International Journal of Grid Distribution Computing
Vol.7, No.5 (2014), pp.1-14
http://dx.doi.org/10.14257/ijgdc.2014.7.5.01

# Multidimensional Aggregation Process in Cloud Computing System

Song Li

*Institute of Engineering, Mudanjiang Normal University*
*Mudanjiang 157011, China*
*Songli2014sl@126.com*

***Abstract***

*This paper presents multidimensional aggregation query processing algorithm in cloud computing system. The existing cloud computing research work in the MapReduce calculation framework lacks effective support to the aggregation of multi-dimensional data. On the other hand, the use of MapReduce computing framework needs to start large computing nodes, and costs huge amounts of energy. For the above problems, this paper presents multidimensional aggregation operation scheme in the cloud computing system, through two layers of index structure to reduce the query computing nodes and Calculation of aggregation operation in single computing nodes. This paper gives an algorithm framework using two layers of index structure to process multidimensional aggregation query, and proposed priority mode of performance and multi-dimensional aggregation algorithms under low-power mode in this framework. In two modes of multidimensional aggregation algorithm proposed query allocation problem, and proves that the two modes query allocation problems are NP-completing problem. This paper presents approximation algorithm of two NP-completing problem and proves the approximate ratio of two approximation algorithms. Theoretical analysis and simulation results prove the validity of the multidimensional aggregate query plan.*

***Keywords:*** *cloud computing, query processing, aggregation, MapReduce, spatial database*

## 1. Introduction

Online data analysis is an important application in decision support system [1-2], to provide decision support for the government, enterprises and staff of scientific research. Multidimensional aggregation operation is query type of representative q in on-line data analysis [3-4], to return the data summary information of given area in large multidimensional data; it is widely used in the real world. In spatial database, several metrics attribute tuple contains the geographical coordinates of the location and description. For example, the sensor measured temperature, humidity, rainfall, noise *etc.,* People tend to care about the data given in the summary of regional temperature, humidity, rainfall, and other information the average noise, not the corresponding to all of tuples in database. On the other hand, objects have multiple attributes in the real world. For example, commodity prices, sales, quality, personnel's age, salary, work experience. Decision makers often need a part object information summary in the choice of strategy [5]. For example, the price is in 100 to 200, sales volume in 10000 to 20000 of the gross sales, or age is average salary of employees between 30-40 years old. The broad application of the query can be achieved through multidimensional aggregation operations.

With the increasing popularization of cloud computing platform and services, more and more applications from a single super server migrate to the cloud computing system [6].

ISSN: 2005-4262 IJGDC

Cloud computing systems provide users with unlimited computing power virtualized compute-intensive applications are many reasonable solutions, which, including a single query on-line data involves a large number of data analysis and a large number of online transaction processing system. Therefore, design various supports OLAP query processing and indexing technology has become the technology problems need to solve urgently in the cloud [7-8]. Unfortunately, there is no research work focus on cloud computing systems in multidimensional aggregation query, existing work by simple extensions for performance aggregation operation of multidimensional data cannot obtain ideal [9].

Aiming to build a data analysis system in the cloud computing system needed to deal with the multi-dimensional aggregation techniques, propose Multi-Dimensional Aggregation in Cloud (MAC) scheme [10]. MAC can not only in cloud computing returns accurately multi-dimensional aggregation results system, you can also return to satisfactory approximation according to user-specified quality threshold of aggregation results, in order to reduce the waiting time of the user and quickly submit aggregation results to the user. MAC on multidimensional aggregation query optimization is including query performance and power consumption of the system. Only traditional methods to optimize query performance, however, the current global computing devices, especially the server energy consumption is very objective, in the economic and environmental aspects are huge costs. Therefore, MAC not only will optimize the query response time, and will the whole energy consumption into consideration, in given the limited to minimize query response time.

## 2. Multidimensional Clustered Index in Cloud System

### 2.1. System Architecture

Figure 1 describes the system architecture of MAC, MAC will computing node is divided into two types of aggregation group and the storage group in cloud system. Aggregation group is a small group of computing nodes, aggregation nodes in the group referred to as the aggregation node. The aggregation will the multidimensional data space divide into several sub areas, each aggregation node is maintained one of the sub region. The aggregation node $N_{AGG}$ maintains a MRAS tree in memory, MRAS tree is a mutation of MRA tree, and random sampling tuples is added a certain quantity of each node in the MRA tree. Random sampling of multidimensional coordinate tuples are in a multidimensional region of the tree node. By MRAS tree, gathering nodes can quickly return partial results of multidimensional aggregate queries. Each leaf node NL storage node also maintains a list of SL (NL) in MRAS tree, storage node in the list saves of the tuple of the Multidimensional region of the leaf nodes, for a given multidimensional aggregation query, if multiple regional leaf node of the NL tree of MRAS is contained in the query region, then the leaves kept aggregation values can be added directly to the final result of aggregation. So for aggregation queries accurately, the aggregation node in the MRAS tree can remove the calculation task within the query region, to reduce the computational time in storage nodes. Random sample estimate the number of storage nodes in a given area of multidimensional tuples in the MRAS tree, in order to estimate the corresponding storage node to calculate the time required for the aggregation queries.

MAC has k storage groups, each storage group to keep a complete set of multi-dimensional dataset D, each tuple has k backup in the system, $k$ is system backup factor. Typically the value is $3,4,5$.After the system receives a multidimensional aggregate query, first, aggregate queries are assigned to the corresponding aggregation node, and the use of aggregation node in memory returns some results query in MRAS tree, if the results meet the needs of users,

then the query processing stop. Otherwise, the aggregation node will send queries to the storage node query of the required data. The storage node in the storage group save the local data and external memory MRA tree index structure, and MRA tree adopts multidimensional aggregation query. For the exact multidimensional aggregation query, the storage node of received query using the MRA tree for accurate results are returned to the user, the results in this part were included in the final results. When the entire storage node receives the query to complete local operation after return partial results, the exact multidimensional aggregation query processing is over.

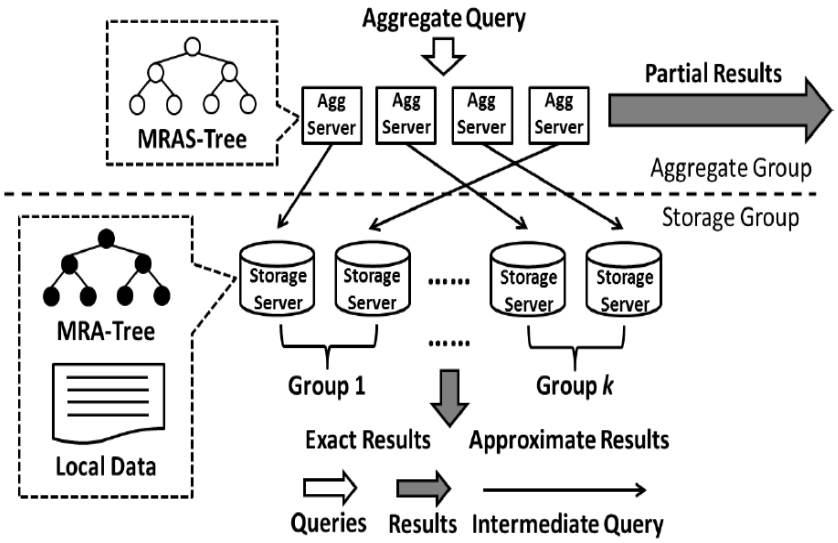



**Figure 1. MAC System Architecture**

### 2.2 Aggregation Process Framework

**Definition1: Multidimensional Aggregation Query:** Aggregation operations include commonly SUM, COUNT, AVG, MIN, MAX in multidimensional aggregate query. Given the multidimensional dataset $D$ and $d$ dimensional space area $R^d$. Each tuple $t \in D$ format: $t = (loc, values)$ .Which $t.loc = (l_{1,\ldots,} l_d) \in R^d$ $t.values = (v_{1,\ldots,} v_m)$ .Multidimensional aggregation query definition is $Q(R, OP_i)$ , which $R \subseteq R^d$ and $OP \in \{\text{SUM, COUNT, AVG, MIN, MAX}\}$ is function of return value.

Aggregation queries $Q(R, OP_i)$ process of MAC is divided into two stages, respectively, using aggregate results of the global index and a local index, and computing result of the global index and the local index is merged as the final clustering results.

The first phase is performed in the aggregation group, system selects an aggregation node as agent node $agent(Q(R, OP_i))$ of the query $Q(R, OP_i)$, and query $Q(R, OP_i)$ sent to the intersection aggregation node of space region and query region, received query set of aggregation node is $G_{AGG}^R = (N_{AGG} \mid N_{AGG} \in G_{AGG}, region(N_{AGG}) \cap R \neq \varnothing\}$ . $N_{AGG}$ of the aggregation node $G_{AGG}^R$ uses memory in the MRAS tree $T_{N_{AGG}}$ to handle query, and Partial results $\text{Re}\, sult_{Partial}$ of aggregation are obtained. MRAS tree computing space area is completely contained leaf sub-node set in the query area R

$LS_{CONTAIN}^{R} = \{n \mid n \text{ is leaf sub} - \text{node,region(n)} \subseteq R\}$, and The aggregate value of all the leaf nodes in the $LS(N_{AGG})_{CONTAIN}^{R}$ merged into partial results $\text{Re}\,sult_{Partial}$.

After the query returns aggregation node to calculate part results $\text{Re}\,sult_{Partial}$.Aggregate operation need to obtain aggregation results of the missing space area in $\text{Re}\,sult_{Partial}$.R and $LS(N_{AGG})_{OVERLAP}^{R}$ are intersecting aggregation results of multidimensional region. The second phase of aggregation query in MAC calculates partial aggregation results. To determine to access storage node $G_{ST}^{R}$, aggregation process goes for second phase. Algorithm 1 describes the framework for MAC processing of multidimensional aggregation query.

In the first phase, algorithm1 use global index to calculate partial results $\text{Re}\,sult_{Partial}$ of aggregation query and query needs to access the storage node set $G_{ST}^{R}$ in aggregation node. In the second phase, the aggregation node in a query adds in the storage node need to handle multidimensional region set, and sends a modified query to the appropriate storage node.

Algorithm1 Aggregation Processing Framework

Input: Aggregation queries $Q(R, OP_i)$

Output: Aggregation Result $\text{Re}\,sult$

1. $Agent(Q(R, OP_i)) = \arg \max_{N_{AGG} \in G_{AGG}} \{region(N_{AGG}) \cap R\}$
2. $G_{AGG}^{R} = (N_{AGG} | N_{AGG} \in G_{AGG}, region(N_{AGG}) \cap R \neq \varnothing\}$
3. $\text{Re}\,sult_{Partial} = \varnothing$
4. $\text{Re}\,sult = \varnothing$
5. FOR $N_{AGG} < G_{AGG}^{R}$ do

   Send $Q(R, OP_i)$ to $N_{AGG}$

   Calculate $LS(N_{AGG})_{CONTAIN}^{R}$ and $LS(N_{AGG})_{OVERLAP}^{R}$

   Aggregate data in the $LS(N_{AGG})_{CONTAIN}^{R}$ included in $\text{Re}\,sult_{Partial}$
6. According to the query type and all $LS(N_{AGG})_{OVERLAP}^{R}$ to calculate $G_{ST}^{R}$
7. $\text{Re}\,sult = \text{Re}\,sult \cap \text{Re}\,sult_{Partial}$
8. Return $\text{Re}\,sult$

## 3. Aggregation Process of Multidimensional Performance Priority in Cloud System

In the second phase the aggregation process， computing $LS_{OVERLAP}^{R}$ of aggregation node are all and leaf nodes of part query region intersects set in the MRAS tree. Exact aggregation need to work with aggregation value of all leaf nodes in multidimensional region to get the final result aggregation. Each leaf node $n \in LS_{OVERLAP}^{R}$ contains a storage node list $SL(n)$ in the $LS_{OVERLAP}^{R}$, storage node contains multidimensional region $region(n)$ tuple of leaf node. In order to obtain $region(n)$ aggregation values, at least one storage node needs to process the query in $SL(n)$. After the query is sent to the storage nodes, the storage node in local processing queries and returns the aggregation results. Finally, a storage node completes the

computation and returns the result, aggregation process is over. Based on the above analysis, in the second stage the time overhead of aggregation process

$$Time_{phase2} = \max_{N_{ST} \in G_{ST}^{R}} \{Time(N_{ST})\} \qquad (1)$$

Processing time last return results of minimization the storage node can improve the efficiency of aggregation query, to obtain the shortest query response time. When system selects $G_{ST}^{R}$ storage nodes to assign to sub-region, need to minimize the load processing time of maximum storage node.

**Definition2: Query Allocation Problem:** Given the storage node set $G_{ST}$, query sub-regional set $SR$, which, the target storage node set of each $sr \in SR$ is $G_{ST}^{sr}$, and $sr$ price is $count(sr)$. The solution of a mapping $H : SR \to G_{ST}$, in order to $\max_{N_{ST} \in G_{ST}} \{\sum_{H(sr)=N_{ST}} counter(sr)\}$ obtain minimization.

**Definition 3: Query Allocation Decision Problem:** Given the storage node set $G_{ST}$, query sub-regional set $SR$, which, the target storage node set of each $sr \in SR$ is $G_{ST}^{sr}$, and $sr$ price is $count(sr)$ .For given positive integer $M$ ,determining whether there is a mapping: $H : SR \to G_{ST}$, to obtain $\max_{N_{ST} \in G_{ST}} \{\sum_{H(sr)=N_{ST}} counter(sr)\} \le M$

Because the query allocation decision problem is NP complete problem, in this paper, using the approximate algorithm to solve the query allocation problem, and does not provide the exact solution of the problem. Algorithm2 gives the greedy algorithm of query sub-regional allocation problem.

Algorithm2 Query Dissemination

Input: The query region set $SR$, the storage node set $G_{ST}^{R}$

Output: Mapping $H : SR \to G_{ST}$

1. $H = \varnothing$
2. FOR $sr \in SR$ do

$N_{ST} = \arg\min_{N_{ST} \in G_{ST}} \{ \sum_{H(q)=N_{ST}} count(q)\}$

Set up $H(sr) = N_{ST}$

3. Return $H$

## 4. Aggregation Process of Multidimensional Low Power Consumption in Cloud System

In the low power consumption mode of MAC, all aggregation nodes maintain open. A small amount of aggregation node in the open power consumption is acceptable. Power consumption of main in the system is caused by the large number of storage nodes. Even if the storage nodes remain idle, power consumption of the individual storage nodes can also reach the peak power of 67% or higher. The computing hardware modern has many working state includes Active, Sleeping, Hibernate and Shutoff, the hardware power are significant differences in the diverse work state. MAC in low power consumption mode will all storage nodes adjust to sleeping. In order to reduce system power consumption during idle time. At the same time, the storage node in the system can be quickly converted to active. To ensure

availability of all data in the system. The first phase the of aggregation process, the aggregation was calculated to be allocated to the query sub-region set $SR$ of the storage node. And through the query sub-regional allocation algorithm will query sub-regional distribution $sr \in SR$ to the storage node $N_{ST}^{sr}$.In this process, $N_{ST}^{sr}$ from sleeping state to active state. And calculate the aggregate value in the query sub-region $sr$ . In the aggregation process, the power consumption of the entire system is determined by the storage node number of active. In order to limit the power consumption of the whole system. MAC limited number of storage nodes M of active. MAC of the query sub-regional allocation and scheduling scheme can be defined by the following in the low power consumption mode.

**Definition 4: Scheduling Branch:** Given the storage node set $G_{ST}$ , query sub-regional set $SR$ , which, the target storage node set of each $sr \in SR$ is $G_{ST}^{sr}$ , and $sr$ price is $count(sr)$ .Scheduling branch $S_{G_{ST}}^{SR}$ is all the elements array $(N_{ST}^{i1},...,N_{ST}^{in})$ in storage node set $G_{ST}^{'} \subseteq G_{ST}$ . Assigned to the query sub-region set of the storage node $N_{ST}^{i_j}$ is $SR_{N_{ST}^{i_j}} \subseteq SR$ . For $\forall u \neq v, SR_{N_{ST}^{u}} \cap SR_{N_{ST}^{v}} = \varnothing$ . The scheduling branch will from $j = 1$ to $n$ assign to individually process the query sub-region $N_{ST}^{i_j}$ . All query sub-region set is composed set $SR(S_{G_{ST}}^{SR})$ in scheduling branch $S_{G_{ST}}^{SR}$ , which, in the composition of all storage nodes are composed set $G_{ST}(S_{G_{ST}}^{SR})$ .

**Definition 5: M Complete Scheduling:** Given the storage node set $G_{ST}$ , query sub-regional set $SR$ , which, the target storage node set of each $sr \in SR$ is $G_{ST}^{sr}$ , and $sr$ price is $count(sr)$ . M complete scheduling is a scheduling branch set $MS_{G_{ST}}^{SR}$ . Which, $\cup_{S \in MS_{G_{ST}}^{SR}} SR(S) = SR$;

$\cup_{S \in MS_{G_{ST}}^{SR}} G_{ST}(S) = G_{ST}$ $\forall S_j \in MS$ , $i \neq j$ $G_{ST}(S_i) \cap G_{ST}(S_j) = \varnothing$ , $SR(S_i) \cap SR(S_j) = \varnothing$ and $|MS| \leq M$ .The processing time of M complete scheduling $MS_{G_{ST}}^{SR}$ is $T(MS_{G_{ST}}^{SR}) = \max_{S \in MS_{G_{ST}}^{SR}} \{T(S)\}$.

M scheduling problem of the shortest complete is a NP complete problem. In this paper, using the approximate algorithm to solve the shortest M complete problem. Algorithm3 gives approximation algorithm of the problem.

Algorithm3 Shortest M Complete Scheduling Algorithm

Input: The query region set $SR$ , the storage node set $G_{ST}^{R}$ , positive integer M

1. $ActiveNode = \varnothing$
2. $ActiveSR = \varnothing$
3. $rest = SR$
4. FOR $i = 1$ to M do

   $N_{ST} = \arg_{N \in G_{ST}} \max\{SR(N) \cap rest\}$

   $N_{ST}$ is converted to active

   $ActiveNode = ActiveNode \cup \{N_{ST}\}$

   $ActiveSR = ActiveSR \cup (SR(N_{ST}) \cap rest)$

$rest = rest \mid SR(N_{ST})$

5. WHILE $rest \neq \varnothing, ActiveSR \neq \varnothing$ do

For $sr \in ActiveSR$ do

The $sr$ is assigned to the first to handle $sr$ node $N_{ST}$ in the $ActiveNode$

$ActiveSR = ActiveSR \mid \{sr\}$

For $N_{ST} \in G_{ST}^{sr}$ do

If $ActiveSR \cap SR(N_{ST}) = \varnothing$ then

$N_{ST}$ is converted to sleeping

If $rest = \varnothing$ then

$N_{ST} = \arg_{N \in G_{ST}} \max\{SR(N) \cap rest\}$

$N_{ST}$ is converted to active

$ActiveNode = ActiveNode \cup \{N_{ST}\}$

$ActiveSR = ActiveSR \cup (SR(N_{ST}) \cap rest)$

$rest = rest \mid SR(N_{ST})$

## 5. Experiment Design and Performance Evaluation

### 5.1. Experimental Setup

In the simulation experiments, we generated local MRA tree for each storage node, and saved in the PC disk simulation. For local search in the each storage node, we process the query in the simulation lab PC, and record the time overhead. Record the time overhead for query execution time in computing system. Because aggregation algorithm presented in this paper is without communication between the various storage nodes, so use local index query time can accurately calculate the query time. Setting strategy of simulation environment and computing resource scheduling research are similar in cloud computing system. In the simulation, MRA tree in the external memory page size is 8KB. Simulation experiment adopts an aggregation node and a large number of storage nodes. The simulation system is realized by JAVA language. Java language version is JDK1.6.0 39. The experimental environment is a PC machine, processor model is Intel Core i5-2400, the processor frequency is 3.10GHz; the memory capacity is 4GB; hard disk is 500GB.

Dataset of simulation experiment adopts multidimensional dataset of artificial synthesis, the TX dataset expanded in Open Street Map open source project to get dataset of simulation experiment. Road information of USA Texas state and the space object are contained in TX dataset. Each space object contains several keywords used to describe the object. A given system storage nodes of each storage group number is $N$ .This paper will TX dataset mapping to $N$ disjoint regions of space to get the dataset of whole system. In the simulation experiment of TX. The number of objects in the system are $140 \times 10^6$. The synthesized dataset name is still TX. TX dataset includes spatial object of $140 \times 10^6$. Each object has a two-dimensional coordinates and some string attribute keywords. In this paper, to use coordinate of space object is multi-dimensional attribute, and use the number of keyword and keywords total length are two aggregation properties. Using query range size is 1% to 20% of the whole data space in the simulation experiment, each query consists of COUNT aggregation values and two aggregation properties on SUM and AVG. In 3-dimensional dataset of synthetic

uniform, we generated $140\times10^6$ 3-dimensional coordinates of uniform distribution, and generating attribute values of two uniform distribution for each coordinate. Coordinate range is $[1,10^9]$. Aggregation attribute value is $[1,10^4]$.

The storage node is divided into the K Group in simulation experiment, each storage group keeps a complete data set. The value of the backup factor K is 1, 2, 3, 4 and 5. The number of storage nodes of each storage is 5, 10, 25, 50, 100 and 200 in the group. Because the overall power of the system in a low power consumption mode depends on the number of storage nodes. Nothing to do with the algorithm in the paper, therefore, simulation results are time overhead of multidimensional aggregate query. In the experiment, the storage node from the sleep state to the open state time is 5 seconds.

This paper will contrast experiment name MAC-random, in the query process of performance priority. MAC-random does not use the query allocation algorithm in the paper. But taken randomly nodes strategy of selected storage. The queries were randomly assigned to treatment storage node. Query processing in a low power consumption mode. When MAC-random selects open node, random selection can calculate Storage node of untreated query. This paper method is named MAC in the two set of experiments.

### 5.2 Aggregation Process of Performance Priority

First, to test aggregation process of performance priority, query response time and the relationship of the amount of storage in each storage group nodes. Figure 1 and Figure 2 are query response time varies with the size of the storage group in TX and 3 dimensional synthetic dataset. Backup factor of experiment is3, search region size for the whole data space is 10%, from the Figure 1-2 shows, increase the number of nodes in the storage group, the query response time is decreased, and the decreased gradually weakened. This is because the query need to access multiple query sub-regional, in the query of sub-regional distribution process, increase the number of nodes in the query can be shared equally overall query time overhead, query response time will show decreasing trend. However, when a large number of storage nodes, each storage node for the query sub-region has very little, so the query time is decreasing and weak. From Figure 1-2 shows, the MAC query response time substantially is better than MAC - random, this shows that query of allocation algorithm is the validity. Three-dimensional synthetic data query time overhead is larger than the query time of TX, because the former data dimension is greater, so the query and more sub-regions are intersecting, and the query and intersecting part of sub-regions are greater. To produces a longer query response time.

Figure 3 and Figure 4 give the backup factor for affection of time response query. Query the size of the area used in the experiment was 10% of the multi-dimensional space, each storage group contains 50 storage nodes. In the backup factor increased from 1 to 5, the query response time to gradually decline. Backup factor is 1, MAC and MAC-random distribution scheme is the same, so it has the same query response time. After increase backup factor, MAC query time is significantly less than MAC-random, this is because the MAC can choose appropriate target in the storage node more to distribute query. The query response time in the backup factor is larger to decline significantly weak. Query factor increased to 4, 5, the storage node cannot be further split the query sub-regions overhead to cause by cost of the query time.

Figure 5 and Figure 6 give change of the query time with the query region size. Experiment query region size is increased from 1% to total 25% of multidimensional space. System has three storage groups, each storage group contains 50 storage nodes. The experiment results show that the query performance gradually is decreased after increase the

query region. Figures 1 and 2 results are consistent, the query response time is less than 3-dimensional dataset in the TX. This is because the query region and low dimension query sub-regional are intersecting. The growth rate of query response time is less than the 3-dimensional dataset in TX dataset. This is because the two-dimensional space, increasing the size of the query region intersects the query sub-regional growth rate is lower than the number of three-dimensional space.

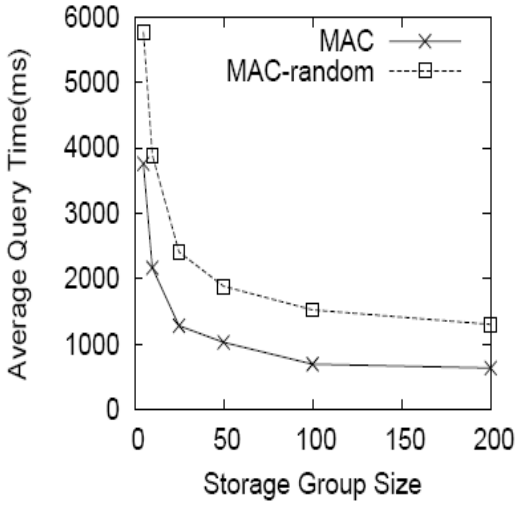



**Figure 1. Effect of Storage Group Size (TX Dataset)**

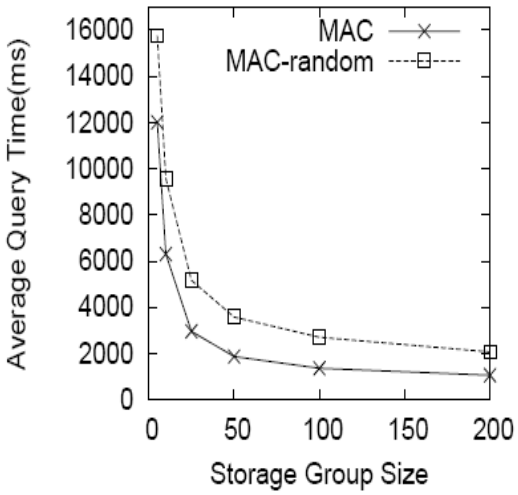



**Figure 2. Effect of Storage Group Size (3d Dataset)**

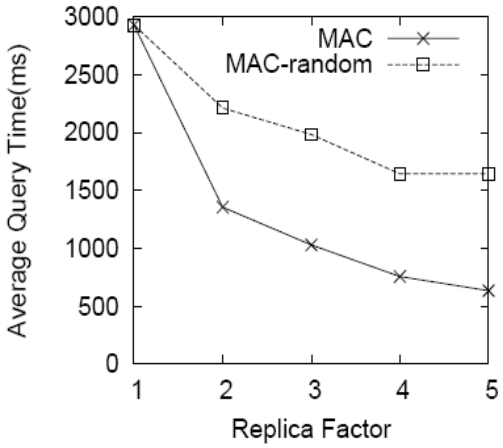



**Figure 3. Effect of Replica Factor (TX Dataset)**

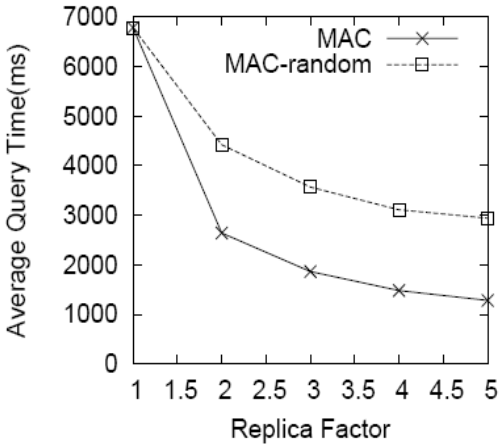



**Figure 4. Effect of Replica Factor (3d Dataset)**

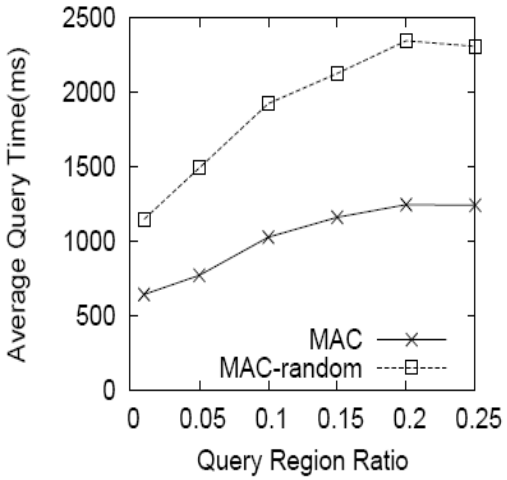



**Figure 5. Effect of Query Region (TX Dataset)**

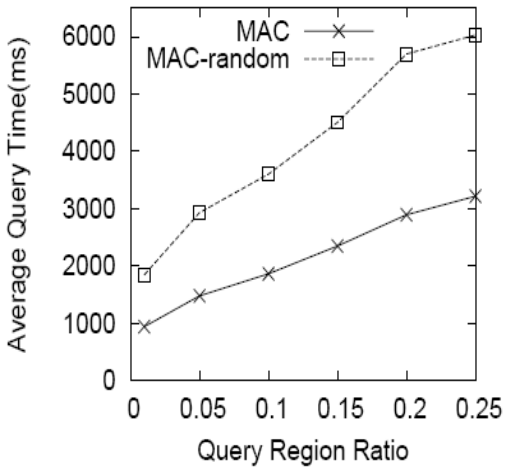



**Figure 6. Effect of Query Region (3d Dataset)**

### 5.3 Aggregation Process of Low Power Consumption

Figure 7 and Figure 8 show the query response time varies with the size of the storage group. Experiments use 3 storage groups; the query area for the whole data space is 10%. The storage node number of query is 60% of node number in single storage group. From the experimental results can be seen, the storage group size is increased from 5 to 200, the response time of query to decrease. The query time of MAC is better than MAC random, but the advantage increases with the size of the storage group is weakened. The reasons for this situation is to increase the storage node can be used, the query can better share each storage node, so MAC-random can also get better performance. However, in the use of the storage node is less, the performance advantage of MAC is obvious. The method is applied to the system of limit energy consumption.

Figure 9 and figure 10 present change of the query response time varies with the backup factor. Each storage group contains 50 storage nodes in the experiment; the query region is 10% of the entire space. The number of queries is limited to 30 storage nodes. From the experimental results can be seen, the response time of MAC is less than MAC-random. In the TX dataset, increasing the duty factor can reduce query time of MAC. MAC slight is increased in the three-dimensional synthetic data. This is because MAC cannot find the optimal query allocation scheme, but it can provide schedule scheme of quality assurance. MAC-random in the face of more choice of storage nodes to use scheduling scheme caused by a longer query time, this is because the choice of random method cannot choose the appropriate storage node.

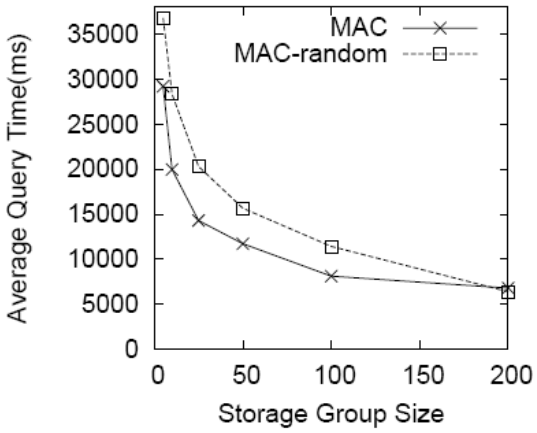



**Figure 7. Effect of Storage Group Size (TX Dataset)**

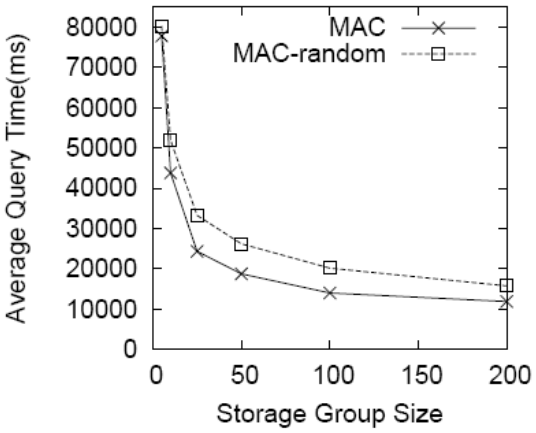



**Figure 8. Effect of Storage Group Size (3d Dataset)**

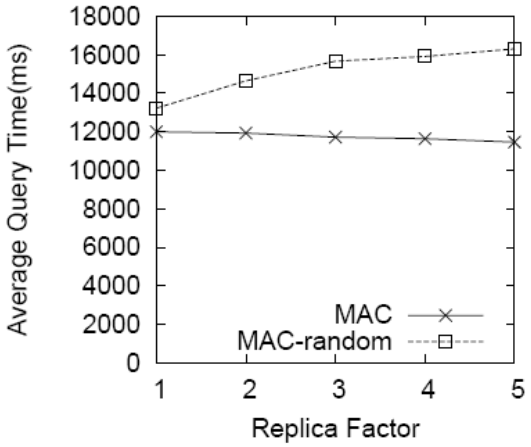



**Figure 9. Effect of replica factor (TX dataset)**

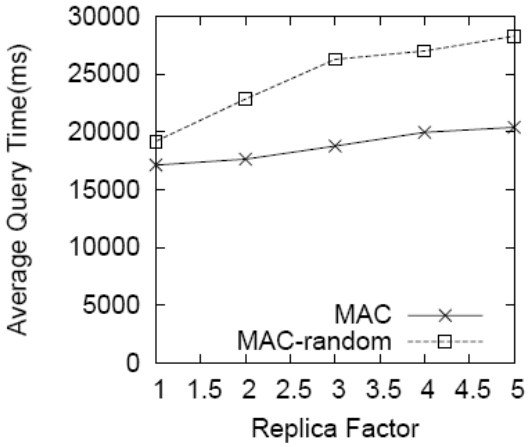



**Figure 10. Effect of replica factor (3d dataset)**

Figures 11 and 12 show that the query time increases with increasing area of the query in the system. Experiments use three storage node, each storage group contains 50 storage nodes**,** a total of 150 storage nodes. The maximum number of open nodes is 30, query region size increased from 1% to 25% of the entire space. Growth trends of query time is less than three-dimensional dataset in the TX dataset. This is consistent with the Figure 5-5 and Figure 5-6 results.

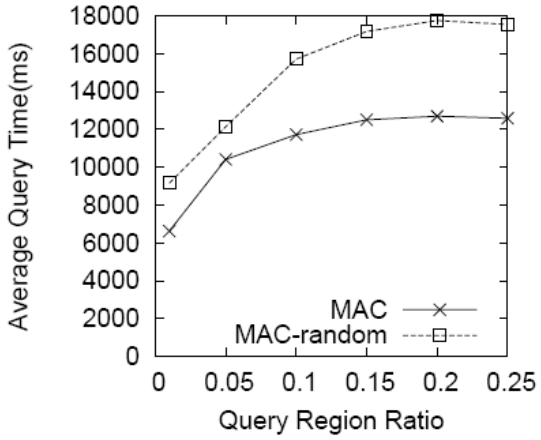



**Figure 11. Effect of Query Region (TX Dataset)**

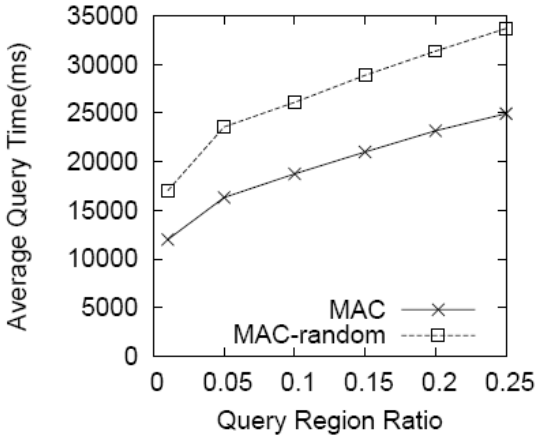



**Figure 12. Effect of Query Region (3d Dataset)**

Figure 13 and Figure 14 shows change of the query response time with the open limit the number of nodes changes. Experiments use three storage node, each storage group contains 50 storage nodes. The query area for the whole data space is 10%. When number limiting of open nodes grow from 10 to 50 in the process, query response time decreases in TX and three-dimensional dataset, and decreasing trend gradually weakened. MAC query time is always less than the MAC-random. This is because there is more storage nodes in parallel query processing, therefore, the query time can be effectively reduced. However, increasing open storage node cannot continue to reduce the query time. This is because the query sub-region only is distributed in the part storage nodes, other storage nodes cannot accelerate query processing.

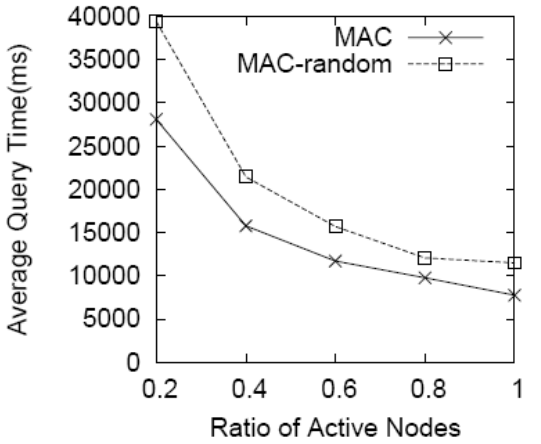
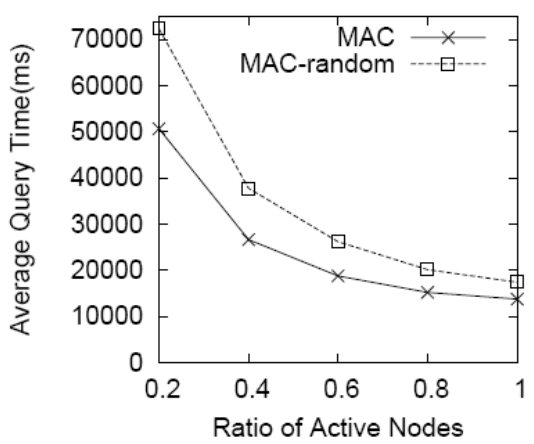



**Figure 13. Effect of Active Node Figure Number (TX dataset)**




**14. Effect of Active Node Number (3d Dataset)**

## 6. Conclusion

This paper proposed MAC schema in cloud computing system, in order to process multidimensional aggregation query. In the storage nodes and external memory effectively cut the search space through two layers of index MAC-Index in MAC. The query algorithm

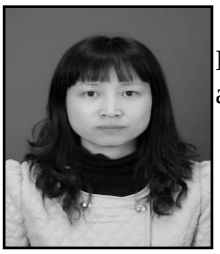
 

of multidimensional aggregation is proposed in MAC performance priority mode and low power consumption mode. This paper proves that query sub-regional allocation and scheduling problem are NP complete problem in the performance priority mode and low power consumption mode, and gives the effective approximate algorithm for query sub-regional scheduling process. Experiments show that MAC can effectively handle multidimensional aggregate query in the cloud computing system, and to support query mode performance priority and low power consumption.

## References

[1] S. Jie, H. Hongying, W. Zhi and Z. Zhiliang, "Energy efficiency improvements in cloud computing environment measurement model", Journal of Zhejiang University (Engineering and Technology Edition), vol. 01, **(2013)**, pp. 44-52.
[2] W. Peng, Z. Lei, R. Chao and G. Youming, "Study on cloud computing model and simulation analysis system of phase space", Chinese Journal of computers, vol. 02, **(2013),** pp. 286-296.
[3] S. Yingjie and M. Xiaofeng, "Review of research on query technology of cloud data management system", Chinese Journal of computers, vol. 02, **(2013)**, pp. 209-225.
[4] W. Jinbao and G. Hong, "Query processing and optimization technology research in cloud computing system", The intelligent computer and application, vol. 04, **(2013)**, pp. 51-54.
[5] L. Chunyan and Z. Xuejie, "Research on high performance computing cloud based on benchmark test", Computer science, vol. 12, **(2013)**, pp. 23-30.
[6] Q. Yurong, W. Weiyuan, S. Hua, L. Bin and Y. Xingyao, "The cloud computing environment, software and hardware of energy-saving and load balancing strategy", The application of computer, vol. 12, **(2013),** pp. 3326-3330.
[7] E. Thereska, A. Donnelly and D. Narayanan, "Sierra: Practical Power-Proportionality for Data Center Storage", Proceedings of the Sixth Conference on Computer Systems, New York, NY, USA: ACM, **(2011)**, pp. 169–182.
[8] J. Chou, J. Kim and D. Rotem, "Energy-Aware Scheduling in Disk Storage Systems. Distributed Computing Systems (ICDCS)", Washington DC: IEEE, **(2011),** pp. 423–433.
[9] B. Guenter, N. Jain and C. Williams, "Managing Cost, Performance, and Reliability Tradeoffs for Energy-Aware Server Provisioning", INFOCOM, Washington D-C: IEEE, **(2011),** pp. 1332–1340.
[10] I. Lazaridis and S. Mehrotra, "Progressive Approximate Aggregate Queries with Multi-Resolution Tree Structure", Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data, New York, NY, USA: ACM, **(2001),** pp. 401–412.

## Authors



**Song Li**, she is an associate professor at Institute of Engineering of Mudanjiang Normal University. She is in the research of computer application.