The Study and Application of Hadoop across Multiple Clusters

Shengtao Sun
School of Information Science and Engineering
Yanshan University
Qinhuangdao, P. R. China
e-mail: ysusst@163.com

Aizhi Wu
College of Vehicles and Energy
Yanshan University
Qinhuangdao, P. R. China
e-mail: ysuwaz@163.com

Xiaoyang Liu
Data Industry Research Institute Limited Company
Data Valley
Qinhuangdao, P. R. China
e-mail: xyliu@163.com

Abstract—Hadoop is a widely applied tool for large-scale data-intensive computing in big data, but it can only be implemented on single cluster environment. In this paper, we focus on the application of Hadoop across multiple clusters and dedicate to solve the key problems of data sharing and task scheduling among clusters. A hierarchical distributed computing architecture of Hadoop across multiple clusters is designed. The virtual HDFS and job adapter are proposed to provide global data view and task allocation across multiple data centers. The job submitted by user to this platform is decomposed automatically into several sub-jobs and then allocated to corresponding cluster by location-aware manner. A prototype based on this architecture is presented and currently applied in the distributed spatial information processing across spatial data centers.

Keywords-multiple clusters; Apache Hadoop; hierarchical distributed computing; virtual HDFS; job adapter

I. INTRODUCTION

The MapReduce paradigm has emerged as a highly successful programming model for large-scale data-intensive computing applications [1]. However, current Hadoop implementations cannot be applied to distributed data processing across data centers. Nowadays, data-intensive computing typically uses modern data center architectures and massive data processing paradigms [2]. The requirements for data-intensive analysis of scientific data across distributed clusters or data centers have grown observably in recent years. This research is devoted to a study on the distributed data processing model across multiple data centers.

In this paper, we try to improve the capability and flexibility of Hadoop. A new hierarchical distributed computing architecture of Hadoop across multiple data centers is proposed. Java Socket is used to transfer these tasks to the corresponding cluster and data center, where the data processing in supposed to be run without data moving. Virtual HDFS (Hadoop Distributed File System) is presented to provide the catalog service of these data centers, which can record the meta-data and access path of all data. The design and application of this architecture are illustrated in this paper and we hope this study may enlighten the attention of Hadoop global implementation.

The rest of this paper is organized as follows: Section 2 discusses related works of our research, Section 3 presents the design of a hierarchical distributed computing architecture of Hadoop, and a prototype system based on this architecture is shown in Section 3. Finally, Section 4 concludes the paper and points out the future work.

II. RELATED WORKS

Cloud computing is a set of network enabled services to allow the centralized data storage and online access to computer services or resources [3]. It provides scalable, quality guaranteed, normally personalized, convenient computing infrastructure on demand. The MapReduce paradigm and its open sourced implementation—Hadoop have been recognized as a representative enabling technique for Cloud computing [4].

In distributed computing frameworks based on Hadoop, one job committed to master node (name node) may be divided into several same tasks and then run on several slave nodes (data nodes). The data of this job is stored in HDFS (Hadoop Distributed File System) which is distributed to data nodes of the same cluster. In the situation of distributed computing crossing clusters, the data copying from multiple sites to the computing center is the traditional method.

But the copy process is tedious and inefficient between global distributed data centers through wide-area network, especially in era of big data. Moving the computation instead of moving the data is the proper way to tackle this problem [5]. By using data parallel processing paradigms on multiple clusters, simulations can be run on multiple computing centers concurrently without the need of copying the data. In G-Hadoop [6], Gfarm and Torque were adopted to manage data files and resources for clusters. In P2P-MapReduce [7], the adaptive MapReduce framework provided a more reliable MapReduce middleware in dynamic Cloud infrastructures. A Federated MapReduce [8] provided a transparent way to run original MapReduce jobs across multiple clusters without any extra programming burden.

The Apache Hadoop MapReduce implementation may be upgraded for a multi-cluster environment with a decision algorithm that would prefer local computers to the remote [9]. The research on Hadoop across data centers or clusters is rare, but we believe this issue is worthwhile. In our works, we try to employ Hadoop in the application of distributed spatial data processing with high performance.

III. THE DESIGN OF HIERARCHICAL DISTRIBUTED COMPUTING ARCHITECTURE OF HADOOP

With the rapid development of manufacture technology and observation technique, human can obtain massive spatial data of variant types from multiple sources. Researchers have devoted to collecting variant spatial data and providing spatial information service. Many spatial data centers have been set up all over the world for different purposes and diverse services. The processing and managing of spatial data must face the status of multiple data centers.

In the design and implementation of distributed computing architecture of Hadoop across multiple spatial data centers, there are mainly three problems need to be discussed and tackled. (1) How to obtain the view of all data in multiple data centers, which may decide the allocation of computing task to proper data center, (2) How to decompose the job into tasks and dispense them to corresponding clusters, which can synergistically control each Hadoop execution engine in multiple clusters, (3) How to deliver tasks transparently across different administrative domains, which need efficient and general way to keep communication among different clusters. In this paper, we try to employ the hierarchical methodology [10] and propose a hierarchical distributed computing architecture based on Hadoop across multiple clusters (named HDC-Hadoop). The overview of this architecture is shown as Fig .1.

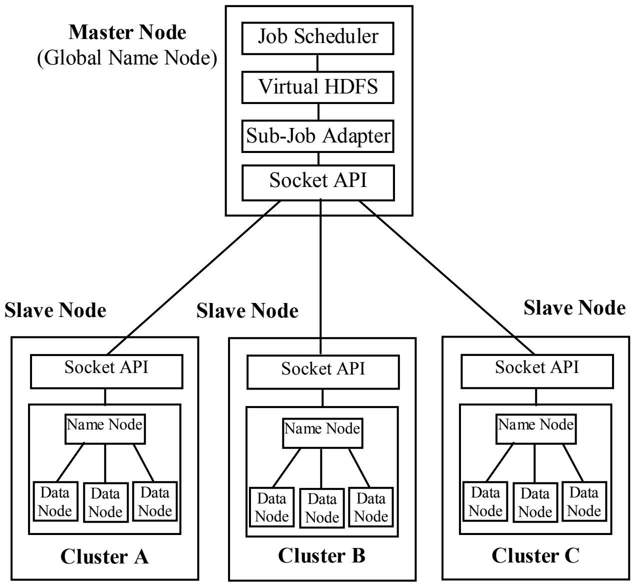


Figure 1. Architecture overview of HDC-Hadoop

The HDC-Hadoop architecture represents a master/slave communication model. The master node is the central entity in this architecture and it is responsible for accepting jobs submitted by user, splitting the jobs into smaller sub-jobs and distributing these sub-jobs to the certain slave nodes where required data are located. The master node is also in charge of managing the metadata of all files available in every data nodes (virtual HDFS view of multiple HDFS). The slave node is installed on each participating cluster and enables it to run sub-jobs allocated by Sub-Job Adapter in the master node (we can also call it global name node).

The virtual HDFS provides a global view of all data files in HDFSs of every cluster. In HDC-Hadoop data must be managed in a location-aware manner in order to provide the required location information for the sub-job adapter on master node. The virtual HDFS just records meta-data of all HDFS files obtained from name nodes, such as data size, file format, access path and so on. When new file is input or existing file is modified, related name node (records the meta-data of HDFS in one cluster) must send the updating meta-data to the virtual HDFS. The relationship of virtual HDFS and HDFSs is shown in Fig .2.

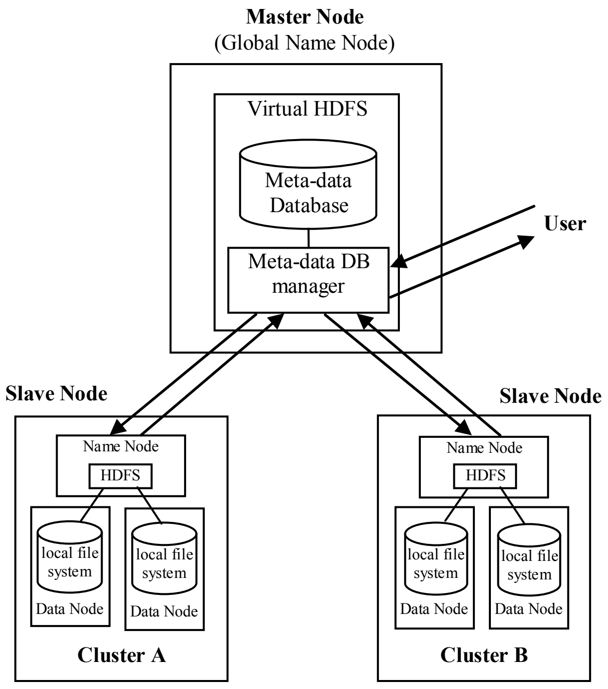


Figure 2. The architecture of virtual HDFS

Submitting a job to HDC-Hadoop is not different from the traditional Hadoop. Users write the MapReduce program for the desired job based on the data files of virtual HDFS. After compiling and decomposing, the file names in Map or Reduction function may be transfer to some locations of files in HDFS. Then the program is allocated and executed on one or several slave nodes where the required data files stored. If all the files in MapReduce program are located in one cluster (data center), the Map and Reduce programs may be transferred to corresponding cluster (slave node). Otherwise, when the files in program are located in several clusters, the program may be recompiled to Map-Map-Reduce program or Map-Reduce-Map- Reduce program, and the program may be executed step by step in different clusters (so as to say, the program is organized into several sequential sub-jobs of different clusters).

The whole workflow of a job execution in HDC-Hadoop is composed of five steps:

Job submission: user submits a MapReduce program and configuration (including parameters, input files and additional resources) to master node of HDC-Hadoop. The job scheduler may set a job ID for the new job and push it in the queue of jobs.

Program compiling and decomposing: job in queue may be split into small location-aware sub-jobs based on required data. Based on meta-data in virtual HDFS, if all files in the MapReduce program are located in one cluster (data center), the program may be transferred directly to corresponding cluster (slave node). Else if the files in the MapReduce program are located in different clusters, the program may be recompiled to Map-Map-Reduce or Map-Reduce-Map-Reduce program and organized into several sequential sub-jobs of different clusters. The file names in Map and Reduce functions based on virtual HDFS are translated into local file path of the HDFS in clusters.

Sub-job localization and assignment: based on virtual HDFS, sub-job may be assigned properly to the name node of related cluster in data-aware manner. This allocation adopts socket communication method, which can inter-transfer message, program and file based on TCP (Transmission Control Protocol). And then a job execution (the method *runJob()* of Hadoop is called) can be run on the cluster based on local file system.

Task submission and Task execution: after the job is localized on the cluster, the JobTracker in name node splits into smaller tasks. When the TaskTraker in data node receives a new task, it localizes the tasks executable and resources in local file system. Then the job is executed by spawning a new JVM (Java Virtual Machine) on the computed node and running the corresponding task with the configured parameters.

Result feedback: after all tasks of a sub-job are completed in one cluster, the name node may return the result files paths to the sub-job adapter in master node. The adapter checks whether this sub-job is a simple job (all required files located in one cluster, that is to say there is only one sub-job corresponding to a user job). If it is a simple job then register the new files in virtual HDFS and return the file name list to the job scheduler. Else if this sub-job has other subsequent or relevant sub-jobs, the output of this sub-job may be used as the input of other sub-jobs which may then be allocated and executed on corresponding cluster. Till all the sub-jobs of one user job are completed, the output files are registered in virtual HDFS and the file name list is returned to job scheduler. Finally, job scheduler returns the results to corresponding user by job ID.

Based on the HDC-Hadoop architecture, distributed computing job across multiple clusters or data centers can be implemented.

IV. APPLICATION AND PROTOTYPE

This section discusses a prototype system of this HDC-Hadoop architecture. In this prototype, two clusters are built on two data centers separately based on Hadoop. The schematic diagram of the deployment is shown in Fig .3.

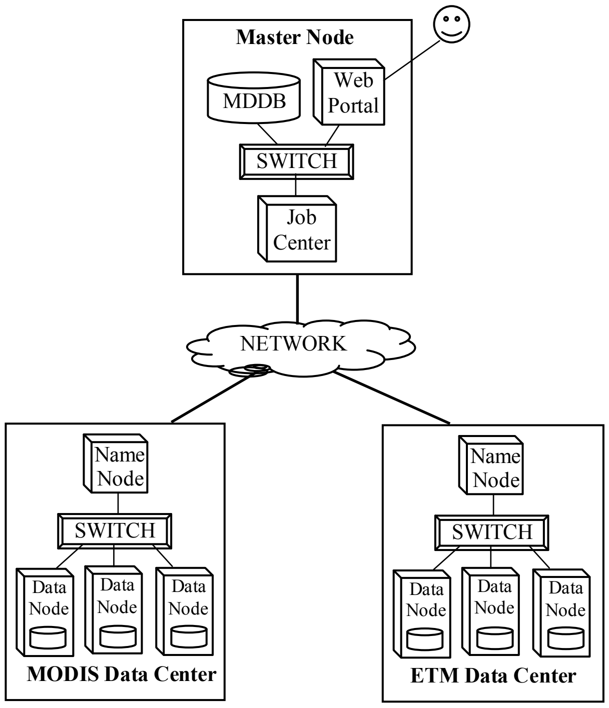
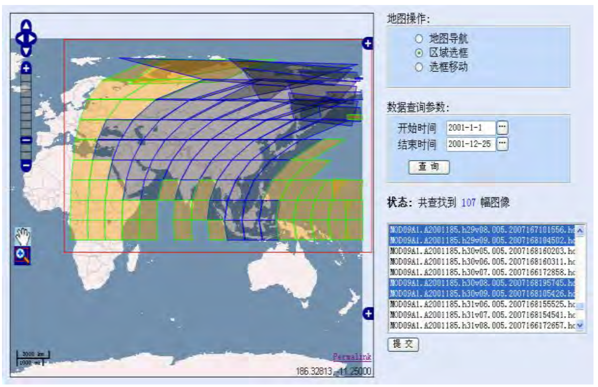


Figure 3. Deployment diagram of the prototype system

In Figure 3, there are two distributed data centers (two data centers for remote sensing raster data, one for MODIS data service and another for ETM data service). On the data storage servers, we installed virtual machines as data nodes of Hadoop and set the disks sharing, which can make the data accessible in Hadoop.

Moreover, in each data center, we added a server as name node of Hadoop system, which takes charge of the sub-job receiving from master node and task assigning to data nodes. In the master node, Web portal provide the interface for users to submit job and obtain results, job center is responsible for the job scheduling and managing, and MDDB (meta-data database) stores the meta-data of virtual HDFS.

When user submits a new job, he firstly searches the required data based on the data view of virtual HDFS, which can make the details of multiple data centers transparent, shown as Fig .4.



数据查询参数:
开始时间 2001-1-1
结束时间 2001-12-25

状态: 共查找到 107 幅图像

```
MOD09A1.A2001185.h29v08.005.2007167101556.hc
MOD09A1.A2001185.h29v09.005.2007168104502.hc
MOD09A1.A2001185.h30v05.005.2007168160203.hc
MOD09A1.A2001185.h30v06.005.2007168160311.hc
MOD09A1.A2001185.h30v07.005.2007166172858.hc
MOD09A1.A2001185.h30v08.005.2007168195745.hc
MOD09A1.A2001185.h30v09.005.2007168105426.hc
MOD09A1.A2001185.h31v06.005.2007168155525.hc
MOD09A1.A2001185.h31v07.005.2007168154541.hc
MOD09A1.A2001185.h31v08.005.2007166172657.hc
```

Figure 4. The query interface of spatial data

After selecting the required spatial data files, user uploads the algorithm runtime environment of virtual machine, which may be transferred to the cluster where the required data is stored. In the job center, the execution of each job can be monitored in Web portal. The workflow overview of job execution is shown as Fig .5.

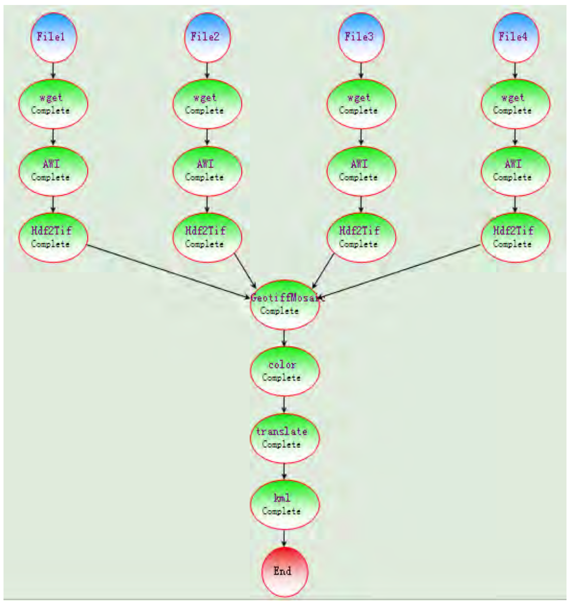


Figure 5. The monitor interface of job execution

The final execution results are expressed in form of KML (Keyhole Markup Language) files, and the access paths may be returned to the user interface in Web portal server. Based on the HDC-Hadoop architecture, the simulation system can realize data distributed computing based on Hadoop across multiple clusters, but this prototype is still a test bed currently and there are many unresolved problems yet.

V. CONCLUSIONS

The requirements for data-intensive computing across distributed clusters and multiple data centers have grown significantly in recent years. However, original MapReduce paradigm and Hadoop platform are developed to operate on single cluster environments and cannot be implemented in large-scale distributed data processing across multiple cluster and data centers. The goal of this research is to apply Hadoop framework in the large-scale distributed computing across multiple clusters. In this paper, a hierarchical distributed computing architecture is designed and presented. This architecture is based on master/slave communication model and virtual HDFS is proposed to provide global view of data sets across distributed clusters. The HDC-Hadoop architecture, virtual HDFS structure and job execution flowchart are illuminated, and also a prototype based on this architecture is built up and presented. The implementation results show that this design is feasible and has application prospect. To make HDC-Hadoop fully functional and implementable, next step we plan to enhance the distributed file system across wide area networks and design the security mechanism across multiple administrative domains for this architecture.

ACKNOWLEDGMENT

The authors would like to thank all the team members of Data Technology Department in CEODE (Center of Earth Observation and Digital Earth) of CAS (Chinese Academy of Sciences). This work is supported by the Science and Technology Research and Development Plan of Qinhuangdao (No. 201401A010) and the Doctoral Foundation of Yanshan University (No. B718).

REFERENCES

[1] Yadav Krishna R. and Purnima Singh. MapReduce Programming Paradigm Solving Big-Data Problems by Using Data-Clustering Algorithm. International Journal of Advanced Research in Computer Engineering & Technology, vol. 3, no. 1, pp. 77-80, 2014.

[2] Lizhe Wang, Jie Tao, Holger Marten, Achim Streit, Samee U. Khan, Joanna Kolodziej and Dan Chen. MapReduce Across Distributed Clusters for Data-intensive Applications. IEEE International Parallel & Distributed Processing Symposium, Shanghai, China, pp. 2004-2011, May 2012.

[3] Ling Qian, Zhiguo Luo, Yujian Du, Leitao Guo. Cloud Computing: An Overview. Cloud Computing, Lecture Notes in Computer Science, vol.5931, pp. 626-631, 2009.

[4] Lu Huang, HaiShan Chen, TingTing Hu. Research on Hadoop Cloud Computing Model and its Applications. The Third International Conference on Networking and Distributed Computing, Hangzhou, China, pp. 59-63, October 2012.

[5] Lizhe Wang, Jie Tao, Yan Ma, Samee U. Khan, Joanna Kolodziej and Dan Chen. Software Design and Implementation for MapReduce across Distributed Data Centers. Applied Mathematics & Information Sciences, vol. 7, no. 1, pp. 85-90, 2013.

[6] Lizhe Wang, Jie Tao, Rajiv Ranjan, Holger Marten, Achim Streit, Jingying Chen, Dan Chen. G-Hadoop: MapRecuce across distributed data centers for data-intensive computing. Future Generation Computer Systems, vol. 29, no. 3, pp. 739-750, 2013.

[7] Fabrizio Marozzo, Domenico Taliaa and Paolo Trunfio. P2P-MapReduce: Parallel data processing in dynamic Cloud environments. Journal of Computer and System Sciences, vol. 78, no. 5, pp. 1382-1402, 2012.

[8] Chun-Yu Wang, Tzu-Li Tai, Jui-Shing Shu, Jyh-Biau Chang and Ce-Kuen Shieh. Federated MapReduce to Transparently Run Application on Multicluster Environment. The IEEE 3rd International Congress on Big Data, Alaska, USA, pp. 296-302, June 2014.

[9] I. Tomasic, A. Rashkovska and M. Depolli. Using Hadoop MapReduce in a Multicluster Environment. The 36th International Convention on Information & Communication Technology Electronics & Microelectronics, Opatija, Croatia, pp. 345-350, May 2013.

[10] Cesar Andres, Carlos Molinero and Manuel Nunez. A Hierarchical Methodology to Specify and Simulate Complex Computational Systems. Computational Science (ICCS 2009), Lecture Notes in Computer Science, vol. 5544, pp. 347-356, 2009.