International Journal of Multimedia and Ubiquitous Engineering
Vol.10, No.10 (2015), pp.375-384
http://dx.doi.org/10.14257/ijmue.2015.10.10.37

Study on the Distributed Crawling for Processing Massive Data in the Distributed Network Environment

Chang-Su Kim

PaiChai University, 155-40, Baejae-ro, SeoGu, DaeJeon, Korea
ddoja@pcu.ac.kr

Abstract

Due to the development of IT, distribution of smart phone, and an increase of use of SNS, various types of contents are being produced and consumed in Internet. Therefore, information searching technology has become important due to a sharp rise in data. However, information searching technology requires much of background knowledge and hence has been recognized as what was difficult to access to. Issues with previous search engine were how many of qualified personnel with background knowledge along with huge amount of development expenses were required. Therefore, search engines have been recognized as what was exclusively possessed by leading IT companies or specialized organizations. This study is intended to suggest a search engine with an index structure for making it convenient to effectively search information by distributed crawling massive amount of websites and web-documents in the distributed environment. Search engine suggested in this study has been realized by Hadoop structure for supporting the distributed processing.

Keywords: *Distributed Crawling, Hadoop, Massive Data, MapReduce, Nutch, Search Engine, Solr, YARN*

1. Introduction

Due to the development of Internet and an increase of use of smart devices, the scale of web has been rapidly rising. Therefore, users tend to find it difficult to find data they want in other massive data. In addition, importance and necessity of a search engine that users are able to internally use in the shopping malls or organizations have been raised [1].

Because of the development of IT, distribution of smart devices, and an increase of use of SNS (Social Networking Service), various types of contents are being produced and consumed in the Internet. Therefore, information searching technology has become important due to a rapidly increase of data. However, information searching technology requires much of background knowledge and hence has been recognized as what was difficult to access to. Therefore, it has been recognized as technology that was exclusively possessed by leading IT companies or organizations specializing in searching technology.

Searching engine is a program for searching sites or documents existing in the web and is divided into web-crawler of websites or web-documents, indexer for indexing process, and searcher for representing the results by comparing contents in the indexing according to the request from the search of users [1].

Issues with previous search engine were how many of qualified personnel with background knowledge along with huge amount of development expenses were required. Therefore, search engines have been recognized as what was exclusively possessed by leading IT companies or specialized organizations. However, emergence of Lucene has made it feasible for small scaled developers to develop search engine by using Lucene and hence lowering the barrier to entry [1-3].

ISSN: 1975-0080 IJMUE

In addition, it became feasible for managers with less amount of development knowledge to establish internal search engine by exploiting open source frameworks including Nutch, Solr, and Hadoop [1-3].

This study is intended to suggest a search engine with index structure for making it convenient to effectively search information by distributed crawling mass amount of websites or web-documents. Search engine suggested in this study has been realized based on Hadoop for supporting distributed process.

2. Related Researches

2.1. Hadoop

Hadoop 2.x has supplemented poorly designed structure and node bottleneck phenomenon of Hadoop 1.x and made it feasible to constitute previously limited Namenode with multiple Namenodes for supporting HDFS (Hadoop Distributed File System) federation [1].

MapReduce of Hadoop 2.x is called as either YARN (Yet Another Resource Negotiator) or Map-Reduce (MRv2), and job tracker has divided a major function of resource management and the job life cycle control into a new component [1,3]. First of all, resource manager deals with the entire cluster resources, assigns resources to applications in need of work, and collects them when the work is done. In addition, application master served as a role of scheduling and adjusting the work of each of the applications. Applications are performed after being distributed into multiple nodes, and there exists a node manager in each node.

2.2. YARN

YARN is comprised of two types of Daemon; resource manager and node manager. There is one resource manager in the master server for managing the entire cluster, and there is a node manager in each node. In addition, there exists an application master in each of the programs. Previous map-reduce has only served as a role of mapping and reducing work in the program. However, YARN is capable of performing both mapping and reducing work and also of creating distributed program in other purposes. Figure 1 is the structure of YARN of Hadoop [1-4].

2.2.1. Structure of Resource Manager: Resource manager is comprised of two main components called scheduler and program manager (application manager). Scheduler manages the conditions of resources of node manager and assigns insufficient resources. In addition, it does not examine or monitor the status of a program but only serves as a role of scheduling work. It does not re-start the program occurring with an issue due to an error of programs or hardware but only deals with the process of functions related to resources required by a program (CPU, Disk, and network *etc.*).

Application manager performs the application master for particular tasks in the Daemon of node manager and controls the status of application master [5].

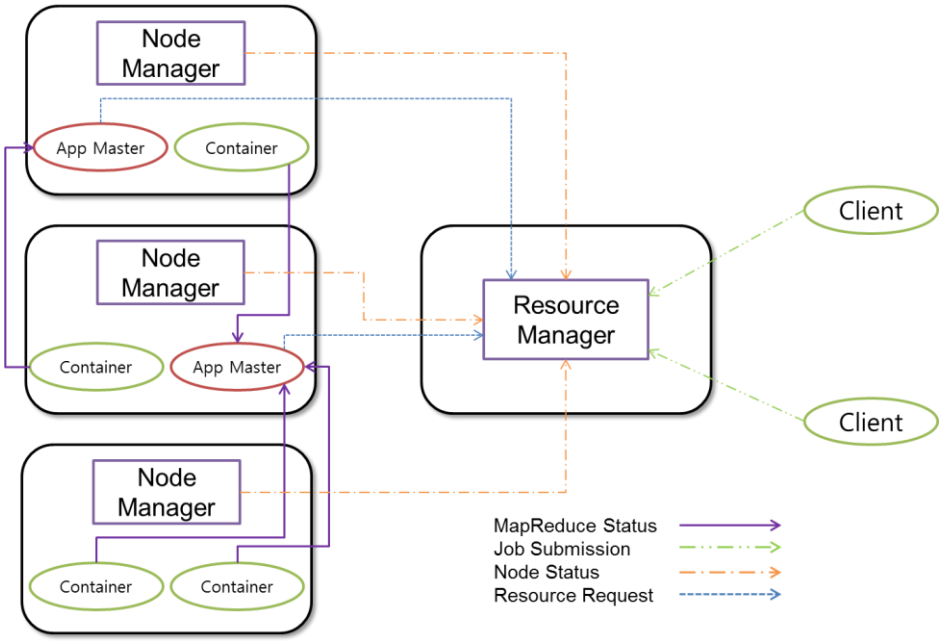

Figure 1. Structure of YARN of Hadoop

2.2.2. Structure of Node Manager: There is one node manager in each node (computer). It monitors the amount of consumed resources in the application container and serves as a role of notifying related information to resource manager. Node manager is comprised of program master (application mater) and application container. First of all, application master serves as a role of master on one program, is assigned with appropriate application container from scheduler, and monitors/manages a status of program performance. Application container represents a resource assigned to the program.

2.3. Lucene

Lucene indicates highly functional information searching IR (Information retrieval) library. IR indicates a course of searching documents or meta information related to documents. When using Lucene, it is feasible to conveniently add an information searching function on the application to be developed [2-6].

As for the strength of Lucene, it not only makes it feasible to easily and rapidly use full-text index and searching function by utilizing the core library but also continuously adds various types of library that an advanced function is to be used.

Lucene has been used as a search engine in various projects and applications and also for two main open source search applications named Solr and Elasticsearch. As Solr has recently been transferred to the Apache Software Foundation, it was integrated to Lucene and developed together. Figure 2 indicates the structure of Lucene/Solr [2, 7-8].

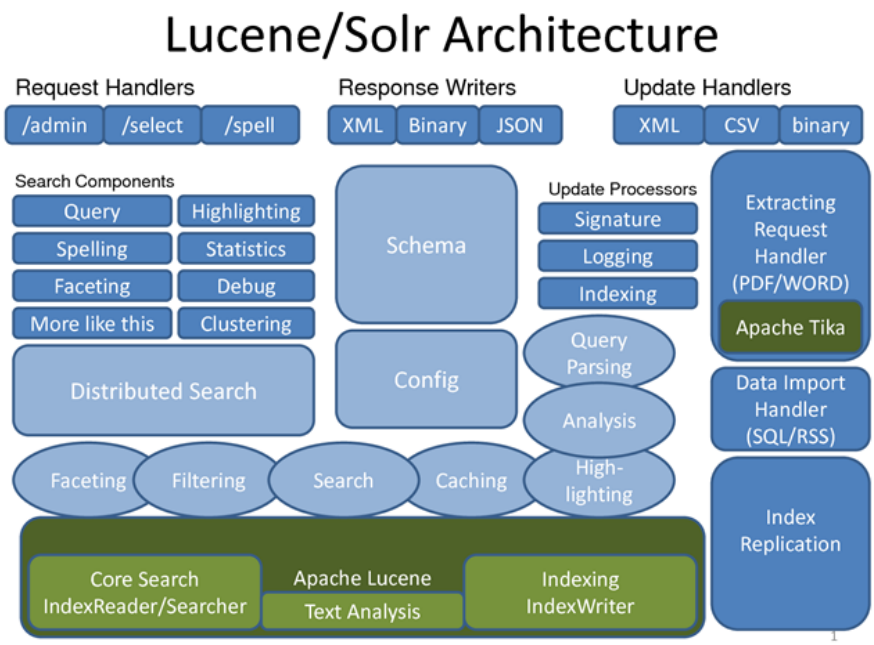

Figure 2. Structure of Lucene/Solr

As big data has recently been receiving much attention, Lucene and search engine has become in the limelight. Lucene is not a search application but only provides a library for offering index and search function.

As for a brief explanation of index and search procedures of application, Lucene extracts original files into an indexing-available text calling Lucene API and adding the converted text to the index. When comprising the index, search words requested by a user in the user interface of the search application are converted to Lucene query receiving a result by performing prepared Lucene query on the index and outputting the result on the screen. Solr and Elastic search are relevant to the representative search application.

2.4. Zookeeper

When designing the distributed system, one of the serious issues is about how to share information between the systems and related to necessity of checking the status of servers in the cluster, and also a problem of processing the lock for synchronization between distributed servers [3-4, 9-10].

A solution of such issues is called coordination service. Apache Zookeeper is the representative example of it. This coordination service maintains an important status information or setup information in the distributed system. Therefore, errors in coordination service cause failure of the entire system. This is the reason why high availability is required through duplex configuration.

Zookeeper well provides such features that it has been widely used for already well-known distributed solution. Representative examples include a type of NoSQL, Apache HBase, and also Kafka, a massive distributed queue system.

Since distributed system is designed for the purpose of coordination, data access shall be swiftly provided and also equipped with an ability of coping with failure with its own function. Therefore, Zookeeper self-provides clustering function and makes it feasible to fail over/fail back without losing data at the time of failure. Figure 3 is a structure of server of Zookeeper and client [3-6].

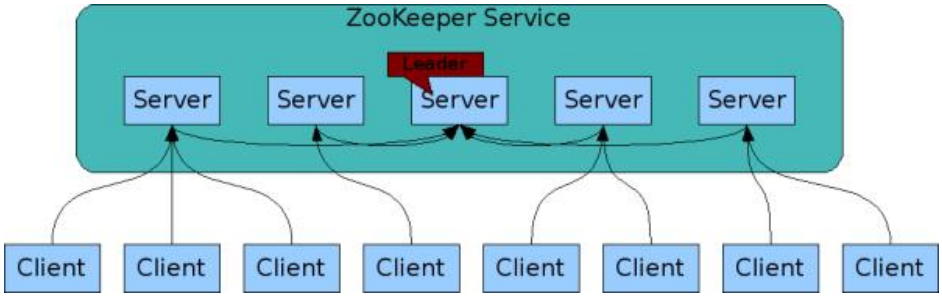

Figure 3. Structure of Server of Zookeeper and Client

3. System Design and Experiment

Major part of the search engine is data crawling, and the suggested design of search engine has been created by considering it. Data crawling is categorized into a method of small scale crawling and a method of massive data crawling in a large scale. The structure of distributed crawling search engine is shown in the Figure 4.

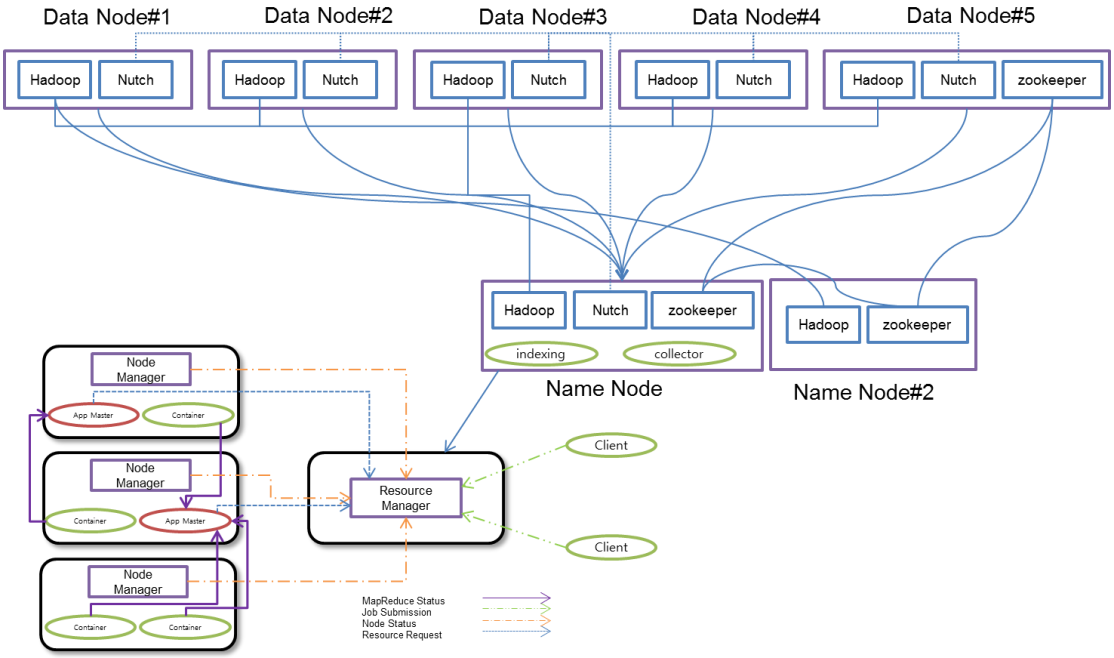

Figure 4. Structure of Distributed Crawling Search Engine

As shown in the Figure 4, physical environment where Nutch is to be operated based on MapReduce is required to effectively crawl the massive data. In addition, three Zookeeper servers are organized for high availability of the server while comprising of Namenode and Secondary Namenode to cope with occurrence of failure. In addition, five Datanodes were organized to process and save data.

Solr was installed in the Namenode making it feasible to proceed indexing work and search for crawled data. In addition, scope of usage of Nutch has been classified making it possible to perform data in a local area or small scale only with Namenode and crawl the data.

3.1. Environment Setup

Frameworks used as a search engine operate according to Java. Therefore, JVM environment shall be setup. JVM is classified into open JDK and Oracle JDK, and this study has used Oracle JDK 7. In addition, it is required to setup JVM and environment variables of frameworks in the .bashrc that is used as Linux an environment variable in each of the frameworks. Table 1 indicates Linux environment variables.

Table 1. Indicates Linux Environment Variables

```
#Java
export JAVA_HOME=/usr/lib/jvm/java-7-oracle
export PATH=$PATH:$JAVA_HOME/bin
#Hadoop
export HADOOP_HOME=/home/hadoop/tools/hadoop-2.2.0
export HADOOP_PREFIX=/home/hadoop/tools/hadoop-2.2.0
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
#Native_Path
export HADOOP_COMMON_LIB_NATIVE_DIR=
${HADOOP_HOME}/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native“
#Zookeeper
export ZOOKEEPER_HOME=/home/hadoop/tools/zookeeper-3.4.6
export PATH=$PATH:$ZOOKEEPER_HOME/bin
#Nutch
#export NUTCH_HOME=/home/hadoop/tools/apache-nutch-1.7
export NUTCH_HOME=/home/hadoop/tools/apache-nutch-1.8
export JAVA_HOME=$(readlink -f /usr/bin/java |
sed "s:bin/java::")
#Solr
export SOLR_HOME=/home/hadoop/tools/solr-4.8.1/solr/example/solr
```

Suggested search engine constitutes distributed environment by using multiple computers. Therefore, each of the computers communicates with one another by using SSH. In addition, computers connected to the network distribute, process, and save data.

3.2. Server Environment Setup

In the Hadoop’s distributed environment, low-priced or general computers are used instead of expensive servers. Therefore, it is exposed to frequent breakdown. Failure of one computer leads to the suspension of the entire computers. Therefore, Zookeeper is used to prevent such a phenomenon and also damage from it. Zookeeper server shall be comprised of odd numbers, and this study has used up to 3 servers, and remaining parts were used with clients. Table 2 is the list of server environment variables.

Table 2. List of Server Environment Variables

```
# The number of milliseconds of each tick
tickTime=2000
# synchronization phase can take
initLimit=10
# sending a request and getting an acknowledgement
syncLimit=5
# example sakes.
dataDir=/home/hadoop/tools/zookeeper-3.4.6/data
# dataLogDir=/home/hadoop/tools/zookeeper-3.4.6/log
# the port at which the clients will connect
clientPort=2181
# increase this if you need to handle more clients
# maxClientCnxns=60
# the port at which the clients will connect
# clientPort=2181
maxClientCnxns=0
maxSessionTimeout=180000
server.1=192.168.0.3:2888:3888
server.2=192.168.0.4:2888:3888
server.3=192.168.0.5:2888:3888
#server.4=datanode2:2888:3888
#server.5=datanode3:2888:3888
#server.6=datanode4:2888:3888
#server.7=datanode5:2888:3888
```

3.3. Experiment

Suggested web crawler and indexing unit of search engine, operation process and functions of searching unit are explained in this chapter.

3.3.1 Web-Crawling Based on Nutch: Web crawler used in the search engine uses Nutch 1.7 and Nutch 1.8. Nutch is separated into the directory that is performed by single node and distributed nodes.

Performance of Nutch with a single node proceeds web-crawling on the small scale, and MapReduce is used to perform Nutch with distributed nodes. Crawling DB of Nutch performed by MapReduce is saved in HDFS.

3.3.2. Solr Indexing: Solr indexing is only performed in the Nutch folder. Solr status is confirmed by the web-browser as shown in the Figure 5.

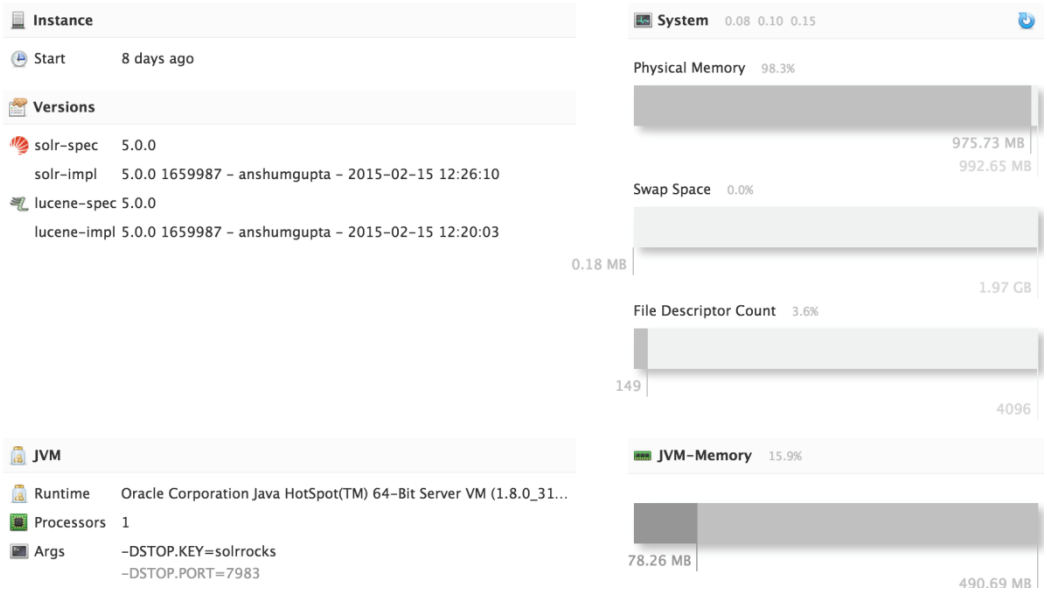

Figure 5. Solr Status

Crawl DB indexed with Solr is searched as shown in the Figure 6, and it is feasible to search with multiple languages and also in a particular file format.

Figure 6. Search with Multiple Languages

4. Conclusion

As the number of website user has been increased, the scale of web started significantly increasing. Because of this phenomenon, importance of search engine as a technology for effectively searching information a user wants to seek for in massive data has been raised. Search engines are classified into the Internet search engine applied on the websites, internal search engine applied to internal space of Internet, and log search engine for collecting log data.

This study has stated how to realize and use search engines for searching massive data. Realized search engine has used the Java frameworks developed as an open source that it represented a high level of transferability and also possibility of development with lower costs. In addition, experiment was conducted to verify the function of suggested search engine and verified appropriateness of the study.

Suggested search engine has used various types of Java frameworks making it feasible to be operated by independent modules and also be connected with other frameworks as shown in the way of realization suggested in this study. Realization of suggested search engine is possible with the least amount of effort and background knowledge about the search engine. Therefore, it is anticipated for suggested search engine to be useful for developers who realize search engine with prototype as well as managers who are in need of internal search engine.

References

[1] H. Karambelkar, "Scaling Big Data with Hadoop and Solr", Packt Publishing Ltd, **(2013)**.
[2] Solr - Apache Lucene, http://lucene.apache.org/solr/.
[3] ZooKeeper, https://zookeeper.apache.org/doc/trunk/zookeeperOver.html.
[4] ZooKeeper, Apache. "What is ZooKeeper.", http://zookeeper.apache.org .
[5] G. Trey, T. Potter, and Y. Seeley, "Solr in action, Manning", **(2014)**.
[6] Nayrolles, Mr Mathieu, "Mastering Apache Solr: A practical guide to get to grips with Apache Solr", inKstall, **(2014)**.
[7] M. Maged, "Scale-up x scale-out: A case study using nutch/lucene", Parallel and Distributed Processing Symposium, 2007. IPDPS 2007, IEEE International. IEEE, **(2007)**, pp. 1-8.
[8] H. Allan, and M. Najork, "Mercator: A scalable, extensible web crawler", World Wide Web 2.4, **(1999)**, pp. 219-229.
[9] B. Liang, W. Guangqiong and D. Xiaoqing, "Research and application of full-text retrieval model based on Lucene", Microcomputer & Its Applications, vol. 1, **(2007)**.
[10] H. B. Lee, "A vertical search engine for school information based on Heritrix and Lucene", Convergence and Hybrid Information Technology, **(2011)**, pp. 344-351.

Author

Chang Su Kim, He received his B.S., M.S., and Ph.D. degrees from the Department of Computer Engineering at Paichai University, Korea, in 1996, 1998, and 2002, respectively. From 2005 to 2012, he worked for the Department of Internet at Chungwoon University as a professor. Since 2013, he has worked in the Department of Computer Engineering at Paichai University, where he now works as a professor. His current research interests include multimedia document architecture modeling, web 2.0, and the semantic web.

