Big Data Processing with MapReduce for E-Book

Tae Ho Hong[2], Chang Ho Yun[1,2], Jong Won Park[1,2], Hak Geon Lee[2],
Hae Sun Jung[1] and Yong Woo Lee[1,2]

[1]The Ubiquitous (Smart) City Consortium
[2]The University of Seoul, Seoul, South of Korea
{ghdxogh, touch011, comics77, withteas}@uos.ac.kr, banyasun@gmail.com,
ywlee@uos.ac.kr

Abstract

Evolution of IT and computer has made e-books popular day by day. In this paper, we are interested in searching a word in e-books. However, it is impossible to search a word in digitized e-books if they consist of image files such as JPG and PDF. Our solution to this problem is to transform the image file based e-books into text files based e-books to enable searching a word in e-books. We use EPUB, a XML-based text file, which is defined by IDPF(International Digital Publishing Forum). That is, we convert the image file based e-books into EPUB format e-books, so that searching a word in e-books can be done without any problem. The converting job should deal with very big data usually and require a lot of computing power. If we do the conversion in an usual personal computer, it would take a lot of processing time or it might be impossible for us to complete it. We used MapReduce model with a cluster system which enables us to perform the conversion successfully and reduce the processing time. This paper presents our Hadoop-based e-book Conversion System which is a distributed computing framework to transform the image based e-books into EPUB format e-books. Our experimental system consists of up to 15 cluster nodes. This paper evaluates the performance of the experimental system which processes the conversion of up to 2TB(Terra Byte) image files into EPUB files with a 15 nodes cluster system. We analyzed the processing time when the number of nodes in the cluster system was varied. We also analyzed the improvement effect when the dpi of the image file was varied. The performance evaluation confirmed us that the Hadoop-based e-book Conversion System successfully processed the big data for e-book.

Keywords: *E-book, Big Data, MapReduce, Hadoop, EPUB, Internet*

1. Introduction

"An electronic book (variously, e-book, ebook, digital book, or even e-edition) is a book-length publication in digital form, consisting of text, images, or both, and produced on, published through, and readable on computers or other electronic devices" [1]. The Oxford Dictionary of English defines the e-book as "an electronic version of a printed book," but e-books exist without any printed equivalent. E-books are usually read on dedicated e-book readers. Personal computers, tablet personal computers and many mobile phones can also be used to read e-books." Evolution of computer systems, multi-media facilities and computer communication technology make the e-book essential in the modern era.

The e-book can be made in various kinds of formats such as txt, html, pdf, jpg and EPUB [7]. However, if the e-book has the format of an image file such as pdf and jpg, then it is impossible for us to search a word in the e-book. Unfortunately, we are interested in searching a word in the e-books. This paper presents our solution to this matter.

Our solution to the problem is to transform the e-book images into e-book texts so that we can search words in e-books. EPUB, a XML-based text file was used. IDPF(International Digital Publishing Forum) defined EPUB which is a good news to the problem. Once the e-book images is converted into EPUB format e-books, we can search a word in the e-book without any problem. The OCR [8] (Optical Character Recognition) is used to recognize the characters from the image and the recognized characters is used to create the EPUB files.

The converting job should deal with very big data usually and require a lot of computing power. If we do the conversion in a usual personal computer, it would take a lot of processing time or it might be impossible for us to complete it. Since the MapReduce model is useful when we process big data, we used the MapReduce model with a cluster system which enabled us to perform the conversion successfully and reduce the processing time.

MapReduce [2, 3] is a programming model to processing big data and usually uses a cluster environment that consists of distributed and parallel computers or recently uses a cloud computing environment. It is based on the ‘map’ and the ‘reduce’ functions commonly used in functional programming. The map function takes a series of input pairs and produces a set of intermediate key/value pairs. The MapReduce framework makes groups of all intermediate values associated with the same intermediate key and passes them to the ‘reduce’ function. The ‘reduce’ function receives an intermediate key and a set of values. Then, it merges these values together to form a possibly smaller set of values.

This paper presents our Hadoop-based E-book Conversion System which is a distributed computing framework to transform the image based e-books into EPUB format e-books. Hadoop[4] is an open-source framework that was derived from Google's MapReduce and Google File System (GFS) [5]. It consists of Hadoop Distributed File System (HDFS) and MapReduce. It is a good solution for big data processing of distributed applications which might require the computing power of thousands of computation-independent computers for over petabytes of data. When the data processing fails or times out, that part of the job is can be rescheduled. If the data is not readable, Hadoop can read the replicated data on the same rack/switch and different racks by using replication of HDFS so that a fault-tolerant processing is possible [6].

We have prepared our own experimental system and evaluated the performance using the experimental system which processes the conversion of up to 2TB (Terra Byte) image files into EPUB files. The performance evaluation confirmed us that the Hadoop-based E-book Conversion System successfully processed the bid data for the e-book.

This paper is organized as follows. Section 2 simply introduces the related works, Section 3 explains our E-book Conversion System. Section 4 presents the performance evaluation. Finally, Section 5 gives conclusions and explains future works.

2. Related Works

INSPIRE [9] and Project Gutenberg [10, 11] are similar to our research. INSPIRE has e-Infrastructures and is a part of D4Science-II Project [12]. It can digitalize and index documents that are created by CERN, DESY, Fermilab and SLAC. OCRopus [13] is used as an OCR tool. It converts the pdf format documents into the hOCR [14] format. In order to do indexing, it uses the Lucene [15]. It uses the MapReduce with Hadoop. The difference from our work is that it is to make digitalized scientific documents by indexing the contents in documents, but not to make the e-book.

Gutenberg project uses DP (Distributed Proofreaders) as a methodology to make the e-book. DP uses ABBYY [16] FineReader as common OCR tools to make the e-book. It monitors and does proves processes. Gutenberg is not to produce images or PDF files. Therefore there is no consideration for distributed and parallel processing. However, our E-book Conversion System uses tesseract [17], as an open source software which is supported by Google and about 10 times faster than OCRopus which is used in INSPIRE.

3. The E-book Conversion System

3.1. Architecture

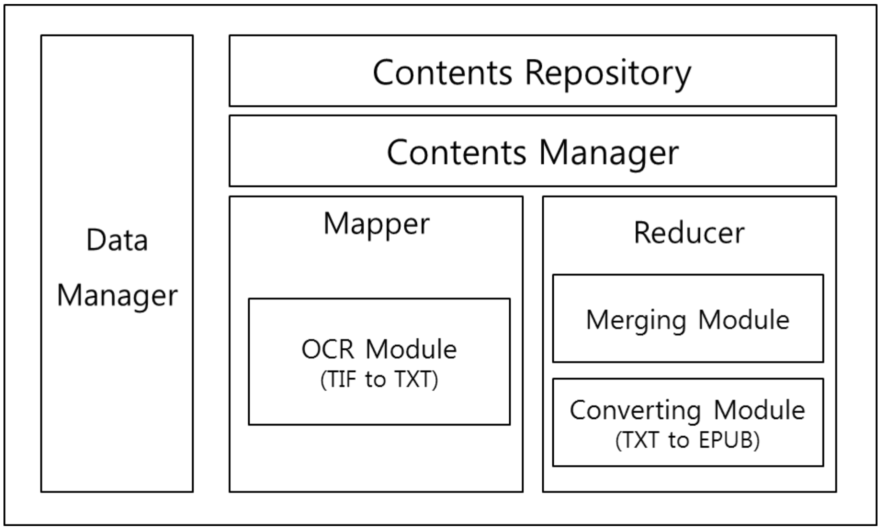


Figure 1. Architecture of the E-book Conversion System

The architecture of the E-book Conversion System is shown in Figure 1. The components of our E-book Conversion System are the “Data Manager”, the “Contents Repository”, the “Contents Manager”, the “Mapper” and the “Reducer”. The “Data Manager” manages input images and passes it to the “Contents Repository”.

The “Contents Repository” is a repository to store all e-book files such as image, text and EPUB format file. It receives the input data from the “Data Manager. It also sends the data to the “Contents Manager” and, from the “Contents Manager”, receives the intermediate data such as the data processed in the ‘map’ step and the ‘reduce’ step and stores them.

The “Contents Manager” gets the image files from “Contents Repository” for the ‘map’ step and the ‘reduce’ step and passes them to the “Mapper” and “Reducer” respectfully. It receives the intermediate data, from the “Mapper” and “Reducer”.

The “Mapper” sorts image files which are received from the “Contents Manager” into a set of intermediate key/value pairs. The “OCR module” of Mapper uses Tesseract OCR software to convert image files into text files.

The “Reducer” merges the set of intermediate key/value pairs which are received from the “Contents Manager”. The “Merging module” of the “Reducer” merges text files of same "book-ID". The “Converting module” of the “Reducer” converts these text files into EPUB format files.

3.2. Operation

The workflow of the E-book Conversion System is shown in Figure 2.

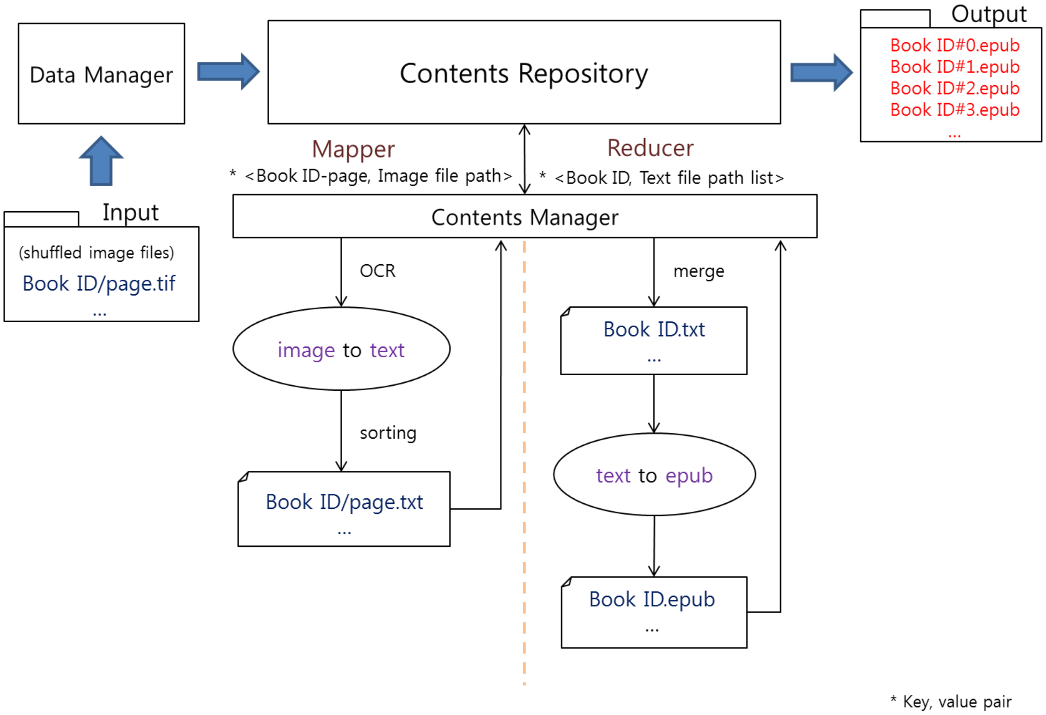
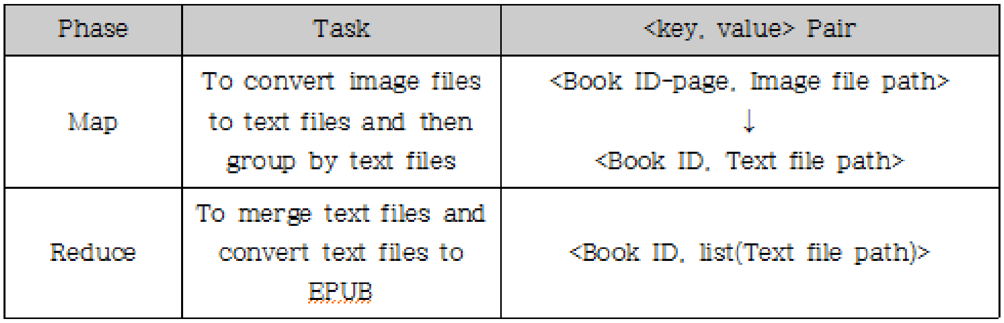


Figure 2. Workflow of the E-book Conversion System

Image files of an e-book are randomly transmitted to the "Data Manager’. The image files are in the format of "tif": for example, page_number.tif under the directory of "book-ID". Image files that are brought into the "Data Manager’ are stored in the "Contents Repository". The "Content Repository" is a storage space similar to the HDFS of Hadoop.

The key of the “Mapper” input is "book-ID" and “page-number”. The value of the “Mapper” input is the path of image files as shown in Table 1. The “Mapper” receives image files according to path of image files through the “Contents Manager”. The “Mapper” converts the image files to text files through “OCR module”. After that, the text files are stored in the “Contents Repository” through the “Contents Manager”. The “Contents Manager” sorts the text files according to the name and provides their path to the “Reducer”. Instead of sending the contents of each text file to the “Reducer”, we store the text files in the “Contents Repository” and just use the path of each text file as the value of key/value pairs. Finally, the e-book in the EPUB format is stored in the “Contents Repository”.

Table 1. The Definition of Key-Value pair in thestep of the 'Map' and the ' Reduce'



Phase	Task	<key, value> Pair
Map	To convert image files to text files and then group by text files	<Book ID-page, Image file path> ↓ <Book ID, Text file path>
Reduce	To merge text files and convert text files to EPUB	<Book ID, list(Text file path)>

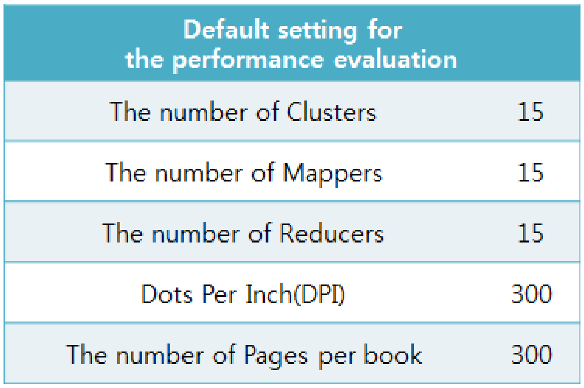
4. Performance Evaluation

4.1. An Experimental Setup

For the performance evaluation experiment, we used a fifteen nodes cluster, where 15 nodes had Intel core i5 760 2.8Ghz Processors and each node had 8 GB of physical memory. There, each node of the cluster was connected through a Giga-bit Ethernet switch and ran a Ubuntu Linux 11.10 64 bit server edition. The JVM version 1.6.0_22 was used for MapReduce with Hadoop version 0.20.

For the default setting in our performance evaluation, the number of the nodes in the cluster system was 15 and the number of mappers and reducers was 15 for each. The resolution of image files was 300dpi and a book consisted of 300pages. The default setting values of our performance evaluation experiments are summarized in Table 2.

Table 2. Default Setting Values for the Performance Evaluation



Default setting for the performance evaluation	
The number of Clusters	15
The number of Mappers	15
The number of Reducers	15
Dots Per Inch(DPI)	300
The number of Pages per book	300

4.2. Performance

4.2.1. Performance according to the capacity of data: We assumed that an e-book consists of 300 pages, each page was an image file which had the resolution of 300dpi and the size of each page was 19.9 mega-bytes (MB). Therefore, the size of an e-book was 5970 (around 6 giga-bytes (GB)). We varied the size of the volume size of processing data by varying the number of e-books to be processed (converted). Table 3 shows the number of books at each case and their corresponding volume size. Figure 3 shows the processing time accordingly. There, x axis denotes the volume size of processed data (terra-bytes (TB)) for the conversion and y axis denotes the processing time (seconds). We increased the volume size of processed data up to 2 terra-bytes. We show the three different cases: the number of nodes in our experimental system was set up at 5, 10 and 15 nodes respectfully. There, we see that the processing time increases linearly when the volume size of processed data is increased in all cases. Consequently, it can be found that the processing time decreases or the processing speed increases when the number of nodes was increased.

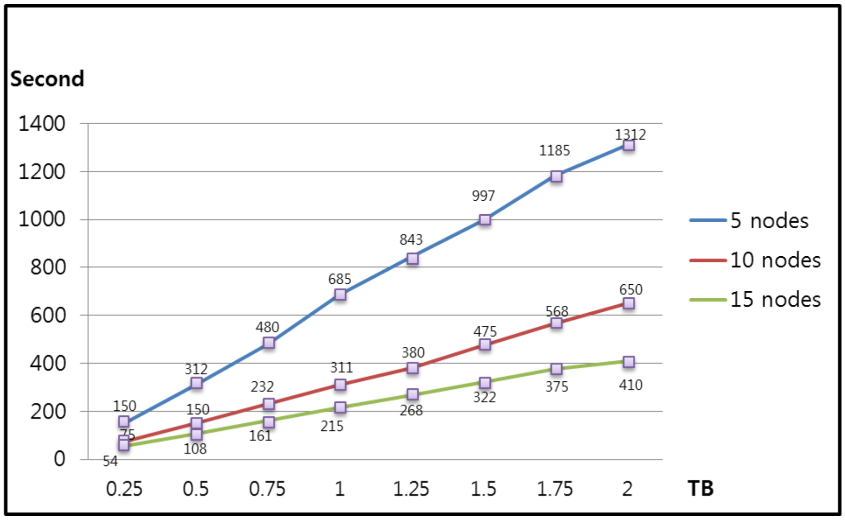


Figure 3. Processing time according to the data capacity

Table 3. The Data Volume Size and its Corresponding Value in the Number of Books

TB	0.25	0.5	0.75	1	1.25	1.5	1.75	2
Number of books	43	86	129	172	215	258	301	344

4.2.2. Performance according to the dpi of the image file: Figure 4 shows the processing time (second) when we varied the resolution of the image file (dpi) in the cluster system which had 15 nodes. When the resolution was under 300dpi, the processing time increased almost linearly. When the resolution was above 300dpi, we saw that there was very little improvement in processing time, that is, the performance improvement was near saturation. Therefore, we can say that our experimental system is reasonably efficient in processing the image below 300dpi.

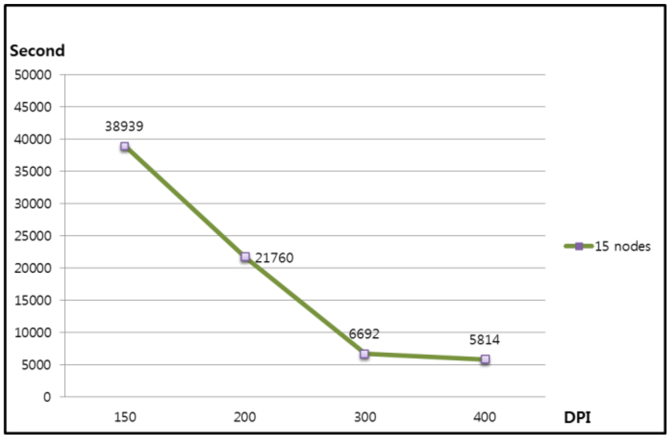


Figure 4. Processing Time According to the Resolution of the Image File

4.2.3. Performance according to the number of mapper: Figure 5 shows the processing time (second) when we varied the number of mapper. There, the resolution of the file was fixed to 300dpi and the cluster system which consisted of 15 nodes was used.

This experiment shows that the processing speed was improved approximately two times when the number of mapper was increased from 15 mappers into 30 mappers and three times when the number of mapper was increased from 15 mappers into 45 mappers. There was little improvement in the processing speed above 60 mappers, that is, the processing speed improvement was almost saturated.

The reason of saturation in the processing speed improvement is analyzed due to the the overhead of administrating multiple mappers in a node at the same time. It can be proved by the fact that when we increased the number of mapper in a node, the processing speed improvement decreased. It is observed in the Figure 5.

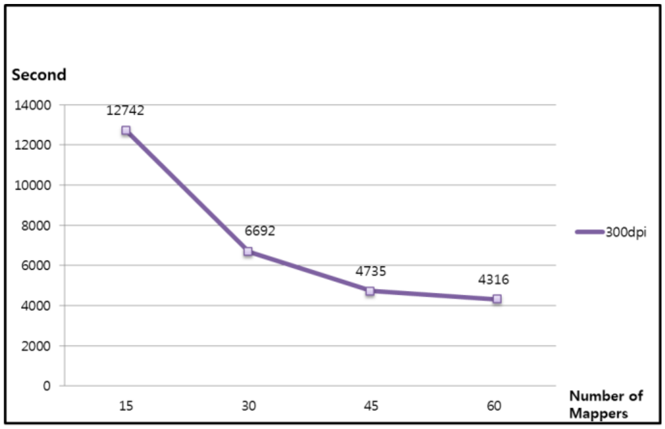


Figure 5. Processing Time According to the Number of Mapper

4.2.4. Performance according to the number of reducer: Figure 6 shows the processing time (second) when we varied the number of reducer. There, the resolution of the file was fixed to 300dpi and the cluster system which had 15 nodes was used. It was seen that when the number of reducer was increased, the processing time linearly decreased. It is because the number of reducer does not exceed the number of nodes, that is, 15 nodes, so that each node is assigned to have one reducer at most.

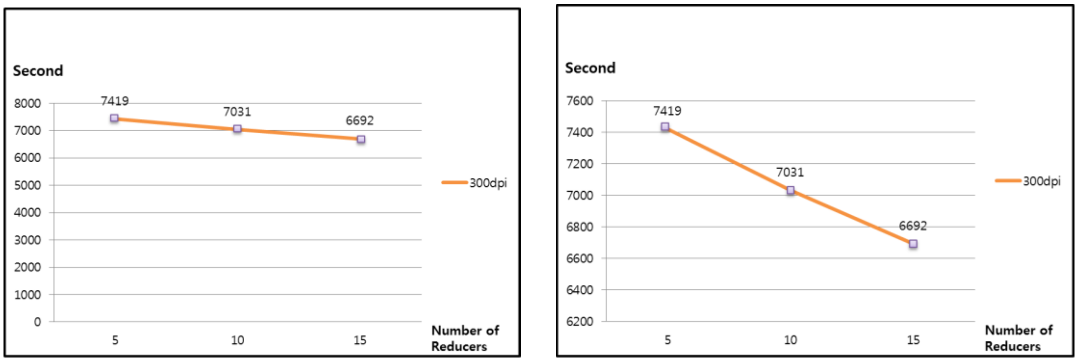


Figure 6. Processing Time According to the Number of Reducer

5. Conclusion

We made our own E-book Conversion System which uses big data processing technique to convert image files into EPUB format files so that we can do searching words in e-books. To recognize the characters in the image files, we used OCR technologies. MapReduce with Hadoop was used for the big data processing.

In performance evaluation, we have confirmed that our E-book Conversion System had successfully worked. From the performance evaluation, we found the following significant results. First, we evaluated the processing time when the number of nodes were varied according to the capacity of data. We varied the size of the volume size of processing data by varying the number of e-books to be processed. There, we saw that the processing time increased linearly when the volume size of processed data was increased in all cases. Consequently, it was found that the processing time decreased or the processing speed increased when the number of nodes was increased.

Second, we have evaluated the processing time according to the resolution of image files. The result of the experiment showed that when the resolution was above 300dpi, there was little improvement in processing time. From this, we found that processing time was efficient at below 300dpi.

Third, we have evaluated processing speed according to the number of mapper. We found that, when the number of mappers was increased, the processing speed also increased, but, processing speed improvement almost saturated above 60 mappers. The reason of the processing speed improvement saturation was analyzed to be due to the overhead of administrating multiple mappers in a node at the same time.

Fourth, and finally, we have evaluated the processing time when we varied the number of reducers. It was seen that when the number of reducer was increased, the processing time linearly decreased. We found that the increasing of number of reducer did not have a significant impact on the speed of processing.

In the future work, we will compare the accuracy of character recognition and speed using another OCR solution such as ABBYY, ARMI.

Acknowledgements

This study was supported by Ministry of Education, Science and Technology in Korea and KISTI (Korea Institute of Science and Technology Information). This study was supported by the Seoul Research and Business Development Program of Seoul, Korea(10561), the Smart(Ubiquitous) City Consortium in the University of Seoul, Seoul Grid Center. We would like to give thanks to Mr. Jun Oh Kim, Mr. Kyoung Gyu Lee, Mr. Eui Dong Hwang, Mr. Sung Min Kim, Mr. Cheol Sang Yoon and the staffs of Seoul Grid Center and the members of Smart(Ubiquitous) City Consortium for their contribution to this research.

References

[1] G. Eileen and R. G. Musto, “The Electronic Book”, in Suarez, Michael Felix and H. R. Woudhuysen, The Oxford Companion to the Book, Oxford: Oxford University Press, **(2010),** pp. 164.

[2] J. Dean and S. Ghemawat, “MapReduce: simplified data processing on large clusters”, Communications of the ACM, vol. 51, no. 1, **(2008)**, pp. 107-113.

[3] J. Lin and C. Dyer, “Data-Intensive Text Processing with MapReduce”, Morgan & Claypool Publishers, **(2010)**.

[4] M. Bhandarkar, “MapReduce programming with apache Hadoop”, Proceedings of the IEEE International Symposium on Parallel & Distributed Processing (IPDPS), **(2010)** April 19-23; Atlanta, USA.

[5] S. Ghemawat, H. Gobioff and S. T. Leung, “The Google file system”, Proceedings of the nineteenth ACM symposium on Operating systems principles, **(2003)** New York, USA.

[6] K. Shvachko, H. Kuang, S. Radia and R. Chansler, “The hadoop distributed file system”, Proceedings of IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), **(2010)** May 3-7; Nevada, USA.

[7] EPUB, http://idpf.org/epub/.

[8] S. Mori, C. Y. Suen and K. Yamamoto, “Historical review of OCR research and development”, The IEEE, vol. 80, no. 7, **(1992)**, pp. 1029-1058.

[9] J. Klem and J. Iwaszkiewicz, “Physicists Get INSPIREd: INSPIRE Project and Grid Applications”, IOP science Journal of Physics: Conference Series, vol. 331, no. 8, **(2011)**.

[10] Project Gutenberg, http://www.gutenberg.org/.

[11] G. B.Newby and C. Franks, “Distributed proofreading”, Proceedings of Joint Conference on Digital Libraries, **(2003)** May 27-31.

[12] D4Science, http://www.d4science.eu/.

[13] T. M. Breuel, “The OCRopus open source OCR system”, Document Recognition and Retrieval XV, vol. 6815, **(2008)**.

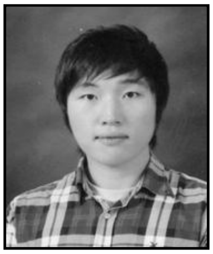
[14] T. M. Breuel, “The hOCR microformat for OCR workflow and results”, Proceedings of Ninth International Conference on Document Analysis and Recognition (ICDAR), **(2007)** September 23-26; Parana, Argentina.

[15] Lucene, http://lucene.apache.org/core/.

[16] E. Mendelson, “ABBYY FineReader Professional 9.0”, http://knowledgebase.abbyy.com/.

[17] R. Smith, “An overview of the Tesseract OCR engine”, Proceedings of Ninth International Conference on Document Analysis and Recognition (ICDAR) **(2007)** September 23-26; Parana, Argentina.

Authors



Tae Ho Hong

Tae Ho Hong received his B.S. in Division of Computer from the Howon University, Korea in 2011 and has been pursuing his Master degree in the Electronic, Electrical, and Computer Engineering from the University of Seoul, Korea. His current research interest includes e-book, OCR, grid computing, cloud computing, distributed computing, and artificial intelligence.

Chang Ho Yun

Chang Ho Yun received his B.S. and Master degree and is now pursuing his Ph.D in Electronic, Electrical, and Computer Engineering from the University of Seoul, Korea. As a research, he have been working for the Smart (Ubiquitous) City Consortium, which is supported by City of Seoul, Korea and is one of the largest and important Ubiquitous-City R&D Projects in the world. His current research interest includes ubiquitous-city, grid computing, cloud computing, ubiquitous computing and semantic web.

Jong Won Park

Jong Won Park received his B.S. and Master degree in the Electronic, Electrical, and Computer Engineering from the University of Seoul, Korea in 2009 and 2011. Since 2011, he has been pursuing his Ph.D degree in same department. He currently works for Smart (Ubiquitous) City Consortium as a researcher. The consortium carries out the U-city project with five million U.S. dollars funded and operated by Seoul Metropolitan Government of Korea. His current research interest includes grid computing, cloud computing, ubiquitous computing, and semantic web.

Hak Geon Lee

Hak Geon Lee received his B.S. and is now pursuing his Master degree in Electronic, Electrical, and Computer Engineering from the University of Seoul, Korea. His current research interest includes ubiquitous-city, grid computing, cloud computing.

Yong Woo Lee

Yong-Woo Lee has been a professor at the School of Electrical and Computer Engineering, the University of Seoul, Korea since 1999. He received his Ph.D. degree in Computer Science from the Department of Computer Science at the University of Edinburgh, United Kingdom and B.S. (1981) degree in Electrical Engineering from Seoul National University, Korea, respectively. Before joining the University of Seoul, he was a senior research scientist at KIST (Korea Institute of Science and Technology) under the Ministry of Science and Technology, Korea, during 1982-1998. He also worked as 30 principal researcher at KERIS (Korea Education and Research Information Service) under the Ministry of Education, Korea, during 1998-1999 and as an international engineer at Schlumberger Technical Services Inc. during 1981. Currently he is the president of the Korean National Standard Committee for ISO JTC1/SC22, supported by the Ministry of Industry and Resource, Korea.

He is also the chairman of the Academic Activity Board of Directors at KSII (Korean Society of Internet Information). He has been the member of Board of Chairs for Grid computing in Korea since 2002. He served many international conferences as the general chair. As the president of the Ubiquitous (Smart) City Consortium which includes SK Telecom, LG-CNS, etc. as industry members and Seoul National University, Korea University, Yonsei University, etc. as academic members, he has been leading the five million U-city project funded and operated by Seoul Metropolitan Government of Korea, since 2005 and receive the Korea Best Award, from the Korea Herald Newspaper, a famous daily newspaper in Korea in 2007. He is a member of Board of Directors at the Korea-British Alumni Association and at the Alumni Association of Seoul National University. His current research interests include ubiquitous computing, grid computing and high speed Internet and applications.