http://ansinet.com/itj

ITJ

ISSN 1812-5638

# INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (6): 1079-1088, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.1079.1088


# A New Massive Data Processing Framework under Cloud Environment for Digital Community

[1,2]Ke Hou and [1]Jing Zhang
[1]School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China
[2]School of Economic and Management, Xi'an Shiyou University, Xi'an 710065, China

**Abstract:** Digital community involves e-government, e-commerce, smart health and other applications. With the increase of customers and types of business, it becomes more important for digital community to process massive data efficiently. Although, the current cloud-based applications can provide some elastic and on-demand calculation abilities to digital community, their underlying programming models still have certain limitations. This study aims to provide a new framework of massive data processing for digital community. In the framework, multiple programming models are adopted and each programming model handles the specific calculations that they do best. These calculations mainly include embarrassingly parallel calculation, iteration calculation and data-dependent complex calculation. To improve the performance of the framework, the programming model connection pool and the virtual subnet are designed and applied. Compared to Hadoop and its modified version, on average, the proposed framework can reduce runtime by 1.32. The experimental results show that the proposed framework has higher generality and efficiency. Moreover, it is reasonable and valuable for digital community to analyze comprehensively trade area on geographical location and business volume.

**Key words:** Multiple programming models, massive data processing, digital community, cloud computing, MapReduce

## INTRODUCTION

The term "Digital Earth" originally occurred in U.S. Vice-President Gore's book (Gore, 1992). Later, he articulated the vision of "Digital Earth" as a digital replica of the entire planet (Gore, 1998). As the microcosmic behavior of digital earth, the term "Digital Community" or "Digital City" refers to a connected community that combines broadband communications infrastructure, flexible and service-oriented computing infrastructure based on open industry standards and innovative services to meet the requirements of governments and their employees, citizens and businesses (Solutions, 2005).

In general, digital community involves e-government, e-commerce and smart health and so on. With their development, data processing techniques applied in digital community have become an important research field. For example, in order to make the work of government and their employees more transparent, participatory and on-line, it is good practice to adopt business intelligence and analytics (BI and A) research in e-government and politics 2.0 applications (Chen, 2009). It is also considered as a severe problem that many countries are facing the lack of people with deep analytical skills (Manyika *et al*., 2011). Most of techniques for e-commerce have been developed by researchers from relational Database Management System (DBMS), data warehousing, Extract Transform and Load (ETL), On-line Analytical Processing (OLAP) and Business Performance Management (BPM) (Chaudhuri *et al*., 2011). With the expanding customers and businesses of digital community, the data presents trend towards "big data" (Hey *et al*., 2009). Instead of traditional database system and parallel computing platform, cloud-based applications (Hayes, 2008) can afford elastic and on-demand calculation ability to process massive data for digital community. Most of these applications are developed from data-intensive computing systems. For the cloud-based applications of digital community, the calculations mainly include embarrassingly parallel calculation, iteration calculation and data-dependent complex calculation. There are many different data-intensive computing systems which are used in digital community but each of them cannot ensure generality and efficiency concurrently. The reason for the

**Corresponding Author:** Ke Hou, School of Economic and Management, Xi'an Shiyou University, 710065, China Tel: +86 29 8329 1415

situation is that the programming models are limited to some specific calculations. For example, MapReduce (Dean and Ghemawat, 2008) and Dryad (Isard *et al*., 2007) are mainstream programming models. They are not suitable to solve all types of calculation problems. In Hadoop (Foundation, 2011), MapReduce is implemented with Java language but iterative calculation can only be completed through generating and invoking a chain of MapReduce jobs (Olston *et al*., 2008; Thusoo *et al*., 2009). Due to its task partitioning mechanism, Hadoop is only suitable for a single calculation with no data-dependence. Some other modified versions of Hadoop are proposed to improve it, such as Haloop (Bu *et al*., 2010), Twister (Ekanayake *et al*., 2010). They modify MapReduce to complete iteration inside the job but they also cannot solve the calculation problems with data-dependence. Using Directed Acyclic Graph (DAG), Dryad can represent complex calculations with data-dependence very well. However, it is not suitable for the iterative calculation.

This study proposes a new massive data processing framework based on multiple programming models which is special for digital community in cloud environment. In this framework, each programming model handles the specific calculations that they do best. Hence, the proposed framework is more efficient than other frameworks.

## CLOUD-BASED MASSIVE DATA PROCESSING

Fundamentally, data management refers to the process of data collecting, storing and handling with computer hardware and software technology. And massive data storing and handling is the core of massive data management system. On the one hand, the increasing data amount in various fields makes massive data management system turn to data-intensive computing (Kouzes *et al*., 2009). On the other hand, the increasingly demand for processing large-scale data makes cloud computing become a successful application of data-intensive computing in the business field. All these trends lead to a new research direction, "data-intensive computing taking massive data processing as center" (Bryant, 2007). Data-intensive computing discussed in the paper refers to Google-like computing that faces the data of the Internet information services. However, data-intensive computing in the field of scientific computing will not be mentioned. Massive data processing cloud is a data-intensive computing platform and it is a system software in Platform as a Service (PaaS) layer. As the middle layer of cloud, PaaS is necessary not only to provide a simple and reliable processing framework for the upper application but to shield the complexity of underlying virtualization facilities.

Cloud computing technology can be used in manufacturing, telecommunications, financial services, energy, e-commerce, education, scientific research and other fields. Figure 1 shows the hierarchical structure of massive data processing cloud. Although there are different types of cloud applications, all of them need a data processing framework with high performance and compatibility. Facing with "big data" problem, massive data processing cloud should be able to provide universal services, such as search engine, Social Network Site (SNS), e-commerce.

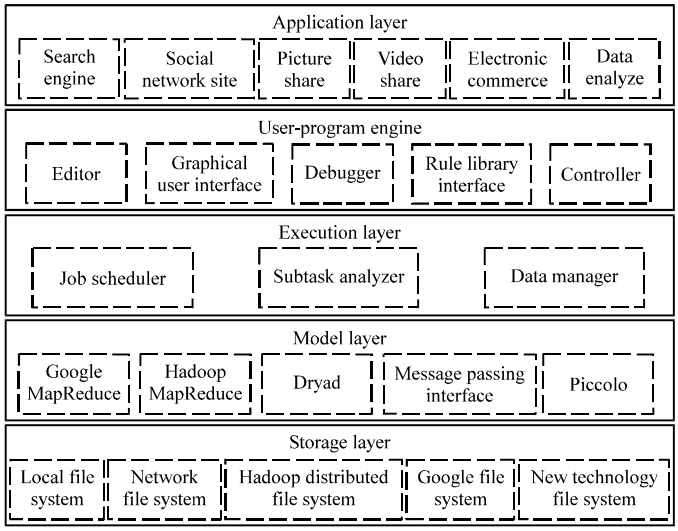


Fig. 1: Hierarchical structure of massive data processing cloud

## MASSIVE DATA PROCESSING FRAMEWORK

**Structure:** To build cloud-based digital community, there are four steps to do. (1) Setting up a data center connecting large-scale cheap servers through computer network, (2) Converting hardware and software resources into the infrastructure that can be unified deployed and managed using virtualization technology, (3) Establishing PaaS platform to manage the vast amounts of data and (4) Providing software as a service (SaaS) applications for data community.

The front end of the framework consists of client computers, operation server and rule library server. The user program is loaded on the client computers and the operation server is responsible for managing and executing jobs and subtasks. The back-end is composed of a lot of nodes which belong to different programming models and constitute the programming model connection pool. The connections between subtasks and programming model nodes always work standby. A small amount of those nodes are called master nodes. Master nodes also belong to different types of programming models. The rest of nodes are slave nodes which run on many physical machines. If the operation server executes a subtask, the framework will generate a calculation instance and assign an idle master node in the connection pool to the subtask. This master node is responsible for managing the instance and it can dynamically select slave nodes to establish a virtual subnet (Sharony, 1996). After the subtask is completed, the virtual subnet will be released immediately. Figure 2 shows physical view and logical view about the network topology of massive data processing framework. The structure of the framework is shown in Fig. 3.

**Workflow:** Before migrating SaaS applications for digital communities to massive data processing cloud, developers must login user-program engine and upload source codes of SaaS applications. The massive data processing framework will convert the code into executable jobs and then the editor of the framework can help developers decompose the job into some interrelated subtasks. The rule library server will generate subtask information, such as the serial number and name of subtask, the serial number of job. According to the situation that programming models are designed for specific calculations, the framework can help the developer utilize the programming model information to define calculation types of subtasks in the job. In this study, calculation types include embarrassingly parallel calculation, iteration calculation, data-dependent complex calculation, etc. Subtask-model information includes the serial number and name of programming model, the address of master nodes and so on. Namely, all of subtasks are associated with appropriate programming models. While SaaS applications start, the user-program engine is still the controller of the entire processing framework. It will query the rule library and call appropriate programming model instances to perform those subtasks until the entire program is finished. Figure 4 shows the workflow of massive data processing framework.

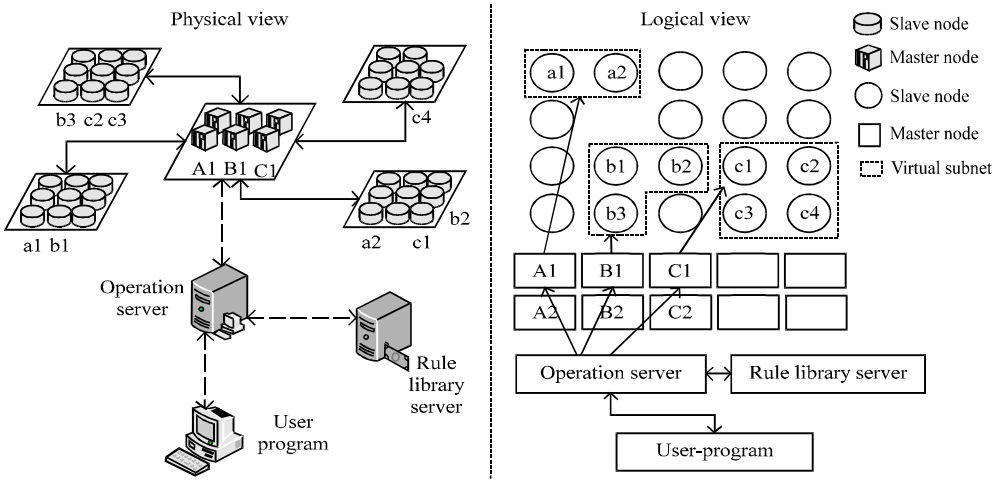


Fig. 2: Network topology of massive data processing framework

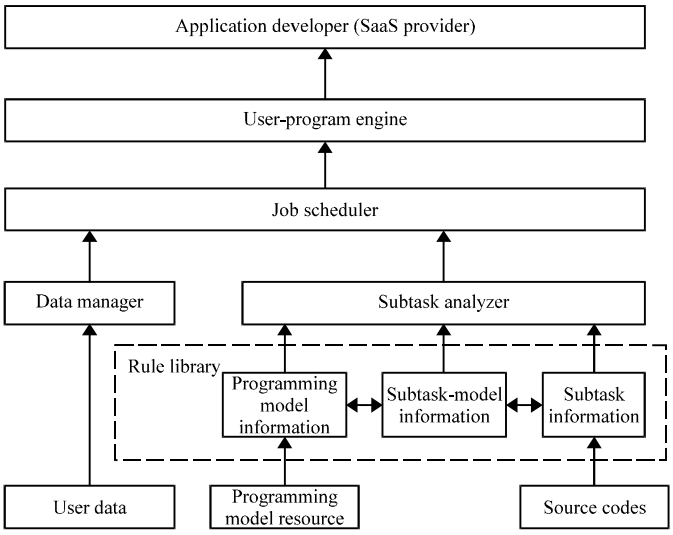


Fig. 3: Structure of massive data processing framework

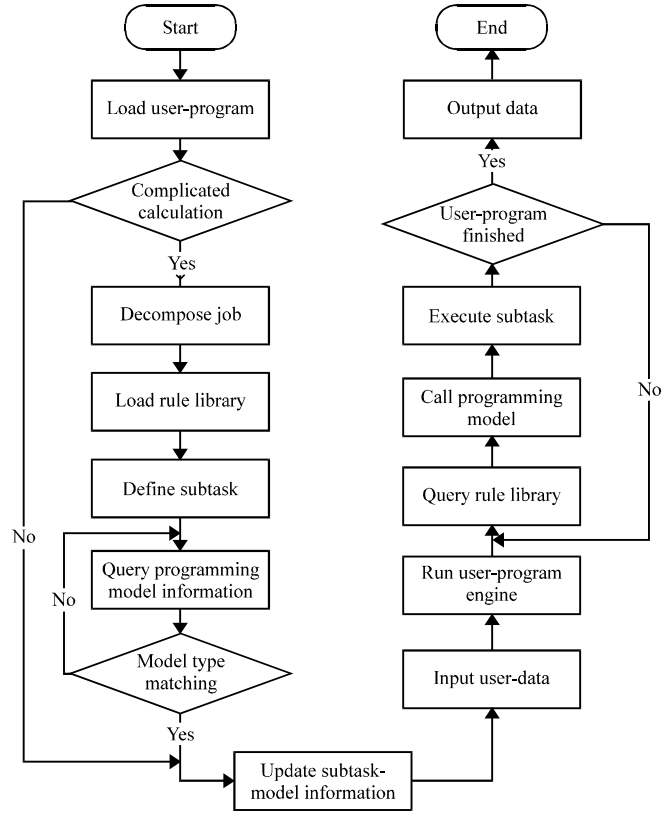


Fig. 4: Workflow of massive data processing framework

**Key technologies**

**Task partitioning:** In this study, each programming model can complete the subtasks which belong to specific calculations they do best, which is called "First level acceleration" for implementation efficiency.

As mentioned above, the basic process of task partitioning is defined as follows:

- Consider the source codes as a set S which consists of multiple lines of statements P ($P_1$, $P_2$, …, $P_n$) and

choose some statements to define as a subtask St. For example, $\{P_j, \ldots, P_k\} \Rightarrow St_m$. These statements must be executed in parallel

Not all of source codes can be defined and splitted into the subtasks. If the statements cannot be parallelized, it is best to execute them on the operation server locally. If they can be defined and splitted into the subtasks, it is necessary to determine which type of calculation they belong to.

**Programming model connection pool:** Inspired by the database connection pool technology, a programming model connection pool is design with connection multiplex technology. In order to realize the connection pool, a set of strategies about utilizing, distributing and managing connection is made. To avoid establishing and closing the connection frequently, the connection is reused efficiently and safely. While designing the programming model connection pool, three issues are considered: establishing the connection pool, using and managing the connection and closing the connection pool.

The programming model connection pool consists of two parts. One is the connection pool class which can obtain or create a connection, return the connection to the pool, close all the connections and release all the resources. The other is the connection pool manager class which is responsible for loading Programming Model (PM) driver, creating the connection pool object according to the defined attributes and tracking how the connection works. In addition, it is also responsible for providing the interface between the connection pool objects. Each connection pool object manages a set of packaged connection (PMConn) objects that, can be shared by any number of subtasks.

Connection objects work at standby state, rather than start when they are needed. So the connection between subtasks and programming model nodes will not be establish and close frequently, which is regarded as "Second level acceleration" of implementation efficiency. When the framework handles many jobs concurrently, there will be more subtasks calling the same type of programming model. The connection pool will enable multiple connection objects, so as to achieve "Third level acceleration".

**Subtask parallelization:** For a single programming model, parallelization occurs only within a subtask but the proposed framework allows users to describe certain subtasks as parallelization tasks and record them into the rule library. Different programming model instances can handle the subtasks within a job in parallel. Parallelization among different subtasks can further improve the efficiency of the job.

**Virtual subnet:** Slave nodes can run on one or many physical machines. When a master node starts a subtask, for task scheduling and load balancing, it can dynamically select slave nodes on same or different physical machines to form a virtual and internal network called virtual subnet. In addition, the master node is responsible for monitoring, managing and adjusting the resources within the virtual subnet dynamically. If the calculation tasks are completed, the master node will immediately release this virtual subnet.

## SIMULATION AND VALIDATION

**Background and system deployment:** In this study, the data came from "The Line Community Charges Supermarket". It is a store network on "digital community service platform" established by Line Group. The location information and the communications agency disbursement data in 2011 are gotten.

A simulation system is built to validate the design's feasibility and it mainly consists of the underlying environment and the upper software. The underlying environment is a small cluster which includes 10 servers and 1 Gb switch and each computer installs Linux and Java environment. In order to ensure security of remote login session, secure shell (SSH) is configured and used. The upper software is a data-intensive computing system with multiple programming models and its deployment is shown in Fig. 5.

**Simulation design:** The simulation procedure includes five stages. The first stage is preparing data (stage A), the second is defining subtask (stage B) and the third is analyzing trade area on geographical location named the natural trade area (stage C). Then, the last two stages are analyzing trade area on business volume (stage D) and analyzing trade area comprehensively (stage E). Among them, the stage C and D can be executed in parallel. To obtain comparison in performance between the proposed framework and the others (Hadoop and its modified version), the user program is executed in three ways separately.

**Stage A: Preparing data:** Firstly, latitude and longitude information of all the stores in town are gathered through Google maps and are saved as a text file. Secondly, through a data access interface every store's communication agency disbursement data are gotten and saved into separate files. The communication agency disbursement data are daily transaction data of China Mobile, China Unicom and China Telecom companies.

**Stage B: Analyzing subtask:** After the source codes for testing are uploaded into the user-program engine, they are converted into a complex job which contains three

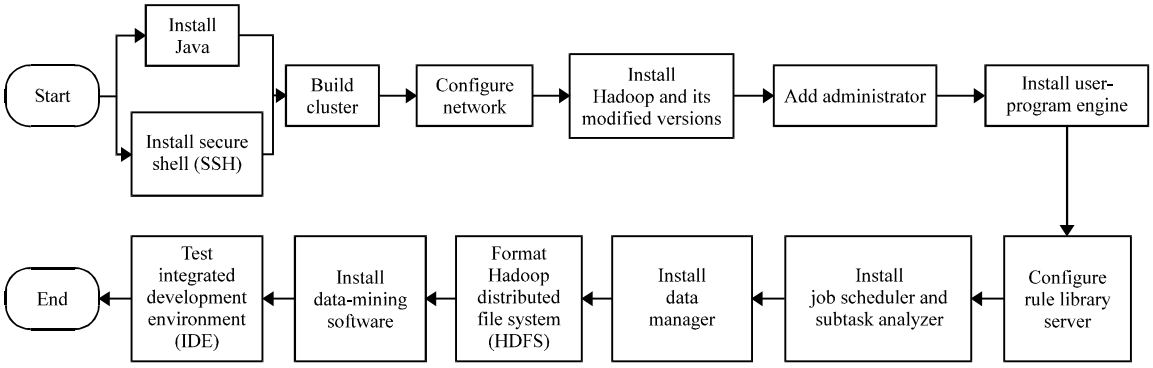


Fig. 5: Deployment of simulation system

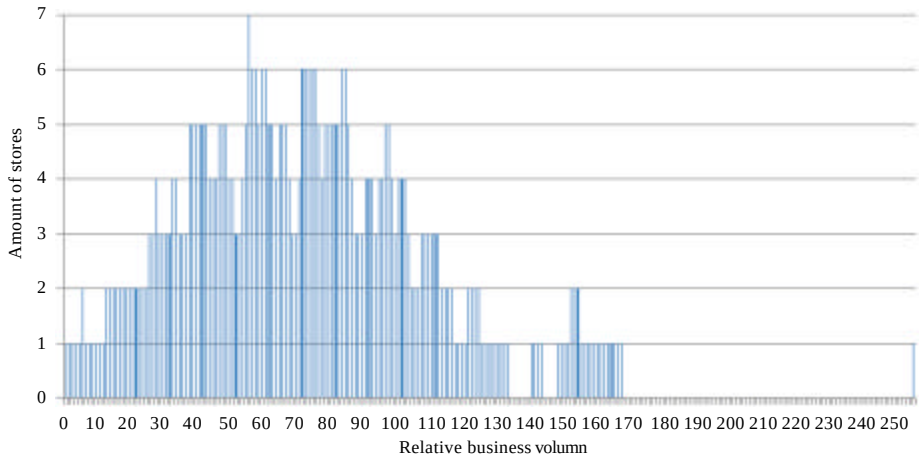


Fig. 6: Business volume histogram

kinds of operation. The first operation is a series of arithmetic operations on business volume, such as sum and ratio which belong to embarrassingly parallel calculation. The second is a k-means algorithm for studying the similarity of stores by loops and dividing them into N clusters. The third is the threshold segmentation which researches the store classification on business volume histogram. The latter two operations belong to iterative calculation. Using the editor to decompose the job into some subtasks and label them with eXtensible Markup Language (XML) tags. Then the right programming models are specified to them and subtask-model information are generated. These operations are ultimately defined as three subtasks called Subtask 1 (St1), Subtask 2 (St2) and Subtask 3 (St3).

**Stage C: Analyzing natural trade area:** To execute St2, the XML file is opened in the user-program engine. The k-means algorithm clusters the stores by their latitude and longitude. It is defined as follows:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2 \quad (1)$$

where, E is the sum of squared error of all the objects in the data set, p is a point in the space and presents a given object, $m_i$ is the mean of cluster $C_i$ (p and $m_i$ are multiple dimensions). To get initial value of k, canopy clustering algorithm is used.

**Stage D: Analyzing trade area on business volume:** This stage includes St1 and St3 and all the stores are viewed as a set S. St1 calculates every store's yearly business volume $b_i$ and their business volume ratio $p(p = b_i/\max(b_i))$. For the sake of simplicity to analyze, p is multiplied by 255 and the result is rounded as relative business volume. Figure 6 shows a business volume histogram with 256 levels which is a function on business level. This histogram describes the number of stores which have any business volume. In Fig. 6, the abscissa is the level of business volume and the ordinate is the appearance frequency of a business volume.

In stage St3, iterative method is used to get the threshold Th (or h) within business volume level $[t_1, t_2]$ and complete threshold segmentation to stores. Function f(x, y) refers to the store's business volume with longitude x and latitude y and its range is [a, b]. The iterative method uses the following formula:

$$Th = \frac{\sum_{a}^{b} f(x,y)}{\sum_{a}^{b} t} \quad (2)$$

Firstly, suppose $t_1 = 0$, $t_2 = 255$, the initial threshold value is calculated and then new thresholds called $h_2$ and $h_3$ are gotten from $[t_1, h_1]$ and $[h_1, t_2]$, respectively. And so on, all the thresholds $h_i$ (i = 1, …, n) are obtained. Secondly, listing $h_i$ in ascending order after adding 0 and 255, an ordered series $\{a_n\} = \{0, \ldots, h_i, \ldots, 255\}$ is gotten. Assigning two adjacent values to $t_1$ and $t_2$ sequentially, the threshold segmentation on S is done. The stores within $[t_1, t_2]$ are classified into one group and then n+1 groups are gotten. Finally, the results are output and saved as text file.

**Stage E: Analyzing trade area comprehensively:** Drawing the natural trade area and the trade area on business volume separately and overlaying them, the comprehensive trade area is gotten.

## RESULTS

According to geographical location and business volume, analyzing trade area gets satisfied results shown in Fig. 7. In Fig. 7a, inverse triangles are the stores, different colors represent different trade areas. In Fig. 7b, circle points are the stores in the trade area on business volume and different gray shades represent different groups. Overlaying the natural trade area and the trade area on business volume, the comprehensive trade area is shown in Fig. 7c. In addition, Fig. 7d is an enlarged view on part of Fig. 7c. One thing to notice here is that Fig. 7a and Fig. 7b are the part of enlarged drawings.

In this simulation, the proposed framework is compared with Hadoop and modified version of Hadoop. Figure 8 shows the comparison in performance as the number of nodes (N) increases from 8 to 64.

## DISCUSSION

Although the simulation result has proved that the proposed framework works as desired, there are some issues needed to be discussed.

The first issue is whether the comprehensive trade area is more applicable to digital community than the natural trade area. Generally, the natural trade area has potential marketing value (Reilly, 1931; Christaller, 1966). It is helpful not only to estimate consumer behavior (Van Leeuwen and Rietveld, 2011) but also to develop the Location Based Services (LBS) (Schiller and Voisard, 2004). However, it is clear that the natural trade area cannot reflect the store's business volume. Combining it with the trade area on business volume, the comprehensive trade area is reasonable and valuable. Besides, traditional tools are just suitable to handle small-scale data, such as Statistic Package for Social Science (SPSS) (Janssens *et al*., 2008).

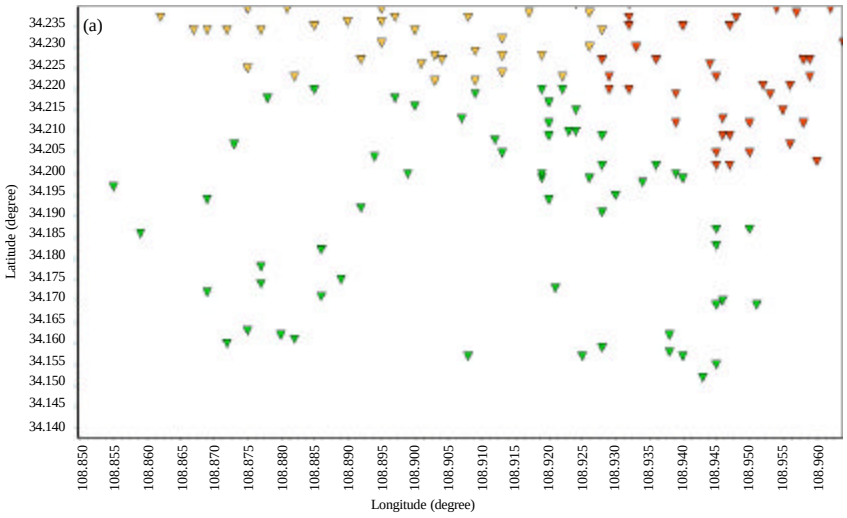


Fig. 7(a-d): Conitnue

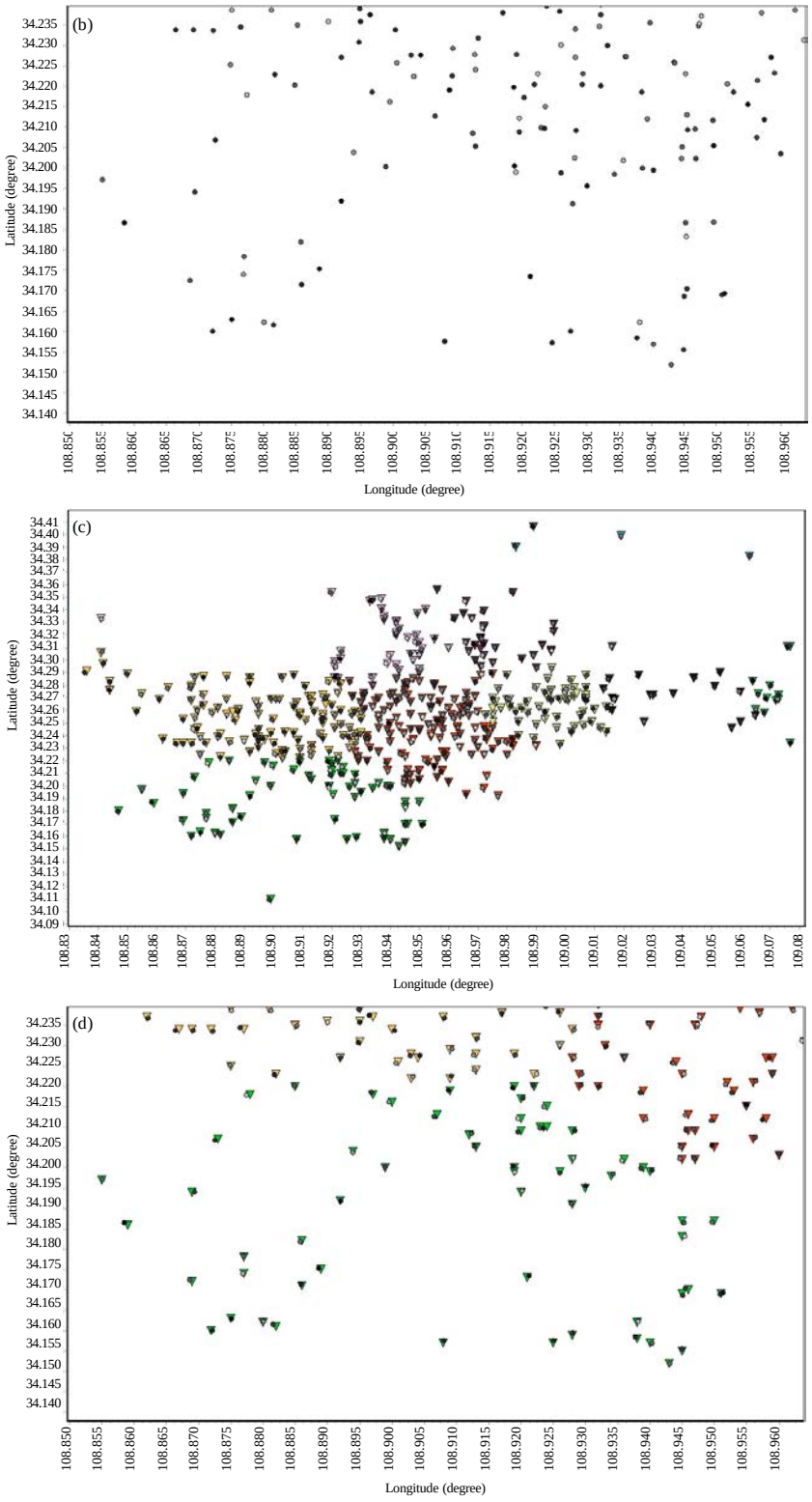


Fig. 7(a-d): Results of analyzing trade area, (a) Natural trade area, (b) Trade area on business volume, (c) Comprehensive trade area and (d) Enlarged view on part of Fig. 7c

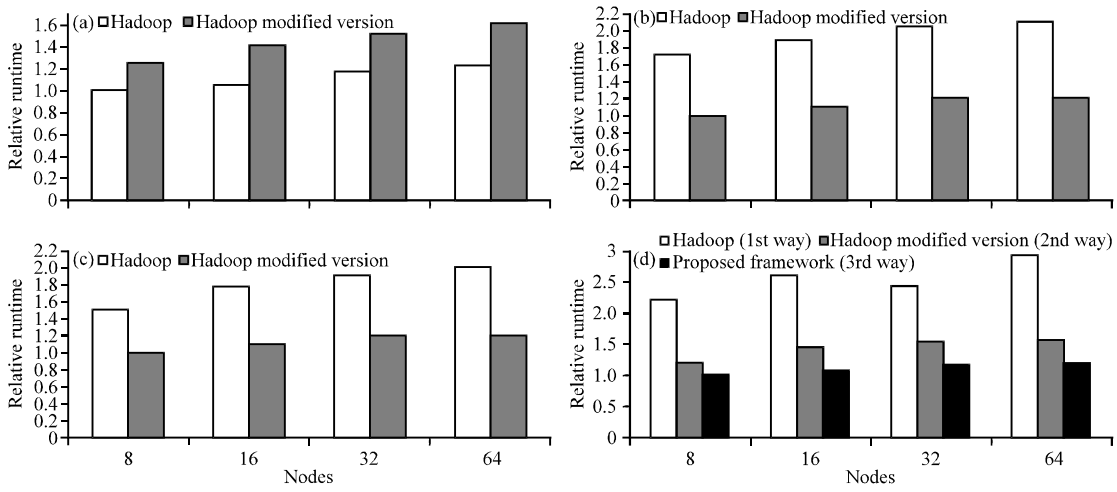


Fig. 8(a-d): Scaling performance, (a) Subtask 1, (b) Subtask 2, (c) Subtask 3 and (d) Three ways

The second issue is about the generality and efficiency of the proposed framework in this study. As noted before, programming models are oriented to specific calculations. In this study, the user program is executed in three different ways separately. In the first way, the user program is executed only with Hadoop. Its higher efficiency during embarrassingly parallel calculation can be seen in the other solutions (Dyer *et al*., 2008; Van Gael *et al*., 2009). However, Fig. 8b and c show that while processing iteration, a chain of MapReduce jobs leads to the degradation of performance. In the second way, Hadoop is replaced by its modified version which originated from Haloop. The modified version of Hadoop provides similar efficiency as well while completing iteration. Figure 8a shows that its performance decreases on embarrassingly parallel calculation, compared with Hadoop. In the third way, multiple programming models were used serially. Hadoop is responsible for executing embarrassingly parallel calculation, while the iteration calculation is allocated to modified version of Hadoop. Figure 8d shows that the last way is the highest one, because each programming model has completed the calculations that they do best. Moreover, applying programming model connection pool and virtual subnet further improves the performance. Overall, as the figures show, the proposed framework lowers the runtime by 1.32 on average. By the way, if the subtasks in stage C and stage D are performed concurrently, the proposed framework will get higher efficiency. While $T_{CD}$ equals to the maximum of $T_C$ and $T_D$, the minimum of $T_C$ and $T_D$ is saved.

## CONCLUSION

This study orients towards massive data processing in cloud-based digital community and proposes the massive data processing framework on multiple programming models. The key technologies of the framework are analyzed and studied, such as task partitioning, programming model connection pool, subtask parallelization. The result of simulation shows that the proposed framework can improve execution efficiency of the user program significantly. With the development of data-intensive computing, more new programming models will appear. Hence, how to apply them into the proposed framework will become the future work.

## ACKNOWLEDGMENTS

Parts of this research are supported by the 2007 Nationality 863 Project "Software Product Line based on Workflow" (No. 2007AA010305), the State Key Laboratory for Manufacturing Systems Engineering Open Project (No. sklms2011011). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers which have improved the presentation.

## REFERENCES

Bryant, R.E., 2007. Data-intensive supercomputing: The case for disc. Technical Report No. CMU-CS-07-128, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA., USA.

Bu, Y., B. Howe, M. Balazinska and M.D. Ernst, 2010. Haloop: Efficient iterative data processing on large clusters. Proceedings of the VLDB Endowment, Volume 3, September 2010, Seattle, WA, USA., pp: 285-296.

Chaudhuri, S., U. Dayal and V. Narasayya, 2011. An overview of business intelligence technology. Commun. ACM, 54: 88-98.

Chen, H., 2009. Ai, E-government and politics 2.0. IEEE Intell. Syst., 24: 64-86.

Christaller, W., 1966. Central Places in Southern Germany: Translated from Die Zentralen Orte in Suddeutschland. Prentice Hall, USA., Pages: 230.

Dean, J. and S. Ghemawat, 2008. Mapreduce: Simplified data processing on large clusters. Comm. ACM, 51: 107-113.

Dyer, C., A. Cordova, A. Mont and J. Lin, 2008. Fast, easy and cheap: Construction of statistical machine translation models with mapreduce. Proceedings of the 3rd Workshop on Statistical Machine Translation, June 9, 2008, Association for Computational Linguistics, pp: 199-207.

Ekanayake, J., H. Li, B. Zhang, T. Gunarathne, S.H. Bae, J. Qiu and G. Fox, 2010. Twister: A runtime for iterative mapreduce. Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, June 21-25, 2010, Chicago, IL, USA., pp: 810-818.

Foundation, A.S., 2011. Hadoop software. http://hadoop.apache.org/

Gore, A., 1992. Earth in the Balance: Ecology and the Human Spirit. Plume, New York, ISBN: 9780395578216, Pages: 407.

Gore, A., 1998. The digital earth: Understanding our planet in the 21st Century. Aust. Surveyor, 43: 89-91.

Hayes, B., 2008. Cloud computing. Commun. ACM, 51: 9-11.

Hey, A.J.G., S. Tansley and K.M. Tolle, 2009. The Fourth Paradigm: Data-Intensive Scientific Discovery. Microsoft Research, Redmond, WA., ISBN: 0982544200, Pages: 284.

Isard, M., M. Budiu, Y. Yu, A. Birrell and D. Fetterly, 2007. Dryad: Distributed data-parallel programs from sequential building blocks. ACM SIGOPS Operat. Syst. Rev., 41: 59-72.

Janssens, W., P. De Pelsmacker, K. Wijnen and P. Van Kenhove, 2008. Marketing Research with SPSS. Prentice Hall/Financial Times, USA., ISBN: 9780273703839, Pages: 441.

Kouzes, R.T., G.A. Anderson, S.T. Elbert, I. Gorton and D.K. Gracio, 2009. The changing paradigm of data-intensive computing. Computer, 42: 26-34.

Manyika, J., M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh and A.H. Byers, 2011. Big Data: The Next Frontier for Innovation, Competition and Productivity. McKinsey Global Inst., USA., pp: 1-137.

Olston, C., B. Reed, U. Srivastava, R. Kumar and A. Tomkins, 2008. Pig latin: A not-so-foreign language for data processing. Proceedings of the ACM SIGMOD International Conference on Management of Data, June 9-12, 2008, Vancouver, BC., Canada, pp: 1099-1110.

Reilly, W.J., 1931. The Law of Retail Gravitation. W.J. Reilly, New York, Pages: 75.

Schiller, J.H. and A. Voisard, 2004. Location-Based Services. Publisher Elsevier, USA., ISBN: 1558609296, Pages: 255.

Sharony, J., 1996. An architecture for mobile radio networks with dynamically changing topology using virtual subnets. Mob. Networks Appl., 1: 75-86.

Solutions, I., 2005. Digital community best practices. Intel Solutions White Paper.

Thusoo, A., J.S. Sarma, N. Jain, Z. Shao and P. Chakka *et al*., 2009. Hive: A warehousing solution over a map-reduce framework. Proceedings of the VLDB Endowment, August 24-28, 2009, Lyoa, France, pp: 1626-1629.

Van Gael, J., A. Vlachos and Z. Ghahramani, 2009. The infinite hmm for unsupervised pos tagging. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Volume 2, August 6-7, 2009, Singapore, pp: 678-687.

Van Leeuwen, E.S. and P. Rietveld, 2011. Spatial consumer behaviour in small and medium-sized towns. Reg. Stud., 45: 1107-1119.