi-dimension is one of the most popular data models in the data warehouse. Measure is the chief basis for decision analysis in this model. In the spatial-temporal data warehouse, measure will be decided with two dimensions which are the region and the time. Fig. 7. shows the number of tracks in region <S1,S2,···.S9> versus the time of <T1,T2, ··· T5>. It is the L-aggregation and R-aggregation value of segments from T1 to T5. And these groupings assume that the aggregation function is average.

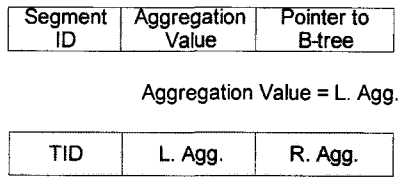


Fig. 5. Segment leaf and Extended B-tree Node Structure.

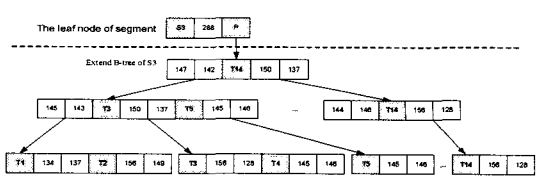


Fig. 6. Example of Extended B-tree.

L-Aggregate results over timestamps

	T1	T2	T3	T4	T5	Average aggregate over segments
S1	21	21	45	33	33	31
S2	143	133	145	122	121	133
S3	134	156	156	154	145	149
S4	56	43	53	45	54	50
S5	55	55	55	55	55	55
S6	32	34	53	24	24	34
S7	31	45	45	45	34	40
S8	45	34	32	32	32	35
S9	22	22	22	54	34	31

R-Aggregate results over timestamps

	T1	T2	T3	T4	T5	Average aggregate over segments
S1	58	61	55	45	45	53
S2	156	145	145	171	168	157
S3	137	149	128	130	146	138
S4	57	49	51	53	57	54
S5	63	63	62	67	67	64
S6	43	41	41	41	42	42
S7	54	54	52	52	56	54
S8	45	57	56	56	51	53
S9	24	24	21	45	31	29

Fig. 7. Average Aggregate over Segments and Timestamps.

Now, we elicit the aCN-RB-Tree to establish index structure. The aggregate aCN-RB-Tree is based on following concept: The polylines of network have the hierarchy characterize in spatial relationship. So we use MBR to connect every polyline in order to query with windows easily. And we depict the extended B-tree to index the temporal aggregation. In particular, each network entry has the form <Segment ID, Lagg[], Ragg[],Timestamp>. We use segment MBR to fix the position, and find the segment ID which accord with MBR. And insert Timestamp and aggregation into extended B-tree. The average aggregation values also update. We make the sum of L.agg. and R.agg., and update the segment block. If the aggregation value is not changed, then, the node will be not updated.

Fig. 8. show the aCN-RB-Tree index about the network of Fig.7. We can see there are the extended B-trees for timestamp in the bottom and the fixed network R-tree in the top. We can query the spatio-temporal information efficiently depending on it. There are extended B-trees in the bottom level, and the middle level contain the segment ID and average aggregation value from B-trees, meanwhile, this value is the sum of L aggregation and R aggregation, the P is the pointer to connect B-tree, every node can store two segment information at most. The top level merges every two node's aggregation of middle level. It contains the MBR of those segments and the segment ID and aggregation, which the aggregation value is bigger. For example, in the S1 and S2, S2's value 290, is bigger than S1's 84. So we choose S2 and its aggregation. Using the method we can

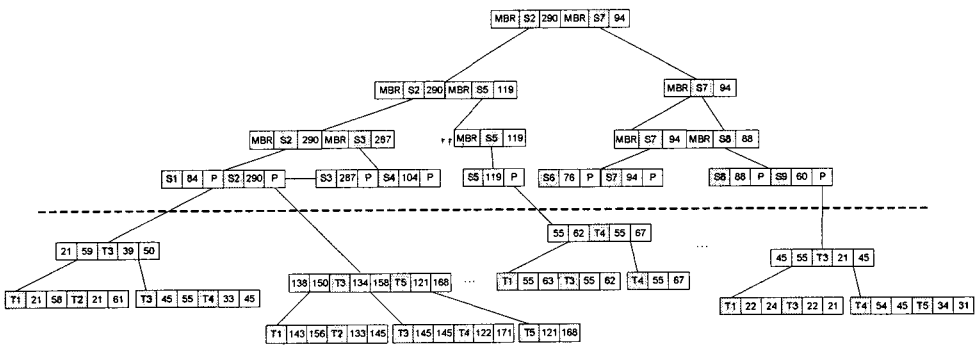


Fig. 8. Example of aCN-RB-Tree.

query which segments is most busy over appointed region.

Next we will introduce the operation of aCN-RB-tree in the following section.

3.2 Insert and Search Processing

The operation of aCN-RB-tree includes two parts: data insert and query processing. The query processing is the operation function, and the data insert is the basic element of operations. Query search is for the clients, and the data insert is for the data warehouse administrators. We will introduce the details in the following sections.

3.2.1 Insert Algorithm for aCN-RB-Tree

The data insert is the most important and basic operation in data warehouse, it is the key to construct the data warehouse. We will introduce how to inset the new value.

The data insert means the every road segment aggregation changes as the time passes. So it is mostly related to the timestamp and aggregation value. The data series are inserted in data warehouse. The data series include many data blocks including segment ID, aggregation value and the timestamp, we just study one data block insert because the others are same. At first, we search whether the segment ID is in aCN-RB-tree, if aCN-RB-tree includes this segment ID, we search the timestamp in the extended B-tree depending on the pointer to extended B-tree and insert some new value in the appropriate position. We will make the average calculation between the new value and the value of latest update until the extended B-tree's root. The problem we have to consider is if the query window include different directions, how to solve the aggregations directions. For example, the collision exists between the horizontal and the vertical directions. We can't define the direction, for this situation, we make some average aggregation denote and get the sum of L and R aggregations. Put the sum into the segment node. The Max aggregation needs re-comparing until build the new balance. The storage space changing happens the extended B-tree, and the another nodes value just changes, needn't change the storage space. Fig. 9.(a) shows the flowchart about Insert algorithm of aCN-RB-tree. The data insert algorithm is shown in Fig. 9.(b).

3.2.2 Search Algorithm for aCN-RB-Tree

A spatio-temporal aggregation query is: give an arbitrary spatial road, compute the average car's flux in this polyline in 2006.

The aCN-RB-Tree processes WIA (window-interval aggregate) queries, by sizing-up the visit nodes which reach most closely.

Take this as an example, suppose that a user is finding all objects with some direction in some network overlapping the query window qw of Fig. 10. during the time interval[T2,T4]. The S5 and S1 are totally in the shaded window, and then we will visit the corresponding extended B-tree to retrieve the temporal information. And there are some parts of S2 also in the query window, because the aggregate function is average, S2 is also available. We can query the information depend on Fig. 11. There are 2 kinds of query in aCN-RB-Tree, the one is region query and the other is point query. For region query, find the window query's MBR in R-tree. We can get the corresponding segment ID and Max aggregation. If query doesn't need direction information, we just find the aggregation in the segment node, needn't query extended B-tree. Else, we depend on TID to find aggregation with directions from extended B-tree. For point query, we need to search the point is in which segment, and search the information about this segment. The method is same as region query. Fig.11.(a).shows Flow chart about search algorithm of aCN-RB-Tree, and Fig.11.(b) expresses search algorithm.

The aCN-RB-Tree operations data insert and

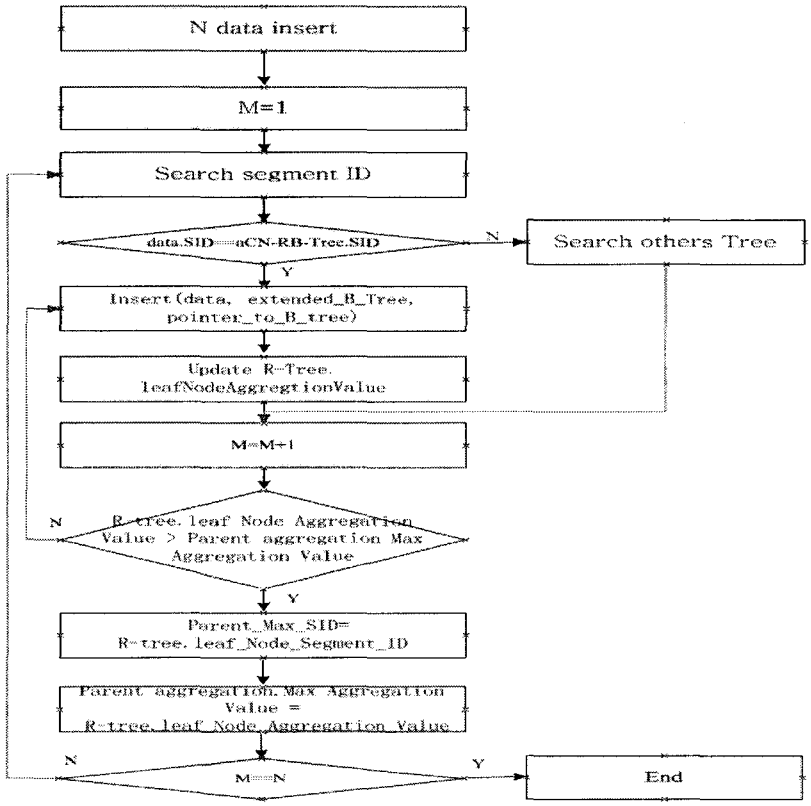


Fig. 9.(a).Flowchart about Insert algorithm of aCN-RB-tree.

```
Insert algorithm of aCN-RB-tree
N data insert into data warehouse;
1  Search segment ID; to insert data
2  if (data.segmentID==aCN-RB-tree.segmentID)
   // if aCN-RB-tree include input data' segment
        Insert(data, extended_B_Tree, pointer_to_B_tree);
         //insert data to extended-B-tree depending on pointer to B-tree.
3  if (insert(data, extended_B_Tree, pointer_to_B_tree)==true)
     //if there is the data inserted in extended B-tree
      R-tree.LeafNodeFlag=true;
      Update R-tree.leafNodeAggregtionValue;
      // the R-tree's leaf node flag= true, else is false;
   else  R-tree.LeafNodeFlag=false;
4  repeat 1,2,3, until N data input;
5  if R-tree.LeafNodeFlag=true;
     Update R-tree.leafNodeAggregtionValue;
6  if (R-tree.leaf_Node_Aggregation_Value>Parent_aggregation.Max_Aggregation_Value)
   Parent_Max_SID=R-tree.leaf_Node_Segment_ID;
   Parent_aggregation.Max_Aggregation_Value= R-tree.leaf_Node_Aggregation_Value;
7  while(
   R-tree.leaf_Node_Aggregation_Value>Parent_aggregation.Max_Aggregation_Value)
   repeat 6;
```

Fig. 9.(b). Insert Algorithm of aCN-RB-Tree.

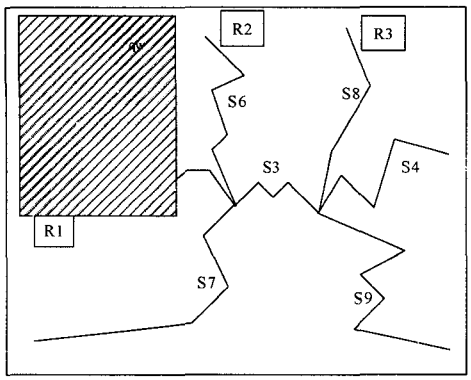


Fig. 10. Example of Window Query.

query processing are introduced. Because of using the segment unit, the can-RB-Tree avoids the dead space, and it make query efficient. And the storage space just increases in the extended B-tree, and the others part keeps the balance of storage, so it can't be affected by storage problem. In conclusion, aCN-RB-Tree is a good index structure. We will make the performance evaluation in the next section.

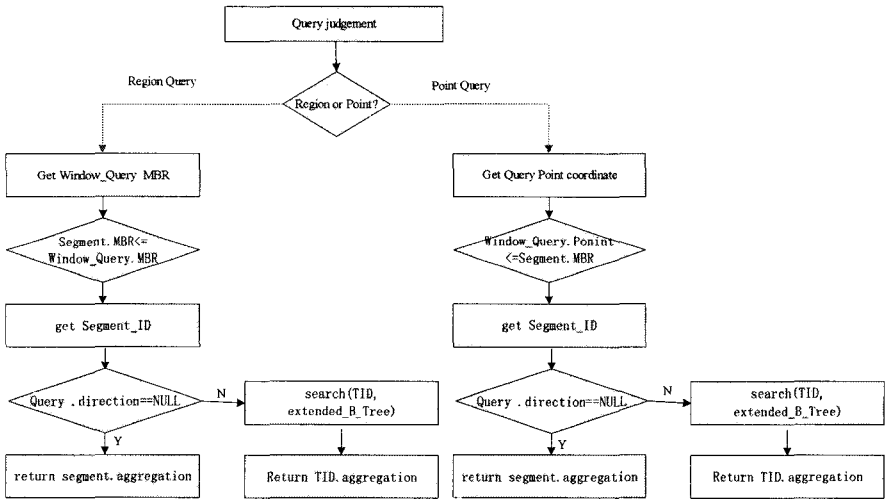


Fig.11. (a). Flowchart about Search Algorithm of aCN-RB-Tree.

```
Search algorithm of can-RB-Tree
1  region query
A    Search (MBR,R-tree);
// find corresponding MBR in R-tree, we can get Max aggregation and Segment ID
B    if (Segment.MBR<=Window_Query.MBR)
            get Segment_ID ;
C    if (Query .direction==NULL)// Query needn't direction information
            return segment.aggregation;
D    else
                    search(TID, extended_B_Tree);
       // else query need the direction information,search TID from extended B-tree
            Return TID.aggregation;
2   point query
A   if (point ==Segment.point)// if the point in the segment
       Search Segment_ID;    //we search this segment ID
C    if (Query .direction==NULL)// Query needn't direction information
            return segment.aggregation;
D    else
                    search(TID, extended_B_Tree);
// else query need the direction information, search TID from extended B-tree
```

Fig.11. (b).Search Algorithm of aCN-RB-Tree.

4. PERFORMANCE EVALUATION

We make the experimental evaluation for determine in which conditions the aCN-RB-tree is efficient. So as to have a direct comparison with another spatiotemporal access method, we have chosen the 3D R-tree. And aCN-RB-tree is designed to query "find the number of cars within a given area during a giving time interval". Performance of various methods is measured by the number of nodes accessed during the processing of workload, each consisting of 500 queries. Every query include two parameters that affect performance: the spatial query window(qs), denotes the percentage of its length over the whole network, and interval length(qt).

In order to compare rationally, we use an ASCII file with the network of Oldenbourg and a file with moving objects. We open the network file firstly, divide it into segments and construct the spatial tree. Then we open the moving objects' file, add the information to the spatial tree, and construct the extended B-tree. Here the page for the leaf and non-leaf node is used with the size of 4096, which can save 200 leaf nodes and non-leaf nodes. 100 roads are concluded in every dataset and are divided into 420 segments by the point of intersection with 1000 timestamps.

234kb spatial data will be used in leaf node, which contain 7035 segments. Each segment has a sign and a MBB value. In the non-lefenod, we have 558kb non-spatial data including the two-direction aggregation value and the volume of timestamp. There are 20275 tuples in total and the timestamp ranges from 0 to 50. We could get the result from three methods of aCN-RB-tree, 3D-Rtree and FNR-tree. (Table 1.)

In the comparison with a3DR-tree, the timestamp increases in percentage with the primary parameters of the memory capacity and the frequency that the node has been accessed.

For example, the aggregate data of 1000 uniformly distributed regions collected over 100 timestamps. At each timestamp, the aggregate data of AA of the regions is modified, where AA is a dataset parameter called the aggregate agility (we will denote AA in following session). AA=5% means the 50 regions update their aggregate data per timestamp.

To the first test, the timestamps we renovated were 15%, 30%, 45%, 60%, 75%, respectively. The variations of the generation of tree size and access times can be monitored. Fig. 13 shows that the relationship between storage and AA.

The variations of 3 trees, FNR-tree, a3DR-tree and aCN-RB-tree, are compared in Fig.12. When AA is relatively low, the storage spaces of 3 trees overhead are quite similar. As the AA increases, the a3DR-tree holds the largest size which followed by the FNR-tree. And the aCN-RB-tree takes up the smallest. That is because in the timestamp, aCN-RB-tree used an extended B-tree thereby sharply minimized the number of the temporal node.

In Fig. 13, the average number of node accesses of query processing with the different values of AA is presented. The condition is qs=5%, qt=50. The node access of aCN-RB-tree is lower than that of

Table 1. Comparison of Data Set

	Number of entries	Number of nodes	Space utilization (%)	Spatial extend		Temporal extent
				x	Y	
aCN-RB-tree	10354	503	50.78	281-23854	3935-30851	0-100
3D R-tree	10354	463	64.89	1821-20671	4005-30851	0-100
FNR-tree	10354	550	55.45	281-23854	3935-30851	0-100

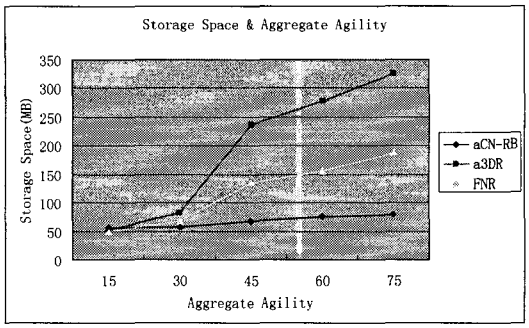


Fig. 12. Storage Space and Aggregate Agility.

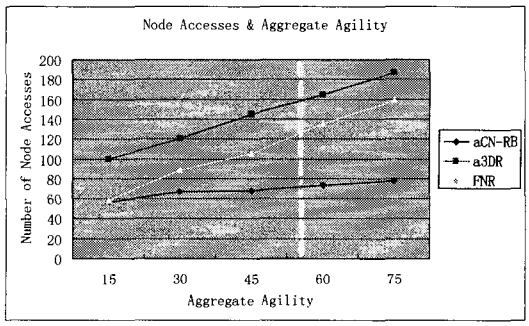


Fig. 13. Node Accesses and Aggregate Agility.

the a3DRtree due to the FNR-tree aims at the router, thus the accuracy is higher than the a3DRtree based on the region. The adoption of extended B-tree also minimized the visiting times.

The reason we omitting the aRB-tree is that the aRB-tree is corresponding with the region. The aRB-tree and aCN-RB-tree are basically same for the node and performance only except the aCN-RB-tree is corresponding with the network. It possesses a direction in each segment which remains with non-leaf node, therefore it cannot be considered with a quantized comparison.

The last one we will make the performance experiment about range queries with a certain temporal extent. We make the timestamp at 1, 50, 100 respectively, and the performances will be analyzed in the query extent and length as follow. First we fix the spatial value in the query window. Then adjusted qs=1%, 5%, 9% and investigate the value changes of the node access with the timestamp range of 0 to 100.(Fig.14)

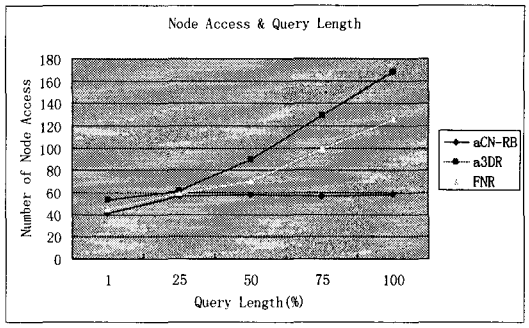


Fig. 14. Node Access and Query Length.

From the node access and query length figure, we can see the node access increases as the query length increases in a3DR, and the query length can not affect the node access. The reason is that a3DR relates the coordinate(x,y,t) and the aCN-RB just relates the MBR and the whole segment. FNR ignores the times interval, so it can't efficient as B-tree in time query aspect.

From the node access and query extent Fig.15, we can see the difference reduces if the query extent increases. Because the number of extended B-tree node is needed to increase as the window increases. The node access of FNR is similar with aCN-RB, but the time nodes also increase more than aCN-RB during the query extent increase.

In summary, aCN-RB tree shows the good performance in range query, for temporal and spatial. Because of the data structure, it is unnecessary to access so many nodes to find the exact value. During the change of time, it will reach veracious value when the data granularity became rough, and

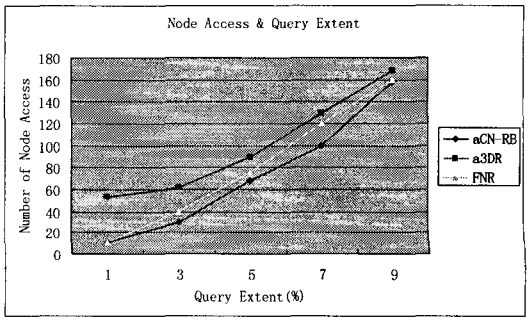


Fig. 15. Node Access and Query Extent.

which is very important for data warehouse. In other words, unlike a3RDtree, aCN-RB-tree is an on-line structure. The aCN-RB-tree increases the overhead of index structure and adds the complexity of spatial-temporal data.

5. CONCLUSIONS

It is critical for applications to access spatio-temporal data. Data warehouse can solve the problem of relational database and non spatial database. In the spatio-temproal aspects, it is challenged to store and get the exact result with the spatial relationship efficiently. In this paper, we proposed aCN-RB tree index, which is used in checking the general instance of trajectory network, in order to get the better decision support. This tree uses the FNR-tree to describe the information of trajectory network. Extended B-tree is the timestamp index, in every node, also includes the aggregation with two directions. aCN-RB tree not only can require the result immediately, but also can control the granularity change in the data warehouse during the time interval. It decreases the query time with little consumption of storage space. Because it is an on-line structure, it could support the applications to the PDA and mobile industry. In summary, it is an efficient indexing structure.

6. REFERENCE

[1] B. M. I. Lopez and R. Snodgrass. "Spatiotemporal aggregate computation: A survey," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2005.

[2] Y. Xia and S. Prabhakar, "Q+Rtree: Efficient Indexing for Moving Object Databases," *The 8th International Conference on Database Systems for Advanced Applications (DASFAA)* Kyoto, Japan, 2003.

[3] K. S Kim, S. Kim, T. Kim, and K. Li, "Fast indexing and updating. method for moving objects on road networks," *Proc.of the Fourth International Conference on Web Information Systems. Engineering Workshops (WISEW'03).*

[4] D. Papadias, Y. Tao, P. Kalnis, and J. Zhang, "Indexing Spatio-Temporal Data Warehouses," *Proc. of International Conference on Data Engineering (ICDE)*, 2002.

[5] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. "The R*-tree: An efficient and robust access method for points and rectangles," *In Special Interest Group On Management Of Data (SIGMOD)*, pp.322-331, 1990.

[6] R. Bayer, "Binary B-Trees for Virtual Memory," *ACM-SIGFIDET Workshop* 1971,

[7] N. Meratnia, "Aggregation and Comparison of Trajectories," *Proc. of the 10th ACM International Symposium on Advances in Geographic Information Systems ACM-GIS*, McLean, pp.6-9, 2002.

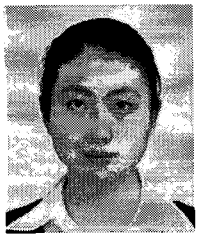
[8] D. Pfoser. "Indexing the Trajectories of Moving Objects," *IEEE Data Engineering Bulletin.* Vol. 25, No. 2, pp. 2-9, 2002.

[9] E. Frentzos. "Indexing objects moving on fixed networks," *Proc.8th Int'l Symposium on Spatial and Temporal Databases.* Berlin: Springer, pp. 289, 2003.

[10] V. Teixeira and R. Hartmut, "Indexing the Trajectories of Moving Objects in Networks (Extended Abstract)," *16th International Conference on Scientific and Statistical Database Management*, 2004.

[11] D. Pfoser "Novel approaches to the indexing of moving object trajectories," *Proc.26th International Conference, Very Large Databases.* San Francisco: Morgan Kaufmann, 2000.

[12] Y. Theodoridis, M. Vazirgiannis, and T. K. Sellis, "Spatio-Temporal Indexing for Large Multimedia Applications," *In Proc. IEEE International Conference on Multi-media Computing and Systems*, 1996.

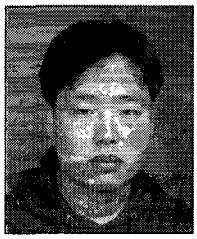


Li JingJing

2004 Department of Computer Science & Engineering, Chongqing University of Posts and Telecommunications (B.S.)

2004~Present Department of Computer Science & Engineering. Inha University (M.S.)

Interesting areas : Spatial Database Warehouse, GIS, Spatial Information Management, Ubiquitous related with SDBMS.



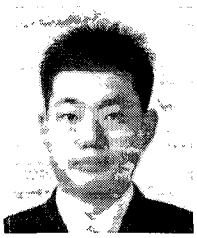
Dong-Wook Lee

2003 Department of Computer Science & Engineering, Sangji University (B.S.)

2005 Department of Computer Science & Engineering, Inha University (M.S.)

2005~Present Department of Computer Science & Engineering. Inha University (PH.D. Course)

Interesting areas : Spatial Database, Spatial data Warehouse, Spatial Information Management, Ubiquitous related with SDBMS.



Byeong-Seob You

2002 Department of Computer Science & Engineering, Inha University (B.S.)

2004 Department of Computer Science & Engineering, Inha University (M.S.)

2004~Present Department of Computer Science & Engineering, Inha University (PH.D. Course)

Interesting areas : Spatial database, Spatial Data Warehouse, Data Stream, Ubiquitous Computing.

Young-Hwan Oh

1993 Department of Computer Science & Engineering, Inha University (B.S.)

1997 Department of Computer Science & Engineering, Inha University (M.S.)

2001 Department of Computer Science & Engineering, Inha University (PH.D.)

1999~2001, Assistant Manager of KGI system develop department.

2002~Present, Assistant Professor of Korea Nazarene University.

Interesting areas : Spatial Database, Database Security, GIS.

Hae-Young Bae

1974 Department of Applied Physics, Inha University (B.S.)

1978 Department of Computer Science & Engineering, Yonsei University (M.S.)

1989 Department of Computer Science & Engineering, Soongsil University (PH.D.)

1985 Visiting Professor of University of Houston

1992~1994 Manager of Electronic Computing Lab of Inha University

1982~Present, Professor of Department of Computer Science & Engineering, Inha University

1999~Present, Manager of Intelligent GIS Research Center

2000~Present, Honorary Professor of Chongqing University of Posts and Telecommunications, China.

2004~2006, President of Department of information and telecommunication, Inha University

2006~Present, President of Graduate School, Inha University

Interesting areas : Spatial database, Distributed Database, GIS, Multimedia database.