J Inf Process Syst, Vol.13, No.4, pp.668~676, August 2017
https://doi.org/10.3745/JIPS.04.0036

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Hadoop Based Wavelet Histogram for Big Data in Cloud

Jeong-Joon Kim*

Abstract

Recently, the importance of big data has been emphasized with the development of smartphone, web/SNS. As a result, MapReduce, which can efficiently process big data, is receiving worldwide attention because of its excellent scalability and stability. Since big data has a large amount, fast creation speed, and various properties, it is more efficient to process big data summary information than big data itself. Wavelet histogram, which is a typical data summary information generation technique, can generate optimal data summary information that does not cause loss of information of original data. Therefore, a system applying a wavelet histogram generation technique based on MapReduce has been actively studied. However, existing research has a disadvantage in that the generation speed is slow because the wavelet histogram is generated through one or more MapReduce Jobs. And there is a high possibility that the error of the data restored by the wavelet histogram becomes large. However, since the wavelet histogram generation system based on the MapReduce developed in this paper generates the wavelet histogram through one MapReduce Job, the generation speed can be greatly increased. In addition, since the wavelet histogram is generated by adjusting the error boundary specified by the user, the error of the restored data can be adjusted from the wavelet histogram. Finally, we verified the efficiency of the wavelet histogram generation system developed in this paper through performance evaluation.

Keywords

Big Data, Histogram, MapReduce, Wavelet

1. Introduction

Big data has large data capacity and data structure is atypical. When managing and analyzing such big data, there is more and more research on generating compressed summary information that represents the most important characteristic of big data than big data itself. Especially, it is possible to save time and system resources by making good use of data summary information in various online analytical processing (OLAP) applications that require quick response results within a short time [1,2]. One of the summary information generation techniques, the histogram [1] can generate optimal data summary information that does not cause a loss of information of the original data. MapReduce [3] is being used as a framework for data processing by Google and has recently attracted attention in various fields due to its excellent scalability and stability. Therefore, research is underway to generate a wavelet histogram in the MapReduce environment.

In the previous research, since the generation of the wavelet histogram is performed more than once,

Manuscript received April 6, 2017; first revision May 22; accepted May 25, 2017.
Corresponding Author: Jeong-Joon Kim (jjkim@kpu.ac.kr)
* Dept. of Computer Science & Engineering, Korea Polytechnic University, Siheung, Korea (jjkim@kpu.ac.kr)

the generation speed is slow and the error tolerance of the data reconstructed from the wavelet histogram cannot be considered. Therefore, there are disadvantages that the error of the reconstructed data cannot be adjusted in advance. However, the wavelet histogram generation system developed in this paper can generate a wavelet histogram within a short time by using MapReduce Job once. In addition, since the user can specify an error boundary in advance before generating the wavelet histogram, the error of the restored data with the wavelet histogram can be adjusted in advance. Finally, we verified the efficiency of the wavelet histogram generation system developed in this paper through performance evaluation. The Hadoop-based wavelet histogram generation system studied in this paper can generate optimum data summary information at low cost and without loss of information than the existing methods. Therefore, it is expected that it can be used in various big data application services that require real-time question and answer in the future.

2. Related Research

2.1 Wavelet Histogram

The wavelet histogram can combine the advantages of wavelet [2] and the histogram to generate optimal data summary information with little information loss of original data. The wavelet histogram transforms original data into wavelet coefficients through a wavelet transform. To summarize the data, only the top K (TOP-K) coefficients of the wavelet coefficients are stored. The wavelet histogram is generated by these TOP-K wavelet coefficients.

Since some of the wavelet coefficients are not stored in order to summarize the data, an error occurs when the data is restored from the stored wavelet coefficients. Therefore, there are two methods to calculate the error for the reconstructed data: average square error method [2] and maximum error method [4]. Since the calculation method can reflect the error condition of the restored data as a whole and the calculation method can reflect the error situation of each data item of the restored data, the calculation method is used more than the calculation method. Therefore, the wavelet histogram generation technique which does not exceed the error boundary is mainly studied considering the error calculated by the calculation method.

2.2 Histogram Estimator for Data in the Cloud

Histogram estimator for data in the cloud (HEDC) [5] is a system that estimates the distribution of data stored in HDFS (Hadoop distributed file system) using equi-width histogram [6] that divides original data into several sections. HEDC shortened the time because it builds equi-width histogram with some sample data of original data. The HEDC has a sampling step for extracting sample data from HDFS, a processing step for constructing equi-width histogram on MapReduce, and a statistical calculation for checking whether the equi-width histogram constructed meets error boundaries.

The equi-width histogram is relatively simple to construct because it is relatively simple to construct. However, if you do not satisfy the error bounds, you need to scan the whole data in advance to calculate the error bounds. This has the disadvantage of being able to take longer to build because it has to be supplemented.

2.3 Wavelet Histograms on Large Data in MapReduce

Wavelet histograms on large data in MapReduce (WHM) [7] generates a local wavelet histogram for data stored in HDFS using MapReduce and then applies a distributed TOP-K algorithm. A typical TOP-K algorithm TPUT (Three-Phase Uniform Threshold) [8] generates a global wavelet histogram for the entire data set. WHM reduces the burden on the reducer nodes by distributing the TOP-K operations performed by the Reducer nodes of MapReduce to each Mapper node. However, it takes a long time to generate the TOP-K wavelet histogram for the entire data because three MapReduce Jobs must be executed. Since the error range cannot be specified before generating the wavelet histogram, there is a high possibility that the error of the restored data becomes large.

3. Wavelet Histogram Generation System using MapReduce

3.1 Overall System Design

A wavelet histogram generation system using MapReduce consists of a data preprocessing manager, a histogram generation manager, and a wavelet histogram generation manager. Fig. 1 shows the structure of the entire system.

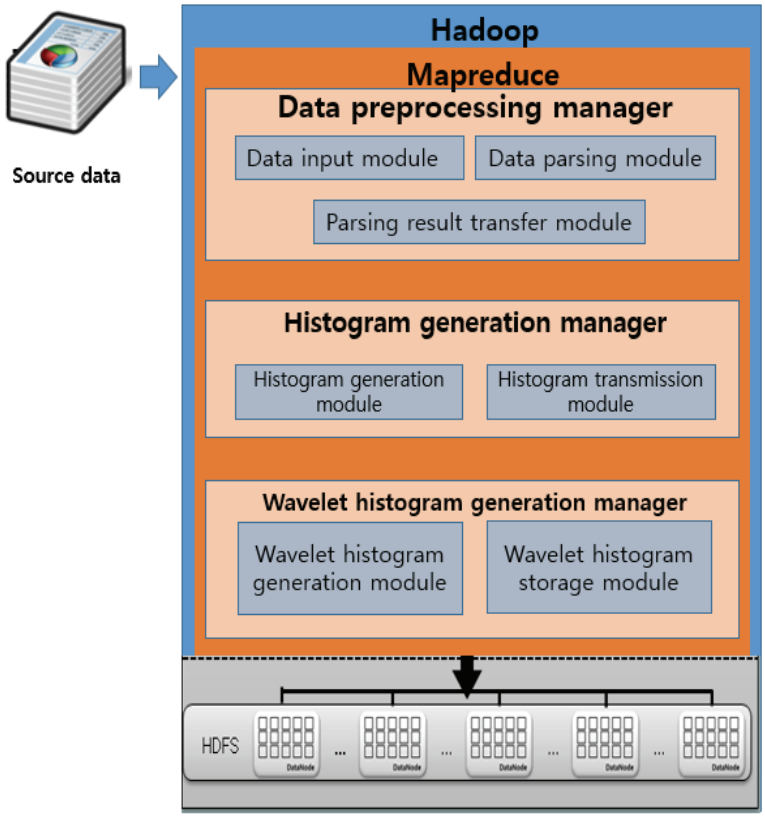

Fig. 1. Structure of the whole system.

The overall process of the system proposed in this paper is as shown Table 1.

Table 1. Overall process of the system

Main process	Detailed process
1. Preprocessing of data	(1.1 Data input module) Reading big data with various saving formats by using the reader of the record (1.2 Data analysis module) Extracts the value of the attribute among the data read from the data input module (1.3 Analysis result transmission module) Generate reception histogram of the result extracted from analysis module, forward to administrator's histogram generation module
2. Create a histogram	(2.1 Histogram generation module) Regional histogram generation in the form of Key (attribute value), Value (appearance frequency) (2.2 Histogram transmission module) Generate a histogram of the area generated by the histogram generation module, generate a wavelet histogram, and send it to the administrator
3. Wavelet histogram generation	(3.1 Wavelet histogram generation module) Histogram generation, administrator receives regional (Local) histogram composed of (Key, Value) pair and merges it. Also, by merging the values of the same Key, the merged (Key, Value) pair constitutes a global histogram (3.2 Wavelet histogram storage module) Distributed all of the keys containing the global histogram in Hadoop HDFS

3.2 Data Preprocessing Manager

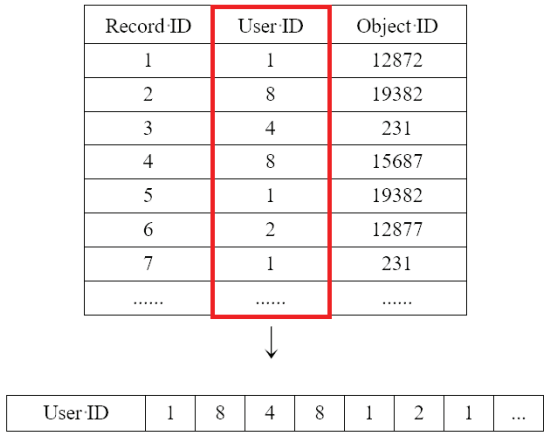

Fig. 2. Example of parsing attribute values.

The data preprocessing manager consists of a data input module, a data parsing module, and a parsing result transmission module. The preprocessing manager is responsible for parsing the value of the attribute in the source data and transmitting the parsing result. Fig. 2 is an example of parsing attribute values from source data.

As shown in Fig. 2, records of users visiting data on a certain server are stored in the form of big data.

In order to estimate the distribution of users connecting to the server, the value of the User ID attribute, which is the user ID that accesses the server among the various attribute values, is parsed.

3.3 Histogram Generation Manager

The histogram generation manager is composed of a histogram generation module and a histogram transmission module. The histogram generation module manages the output and transmission of the attribute values received from the data preprocessing manager using a map deuce mapper, into a pair of key and value. The output key of the mapper is the only value that appears in the User ID, and the output Value represents the number of times the User ID appears. Fig. 3 shows an example of generating the histogram by constructing the User ID attribute value as a pair of Key and Value.

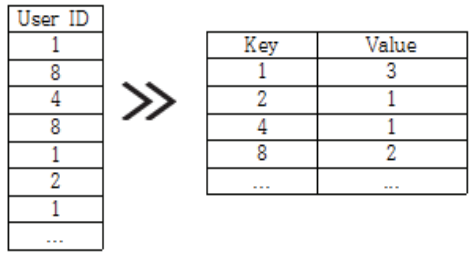

Fig. 3. Histogram of User ID attribute values.

As shown in Fig. 3, Key 1, Key 2, Key 4, and Key 8 are 3, 1, 1, and 2, respectively, so the values are 3, 1, 1, and 2, respectively.

3.4 Wavelet Histogram Generation Manager

The wavelet histogram generation manager is composed of a wavelet histogram generation module and a wavelet histogram storage module. The histogram generation manager generates a wavelet histogram that satisfies an error boundary using a MapReduce reducer. The wavelet histogram includes the wavelet histogram coefficients, and an example of generating the wavelet histogram coefficients is shown in Fig. 4.

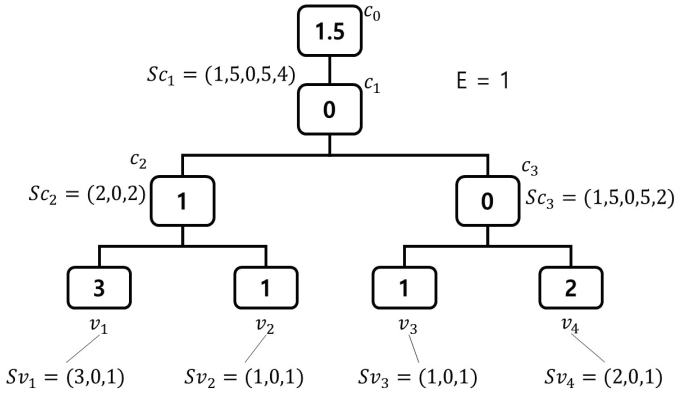

Fig. 4. Example of wavelet histogram coefficient generation.

Table 2 shows the terms used when calculating the wavelet histogram coefficients.

Table 2. Terms used in (1) and (2)

Term	Explanation
$v_1 - v_i$	Node with original data value
$c_0 - c_i$	A node having a wavelet histogram coefficient
E	Error boundary
n_l, n_r	n_l and n_r are the n values of s for the left / right child nodes of a node, respectively
a_l, a_r	a_l and a_r are the a values of s for the left / right child nodes of a node, respectively.
d_l, d_r	d_l and d_r are the d values of s for the left / right child nodes of a node, respectively.
$v_{\max_l}, v_{\max_r}$	$v_{\max_l} = a_l + d_l$, $v_{\max_r} = a_r + d_r$
$v_{\min_l}, v_{\min_r}$	$v_{\min_l} = a_l - d_l$, $v_{\min_r} = a_r - d_r$
v_{max}, v_{min}	$v_{max} = \max\{v_{\max_l}, v_{\max_r}\}$, that is, the maximum value among $v_{\max_l}$ and $v_{\max_r}$, and $v_{min} = \min\{v_{\min_l}, v_{\min_r}\}$, that is, the minimum value among $v_{\min_l}$ and $v_{\min_r}$.
s	s is the structure used to calculate the wavelet histogram coefficients. The values of $s = (a, d, n)$, a and d are determined by the values of $(v_{max} - v_{\min})/2$ in (1.1), and n is the number of nodes belonging to the subtree of one node
S	Set of s
c	Wavelet histogram coefficient

As shown in Fig. 4, since the subtree of the v_1 node is the v_1 node itself starting from the v_1 node, s_{v_1} for the leaf node such as v_1 is calculated as (v_1, Node value, 0, 1). Therefore, $s_{v_1} = (3,0,1)$, $s_{v_2} = (1,0,1)$ for v_1 node and v_2 node, which are leaf nodes, are created and added to S. In this case, the $s_{v_1} = (a_l, d_l, n_l) = (3,0,1)$ for the left child node v_1 of c_2 satisfying $n_l = n_r = 1$ and the child node v_2 of the right child node of c_2 exist in S. Therefore, the wavelet histogram coefficient is calculated using (1).

$$(v_{\max} - v_{\min})/2 \begin{cases} \geq E, & \begin{matrix} c = (a_l - a_r)/2, d = \max\{d_l, d_r\}, \\ a = a_l - c,\ n = 2n_l \end{matrix} \\ < E, & \begin{matrix} c = 0, d = (v_{\max} - v_{\min})/2, \\ a = (v_{\max} + v_{\min})/2,\ n = 2n_l \end{matrix} \end{cases} \tag{1}$$

Since v_{max} and v_{min} for c_2 nodes are 3 and 1, respectively, $d = 0$, $a = 1$, $n = 2$ and $c = 1$ because of $(v_{max} - v_{min})/2 \geq E$. Thus, $s_{c_2} = (2,0,2)$ for c_2 is added to S and s_{v_1} and s_{v_2} are removed from S. If one of s remains in S, we calculate c_0 by applying (2).

$$IF\ |a| \geq |E - d|, THEN\ c = a \tag{2}$$

According to (2), since $s_{c_0} = (1.5,0.5,4)$ remaining in S is |1.5|≥|1-0.5|, $c_0 = 1.5$. Finally, non-zero c_0 and c_2 are stored as final wavelet histogram coefficients. (3) is a formula for restoring data from a wavelet histogram, and Table 3 is a term used in (3).

$$D(i) = \sum_{j \in path(i)} sign(i,j) \times c_j,$$
$$sign(i,j) = \begin{cases} 1, & j = 0 \text{ or } v_i \in left\ \ subtree\ \ of\ \ c_j \\ -1, & v_i \in right\ \ subtree\ \ of\ \ c_j \end{cases} \tag{3}$$

Table 3. Terms used in (3)

Term	Explanation
$D(i)$	Restoration value of v_i
c_j	Wavelet histogram coefficient
$sign(i,j)$	Variable representing the sign of c_j
$path(i)$	The path from one leaf node to the root node

In order to recover the v_2 node value in Fig. 4, $D(2)$ is calculated by performing an operation between c_j belonging to $path(2)$. Therefore, the c_j belonging to $path(2)$, which is the path from v_2 to c_0, is $< c_0, c_1, c_2 >$. If $j = 0$, it is $sign(2,0) \times c_0 = 1.5$. And v_2 are $sign(2,1) \times c_1 = 0$ and $sign(2,2) \times c_2 = -1$, respectively, because they belong to the left subtree of c_1 node and the right subtree of c_2 node, respectively. Therefore, $D(2) = 1.5 + 0 - 1 = 0.5$. In this way, the restored values of the nodes $v_1 - v_4$ are <2.5, 0.5, 1.5, 1.5>, and the errors calculated by the L_∞ calculation method are <0.5, 0.5, 0.5, 0.5>, respectively, so that they do not exceed the specified error boundaries (E=1).

4. Performance Evaluation

4.1 Performance Evaluation Environment

For performance evaluation, Ubuntu 14.04.2 LTS 64 bit and Hadoop 1.2.1 were used as the operating system. Eclipse JUNO version was used as development tool. A total of 16 cluster environments were built. NameNode of Hadoop consisted of Intel i5 CPU and 4G RAM. The remaining 14 DataNodes were configured with Intel i5 CPU and 2G RAM.

4.2 Performance Comparison Evaluation

To verify the efficiency of the wavelet histogram generation system developed in this paper, we used the Page Traffic statistical information data (180 million) accumulated during the week of the Wikipedia Hits Log [9]. The Page Traffic statistical information data used the value of the language attribute of the visitor page among the four attributes (visitor page language, page name, number of page requests, response contents size) as the input data of the performance evaluation. The generation size of the histogram (the number of coefficients included in the histogram) and the generation time of the wavelet histogram were measured.

4.2.1 Histogram generation size

As shown in Fig. 5, the size of the histogram generated by the proposed system is 45% smaller than that of WHM and 54% smaller than that of HEDC. The reason is that the method proposed in this paper has a small histogram generation size because it generates a histogram within a predetermined error boundary.

And the larger the error bound, the larger the histogram generation size for all systems Diminish. However, even if the error boundary increases from the error boundary of 200 or more, the histogram generation size does not decrease any more in all three systems.

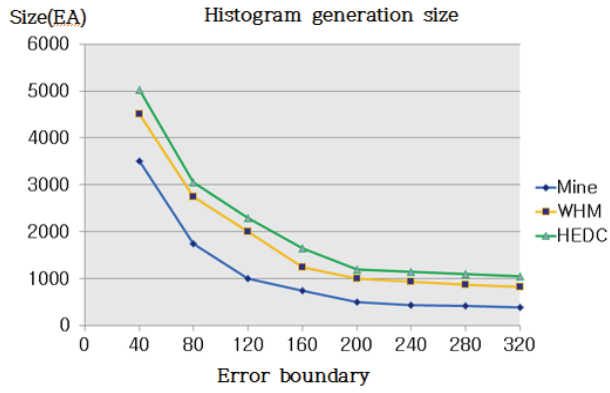

Fig. 5. Histogram generation size.

4.2.2 Histogram generation time

As shown in Fig. 6, when the proposed system is used, the time required to generate the histogram is reduced by 19% from WHM and by 13% from HEDC. Therefore, the overall creation time is increased. The HEDC executes the MapReduce Job when the generated histogram does not satisfy the error boundaries, it executes the MapReduce Job to supplement the existing histogram again.

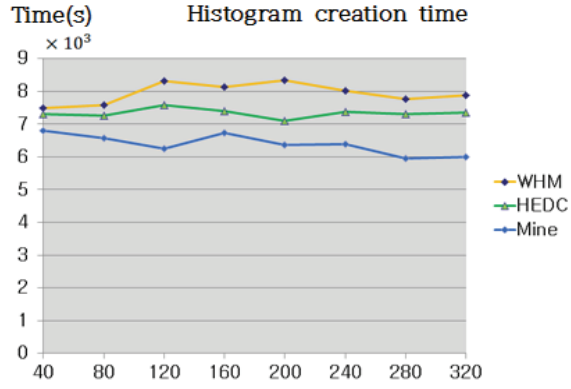

Fig. 6. Histogram generation time.

However, the method proposed in this paper generates histogram by executing MapReduce Job only once. Therefore, it has the advantage of less execution time.

5. Conclusion

Since big data has a large amount, fast creation speed, and various properties, it is more efficient to process big data summary information than big data itself. The wavelet histogram, which is one of the summary information generation techniques, can generate optimal data summary information that does not cause loss of information of original data. MapReduce is being used as a framework for Google's data processing, and has recently gained popularity in a variety of areas due to its excellent

scalability and stability. Therefore, efficient wavelet histogram generation system is needed by using MapReduce.

In this paper, we propose a wavelet histogram generation system in a distributed mapreduce environment. The wavelet histogram generation system developed in this paper can generate the wavelet histogram through only one mapping task, so the entire wavelet histogram generation time is reduced. In addition, since the user can specify an error boundary in advance before generating the wavelet histogram, the error of the restored data from the wavelet histogram can be adjusted in advance.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2017R1A2B4011243).

References

[1] Y. Matias, J. S. Vitter, and M. Wang, "Wavelet-based histograms for selectivity estimation," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, New York, NY, 1998, pp. 448-459.

[2] E. J. Stollnitz, T. D. DeRose, and D. H. Salesin, *Wavelet for Computer Graphics: Theory and Applications*, San Francisco, CA: Morgan Kaufmann, 1996.

[3] MapReduce Tutorial [Online]. Available: https://hadoop.apache.org/docs/r1.2.1/mapred-tutorial.html.

[4] M. Garofalakis and P. B. Gibbons, "Probabilistic wavelet synopses," *Journal of ACM Transactions on Database Systems*, vol. 29, no. 1, pp. 43-90, 2004.

[5] Y. Shi, X. Meng, F. Wang, and Y. Gan, "HEDC: a histogram estimator for data in the cloud," in *Proceedings of the 4th International Workshop on Cloud Data Management*, Mui, HI, 2012, pp. 51-58.

[6] V. Poosala, Y. E. Ioannidis, P. J. Haas, and E. J. Shekita, "Improved histograms for selectivity estimation of range predicates," in *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, New York, NY, 1996, pp. 294-305.

[7] J. Jestes, K. Yi, and F. Li, "Building wavelet histograms on large data in MapReduce," *Proceedings of the VLDB Endowment*, vol. 5, no. 2, pp. 109-120, 2011.

[8] P. Cao and Z. Wang, "Efficient top-k query calculations in distributed networks," in *Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing*, New York, NY, 2004, pp. 206-215.

[9] Wikipedia Page Traffic Statistics [Online]. Available: http://aws.amazon.com/datasets/2596.

Jeong-Joon Kim http://orcid.org/0000-0002-0125-1907

He received his B.S. and M.S. in Computer Science at Konkuk University in 2003 and 2005, respectively. In 2010, he received his Ph.D. in at Konkuk University. He is currently a professor at the department of Computer Science at Korea Polytechnic University. His research interests include Database Systems, Big Data, Semantic Web, Geographic Information Systems (GIS) and Ubiquitous Sensor Network (USN), etc.