

A New Approach to Web Data Mining Based on Cloud Computing

Wenzheng Zhu* and Changhoon Lee

School of Computer Science, Konkuk University, Seoul, Korea
wenzheng@live.co.kr, chlee@konkuk.ac.kr

Abstract

Web data mining aims at discovering useful knowledge from various Web resources. There is a growing trend among companies, organizations, and individuals alike of gathering information through Web data mining to utilize that information in their best interest. In science, cloud computing is a synonym for distributed computing over a network; cloud computing relies on the sharing of resources to achieve coherence and economies of scale, similar to a utility over a network, and means the ability to run a program or application on many connected computers at the same time. In this paper, we propose a new system framework based on the Hadoop platform to realize the collection of useful information of Web resources. The system framework is based on the Map/Reduce programming model of cloud computing. We propose a new data mining algorithm to be used in this system framework. Finally, we prove the feasibility of this approach by simulation experiment.

Category: Smart and intelligent computing

Keywords: Web data mining; Cloud computing; Hadoop; Map/Reduce programming model

I. INTRODUCTION

We live and operate in the world of computing and computers. The Internet has drastically changed the computing world from the concept of parallel computing to distributed computing, to grid computing, and now to cloud computing [1]. With the rapid development of Internet technology, the data in the Internet is growing exponentially, so how to find and mine valuable information has become a hot area of research.

Web data mining [2] aims to discover useful information or knowledge from Web hyperlinks, page contents, and usage logs. Based on the primary kinds of data used in the mining process, Web data mining tasks can be categorized into three main types: Web structure mining, Web content mining, and Web usage mining. Web structure mining discovers knowledge from hyperlinks, which rep-

resent the structure of the Web. Web content mining extracts useful information or knowledge from Web page content. Web usage mining mines user access patterns from usage logs, which record the clicks made by every user.

Basically, data mining technique is used in Web mining. But there are some differences. In traditional data mining, the data is often already collected and stored in a data warehouse. For Web data mining, data collection can be a substantial task especially for Web structure and content mining, and involves crawling a large number of target Web pages. Web data mining is an extended version of data mining.

As we observed, the Internet has now changed computing to cloud computing. Map/Reduce is a great programming model in cloud computing [3] that was introduced by Google. It is well suited to the execution of

Open Access <http://dx.doi.org/10.5626/JCSE.2014.8.4.181>

<http://jcse.kiise.org>

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Received 12 May 2014; Revised 17 August 2014; Accepted 18 August 2014

*Corresponding Author

large distributed jobs in a cloud infrastructure. In brief, a Map/Reduce [4] computation executes as follows: some map tasks are given one or more chunks from a distributed file system. Each of these map tasks turns the chunk into a sequence of key-value pairs, and these pairs are written to local disk as intermediate files partitioned into R (the number of reduce tasks) regions by the partitioning function. The locations of these regions are passed back to the master, who is responsible for forwarding these locations to the reduce tasks. Each of the R reduce tasks is responsible for one of these regions applying reduction. So, all key-value pairs with the same key wind up at the same reduce task. The reduce tasks work on one key at a time, and combine all the values associated with that key in a user-defined way. In this paper, we propose a Map/Reduce approach to realize Web data mining.

II. WEB DATA MINING

Web data mining techniques are the result of a long process of research and product development. Web data mining is based on knowledge from the Web; it aims to discover useful information or knowledge from Web hyperlinks structure, page contents, and usage data [5]. Although Web data mining uses many data mining techniques, it is not purely an application of traditional data mining, due to the heterogeneity and semi-structured or unstructured nature of the Web data. Many new mining tasks and algorithms have been invented in the past decade. Based on the primary kinds of data used in the mining process, Web data mining tasks can be categorized into three types as shown in Fig. 1.

We can graphically define Web structure mining. The Web pages are represented as nodes, and hyperlinks are represented as edges. Basically, the graph shows the relationship between user and Web. The motive of Web structure mining is generating structured summaries about information on Web pages. The summaries show

the links of one Web page to another Web page. Traditional data mining does not perform such tasks, because there is usually no link structure in a relational table.

Web data mining is basically extracting the information on the Web. The process that accesses the information on the Web is Web content mining. Many pages are open to information access on the Web. These pages are the content of the Web. Searching the information and open search pages is also the content of the Web. Finally, accurate results are defined as the result pages of content mining. These tasks are similar to those in traditional data mining. However, we can also discover patterns in Web pages to extract useful data for many purposes, such as descriptions of products, or postings of forums. Furthermore, we can mine customer reviews and forum postings to discover consumer sentiments. These are not traditional data mining tasks.

Web usage mining is the discovery of meaningful pattern from data generated by client server transactions on one or more Web localities. A Web is a collection of inter-related files on one or more Web servers. It is automatically generated data stored in server access logs, reference logs, agent logs, client side cookies, user profiles, metadata, page attributes, page content, and site structure. One of the key issues in Web usage mining is the preprocessing of click stream data in usage logs in order to produce the right data for mining.

III. CLOUD COMPUTING AND MAP/REDUCE MODEL

Hadoop is an open source distributed computing framework [6], which is used for distributed processing of large data sets and designed to satisfy clusters scaled from a single server to thousands of servers. Hadoop is the most widely used cloud computing platform in recent years and has been adopted by major Internet companies and research institutions [7]. A Hadoop cluster is composed of two parts: the Hadoop distributed file system and Map/Reduce. Hadoop is the optimal choice to realize our approach. The Hadoop and Map/Reduce communities have developed a powerful framework for performing predictive analytics against complex distributed information sources [8]. So in this paper, our simulation experiment is designed based on it.

A. Cloud Computing

Cloud computing is a new term for a long-held dream of computing as a utility [9], which has recently emerged as a commercial reality. Cloud computing refers to both the application delivered as services over the Internet, and the hardware and system software in the data centers that provide these services.

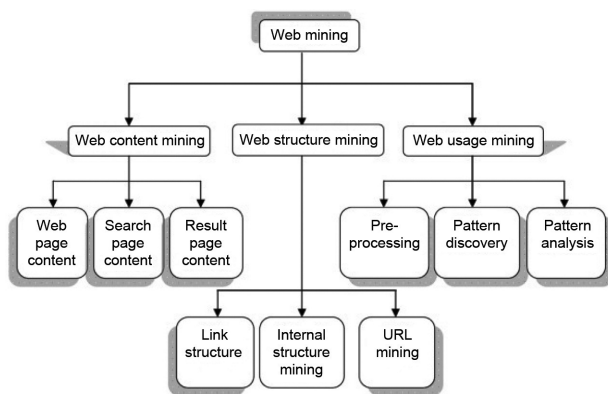


Fig. 1. Classification of Web data mining.

The main objective of cloud computing is to make better use of distributed resources and to solve large-scale computation problems [10]. For example, cloud computing can focus the power of thousands of computers on one problem, enabling researchers to do their work faster than ever. For Web data mining, we use this function of cloud computing to aim at mass data.

B. Map/Reduce Architecture

Map/Reduce is a programming model for processing large data sets [11], which was originally proposed by Google [12]. The framework is designed to orchestrate the work on distributed nodes, and run various computational tasks in parallel providing at the same time for redundancy and fault tolerance. Distributed and parallelized computations are the key mechanisms that make the Map/Reduce framework very attractive to use in a wide range of application areas that include data mining, bioinformatics, and business intelligence. Nowadays, it is becoming increasingly popular in cloud computing.

The Map/Reduce programming model is used for parallel and distributed processing of large data sets on clusters [13]. There are two basic procedures in Map/Reduce: Map and Reduce. Fig. 2 shows an execution overview.

Typically, the input and output are both in the form of key-value pairs. After the input data is partitioned into splits of appropriate size, the map procedure takes a series of key-value pairs and generates processed key-value pairs, which are passed to a particular reducer by a certain partition function; later, after data sorting and shuffling, the reduce procedure iterates through the values that are associated with a specific key and produces zero or more outputs.

IV. SYSTEM FRAMEWORK

In this paper, we built a new system framework to implement Web data mining. First, after the data is collected from the Web, the mass data on the Web must be filtered [14], cleaned, transformed, and combined into

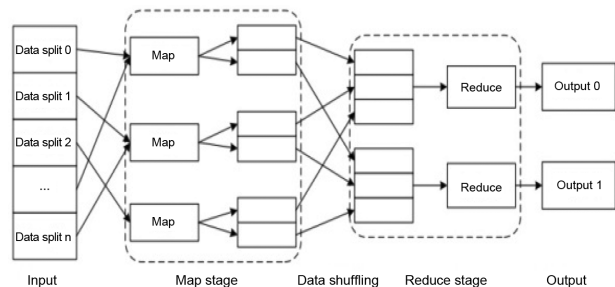


Fig. 2. Execution overview of Map/Reduce.

XML files; and then the files are saved on the distributed data nodes. Each file must be divided into small fixed-size blocks, copied and stored on different cluster node disks for backup. In this work, we define that each file can be copied two times and the two copies are stored onto two different cluster nodes. This system framework can solve the general problems of data missing in storage capacity expansion, and server failure caused by Web data mining [15]. Second, the master is responsible for controlling the entire works, creating the attached task to idle data nodes on the Web. The data node reports the status and result to the master. Then, the master is responsible for combining all the results to the client.

A. System Structure Overview

In our system framework, there are five types of nodes: client, master, name node, algorithm node, and data node as shown in Fig. 3.

The Client as a user submits a task and receives a result. The Master controls the whole workflow, creates the attached task, invokes the Map/Reduce scheduler to assign tasks to the data nodes, and regulates client accesses to the data. The name node divides the XML file into 64 M fixed-size blocks to idle data nodes, sends the IP address of the data nodes to the master, copies and stores the blocks into the other data nodes, and maintains a mapping table (the information of data blocks mapped onto the data node) in order to process write and read requests from the client, just like the name node stores the metadata of each XML file. But the real metadata is not stored on the name node, as it is just the IP addresses of XML files and the information of XML file copies, etc. The algorithm node stores the algorithms for supporting the requests from the master and sends the appointed algorithm to the data nodes. The data node stores the data blocks of XML files in its local disk and executes instructions.

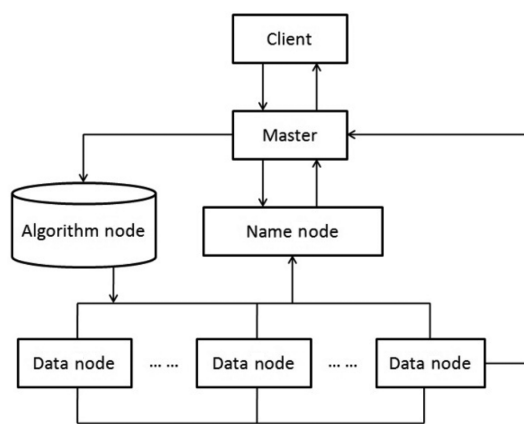


Fig. 3. System structure.

A data node periodically reports its status (idle, in progress, or completed) through a heartbeat message, and asks the name node for instructions. The heartbeat can also help the name node to detect connectivity with its data node. If the name node does not receive a heartbeat from a data node in the configured period of time, it marks the node down. The data blocks stored on this node will be considered lost, and the name node will automatically replicate those blocks of this lost node onto some other data nodes.

In this system framework, the data processing is executed and the data is stored on data nodes rather than transferring the executed data to the master. The master just receives the results, after the Reduce scheduler. So this means that mass data flow is not transferred in the network; and as a result, a lot of time can be saved.

B. A New Presented Algorithm

In an ordinary data mining algorithm, there are two steps: generate all frequent item sets and generate all confident association rules from the frequent item sets. The most important part is the frequent item sets in data mining. As for computing the frequent item sets, there is a method presented as: first generating frequent item set 1-item set L_1 , then generating frequent item set 2-item set L_2 , the algorithm will be continued until some value of K can be supported for L_k to be null set. When it needs to strive for L_k , candidate items C_k can be computed by L_{k-1} . Then by checking every item of C_k , we can get the item that belongs to L_k which can satisfy the minimum support threshold by the client defined. But since the web data is mass data, it will take a lot of time and space to determine C_k . So in this section, we present a new algorithm to confirm that all frequent item sets achieve high efficiency.

We use the processing method presented in [7] to get the processed document d_i . The document will be the content input, and the set of blocks is $\langle client, doc_idl \rangle$. By discovering the frequency of the feature in document, the output can be indicated to be $\langle t, \langle n, f \rangle \rangle$. After the first time of Map/Reduce scheduler, the key/value becomes $(key_1, value_1)$, $key_1=term_1$, $value_1=\langle n_1, f_1 \rangle \langle n_2, f_2 \rangle \dots$. This set would be executed by Map/Reduce as the data input, where key is critical value, and value is the local frequent item sets. In the second processing, by using the local frequent item set computed by the first step, we can compute the second result $(key_2, value_2)$, $key_2=term_2$, $value_2=\langle n'_1, f'_1 \rangle \langle n'_2, f'_2 \rangle \dots$, where we get an improved local frequent item sets again. Then, after the third time, we can compute the global frequent item set. This algorithm can improve the efficiency of data mining.

In this algorithm, we use the following formula:

$$T_{ij} = F_{ij} * e^{\log(n/t)}$$

T_{ij} is the threshold of the present data mining, F_{ij} is the

threshold of the past data mining, n is the number of feature extraction in this document, t is the frequency of feature extraction, and $e^{\log(n/t)}$ is the ratio of last frequent item set.

C. Algorithm Implementation

The implementation of the algorithm we presented can be described as:

```

The first time: Map
for all term1 ∈ client.doc do;
Frequency(t)=Frequency(t)+1;
Output (term1, record<client, doc_id1, Frequency(t)>);
The first time: Reduce
Input term1, record (<n1, f1><n2, f2> ... ..)
Build list R
for all record <n, f> ∈ record (<n1, f1><n2, f2> ... ..)
Append (R, <n, f>)
Find (R)
Output (term1, recordR)
The second time: Map
for all term2 ∈ term1 do
for <clientn, Frequency(t)> ∈ record R do h=h+1
T(t,n)=frequency(t)*elog(n/t)
Append (term2, T(<n1, t1><n2, t2> ... ..))
The second time: Reduce
for <client.doc1, T(t)> ∈ T(W)
if T(W)>Y
Append (T_new<n, T(t,n)>)
Find (T_new)
The third time: Map
for all term3 ∈ term2 do
for <client, Frequency(t)> ∈ record R do h=h+1
T(t,n)=frequency(t)*elog(n/t)*elog(n/t)
Append(term3, T(<n1, t1><n2, t2> ... ..))
The third time: Reduce
for <client.doc1, T(t)> ∈ T(W)
if T(W)=Y
Find(0)
    
```

Through the three time applications of Map/Reduce, we achieve the minimum support threshold.

Finally, the association rule can be generated using the threshold. The result will be delivered to the client.

V. SIMULATION EXPERIMENT

In this section, we design a simulation experiment to prove the possibility of the Web data mining we presented.

This simulation experiment testing environment is in a local area network; the platform is Hadoop [9], and consists of seven computers. The computer configuration is: Intel core duo 2.7 G CPU, 2 G DDR3 memory, and Linux operating system. One of the computers is assigned to be the master, one to the name node, one to the algorithm node, and the others to data nodes.

The name node divides the data into 10 sub-files and

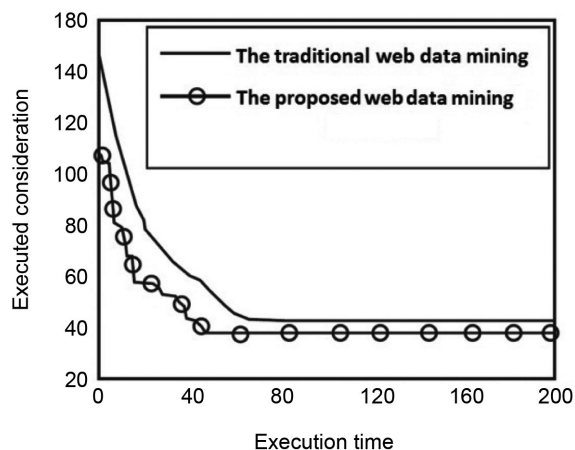


Fig. 4. Experimental result.

copies and stores the files to idle data nodes. The algorithm node stores the algorithms and sends the appointed algorithm to data nodes. The master controls the whole workflow. We presented the function of each node in Section IV above.

First test: We define that the experiment must implement traditional data mining, and record the execution time.

Second test: We define that the experiment must implement the data mining using the three time applications of Map/Reduce that we have presented. Then we record the execution time.

Finally, when we get the experimental result, we compare the executed consideration and execution time of the two tests. We can get the experimental result as shown in Fig. 4.

VI. CONCLUSION

Through comparing the two tests, the experimental result shows that this new approach can improve the execution efficiency and reduce the execution time. The new algorithm can work well and there is no association rule lost. It can be well used in business.

In this work, we noticed that we can aim to find a more accurate and faster approach for Web data mining, also based on cloud computing. We will keep on improving the algorithm we presented.

ACKNOWLEDGMENTS

This research would not have been possible, if we had not been provided with the opportunity to use the computer lab at Konkuk University, Seoul. Therefore, we would like to sincerely thank all the staff and lecturers who helped conduct this research and who made this paper possible.

REFERENCES

1. M. Armbrust, A. Fox, G. Rean, A. Joseph, R. Katz, A. Konwinski, L. Gunho, P. David, A. Rabkin, I. Stoica and M. Zaharia, "Above the clouds: a Berkeley view of cloud computing," Department of Electrical Engineering and Computing Sciences, University of California at Berkeley, *Tech. Rep. UCB/EECS-2009-28*, 2009.
2. C. H. Chang, M. Kayed, M. R. Girgis, and K. F. Shaalan, "A survey of Web information extraction systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1411-1428, 2006.
3. Wikipedia, "Cloud computing," http://en.wikipedia.org/wiki/Cloud_computing.
4. J. Dean and S. Ghemawat, "MapReduce simplified data processing on large clusters," in *Proceedings of the 6th Symposium on Operating System Design and Implementation*, San Francisco, CA, 2004, pp. 137-150.
5. R. Cooley, B. Mobasher, and J. Srivastava, "Web mining: information and pattern discovery on the World Wide Web," in *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence*, Newport Beach, CA, 1997, pp. 558-567.
6. Hadoop, <http://hadoop.apache.org>.
7. Y. Tao, W. Lin, and X. Xiao, "Minimal MapReduce algorithms," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, New York, NY, 2013, pp. 529-540.
8. M. J. Fischer, X. Su, and Y. Yin, "Assigning tasks for efficiency in Hadoop: extended abstract," in *Proceedings of the 22nd ACM Symposium on Parallelism in Algorithms and Architectures*, Santorini, Greece, 2010, pp. 30-39.
9. W. W. Lin, "An improved data placement strategy for Hadoop," *Journal of South China University of Technology: Natural Science*, vol. 40, no. 1, pp. 152-158, 2012.
10. C. Gong, J. Liu, Q. Zhang, H. Chen, and Z. Gong, "The characteristics of cloud computing," in *Proceedings of the 39th International Conference on Parallel Processing*, San Diego, CA, 2010, pp. 275-279.
11. D. Jiang, B. C. Ooi, L. Shi, and S. Wu, "The performance of MapReduce: an in-depth study," *Proceedings of the VLDB*, vol. 3, no. 1-2, pp. 472-483, 2010.
12. X. L. Lu and J. M. He, "Study on cloud storage model of Map/Reduce-based index data," *Journal of Ningbo University*, vol. 24, no. 3, pp. 29-33, 2011.
13. R. Lammel, "Google's MapReduce programming model - revisited," *Science of Computer Programming*, vol. 70, no. 1, pp. 1-30, 2008.
14. M. S. Chen, J. Han, and P. S. Yu, "Data mining: an overview from a database perspective," *IEEE Transaction on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 866-883, 1996.
15. Z. Bar-Yossef and S. Rajagopalan, "Template detection via data mining and its applications," in *Proceedings of the 11th International Conference on World Wide Web*, Honolulu, HI, 2002, pp. 580-591.



Wenzheng Zhu

Wenzheng Zhu received his B.S. degree from the Department of Computer Science of Wonkwang University, Korea, and his M.S. degree from the Department of Computer Science of Konkuk University, Korea, where he is currently working towards his Ph.D. degree. His research interests include cloud computing, OS, and information security.



Changhoon Lee

Changhoon Lee is a professor in the Department of Computer Science at Konkuk University. He received his B.S. degree from the Department of Mathematics of Yonsei University, Korea, and his M.S. and Ph.D degrees from the Department of Computer Science of KAIST, Korea. His research interests are in the areas of AI, OS, and information security.