Advance Journal of Food Science and Technology 5(10): 1331-1336, 2013
ISSN: 2042-4868; e-ISSN: 2042-4876

Submitted: May 29, 2013 Accepted: July 04, 2013 Published: October 05, 2013

Mobile Storage and Search Engine of Information Oriented to Food Cloud

Lifeng Wei and Bingmei Zhao
Shenyang Aerospace University, Shenyang 110136, China

Abstract: The aim of this study is to establish food cloud information search architecture. Food information search engine based on cloud computing architecture can not only achieve more personalized and intelligent search, but also can solve problems of data processing and storage centralization caused by mass of food and custom information. According to design idea of mobile search engine based on cloud computing architecture, the Map/Reduce algorithm and HDFS were thoroughly analyzed and researched. The cloud parallel storage technology under cloud computing architecture was introduced into mobile search engine for design and implementation mobile search engine under open-source Hadoop framework based on cloud computing architecture. The system achieves parallel storage of mass information and mass data to overcome unbalance problem of data centralization and overhead in storage server load caused by mass food data in traditional search engine, thus achieving high efficient mobile device search result and good user satisfaction.

Keywords: Food cloud, hadoop, internet of things, mobile search

INTRODUCTION

Food cloud is product combining Internet of Things and cloud computing in the food field. Where, Internet of things undertakes front-end information collection and cloud computing solve system traceability problem in the background. With the perfection of the traceability system, it is bound that mass data for storage. Cloud computing virtual and distributed storage manner can reduce cost as possible while storing large amount of data. Its unique delivery manner can also greatly improve data efficiency (Qu, 2012; Kumar, 2010). The search results with engine based on cloud computing is far beyond general engines. It also has shorter time and more fluent content. Currently, search engine especially for public food service is far from meeting needs of food information development, thus it is imperative to build cloud-based food network professional search engine. The cloud platform will change traditional information retrieval mode to provide integrated searching for consumers. Client sends retrieval request from vendors to cloud and resource scheduling center performs dynamically allocation and computation. The cloud computing ability can extend with requirements unlimited. Information search is no longer limited by hardware conditions, the speed and accuracy of which also greatly improved. At the same time, cloud computing has ability to regulate mass of information to provide regulatory authorities with comprehensive and unified food safety information. The integrity of information can be ensured. The DAMA service of cloud computing has unique advantage to protect interests of customers and fully meet individual needs of consumers. Application of cloud computing in food safety is not just to solve technical problems, but also shows development trend of food information.

The objective of the study is to describe file storage system, parallel storage system and parallel indexing system in the food information mobile searching engine.

TECHNIQUES AND FRAMEWORK

Core technologies: The most important parts in cloud computing architecture are open-source Linux operation system, Hadoop technology and etc. Hadoop is basic structure of distributed system. It was developed by Apache foundation, which takes full advantage of high-speed computation and storage. The core designs in Hadoop framework are Map/Reduce and Hadoop Distributed File System (HDFS).

Map/reduce algorithm: Map/Reduce is a programming mode that evolves from functional programming idea, which is used from large-scale data distributed and parallel computation. In the Map/Reduce mode, Map method mainly divides data to analyze data of interest to users. It marks a tag key on data belongs to same class. The system collects all data with same key tag to a machine and then calls Reduce method from monitoring and scheduling system to conduct data statistical analysis as well as merge. For example, there is a group of numbers (1, 2, 3, 4), each component multiple 2 (map operation) and the result is

Corresponding Author: Lifeng Wei, Shenyang Aerospace University, Shenyang 110136, China

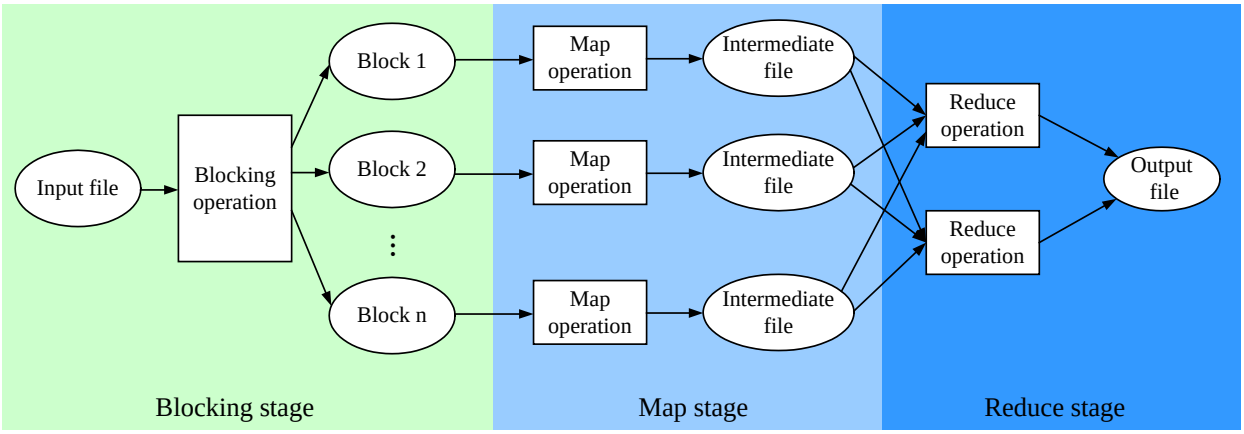

Fig. 1: Google map/reduce data flow diagram

(2, 4, 6, 8) (intermediate file). Then the average result is 5 after reduce operation.

There are one master and several workers in the Map/Reduce architecture of Google. The master is unique that responsible for unified task scheduling and maintain status information of each data node. There are many workers that used for distributed processing of data. The data flow in architecture is shown in Fig. 1.

The whole computation firstly divides input into M jobs and then allocates them to M maps from master to conduct map operation, then output to intermediate file. To a certain extent, it is written to disk and then the data is read into memory. The master integrates same key, namely divides R partitions and then distributes to R reduce workers. After Reduce received requests, it download needed all data according to resource list sorted by key. Then, it conducts reduce operation on them. According to different key grouping, reduce worker returns all key output pair groups to master. The master combines all returned data from reduce worker and generates final result.

HDFS cluster under hadoop framework: Hadoop implements Map/Reduce of Google. As general scheduler of Map/Reduce, *JobTracker* runs on master. *TaskTracker* runs on each machine to execute task. Meanwhile, HDFS of Hadoop achieves Google File System (GFS) of Google (Kumar, 2010; Lagerspetz and Tarkoma, 2011; Simoens *et al*., 2011; Tian *et al*., 2011). HDFS provides underlying support for distributed computing and storage. There are three important roles in the HDFS, namely *nameNode*, *dataNode* and client. *NameNode* can be seen as the manager in the distributed file system, primarily responsible for managing file system namespace, cluster configuration information as well as storage block replication and etc. *NameNode* will store meta-data of file system in the memory, including file information, block information corresponding to each file as well as location of each file block in *dataNode*. *dataNode* is basic unit of file storage, which stores block into local file system. It keeps meta-data of block and periodically sends information in block to *nameNode*. Client needs to obtain the application of the distributed file system.

As to a large file, Hadoop cut it into blocks whose size is 16 MB. These blocks are stored in each node in the form of ordinary file. In this way, data security and reliability can be achieved. For example, in case of query operation in the client, it only needs to interact with *nameNode* to obtained need file operation information and then communicate with *dataNode* to accomplish actual data transmission. When master initiates, it construct structural tree of file system by re-executing original operations. As the structural tree is in memory directly, the query efficiency is high.

System architecture design: Based on developed mobile searching engine and combining with advantages of cloud computing architecture and its core technologies, the study focuses on food information mobile searching engine based on cloud computing so that performance of mobile searching engine can be further improved.

Combining with technical theories of Map/Reduce and HDFS under open-source Hadoop framework, cloud computing architecture idea as well as cloud parallel data storage technology, the study gives food information mobile searching engine system based on cloud computing architecture. The searching engine system mainly consists of three parts, namely cloud information layer, could computing cluster system as well as user query dialog. The cloud computing cluster system is made up of collection layer, processing layer, indexing layer, query layer, interface layer as well as cloud computing file storage system and cloud computing monitoring scheduling system, the architecture of which is shown in Fig. 2.

The food information mobile searching engine based on cloud computing not only has personalized and intelligent searching characteristics of developed

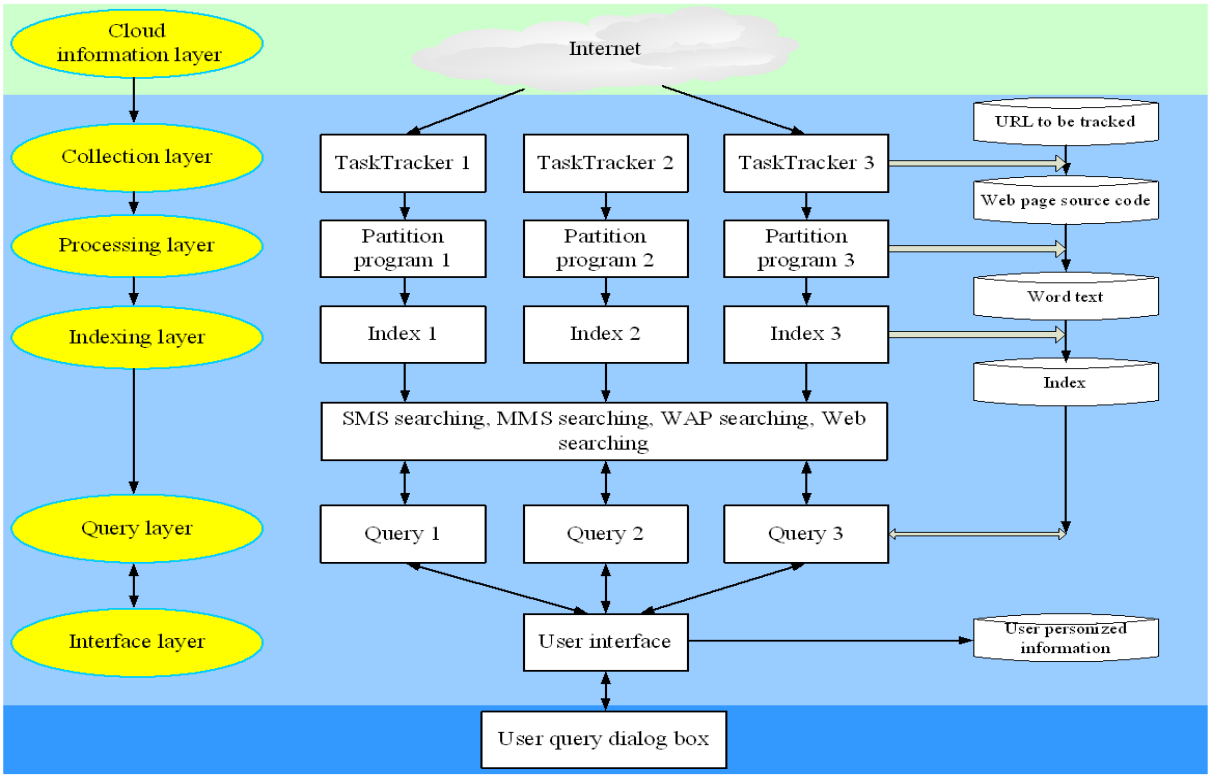

Fig. 2: Diagram of food information mobile searching engine system based on cloud computing

mobile searching engine, but also uses Hadoop as distributed framework, Lucene as indexing tool, cloud parallel storage technology as data file storage manner. The whole system design shows cloud computing architecture idea. In addition, the system can provide different kinds of searching service based on features of different subscriber users. The query layer determines searching service type according to form of needed data from users. The interface call user personalized information database and then perform personalized searching service.

In the distributed data processing on Map/Reduce using Hadoop, the core is control of cloud computing monitoring scheduling system on each layer. The cloud computing monitoring scheduling system *JobTracker* firstly divide searching task to several *dataNode*s based on given URL lists. The *TaskTracker* conducts page crawling according to new task and be added to task list as new. It is managed by *JobTracker* to arrange new Map/Reduce operation. The indexing layer mainly adds segmentation processing using indexing mechanism of Lucene.

The system receives user indexing input and hands it to interface layer for processing. *JobTracker* distributes indexing requirements to *dataNode* in the manner of task for separate indexing and then statute, sort and cache results. The most relevant records are returned to users with user selected personalized indexing type.

The food information mobile searching engine based on cloud computing has different system architecture with traditional mobile searching engine, the hardware architecture of which is also different. The hardware architecture mainly include main service control cluster, storage node cluster, application node cluster, computing node cluster, input device and output device. The traditional CPU and memory device are replaced by clusters.

IMPLEMENTATION AND RESULTS

File storage system implementation: As to any system, the organization structure of underlying file system is the most important infrastructure. The research topic of mobile searching engine is also how to implement good file organization structure (Kumar and Lu, 2010). Referring to GFS and HDFS, the file storage system based on cloud computing was designed using idea of cloud computing architecture and cloud parallel data storage technology.

Simple and fast file structure uses four tables, namely memory table, folder table, file table and block table. To solve problems of unbalance and transmission bottleneck in traditional storage server, the system uses cloud parallel data storage technology. It means the system works in parallel manner, which takes full advantage of each node in backbone server cluster. In case of file upload and download, cloud computing

monitoring scheduling system interacts with main program according to file information, block information in the database and return storage node information of system file to achieve parallel transmission file and data. It not only implements load balance of server, but also improves transmission rate.

Food information mobile searching engine file storage system based on cloud computing refers storage method of GFS and HDFS, namely using data copy storage method. It assumes computation element and storage fails, so it maintains multiple copies of data to ensure re-distribution on failure node. The copy number can also be set by adjusting parameters so that file storage system has high fault tolerance. For example, each file copy of system is stored in different working node *dataNode*. They are managed by cloud computing monitoring scheduling system *nameNode*. In this way, even one working node *dataNode* fails, the file on it can also be read normally (Song and Su, 2011; Liang *et al*., 2012). Especially considering unbalance load and transmission bottleneck in storage server, the system not only set copy for working node *dataNode*, but also copy *nameNode* so that the file storage system be more reliable.

In the mobile searching engine file storage system based on cloud computing, *nameNode* can be seen as manager of distributed file system. The *nameNode* will store meta-data of file system into memory. *DataNode* is basic unit for file storage. It saves block in the local file system and keeps meta-data of block. Meanwhile, it periodically sends all existing block information to *nameNode*. Client is application program that needs to access to distributed file system. In case of an operation of client, the command is not immediately sent to *nameNode* but cache locally until file size reaches certain level. When, client will notify *nameNode* to request for executing operation, so as to improve efficiency of file read and write.

Implementation of parallel indexing system: The parallel indexing system is the basis that cloud computing cluster system in food information mobile searching engine can quickly find the needed information. Construction of inverted indexing in mobile searching engine can be completed by Map/Reduce in Hadoop framework. In case of large data processing, Map/Reduce effectively solve bottleneck of performance decline in centralized system with following methods:

- The map() function analyzes a snapped positive sorted indexing document and processes it to generate (word, doc ID) sequence.
- The reduce() function conducts merge arrange processing on all sequence groups in same word to generate (word, list (doc ID)).

In this way, all output groups produce inverted indexing document.

In the construction of inverted indexing document, splitter divides large-scale positive indexing document dataset and then distributes to different machine for map processing to generate own inverted indexing document. At last, reduce merge these indexing documents to generate unified inverted indexing document, thus to solve bottleneck of efficiency in centralized processing with distributed way. Where, the input of map operation is each document to output merges of each word in the input document to intermediate file. The input of reduce operation is the word and location information as well as sequence of word sequence. The study number and location information can be obtained according to each word.

Based on through research on inverted indexing technology, the study proposed improved schema, namely aggregate address inverted index. Aggregate addresses inverted indexing technology can effectively determine multiple continuous keyword indexing. It uses words and phrases as index entries and designs efficient indexing structure. Aggregate addresses inverted indexing technology can also expand each indexing database to different storage node to facilitate using parallel distributed database in the system and improve concurrent performance. It can not only ensure high data density, but also enhance flexibility of indexing as well as reduce difficulty of indexing technology, while greatly improving efficiency of indexing.

The idea just divides index body into two parts as following:

- Part index of each word independently to constitute a database that can be called as indexing dictionary. It is mainly used to quickly locate to position of corresponding keywords in the inverted list that saved in memory, which alleviate I/O burden to some extent.
- The inverted list marked by file name, the content of which is inverted file corresponding to keywords. The inverted file is stored in system disk of indexing classification according to some mechanism. The system saves indexing type on different node with distributed storage strategy so that it can quickly search needed indexing information/it decrease problem that too heavy I/O burden caused by too large inverted list.

Implementation of parallel storage: The cloud storage system plays a supporting role in the cloud indexing system of food information mobile searching engine. It realizes separately storage of program and data as well as cloud of data in each layer. The main body to implement whole mobile searching engine includes cloud collection layer, cloud processing layer as well as data interaction between cloud indexing layer and cloud query layer. Cloud storage system uses parallel storage manner to improve storage efficiency and decrease complexity of mass data management. It

sees user as core to improve application performance and availability (Yang *et al*., 2012; Zhao *et al*., 2012). Regardless of user terminal, regardless of the user where cloud retrieval cluster system of the mobile search engine recognition only its logical mirror, which makes cloud storage system can focus on the management and operation of the data but not worrying about re-configuration requires data loss and other problems caused by the shutdown, so as to ensure normal use.

In the cloud collection layer, the system will use URL repository to extract URL information database and then store original page document information. After word partition in cloud processing layer, the system divides inputted original page file into individual words by/division. The final word partition result will be stored in separate word database, which is complemented by specific Word out File according to specific format storage. In the cloud indexing layer, system generates file index entries for the input sequence through the indexing process to improve the Finder Find the location of the original data storage performance to extract metadata and generate summary information. In the cloud query layer, system firstly determines searching service type by the query in querying layer according to form of needed data from users and then select personalized searching service. The system to find the entry as a keyword search read a list of documents containing the entry and press the number of occurrences in the document can be sorted from the index library. The upload and download algorithms of massive data are implemented as follows:

- **Parallel upload algorithm:** The specific steps of parallel upload algorithm are as following:

Step 1: **Send upload request:** Select local file to be uploaded. In case of upload operation, firstly send file information as file name, file size of upload file and other information as user ID to main server

Step 2: **Partition computation and node allocation:** After server received file information, firstly conduct partition computation according to file size. The study computes based on size of 5 MB/block. Then allocation file block to nodes according to node information provided by monitoring system. Finally insert file information and block information as file ID, block number, block name, block size and corresponding node IP to database

Step 2.1: Partition computation

Step 2.2: Balance allocation

Step 2.3: Insert database

Step 3: **Response to upload request:** The server send file blocks information to client in format of XML file

Step 4: **Upload file blocks in parallel:** The client establishes a file block queue for each storage node and uploads file blocks to corresponding node in parallel

Step 5: **Upload successful confirmation message:** After storage node received a file block, it sends a confirmation to server. The server will change file block status in database to 1 to show successful uploading. After all file blocks been successfully uploaded, the server change file status in database to 1

Step 6: **Single node failure:** When the server finds some node fails from real-time monitoring information, it immediately re-allocates part of file in uploading

- **Parallel download algorithm:** The specific steps of parallel download algorithm are as following:

Step 1: **Send download request:** Select file to be downloaded and send a download request to server, including file ID to be downloaded and user ID.

Step 2: **Find file block information:** After the server received download request, find file block information from database.

Step 3: **Response to download request:** Server send file block information to client in the format of XML.

Step 4: **Download file block in parallel:** The client creates a thread for each storage node according to received block information and download file block to local temporary folder in computer.

Step 5: **File integration:** After client downloaded all file blocks, integrate them into a complete file and delete the file blocks.

Step 6: **Single node failure:** If the server finds some node fails through real-time monitoring information, it immediately re-allocates part files in downloading.

The design can avoid mass of food data be centralized and unbalance overhead in the cloud storage server. In this way, the food information mobile searching engine based on distributed and cloud computing architecture can re-use Hadoop framework to store data generated in different phases and perform further effective data management as well as personalized searching according to user analysis.

CONCLUSION

Currently, the mobile searching engine based on cloud computation architecture has become important direction in the searching engine filed. The study mainly focuses on cloud computation architecture based on current technologies. Combining with extraordinary advantages and its core technologies, food information searching engine based on cloud computing was designed so that the performance of searching engine

can be further improved. The research emphasis lies on architecture of food information searching engine as well as implementation of file storage system, parallel indexing system and parallel storage system. On the basic infrastructure, the Map/Reduce algorithm and HDFS were analyzed and researched in detail. The system can not only achieve more personalized and intelligent searching, but also implements mass food information processing and parallel storage. It solves problems of data centralization and unbalance overhead in storage server caused by mass information. Meanwhile, an effective and reliable food information searching platform based on Internet and mobile communication was established to improve searching performance and provide better service for users, so as to obtain high effective searching result and higher customer indexing satisfaction degree.

REFERENCES

Kumar, K., 2010. Cloud computing for mobile users: Can offloading computation save energy. Computer, 43(4): 51-56.

Kumar, K. and Y. Lu, 2010. Cloud computing for mobile users. Computer, 83(99): 1.

Lagerspetz, E. and S. Tarkoma, 2011. Mobile search and the cloud: The benefits of off-loading. Proceedings of 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), pp: 117-122.

Liang, D.H., L. Peirchyi and D.S. Liang 2012. Risk management of land use on cloud computing. J. Converge. Inform. Technol., 7(1): 122-129.

Qu, Z.X., 2012. Research on semantic processing for internet of things based on cloud computing. Int. J. Adv. Comput. Technol., 4(16): 339-346.

Simoens, P., F.D. Turck, B. Dhoedt and P. Demeester, 2011. Remote display solutions for mobile cloud computing. Computer, 44(8): 46-53.

Song, W.G. and X.L. Su, 2011. Review of mobile cloud computing. Proceedings of 2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN), pp: 1-4.

Tian, Y., B. Song and E.N. Huh, 2011. Towards the development of personal cloud computing for mobile thin-clients. Proceedings of 2011 International Conference on Information Science and Applications (ICISA), pp: 1-5.

Yang, J., J.L. Jing, D.L. Zhang, L. Chen and Y.H. Yuan, 2012. Dynamical spectrum allocation of WSNs oriented to cloud computing. J. Converge. Inform. Technol., 7(2): 262-268.

Zhao, J.F., W.H. Zeng, M. Liu and G.M. Li, 2012. A model of virtual resource scheduling in cloud computing and its solution uses EDAs. Int. J. Adv. Comput. Technol., 6(4): 102-113.