International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.5 (2014), pp.263-272
http://dx.doi.org/10.14257/ijmue.2014.9.5.26

# Enhancing User-friendliness of the User Taste Prediction Service Using MapReduce Framework

YoonDeuk Seo and Jinho Ahn*

*Dept. of Comp. Scie., Kyonggi Univ., Iuidong, Yeongtong, Suwon 443-760 Gyeonggi, Republic of Korea*
*{seoyd,jhahn}@kgu.ac.kr*

***Abstract***

*In our previous work, a user similarity-based contents recommendation service using NFC was proposed for the same goal. This service used a small sample because it used only information of the user who had watched a museum. However, it has been shown that there are some limitations resulting from the difficulty of accurately predicting the user's preference. In order to lift this drawback, this paper introduces a user taste prediction service using big data for improving user-friendliness to a maximum. The proposed service predicts the user's taste using big data such as Twitter and blogs. It is possible to predict the exact user's preference and might recommend more suitable contents to the user's taste because it predicts the user`s taste using big data with a variety of user's social network information. So, it can recommend contents that match the user's taste. Our simulation results show the proposed big data-based approach can give each museum visiting user more accurate recommendation service appropriate to his or her taste compared with the previous one in terms of user preferences to exhibition-related contents.*

***Keywords:*** *Museum Viewing, Contents Recommendation, User-friendliness, Social Network, MapReduce*

## 1. Introduction

Big data concept means a datasets which continues to grow so much that it becomes difficult to manage it using existing database management concepts and tools [1]. The difficulty can be related to data capture, storage, search, sharing, analytics and visualization etc. Big data can help to gain insights and make better decisions. Big data present an opportunity to create unprecedented business advantage and better service delivery. It also requires new infrastructures and ways of thinking about the way business and IT industry works. Much of the current discussion about big data analytics today focuses on managing and analyzing unstructured data from business and social sources such as e-mail, videos, tweets, Facebook posts, reviews, and Web behavior [2, 3]. While this type of big data analytics promises to provide significant value to organizations, data generated at the edge of the network from sensors and other devices represents another huge, untapped resource with the potential to deliver insights that can transform the operations and strategic initiatives of public and private sector organizations [4].

* Corresponding author: Tel.:+82-31-249-9674; Fax:+82-31-249-9673.

ISSN: 1975-0080 IJMUE

We presented a user similarity-based contents recommendation service using NFC [5]. This service used a small sample because it used only information of the user who had watched a museum. So it had some drawback that it is difficult to accurately predict the user's preference. It had a problem that it cannot recommend the exact content that matches the user taste because the accuracy of the prediction of user taste may be low.

To solve the problem of the previous service method, we present a new method of predicting the user's taste using big data. The proposed service predicts the user's taste using big data such as Twitter and blogs. It can recommend contents that match the user's taste. It is possible to predict the exact user's preference and might recommend more suitable contents to the user's taste because it predicts the user`s taste using big data with a variety of user's social network information. Also it can recommend a personalized content appropriate to the user, regardless of the number of users who have watched a museum. So, it can provide smooth services in museums the number of users is very high.

The rest of the paper is organized as follows. In section 2, we review the related work and in section 3, present our big data-based user taste prediction method. Sections 4 and 5 show experimental system architecture and evaluation results respectively. Finally, section 6 concludes this paper.

## 2. Related Work

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, 7]. The Job is referred to here as a full MapReduce program, which is the execution of a Mapper or Reducer across a set of data. A Task is an execution of a Mapper or Reducer on a slice of data. So, the MapReduce Job usually splits the input data set into independent chunks, which are processed by the map tasks in a completely parallel manner. Fig. 1 illustrates the Hadoop architecture. The Hadoop MapReduce framework consists of a single Master node that runs a JobTracker instance which accepts Job requests from a client node and Slave nodes each running a TaskTracker instance [8]. The JobTracker assumes the responsibility of distributing the software configuration to the Slave nodes, scheduling the job’s component tasks on the TaskTrackers, monitoring them and reassigning tasks to the TaskTrackers when they failed. It is also responsible for providing the status and diagnostic information to the client. The TaskTrackers execute the tasks as directed by the JobTracker. The TaskTracker executes tasks in separate java processes so that several task instances can be performed in parallel at the same time.

The MapReduce input data typically come from the input files loaded into the HDFS. These files are evenly distributed across all the nodes in the cluster. In Hadoop, computer nodes and data nodes are all the same, meaning that the MapReduce and HDFS run on the same set of nodes. At the mapping phase, the input file is divided into independent Input Splits and each split of these Splits describes a unit of work that comprises a single map task in the MapReduce job. The map tasks are then assigned to the nodes in the system based on the physically residence of the input file splits. Several map tasks can be assigned to an individual node, which attempts to perform as many tasks in parallel as it can. When the mapping phase has completed, the intermediate outputs of the map tasks are exchanged between all nodes; and they are also the input of the reduction tasks. This process of exchanging the map intermediate outputs is known as the shuffling. The reduce tasks are spread across the same nodes in the cluster as the mappers. The output of the reduce tasks is stored locally on the slave node. Technically, a MapReduce system is a framework for processing data in chunks.

 

A framework is a collection of functions that can be called by user code, and that may also call user-defined functions, which are then named callback functions. A MapReduce system has the following properties:

1. The format of the input data can be chosen freely.
2. The output data consist of pairs of arbitrary keys and values.
3. Processing happens in two consecutive phases, using two user-defined functions: mapper and reducer.
4. The mapper function creates intermediate results (pairs of keys and values of arbitrary type each) from each input chunk.
5. The reducer function is applied in unison to all intermediate results with the same key, and produces arbitrarily many final results.

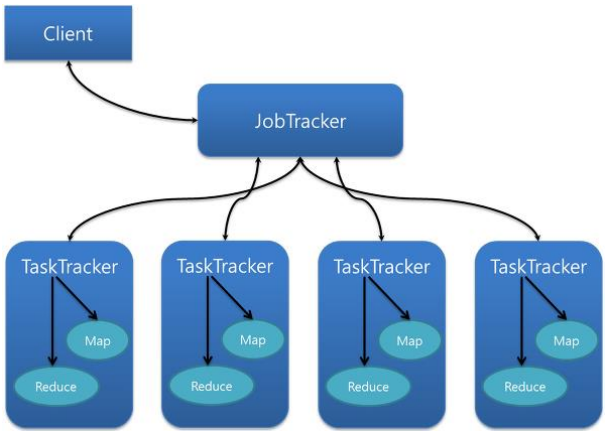


**Figure 1. Hadoop Architecture**

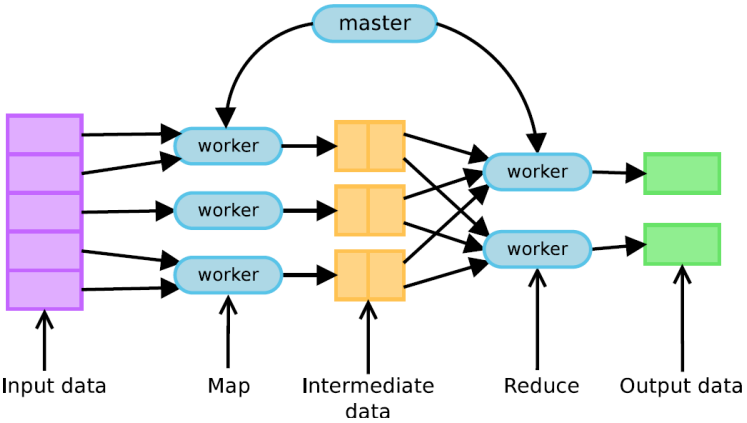


**Figure 2. Schematic overview of MapReduce processing**

Additionally, there is a main function in a MapReduce framework that has to be called by the user to specify the MapReduce program to be run as a job: which mapper and reducer function to execute and which data to process. MapReduce employs multiple user-defined functions operating on arbitrary data types. For a MapReduce program to be correct, the types of the data items and of the functions need to be compatible with each other. MapReduce owes its name to the two main phases into which its execution can be divided: the Map phase (in which mainly the mapper

function executes) and the Reduce phase (executing the reducer function). This is illustrated in Figure 2. Rectangles represent chunks of distributed (input, intermediate, and output) data, and ovals labeled "worker" represent nodes executing user-defined functions. In each phase, processing can happen in parallel. In contemporary applications, the computation is typically distributed over a cluster of hundreds to thousands of worker nodes, controlled by a single master node. In this distributed setting, large sets of data have to be serialized (converted to a representation suitable for transport over a network), communicated over the network, and de-serialized during a MapReduce computation. As this may incur huge communication costs, MapReduce also contains a feature that optimizes locality in the Map phase: the mapper function, operating on a particular chunk of input data, is typically computed on the same node on which the chunk is stored.

There is an extension to MapReduce that enable communication to be reduced even further: the combiner function. A combiner function is a third user-defined function that processes the intermediate data after the mapper function has produced it and that is executed on the same node using the data in memory [7]. This leads us to subdivide the two phases of a MapReduce computation further, as illustrated in Figure 3. The outer rectangles represent the two phases and the inner ovals sub-phases in the execution of a distributed MapReduce program with combiner function.

In the Map phase, the input is split up into chunks (split1), which are fed to the mapper function (map). The intermediate results produced are then split up again (split2; this is needed to create the partitioning desired for the final output) and sorted (sortA) to guarantee a deterministic order of processing. All intermediate results stored on one node can then be pre-reduced (*a.k.a.* combined by the combiner function, combine), which may reduce the size of the data to be sent over the network.

In the Reduce phase, the pre-reduced intermediate results received by a node have to be sorted (or merely merged, sortB) to establish the desired order, and can then be processed by the reducer function (reduce), which in turn produces a partition of the final output. To make use of this optimization, the programmer has to write a third function besides a mapper and a reducer function. Since it has been noted that the reducer function can often be reused as the combiner function in the same MapReduce program [7], additional code might not be necessary.

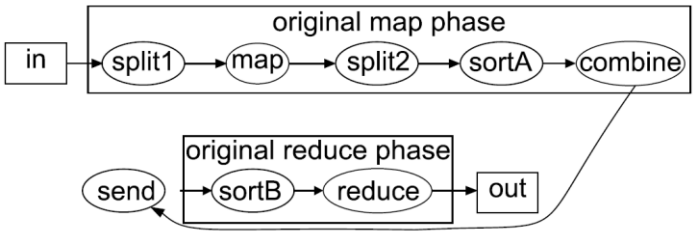


**Figure 3. Phases and sub-phases in MapReduce processing**

## 3. The User Taste Prediction Method

In this paper, we propose a user taste prediction services using big data. It predicts the taste by using the information of many users. The proposed method is composed of three stages. First, it extracts the users to be used for the method for predicting the user's taste. In order to extract the users, it uses the viewing information of the current user. It extracts the users with viewing information through the analysis of big data. It selects the users with more than half the viewing information from the user that has

 

been extracted. In Figure 4, *user[i]* represents user *i* in the big data. *User.Count* represents the total number of users. *ViewingInfo[j]* represents the viewing information of current user *j*. *SimilarTasteUserList* is a list of users who have a taste similar to the current user.

**Algorithm 1** extracting the **users**

```
1: Procedure extractUsers
2: for each user[i] in the big data, i= 1,2,…,User.Count do
3:   for each ViewInfo[j] in the ViewingInfo of current user,
       j= 1,2,…, ViewingInfo.Count do
4:    if ViewInfo[j] exists in the Taste list of user[i] then
5:      SimilarTasteUserList.add(user[i]);
6:    end if
7:   end for
8: end for
9: return SimilarTasteUserList
```

**Figure 4. Extracting the Similar Taste User**

Second, by using a user that is extracted, it generates a prediction contents list for the prediction of user taste. Because it is the users who have a taste similar to the user, the user also prefers the contents they prefer. So, in this paper, we extract the contents that users with similar tastes like in this way. In order to extract these contents, we use a frequency analysis method. It performs a frequency analysis method. And it extracts the contents frequently. After applying this process to all users extracted, it generates the predicted contents list. In Figure 5, *FREQ()* is the frequency analysis method. *MODE()* extracts the contents frequently. *PredictionContentsList* is a list of the predicted.

Finally, in order to recommend appropriate contents, it generates the contents recommendation list. The content recommendation list is determined by comparing the analyzed content recommendation list which is obtained by using the viewing information of the user with the prediction contents list obtained in the previous step. It extracts the contents that exist at the same time in the two lists. Then, in the remaining list, it extracts the contents that have been preferred by many users. It recommends the contents that are extracted through these processes to the user.

**Algorithm 2** extracting the **Prediction**Content**s**

```
1: Procedure extractPredictionContents
2: for each user[i] in the SimilarTasteUserList,
     i= 1,2,…, SimilarTasteUserList.Count do
3:   var result = FREQ(user[i]);
4:   PredictionContentsList.add(MODE(result));
5: end for
6: return PredictionContentsList
```

**Figure 5. Extracting the Prediction Contents**

## 4. Experimental System Architecture

Figure 6 shows the system architecture of the content recommendation service based on the prediction of user taste using big data. The proposed system consists of two parts: Museum Viewing System and Big data Platform. Museum Viewing System collects viewing information, recommends content to users. Big data Platform collects big data information and predicts user's taste. User Viewing Info Collector collects user`s viewing information that is viewing exhibits and viewing path. Viewing Taste

Analyzer through the collected information analyses the user's taste. By using predicted user taste information, Contents Recommender recommends customized content to the users.

Figure 7 shows the Data Collector Module that collects information on big data that is used to predict user taste. By using Crawler, it collects the contents of the Web pages such as blogs and social media such as Twitter. By using Flume, it stores the collected data in HDFS.

Figure 8 shows the Data Processing Module that analyses data collected from the Data Collector by using NLP (natural language processing). It performs NLP process using the user`s viewing information. At this time, NLP requires a long analysis time. So, it uses the MapReduce that can shorten the NLP processing time. By using Data Mining, it extracts the available data to predict user taste. The User Taste Predicting Module predicts a user taste through comparative analysis of the information processed by the data processing module. Through the predicted information, the proposed service recommends a personalized content to the user.

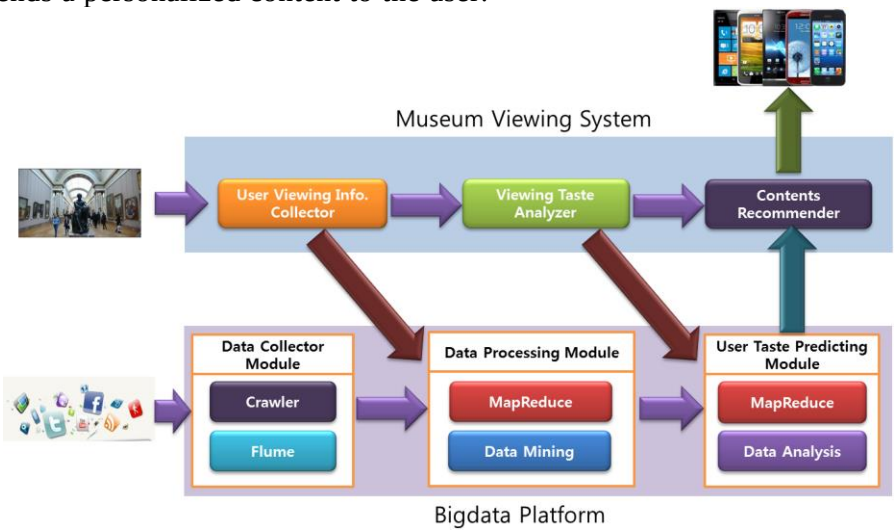


**Figure 6. User Taste Prediction System Architecture**

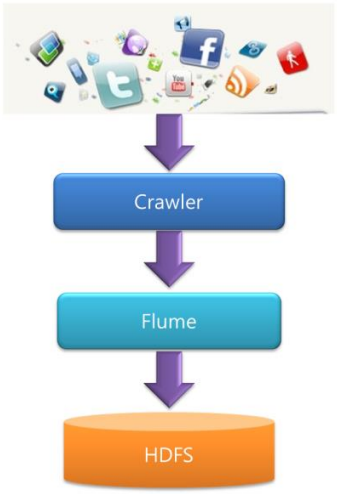


**Figure 7. Data Collector Module**

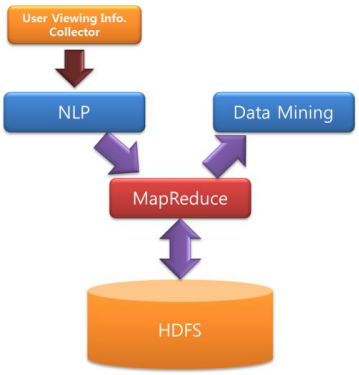


**Figure 8. Data Processing Module.**

## 5. Performance Evaluation

In this section, we can show how effectively the proposed service can recommend users their favorite contents and solve the problems of existing service. Our experimental environment is in Table 1.

We assume that the number of exhibition rooms is ten and each room has three areas. So, there are a total of 30 areas. Each area has two exhibits. So, there are a total of 60 exhibits. Each exhibit has one related contents. So, the total number of contents is 60. The number of previous visitors is 100. The number of user in the big data is 500. We assume that previous visitors watch at least three exhibitions, stay at least five areas, and watch long at least five exhibits.

**Table 1. Experimental Environment**

| Parameter | Value |
|---|---|
| No. of Exhibition Rooms | 10 |
| No. of Area | 30 |
| No. of Exhibits | 60 |
| No. of Contents | 60 |
| No. of Previous Visitors | 100 |
| No. of Big data User | 500 |

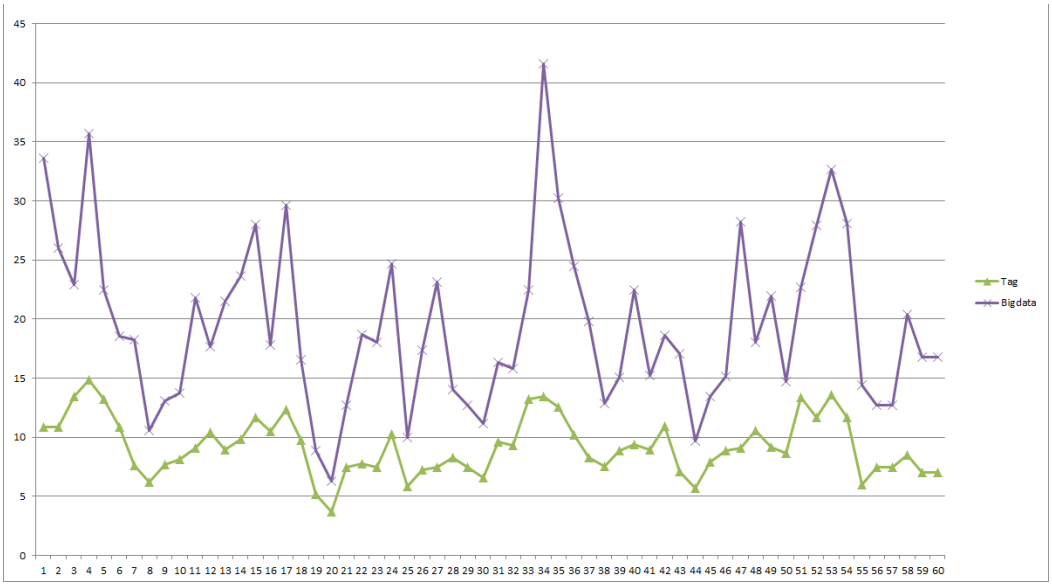


**Figure 9. Contents Preference of User A**

Experiments are performed to find out how the preference values for the present user are changed depending on which services are applied, that is, users similarity method based on tagging patterns [5] and proposed method. In order to evaluate the proposed method, we set up two users. The first user has watched rooms 1, 3, 5, 7 and 9 and stayed at area 3, 8, 15, 20 and 27 and tagged exhibits 2, 3, 14, 17 and 27. His or her favorite contents would be 1, 3, 17, 34 and 53. The last user has watched rooms 1, 2, 4, 8 and 9 and stayed at area 3, 6, 12, 22 and 26 and tagged exhibits 3, 5, 10, 22 and 25. His or her favorite contents would be 4, 5, 24, 44 and 50.

In the first experiment, Figure 9 shows that each method is applied about the first user. The contents 1 which is the first user`s favorite contents have low preference values in the tagging pattern based methods, but high preference values in the proposed method.

In the second experiment, Figure 10 shows that each method is applied about the second user. The contents 44 which is the third user`s favorite contents have low preference values in the tagging pattern based methods, but high preference values in the proposed method.

Through experiments, we can show that the proposed method is better than the tagging patter method. So, the proposed service can recommend appropriately contents more than tagging pattern method.

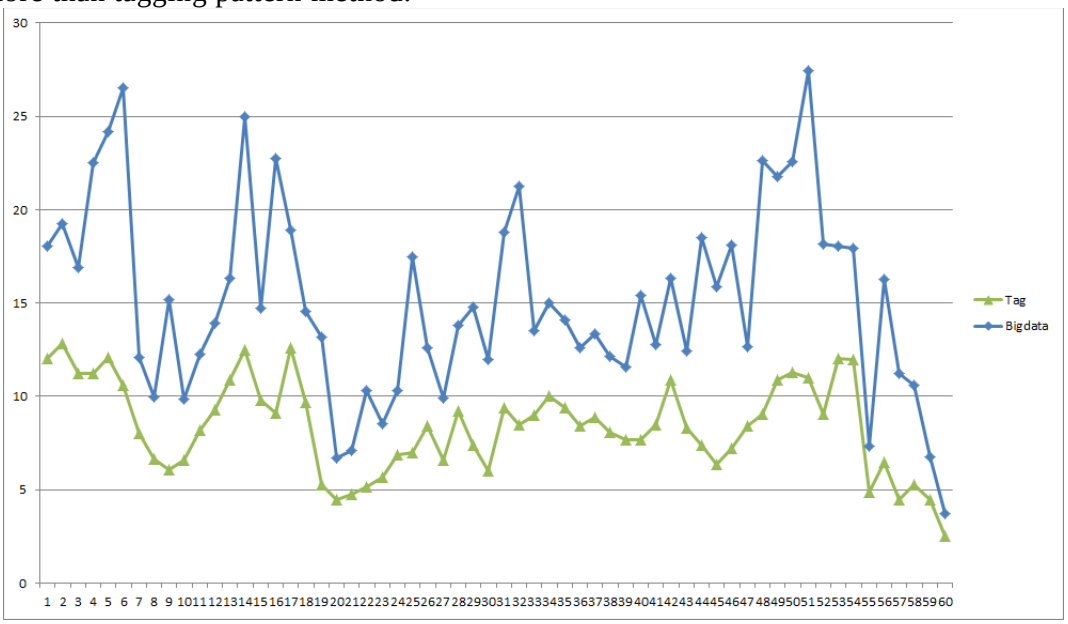


**Figure 10. Contents Preference of User B**

## 6. Conclusions

This paper introduced a method of predicting the user's taste using big data. The proposed service predicts the user's taste using big data such as Twitter and blogs. It recommends a personalized content to user based on the predicted taste. It is possible to predict the exact user's preference and might recommend more suitable contents to the user's taste because it predicts the user`s taste using big data with a variety of user's information. Also it can recommend a personalized content appropriate to the user, regardless of the number of users who have watched a museum. So, it can provide smooth services in museums the number of users is small. Through experiments, we can

 

show that the proposed method is better than the tagging pattern method in terms of user preferences to exhibition-related contents.

## Acknowledgements

This work was supported by the Gyeonggi Regional Research Center (GRRC) and Contents Convergence Software (CCS) research center in Korea(Project No.: GRRC Kyonggi 2013-B04).

## References

[1] D. Agrawal, S. Das, and A. E. Abbadi, "Big data and cloud computing: New wine or just new bottles?", PVLDB, ACM Computing Surveys, vol. 3, no. 2, **(2010).**
[2] N. Marz and J. Warren, "Big Data: Principles and best practices of scalable real-time data systems", Manning Publications **(2013).**
[3] A. Bifet and E. Frank, "Sentiment knowledge discovery in Twitter streaming data", In Proceedings 13th International Conference on Discovery Science, **(2010),** pp. 1-15.
[4] H. Chen, R. Chiang and V. Storey, "Business Intelligence and Analytics: From Big Data to Big Impact", MIS Quarterly, vol. 36, no. 4, **(2012).**
[5] Y. D. Seo and J. H. Ahn, "Reliable Contents Recommendation Service Based On Similarity Of Users Tagging, Proceedings of the second International Conference on Conference On Computers", Networks, Systems, and Industrial Engineering (CNSI2012), **(2012),** pp. 474-479.
[6] J. Berthold, M. Dieterle and R. Loogen, "Implementing Parallel Google Map-Reduce in Eden",s In Proc. Euro-Par, LNCS 5704, **(2009),** pp. 990-1002.
[7] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters", Communications of the ACM, vol. 51, **(2008).**
[8] Apache Hadoop project, Web Page. http://hadoop.apache.org/.

## Authors

**Yoon-Deuk Seo**, he received his B.S. and M.S. degrees in Computer Science from Kyonggi University, Korea, in 2008 and 2010, respectively. He has been a Ph.D. student in Department of Computer Science, Kyonggi University from 2010. His research interests include distributed computing, RFID systems, P2P networks and group communication.

**Jinho Ahn**, he received his B.S., M.S. and Ph.D. degrees in Computer Science and Engineering from Korea University, Korea, in 1997, 1999 and 2003, respectively. He has been a full professor in Department of Computer Science, Kyonggi University. He has published more than 70 papers in refereed journals and conference proceedings and served as program or organizing committee member or session chair in several domestic/international conferences and editor-in-chief of journal of Korean Institute of Information Technology and editorial board member of journal of Korean Society for Internet Information. His research interests include distributed computing, fault-tolerance, sensor networks and mobile agent systems.