International Journal of Database Theory and Application
Vol.8, No.2 (2015), pp.81-94
http://dx.doi.org/10.14257/ijdta.2015.8.2.09

Spatial Approximate Keyword Query Processing in Cloud Computing System

Zuping Liu

Sichuan Vocational and Technical College,
Suining 629000, china
641433894@qq.com

Abstract

The paper proposes spatial approximate keyword query algorithms for cloud systems. Existing work targets on single server solutions, and an exact algorithm is given in memory while another approximate algorithm is given for disk resident datasets. However, a single server fails to provide reasonable throughput due to the limited CPU time and disk bandwidth. Facing the above challenges, this paper gives a two-layered index consisting of global index and local index, which works in a shared nothing cluster for larger query throughput. This paper designs a novel external memory index as local index, which returns exact answer within disks efficiently. It is equipped with keyword set signature and multiple optimizing strategies to reduce I/O cost. The global index partitions the entire spatial space, and each computing node in system maintains a partition. A global index selection algorithm is given. This paper also provides spatial approximate keyword query algorithms based edit distance, including range and the nearest neighbor spatial conditions. An experiment in a shared nothing cluster illustrates the efficiency and effectiveness of our proposed index and query algorithms.

Keywords: *Spatial Keyword Query, Cloud computing, Query processing, Aggregation*

1. Introduction

At present, more and more sites begin to support spatial Keyword query, the query includes a space conditions and a group of key words as text. Space key words query $Q(R, k_1, k_2, ..., k_m)$ in spatial database returns to meet the condition of space R, and contains the given keyword object $k_1, k_2, ..., k_m$ in query. Spatial Keyword query is widely concerned by researchers, and are studied, the existing various index structure used to support the query. However, spatial Keyword query requires the user to use the exact keyword query, if the user is using keywords and keyword objects cannot exact match in database, the object will be filtered out, this restriction reduces the availability of query. In the current application, database may contain error text input; the user can contain error keywords when a user submits a query. These errors will be more prevalent at the scale increased of data set [1-2].

This paper designed the RBF index, the two layer of the index structure composed of a local index and the global index using cloud computing system, support space approximate keyword search in a shared nothing cluster environment, improve the system throughput of the corresponding query. In this paper, the design of a novel R-tree equipped with Bitmaps, support space approximate keyword query in storage, and effective return to complete the query results. The RB tree by adding two kinds of bitmap in the R tree, RB tree node is used to judge whether the given keyword may contain similar enough keywords. Two kinds of bitmap RB tree node were length of fixed length and Gram bitmap. Among them, the length of RB bitmap for instructions indexes in tree node key length, make a quick first step pruning during query processing [3-4].

Gram bitmap is used to judge whether a RB tree node indexing keywords contained in a given gram, in order to determine whether the node indexing and keyword targeted enough similar keywords [5-6]. In order to further improve the efficiency of the query, add contains the string length of q-gram on the structure of RB tree, and design a more accurate pruning strategy [7]. The RBF index using RB tree as a local index in the system, each computing node contains a RB tree, named the index for the RB Forest Index. RBF index of the whole space is divided into a plurality of divided; each computing node maintains a space partition [8-9]. Based on the division of space, the global index RBF index will choose on the local index maintenance computing nodes in each memory, used to speed up query processing. This paper presents a global index RBF index selection method, is used to initialize and periodic global index maintenance [10-11]. In the aspect of query processing, this paper presents the use of RBF index processing range edit distance based on approximate keyword query and nearest neighbor approximation method of keyword query. The RBF index to support a variety of space conditions (such as KNN, Skyline etc.) and keyword similarity measure (such as Dice, Cosine), specific treatment can be obtained by the method of extension is given in this paper [12-13].

2. System Framework

This paper introduces the system framework of RBF index, the index and the global index structure and system including the RBF index of the work flow. RBF index in shared nothing cluster of cloud computing systems, and the use of cloud computing in the system composed of a local index and global index two index structure. Figure 1 describes a RBF index of cloud computing system, the system is composed of 4 computing nodes.

Space data set D in system is partitioned into $4 \times k$ blocks, then the blocks of data are distributed to every system of computing nodes, each computing node receives the k data block. Then, computing node N_i stores the data block of allocation in disk as the local data D_i, where $i = 1, 2, 3, 4$

The local module of the computing node N_i included local data D_i in CN_i and local RB tree T_i. The RB tree added into plurality of the bitmap in the R tree node, and can be pruned conditions of space and approximate keyword conditions. CN_i used T_i index local data of RB tree and accelerate the local query processing. It is shown in Figure1, local data D_i of N_i and the local index T_i are stored in the disk. After built the RB tree index T_i, CN_i selected part of RB tree node as a global index from T_i, and will release these global indexes into the system. The global index is the summary information of local data, it achieved the location data and queries in the system.

Using the RBF index to query the required data is divided into two stages. In the first stage, the query in the system is divided between the nodes that satisfy the query to find global index query according to the computational space. Determine the query needs access to what global index corresponding to the data in the calculation node maintains a global index. In the second phase, the query based on *server* and *addr* Field of global index, in order to find the corresponding data owners, and use the data owner's local index query and returns the query result.

3. External Index RB Tree

3.1. The Structure of RB Tree

RB tree in the R tree node added auxiliary data structure, used to judge whether the object index in tree node may contain that satisfy the query keywords, in order to support keyword queries of space approximation. It is different the LBAK tree and MHR tree, RB tree stored two bitmap, it is respectively length Bitmap LB and Gram Bitmap GB, length Bitmap LB and Gram bitmap GB is used to judge whether the key word set may contain similar elements with a given key enough. Due to the addition of the bitmap in the R tree (Bitmap), we named it RB tree.

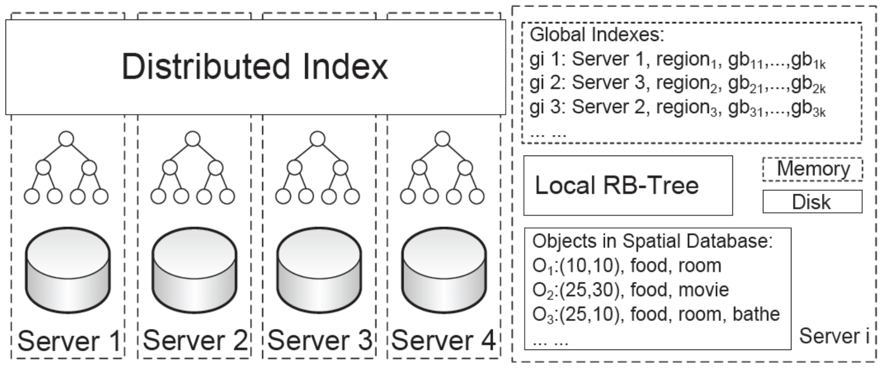
Distributed Index
Server 1
Server 2
Server 3
Server 4
Global Indexes:
gi 1: Server 1, region$_1$, gb$_{11}$,...,gb$_{1k}$
gi 2: Server 3, region$_2$, gb$_{21}$,...,gb$_{2k}$
gi 3: Server 2, region$_3$, gb$_{31}$,...,gb$_{3k}$
... ...
Local RB-Tree
Memory
Disk
Objects in Spatial Database:
O$_1$:(10,10), food, room
O$_2$:(25,30), food, movie
O$_3$:(25,10), food, room, bathe
... ...
Server i

Figure 1. Overview of Two Layer Index

3.2. RB Tree Query Processing

RB tree at the root node to handle algorithm1 to process approximate keyword query Q^{RAK} , algorithm 1 describes the process of node N treatment Q^{RAK} . The result set is initialized to the empty set. If N is leaf node, the algorithm1 inspects the data i in the N , if the data satisfies the query, then put the result set. For each data i examination including 1) coordinate is located in the query area of Q^{RAK} , 2) for each k_j keyword in the query, such that $ED(k_j, k) \le \theta_j$

Algorithm1 RANGEQUERY

```
Input: RB tree node N, Range query $Q^{RAK}(R, k_1, \theta_1, ..., k_m, \theta_m)$
Output: Spatial database object Result
1  Result =$\varnothing$
2 If N is leaf node then
3    For $i \in N.DS$ do
4      If $i$ meets $Q^{RAK}$ query      Result =Result $\cup i$
5 Else
6   For $i \in N.CS$ do
7       $i$ meets $Q^{RAK}$ query
8         Result =Result $\cup RANGEQUERY(i, Q^{RAK})$
9 Returns Result
```

3.3. RBF Index Query Processing

3.3.1. Query Processing of the Range Approximate Keyword

Processing method of the range approximate keyword query is given by algorithm 2. After the system receives the query $Q(R,K)$, the node Q is sent to all regions of space and computing node of $Q.R$ intersection The global index memory of each received Q compute node i , and returns the space region and $Q.R$ intersect and meet the global index

G_i approximate keyword condition K, then computing node i sends Q to each element owner of G_i. Figure 2 shows query Q^{RAK} search space of range approximate keyword. The system consists of 8 computing nodes; space division is shown in figure 2.The system has 7 global indexes A to G, where B, C and G are stored on multiple computing nodes in memory. In Figure 2, computing node 2 is sending data request to B data owners, and computing nodes 4 and 5 for the query QRAK is to ignore B, because of the area 2 included query region.

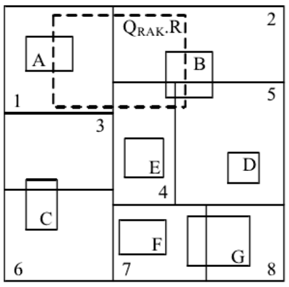

Figure 2. Search Space of RAK Query

Method 2 described to be applied to other distributed system based on space partition, such as CAN, BATON etc. The lookup method for obtaining and query method all node regions is intersected by the distributed system, this method can obtain in CAN and BATON. On the other hand, the regional space in algorithm2 can be rectangle, circular or annular region.

Algorithm2 RBF RANGE QUERY

```
Input: RB tree node N, Range query $Q^{RAK}(R, k_1, \theta_1, ..., k_m, \theta_m)$
Output: Spatial database object Result
1 Result =$\varnothing$
2 $indexOwners$ =lookup ($Q^{RAK}.R$)
3 For $i \in indexOwners$ do
4     Send $Q^{RAK}$ to $i$
5     $i$ used $Q^{RAK}$ to search the global index of memory, obtained set $G_i$
      Send $Q^{RAK}$ to the owner $DataOwner_j$ of $j$
6  $DataOwner_j$ processed $Q^{RAK}$ in the local, obtained the result $R_j$
7 Result =Result $\cup R_j$
8 Returns Result
```

3.3.2. Query Processing of Nearest Neighbor Approximation Keyword

System used the RBF index to perform nearest neighbor approximation keyword queries, and until query target object recent satisfies the approximate keyword conditions were found, in conditions of the process of object search, the query needs to perform some round. Each round of the implementation process is divided into two stages: the index of exploration and data acquisition. In the indexing stage, Q^{NAK} query is sent to the spatial region calculation node N, and appointed N as the query agent. Then the N in memory searched and queried target recently, and meet the global index approximate keyword conditions. If the index does not exist in the N memory, N expand the search radius and the other nodes search until obtaining global index of meeting the conditions. Subsequently, the query got into the data acquisition stage, the system launched a range approximate keyword query Q^R, where, R is an annular region; lower radius of the annular region is the minimum distance of Q^{NAK} query object, while the sharp upper bound is maximum distance of Q^{NAK} query target. It is shown in Figure 3, assumed that all global index satisfied the approximate keyword query conditions, in the current round of indexing stage found index B. in the data acquisition stage, the query region is annular zone of figure3. In Figure 3, a new round index of exploration will obtain D, the upper ring radius of query is Q^{NAK}. The nearest neighbor approximation method of keyword queries can be directly extended to k nearest neighbor approximation keyword processing method. Query processing continues, until the k meet key condition, and the query nearest object is found. Algorithm 3 describes the system to deal with the nearest neighbor approximation process of keyword query

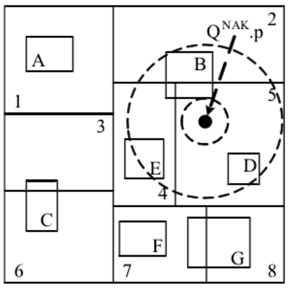

Figure 3. Search Space of NAK Query

Algorithm3 RBF RANGE QUERY

```
Input: The nearest neighbor approximation keyword query $Q^{NAK}(loc, k_1, \theta_1, ..., k_m, \theta_m)$
Output: Object set Result
1 Result =$\varnothing$
2 $d_{\min} = 0$
3 $d_{\max} = 0$
4 While Result $| < 1$ do
5     $G^p = \{gi \mid \bar{d}(gi,Q) > d_{\max}, filter^{MGB}(gi,Q) = true\}$
6     $gi = \arg_{gi \in G^p} \min\{d(gi,Q)\}$
7     $d_{\min} = \underline{d}(gi,Q)$
8     $d_{\max} = \bar{d}(gi,Q)$
9     $G = \{gi \mid d_{\min} < \bar{d}(gi,Q), \underline{d}(gi,Q) < d_{\max}\}$
10    $G^T = \{gi \mid gi \in G, filter^{MGB}(gi,Q) = true\}$
11    For $gi \in G^T$ do
12        $R_{gi} = getData(gi, Q, d_{\min}, d_{\max})$
13        Result $=$ Result $\cup R_{gi}$
14 Returns Result
```

4. Experiment Design and Discussion

4.1. The Experimental Setup

The paper tests query performance of the RBF index, experimental environment is unshared cluster. Each node configuration is as follows: Linux operating system, Xeon CPU, frequency is 2.66GHz, memory is 16GB. The disk is 80GB. The programming language is Java, JDK version is 1.6. The 4.1 section introduces the performance test of the local RB tree, the 4.2 section gives RBF index query throughput testing in the cloud computing system. Table 1 gives the parameter and its value of 4.1 sections and the 4.2 section in testing. Experiment used data Set TX and CA from the open source project Open Street Map.

Table 1. Experiment Parameters

Parameters	Value
Page size	4KB
The type of query	The range of approximate query Nearest neighbor approximation keyword query
Selection of spatial query local index test	1%, 5%, 10%, 15%, 20%
Selection of spatial query global index test	0.01%, 0.05%, 0.1%
Normalized edit distance	0.1, 0.2 ,0.3
The number of tuples	2, 4, 6, 8, 10($\times 10^6$)
The size of the cache query processing local index test	0(MB)
The size of the cache query processing global index test	16(MB)
Whether to use the global index selection algorithm	YES/NO
Calculation of the number of nodes	8, 16, 24, 32

4.2 The Performance of Local RB Tree

This paper used the TX data set CA data set to test RB tree nodes in the query I/O performance of single node. The query range is approximate keyword query, generating process is as follows: randomly selected from the data set of 100 objects, and the coordinates of the object is as the center, according to the size of given query region (denoted as P) to generate a square query area, size is the entire area of 1%, 5%, 10%, 15% and 20%, used keywords of The object as query keywords, each keyword is corresponding to similar to the edit distance threshold, value is 10%, 20% and 30%. In the following test, it compares the RB tree and the existing in the file memory space approximation accurately processing keyword queries the R tree I/O cost, in order to validate to effectiveness of the bitmap information and optimization strategy in the RB tree. In the implementation of the experiment used RB tree. RB-tree showed without the use of additional length q-gram and asymmetric gram model pruning strategy, RB-tree (A) said the implementation of two kinds of optimization strategy. The following tests, Gram Bitmap of RB tree and RB tree (A) is1000, the length of the bitmap length is 24.

It is shown in Figure 4 and Figure 5, we normalized edit distance is fixed at 0.2, and to test different query region size effect on query IO. This paper selected the query region size is the whole data space of 1%, 5%, 10%, and 15% and 20%. Experiment results are shown from TX and CA data set; R tree query cost with IO price query region growth to increase, and RB tree IO cost remains unchanged. Because the R tree is only pruning by spatial information, so IO cost with the query region is growth. The RB tree can effectively use the keyword information pruning, so query IO cost does not growth with the query region.

The experiment results show that the RB tree of the query I/O cost is only8% to 9% of R tree query I/O cost. RB tree (A) I/O cost is 14% to 30% of RB tree I/O cost, and the query I/O cost with the query region size grows more slowly. This showed that the RB tree (A) using additional length of q-gram and a more accurate pruning conditions substantially to improve the query efficiency.

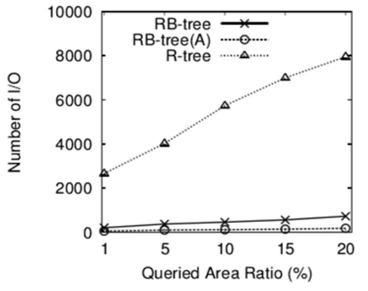

Figure 4. Effect of Spatial Region (CA dataset)

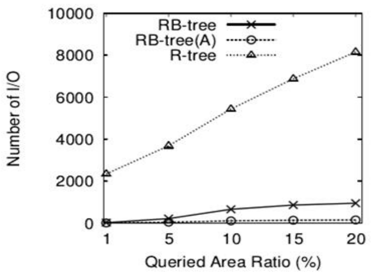

Figure 5. Effect of Spatial Region (TX dataset)

Figure 6 and Figure 7 give the effect of keyword similarity to the query IO cost, it is shown in the Figure 6 and Figure 7, expand the similarity threshold will increase of the query IO cost of the RB tree. R tree IO cost of query does not change with the keyword similarity threshold, because the R tree used only spatial conditions to prune the query, so the edit distance will not increase the I/O cost query of R tree in expanding Standardized query. Compared to the R tree, RB tree in TX and CA data set is reduced by 90% and 85% IO cost query, bitmap information in the RB tree can effectively provide the ability to query pruning. Comparing Figure 6 and Figure 7 in the RB tree and RB tree (A) query performance, RB tree (A) I/O cost is only 16% of the RB trees, which proves that the RB tree used two optimization strategies are valid under different approximate key conditions.

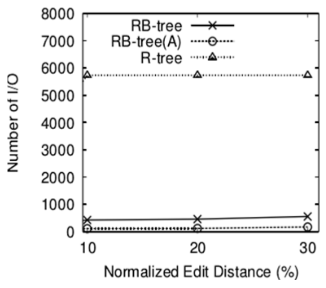

Figure 6. Effect of Skewness (CA dataset)

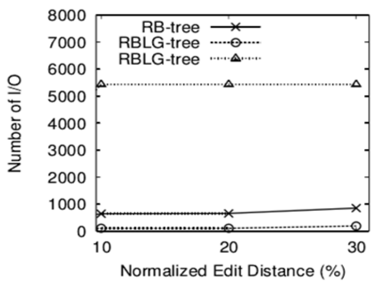

Figure 7. Effect of Skewness (TX dataset)

Figure 8 and figure 9 used TX and CA data set of tuple different number to build two groups containing 2×10^6 to 10×10^6 object RB tree, and test their IO cost query. It is shown in Figure 8 and figure 9, R tree, RB tree and RB tree (A) query IO costs with the amount of data the linear are growing, compared to the R tree, RB tree is reduced by 89% and 92% of the IO cost in TX and CA.

This shows that RB tree query processing used bitmap information has significant pruning ability, but also has good scalability. Compared with the RB tree, RB tree (A) I/O cost is only 16% to 25% of RB tree. Experiment results showed that the local index RB (A) tree of this paper used effectiveness of bitmap information and optimization strategy.

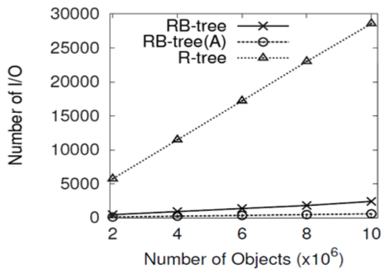

Figure 8. Effect of Data Size (CA dataset)

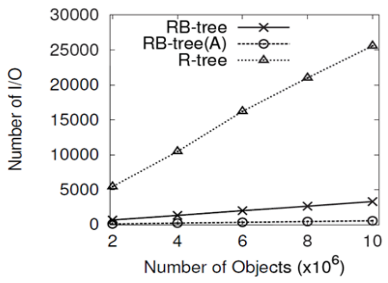

Figure 9. Effect of data size (TX dataset)

4.3. Global Query Performance

4.3.1. Query Performance of Range Approximate Keyword: Figure10 and Figure 11 give the TX dataset and CA dataset query throughput, changes of different query region size case. The query area increased from 0.01% to 0.05% and the whole area 0.1%, the throughput of the system decreases.

Because the query area lead to more greater global index to access, each query carried out the local RB tree search procedure also increased in the treatment process, therefore the throughput of the system decreases. Through the experiment results, the query throughput and calculation of the number of nodes are in proportion, which indicates that the RBF index has good scalability, applied by lots of computing nodes of the cloud computing system.

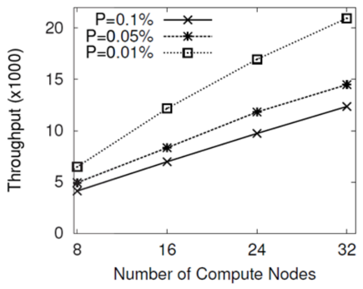

Figure 10. Effect of Spatial Region (CA dataset)

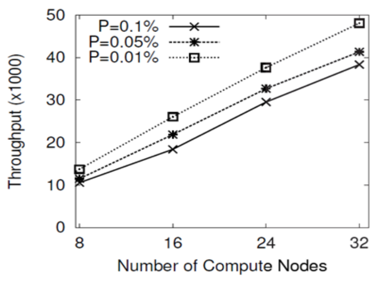

Figure 11. Effect of Spatial Region (TX dataset)

Figure 12 and Figure 13 compare the system throughput of RBF index and the existing cloud computing multidimensional index RT-CAN. In the keyword using the similarity measure normalized edit distance are respectively 10%, 20% and 30%, the query throughput of RBF index in the TX data set and CA data set were 35 times of RT-CAN.

This proves the validity of RBF index of global index and the local index, a global index reduced the number of global index was obtained in the index owner, thereby reducing the number of local search; so the RT-CAN pruning only in space conditions, so the query performance and keyword similarity threshold are independent, and RBF normalized throughput decreases with increasing distance editor.

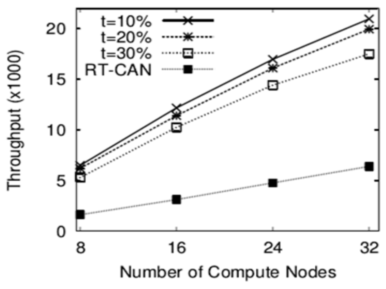

Figure 12. Effect of Normalized Edit Distance (CA dataset)

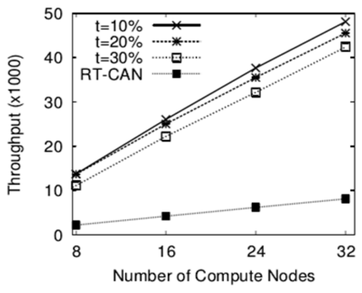

Figure 13. Effect of Normalized Edit Distance (TX dataset)

Figure 14 and figure 15 show selection method of the global index. In the TX data set and CA data set, open the global index selection, system throughput is higher than random selection of a global index throughput. Because the global index selection algorithm based on query distribution mode at present, choose effective global index from the local index. Used the global index to select the global index algorithm provides better pruning ability query nodes. Therefore, query can access less computing node, and have access to less disk pages of the disk in the data, so it has better query throughput system.

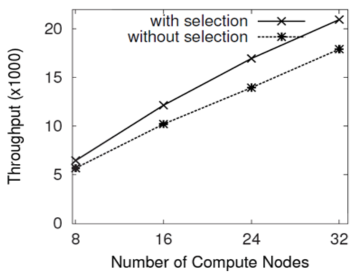

Figure 14. Effect of Global Index Selection (CA dataset)

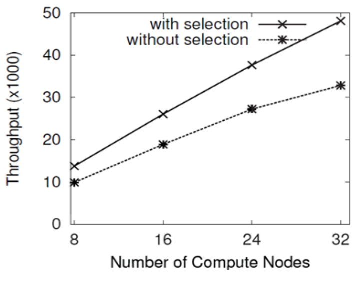

Figure 15. Effect of Global Index selection (TX dataset)

4.3.2. Query Performance of Nearest Neighbor Approximation Keyword: Figure 16 and Figure 17 give performance query of the system processing nearest neighbor similarity query. In the smallest search range query from the whole area increased from 0.01% to 0.05% and 0.1%, query throughput decrease gradually. Due to the minimum area lead to more global index search in the query, but more data owners in local processed range queries, so query throughput of the system with minimal search region decreases. Combined with the range of approximate query test results, in the least search area and keyword query of approximate range search areas are at the same, nearest neighbor query throughput approximate keyword is about 50% range query. Because the nearest neighbor query used the index of exploration stage to get annular range query results, and explore the resulting annular index must be greater than minimum searching area, because the query results included in a global index, query processing can only determine the size of the search space based on the global index. Therefore unable to accurately search space will be reduced to the minimum size of the search space.

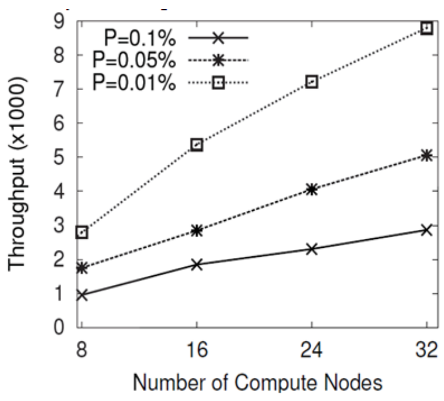

Figure 16. Effect of Minimal Search Region (CA dataset)

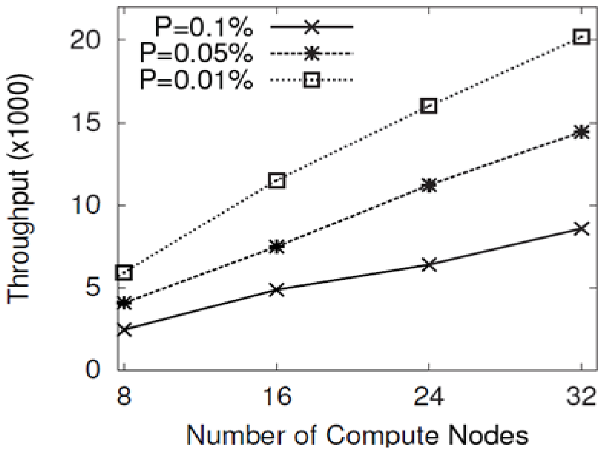

Figure 17. Effect of Minimal Search Region (TX dataset)

Figure 18and Figure 19 compare the query throughput of RBF index and RT-CAN, the minimum search region size of nearest neighbor in query test is10% of the entire space. Query used the edit distance is10%, 20% and 30%

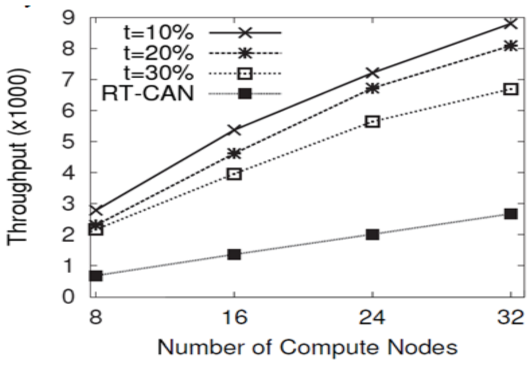

Figure 18. Effect of Normalized Edit Distance (CA dataset)

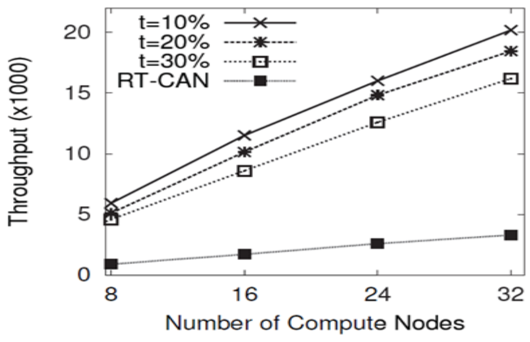

Figure 19. Effect of Normalized Edit Distance (TX dataset)

Figure 20and figure 21 test the effects of the global index selection method for query performance. In the TX and CA data set, used the global index selection method can improve query throughput, and is the same. This is consistent with figure 14 and figure 15. Used the global index selection algorithm, it can make the underlying RB tree node contains the query result is selected as the global index, Thereby reducing the size of the query in the index exploratory phase generated search area. Therefore, used the global index selection system throughput method is greater. The experiment results demonstrated the effectiveness of the global index selection method.

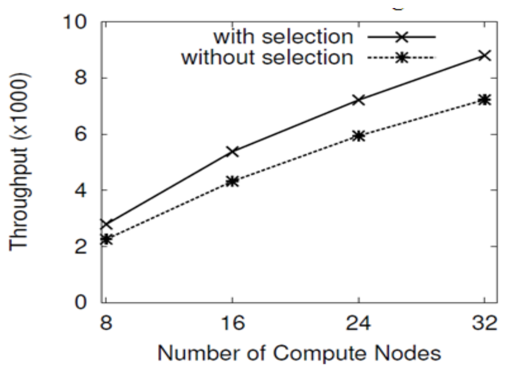

Figure 20. Effect of Global Index Selection (CA dataset)

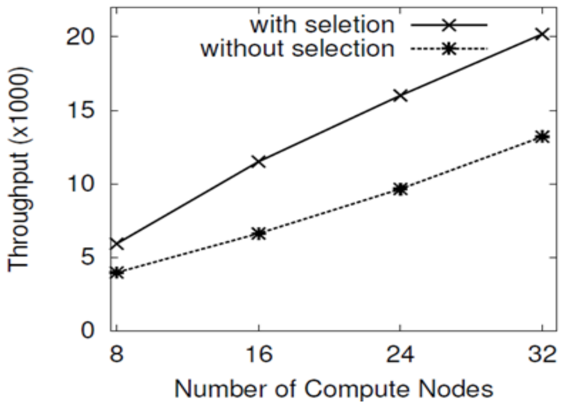

Figure 21. Effect of Global Index Selection (TX dataset)

5. Conclusion

In this paper processed approximate keyword query in cloud computing, and designed the RBF index. In order to effectively support space approximate keyword query in single computing node, this paper designs the RB tree as a local index of RBF index, and improve local approximation keyword query processing space efficiency by an additional string length q-gram and more accurate pruning strategies. RBF query performance index is tested by a cluster of real experiment data, experiment results showed that the query performance of RBF index and local RB tree pruning ability. Compared with the existing methods of RT-CAN, the RBF index has better pruning ability in the global index and the local index, it has good scalability, it can effectively support approximate keyword query of the cloud computing system space.

References

[1] B. Yao, F. Li and M. Hadjieleftheriou, “Approximate String Search in Spatial Databases”, Proceedings of the 22nd International Conference on Data Engineering. Washington, DC, USA: IEEE Computer Society, (**2010**), pp. 545–556.
[2] S. Alsubaiee, A. Behm and C. Li, “Supporting Location-Based Approximate-Keyword Queries”, Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems. New York, NY, USA: ACM, **(2010)**, pp. 61–70.
[3] W. Hongbin, Z. Lianke and W. Nianbin, “Deng Shengchun. Study on spatial index method based on iMeMex data model”, Computer science and exploration, no. 1, **(2014)**, pp. 61-72.
[4] Z. Yanqin, W. Qinqin, W. Tingting and L. Xizhao, “Cloud computing can query based on encryption research”, Journal of Nanjing Normal University (NATURAL SCIENCE EDITION), no. 01, **(2014)**, pp. 8-16.
[5] L. Yanhong, L. Guohui and Z. Bin, “The road network moving object space of continuous Top-k query keywords”, Journal of Huazhong University of Science and Technology (NATURAL SCIENCE EDITION), vol. 6, **(2014)**, pp. 127-132.
[6] S. Fengjun, “To talk about her”, Design news cloud scheme based on the platform of Hadoop. Journal of Chongqing University of Posts and Telecommunications (SOCIAL SCIENCE EDITION), **(2014)**, pp. 115-120.
[7] Z. Yongbo, C. Shudong, J. Hua, Z. Zhen and Y. Haiyun, “collaborative Internet of things based on large data management”, the integrated technology, no. 3, **(2014)**, pp. 49-60.
[8] C. Yanhong, Z. Taihong and M. Jian, “English embedded in cross language database query processing research”, Computer applications and software, no. 6, **(2014)**, pp. 244-247.
[9] S. Yingjie and M. Xiaofeng, “Review of research on query technology of cloud data management system. Chinese Journal of computers, no. 3, **(2013)**, pp. 209-225.
[10] S. Derong, Y. Ge, W. Xi, N. Tiezheng and K. Month, “A review of NoSQL system to support high data management. Journal of software, no. 8, **(2013)**, pp. 1786-1803.
[11] W. Jinbao and G. Hong, “Query processing and optimization technology research in cloud computing system. The intelligent computer and application, no. 4, **(2013)**, pp. 51-54.
[12] L. Yanhong, L. Guohui and Z. Cong, “Road network space key continuous k nearest neighbor query algorithm”, Journal of Huazhong University of Science and Technology (NATURAL SCIENCE EDITION), no. 12, (**2013**), pp. 54-58.
[13] C. Chong, C. Chunan and S. Weiwei, “Spatial keyword wireless data broadcast environment under inquiry”, Research and development of computer, no. 1, **(2013)**, pp.145-153.

Authors

Zuping Liu, She received her master's degree of Sichuan University. Now she is an associate professor in Sichuan vocational and technical college. Her major fields of study are computer application.