Send Orders for Reprints to reprints@benthamscience.ae

The Open Cybernetics & Systemics Journal, 2015, *9*, 131-137

Open Access

The Study of a Hierarchical Hadoop Architecture in Multiple Data Centers Environment

Sun Shengtao[1,4,*], Wu Aizhi[2] and Liu Xiaoyang[3]

[1]*School of Information Science and Engineering, Yanshan University, Qinhuangdao, Hebei, 066004, P. R. China;* [2]*College of Vehicles and Energy, Yanshan University, Qinhuangdao, Hebei, 066004, P. R. China;* [3]*Data Industry Research Institute Limited Company, Qinhuangdao, Hebei, 066004, P. R. China;* [4]*Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Qinhuangdao, Hebei, 066004, P. R. China*

Abstract: Hadoop is a reasonable tool for cloud computing in big data era and MapReduce paradigm may be a highly successful programming model for large-scale data-intensive computing application, but the conventional MapReduce model and Hadoop framework limit themselves to implement jobs within single cluster. Traditional single-cluster Hadoop may not suitable for situations when data and compute resources are widely distributed this paper focuses on the application of Hadoop across multiple data centers and clusters. A hierarchical distributed computing architecture of Hadoop is designed and the virtual Hadoop file system is proposed to provide global data view across multiple data centers. The job submitted by user can be decomposed automatically into several sub-jobs which are then allocated and executed on corresponding clusters by location-aware manner. The prototype based on this architecture shows encouraging results.

Keywords: Across clusters, apache Hadoop, hierarchical distributed computing architecture, multiple data centers, virtual Hadoop file system.

1. INTRODUCTION

The requirements for data-intensive analysis of scientific data across distributed data centers have grown significantly in recent years. The MapReduce paradigm has emerged as a highly successful programming model for large-scale data-intensive computing applications [1]. However, current Hadoop implementations are developed to operate on single cluster environments and 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]. As job sizes become larger, single-cluster solution grows inadequately. This research is devoted to the study on the distributed computing model across 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. The investigations and researches on these three issues have been started recently and made some progress.

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. Socket is used to transfer these tasks to the corresponding cluster (data center), where the data processing in supposed to be run without data moving. Moreover, the virtual HDFS (Hadoop Distributed File System) is presented to provide global view of all data among these data centers, which can record the meta-data and access path of each data center. The design and application of this architecture is illuminated 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 the background and 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 works.

2. BACKGROUND AND 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. So to say, Cloud computing is a reasonable and prospective way to solve many challenges in era of large data. The MapReduce paradigm and its open sourced implementation-Hadoop have been recognized as a representative enabling technique for Cloud computing [4].

1874-110X/15

The MapReduce programming model is inspired by two main functions commonly used in functional programming: Map and Reduce [5]. The Map function process key/value pairs to generate a set of intermediate key/value pairs and the Reduce function merges all the same intermediate system. The most popular implementation of the MapReduce model is the Hadoop framework, which allows applications to run on large cluster built from commodity hardware. The Hadoop framework transparently provides both reliability and data transfer.

The MapReduce programming model is designed to process large volumes of data in parallel by dividing the job into a set of independent tasks [6]. The Hadoop framework consisted of a single Master node and several Slave nodes. In distributed computing frameworks based on Hadoop, one job committed to master node (name node which runs a *JobTracker* process and accepts job requests from client node) may be divided into several same tasks and then run on several slave nodes (data node which runs a *TaskTracker* process and executes task in separate java processes). 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 [7]. 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 [8], Gfarm and Torque were adopted to manage data files and resources for clusters, and a security framework in G-Hadoop has been studied [9]. In P2P-MapReduce [10], the adaptive MapReduce framework provided a more reliable MapReduce middleware in dynamic Cloud infrastructures that can be effectively exploited to manage node churn, master failures, and job recovery in a decentralized but effective way. A Federated MapReduce [11] provided a transparent way to run original MapReduce jobs across multiple clusters without any extra programming burden. In the hierarchical MapReduce framework [12], a Map-Reduce-GlobalReduce model was adopted, and Gfarm file system was used to avoid cross-cluster data movement.

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 [13]. The research on Hadoop (MapReduce) across data centers or clusters is rare, but we believe this issue is worthwhile. In our research, we strive to employ Hadoop in the application of spatial data processing across multiple data centers based on the location-aware manner.

3. THE DESIGN OF HIERARCHICAL DISTRIBUTED COMPUTING ARCHITECTURE OF HADOOP

With the rapid step 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. Users cannot deploy a Hadoop on top of multiple clusters to form a single lager MapReduce cluster because the internal nodes of a cluster are not directly reachable from outside.

There are two possible approaches to addressing the challenge of internal nodes not being reachable from outside [6]. The first is to unify the underlying physical clusters as a single virtual cluster by adding a special infrastructure layer, and run MapReduce on top of this virtual cluster. The other is to make the MapReduce framework directly with multiple clusters without additional special infrastructure layers. We adopt the first approach and study the architecture of Apache Hadoop application. In this paper, we try to employ the hierarchical methodology [14] 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**).

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).

Virtual HDFS provides a global view of all files in HDFSs of every cluster. In HDC-Hadoop framework, data must be managed in a location-aware manner in order to provide the required location information for the task adapter on master node. 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) sends the updated meta-data to virtual HDFS. The architecture of virtual HDFS is shown in Fig. (**2**).

User can query and obtain the filename in virtual HDFS based on metadata in global name node. The filename selected by user in MapReduce program will be transferred automatically into more specific address for file operation, which includes: cluster address, HDFS directory and block number. Cluster address is used to locate program execution location (the name node's address of a cluster), HDFS directory is used to indicate the paths of file accessing (some data nodes' addresses of the same name node), and block number is used to find the partition of data addressing (disk partition of the local file system in a data node).

4. THE JOB EXECUTION WORKFLOW BASED ON HDC-HADOOP

Submitting a job to HDC-Hadoop is not different from the traditional Hadoop. Users write the MapReduce application for the desired job, the application is compiled and then run on the client nodes.

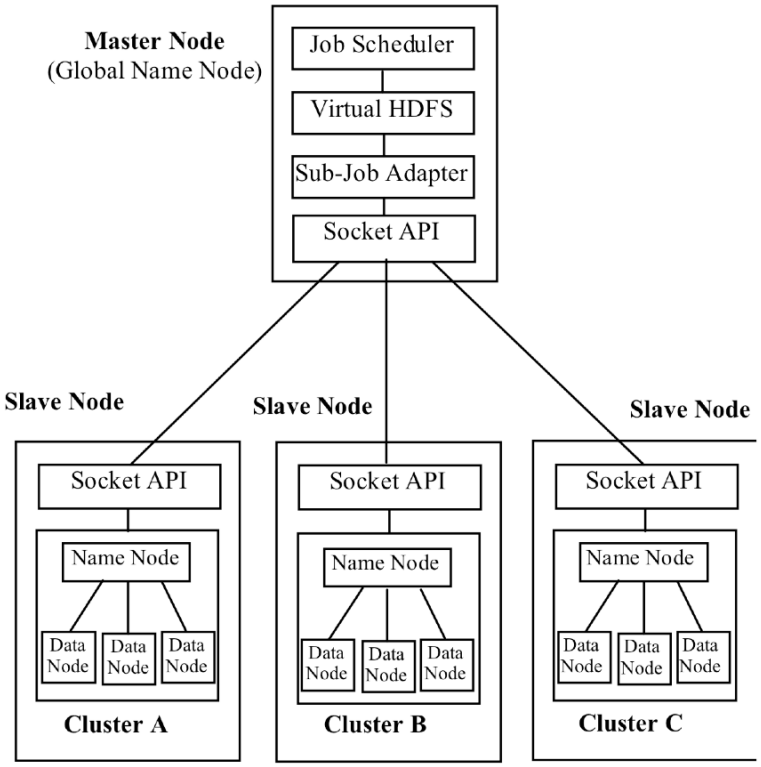

Fig. (1). Architecture overview of HDC-Hadoop.

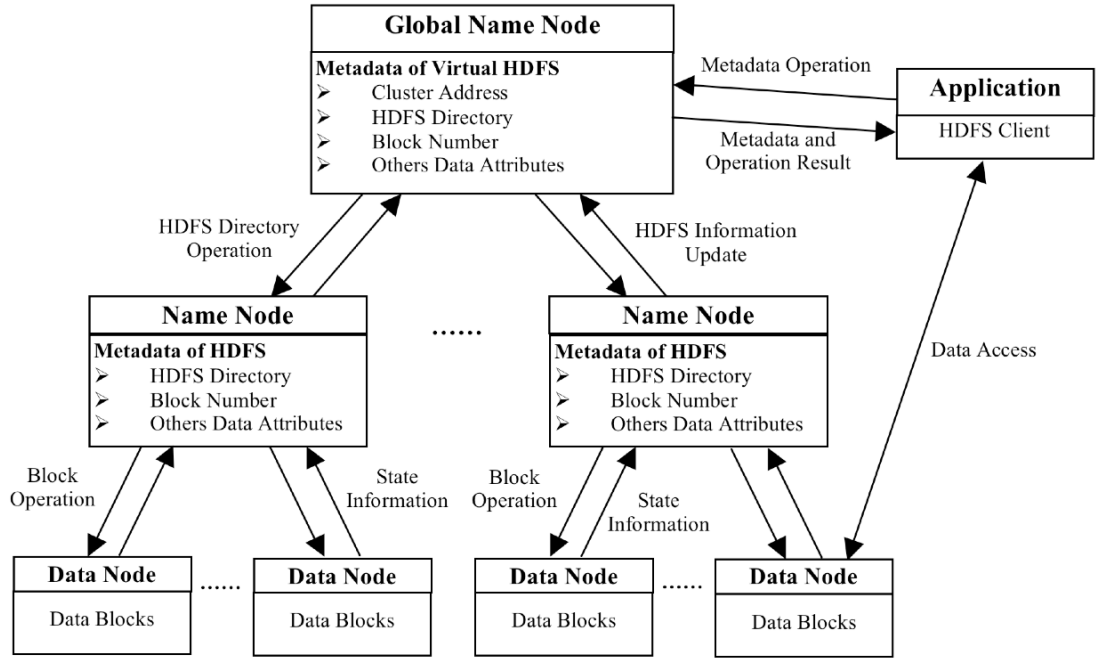

Fig. (2). Architecture of the virtual hadoop file system.

In HDC-Hadoop framework, user writes the MapReduce program for the desired job based on the data files in virtual HDFS. After compiling and decomposing, the file names in Map or Reduce function may be transfered to certain locations of files in HDFS. Then, the program is allocated and executed on one or several slave nodes where the required data files are resident. 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 or parallel sub-jobs of different clusters). The job execution flow in HDC-Hadoop framework is shown in Fig. (**3**). The whole workflow is composed of seven steps as below.

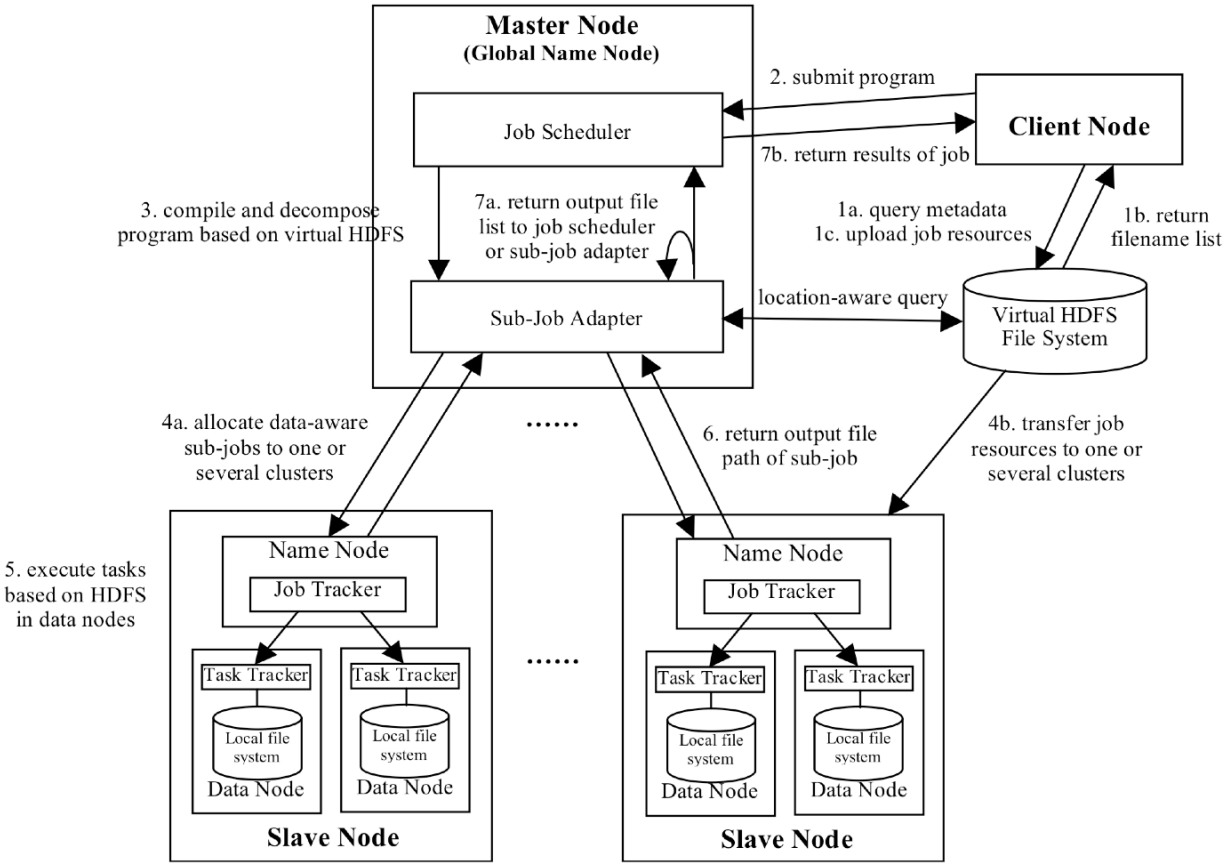

Fig. (3). Execution flowchart of a MapReduce Job in the architecture of HDC-Hadoop.

1) **Data selection**: Before submission of a job, the required data need to be query and selected. Based on the metadata in virtual HDFS, user can find the desired data file from data centers and add the file names in Map program as the input of job.

2) **Job submission**: From client node, user submits a job (programming in MapReduce paradigm based on Java language) and its configurations (including parameters, input file names and additional resources) to the master node of HDC-Hadoop. The job scheduler in master node may set a job ID for the new job and push it in the queue of jobs. Upon a successful request the job client copies the MapReduce executable (JAR) and its configurations to a designated working directory on the file system of master node.

3) **Program compiling and decomposing**: On the master node, jobs 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.

4) **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.

5) **Task submission and 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.

6) **Result of sub-job return back**: Name node monitors the status of all tasks in data nodes of the same cluster. After all tasks of a sub-job are completed in the cluster, the output file is stored in the designated directory of HDFS. Then, the name node obtains and returns the result files paths to the sub-job adapter in master node.

7) **Result of job return back**: The sub-job adapter in master node checks whether this sub-job is a simple job (which means all required data 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 output file 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 or parameter of other sub-jobs which may then be allo-

cated and executed on corresponding clusters. Till all the sub-jobs of one 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.

5. APPLICATION AND PROTOTYPE

In order to illuminate the feasibility of the HDC-Hadoop architecture, this section discusses a prototype example of spatial data distributed processing platform based on this architecture.

In this prototype, two clusters are built on two data centers (two data centers of remote sensing raster data, one for MODIS data service and another for ETM data service) separately based on Hadoop. The schematic diagram of the deployment is shown in Fig. (**4**).

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. (**5**).

After selecting the required spatial data files, user uploads the algorithm runtime environment of virtual machine (Virtual machine in public cloud or private data centers have become the norm for running transactional applications [15]), which may be transferred to the cluster where the required data are resident. In the job center, the execution of each job can be monitored in Web portal, shown as Fig. (**6**).

The final execution results are expressed in the 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

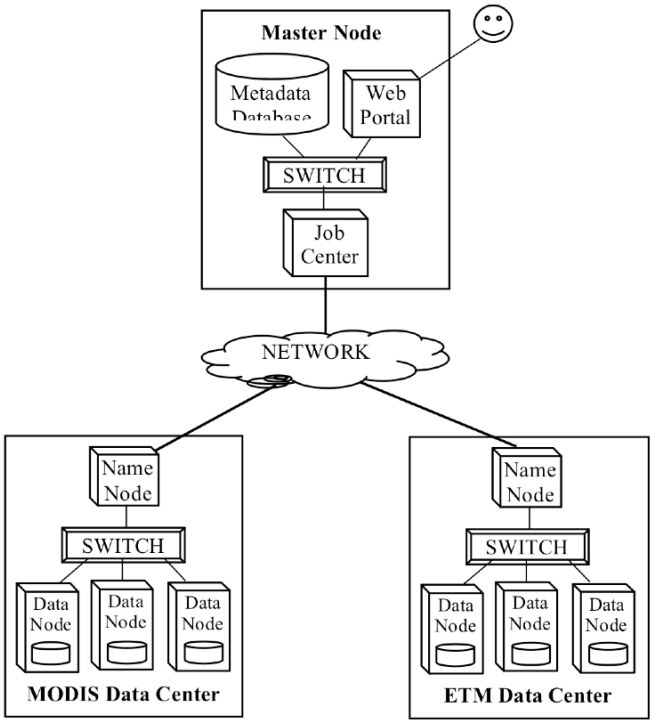

Fig. (4). Deployment diagram of the prototype system.

Hadoop across multiple clusters, but this prototype is still a test bed currently and there are many unresolved problems yet.

CONCLUSION

The requirements for data-intensive computing across distributed clusters and multiple data centers have grown significantly in recent years. However, original MapReduce

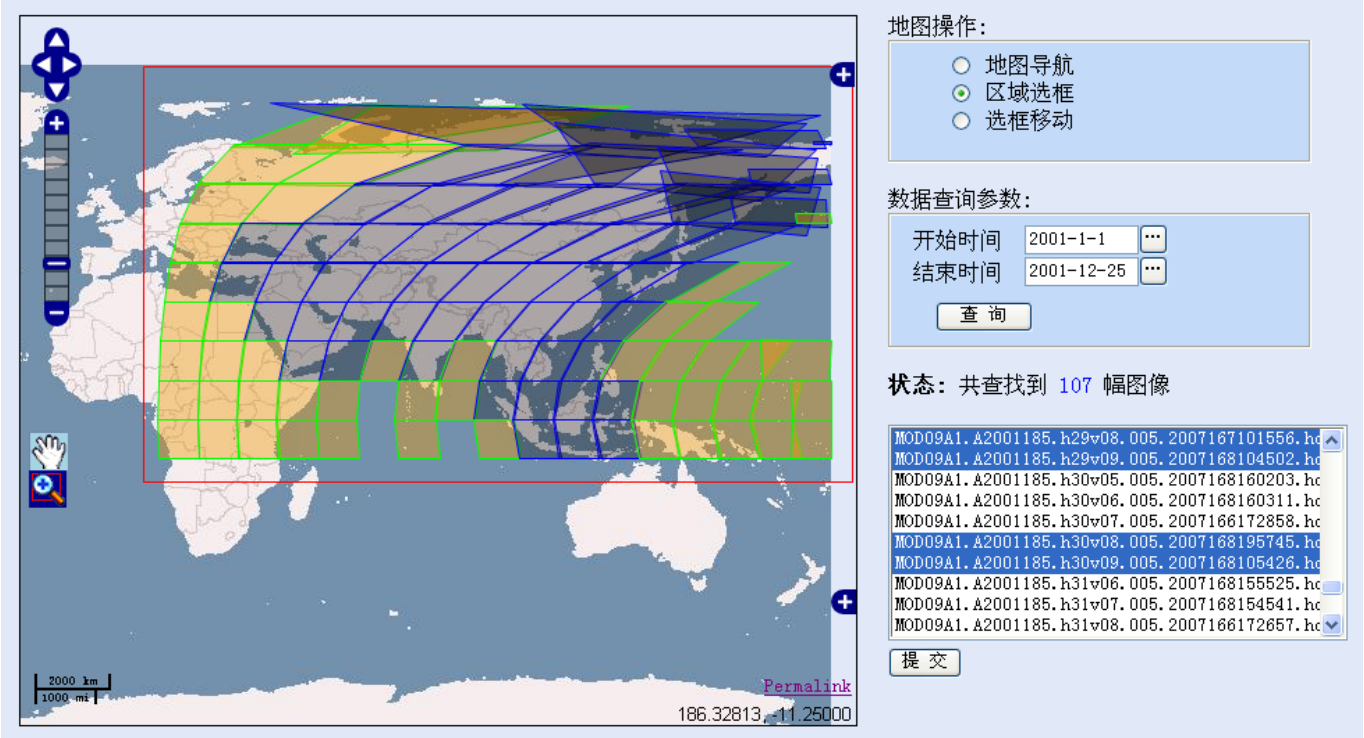

Fig. (5). The query interface of spatial data.

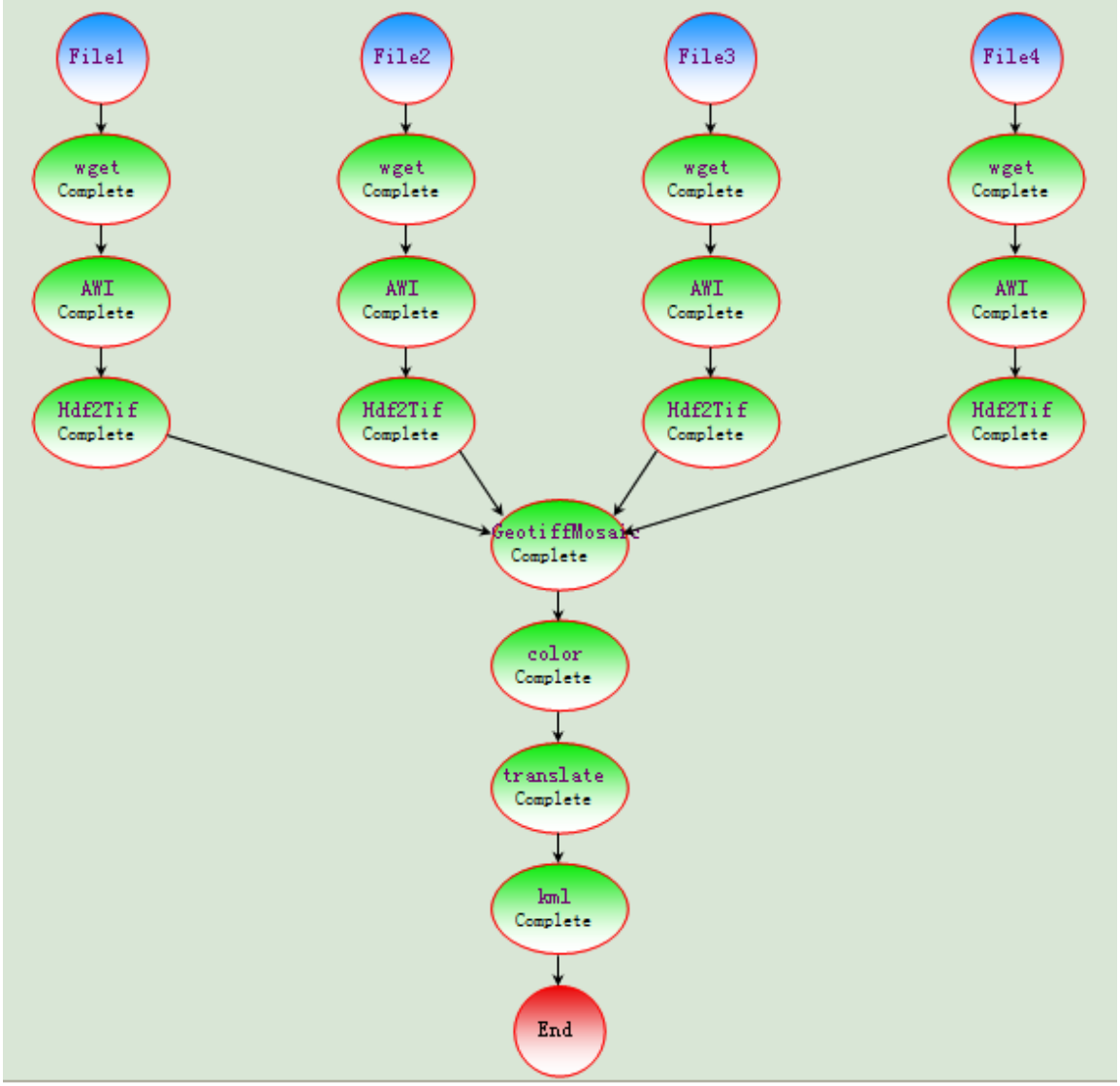

Fig. (6). The monitor interface of job execution.

paradigm and Hadoop framework are developed to operate on single cluster environments, which cannot be implemented in large-scale distributed data processing across multiple clusters and data centers. The goal of this research is to apply Hadoop framework in the large-scale distributed computing across multiple data centers without data moving (so as to say, bringing computation to where data is resident). 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 results show roughly 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, design the programming model and scheduling algorithm of this framework, support workload balance of tasks among heterogeneous clusters, and provide the security mechanism across multiple administrative domains for this architecture.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

The authors would like to thank the advices of Professor Lizhe Wang and the help from team members of Data Technology Department in CEODE (Center of Earth Observation and Digital Earth) of Chinese Academy of Sciences.

This work is supported by the National Natural Science Foundation of China (No. 61303130), the Science and Technology Research Plan of Qinhuangdao (No. 201401A010) and the Natural Science Foundation of Hebei Province (No. F2014203093).

REFERENCES

[1] R.Y. Krishna, and P. 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] L. Wang, J. Tao, H. Marten, A. Streit, S. U. Khan, J. Kolodziej, and D. Chen, "MapReduce across distributed clusters for data-intensive applications", In: *Proceedings of IEEE International Parallel & Distributed Processing Symposium,* Shanghai, China, pp. 2004-2011, 2012.

[3] L. Qian, Z. Luo, Y. Du, and L. Guo, "Cloud computing: an overview. cloud computing", *Lecture Notes in Computer Science*, vol. 5931, pp. 626-631, 2009.

[4] H, Lu, H. S. Chen, and T.T. Hu, "Research on hadoop cloud computing model and its applications", In: *Proceedings of 3rd International Conference on Networking and Distributed Computing,* Hangzhou, China, pp. 59-63, 2012.

[5] R. Lammel, "Google's MapReduce programming model- Revisited", *Science of Computer Programming*, vol. 70, no. 1, pp. 1-30, 2008.

[6] M. Cardosa, C. Wang, A. Nangia, A. Chandra, and J. Weissman, "Exploring mapReduce efficiency with highly-distributed data", In *Proceedings of 2nd International Workshop on MapReduce and its Applications*, California, USA, pp. 27-33, 2011.

[7] L. Wang, J. Tao, Y. Ma, S. U. Khan, J. Kolodziej, and D. Chen, "Software design and implementation for MapReduce across distributed data centers", *Applied Mathematics & Information Sciences*, vol. 7, no. 1, pp. 85-90, 2013.

[8] L. Wang, J. Tao, R. Ranjan, H. Marten, A. Streit, J. Chen, and D. Chen, "G-Hadoop: MapReduce across distributed data centers for data-intensive computing", *Future Generation Computer Systems*, vol. 29, no. 3, pp. 739-750, 2013.

[9] J. Zhao, L. Wang, J.Tao, J. Chen, W. Sun, R. Ranjan, J. Kolodziej, A. Streit, and D. Georgakopoulos, "A security framework in G-Hadoop for big data computing across distributed Cloud data centres", *Journal of Computer and Systems Sciences*, vol. 80, no. 5, pp. 994-1007, 2014.

[10] F. Marozzo, D. Taliaa, and P. Trunfio, "P2P-MapReduce: Parallel data processing in dynamic Cloud environments", *Journal of Computer and System Sciences*, vol. 78, no. 5, pp. 1382-1402, 2012.

[11] C. Wang, T. Tai, J. Shu, J. Chang, and C. Shieh, "Federated mapReduce to transparently run application on multicluster environment", In: *Proceedings of IEEE 3rd International Congress on Big Data*, Alaska, USA, pp. 296-302, 2014.

[12] Y. Luo, B. Plale, Z. Guo, W. W. Li, J. Qiu, and Y. Sun, "Hierarchical MapReduce: towards simplified across-domain data processing", *Concurrency and Computation: Practice and Experience*, vol. 26, no. 4, pp. 878-893, 2014.

[13] I. Tomasic, A. Rashkovska, and M. Depolli, "Using Hadoop MapReduce in a Multicluster Environment", In: *Proceedings of 36th International Convention on Information & Communication Technology Electronics & Microelectronics*, Opatija, Croatia, pp. 345-350, 2013.

[14] C. Andres, C. Molinero, and M. Nunez, "A hierarchical methodology to specify and simulate complex computational systems", *Lecture Notes in Computer Science*, vol. 5544, pp. 347-356, 2009.

[15] B. Sharma, T. Wood, and C. R. Das, "Hybrid: A Hierarchical MapReduce Scheduler for Hybrid Data Centers", In: *Proceedings of IEEE 33rd International Conference on Distributed Computing Systems,* Pennsylvania, USA, pp. 102-111, 2013.

Received: September 16, 2014 Revised: December 23, 2014 Accepted: December 31, 2014