Implementation of Map-Reduce Based Distributed System

Wang Yidan[1, a], Liu Yi[2, b],Gao Boqi[3]

[1]University of Birmingham, Birmingham, UK

[2]Beijing Institute of Fashion Technology, Beijing, China

[3]College of Economic Management, Dalian Jiaotong University, Dalian,China

[a] yidan12.wang@gmail.com, [b]mableliuyi@163.com

Keywords: Distributed System; Map-reduce; System Performance

Abstract: A distributed system consists of a collection of computing components. It enables computers to coordinate their behavior and functions. Also, as a model that processing large sets of data with a parallel, distributed algorithm on a cluster, Map-Reduce has been widely used during the past decades. In this project, a distributed system as the implementation of Map-Reduce proposed to remove duplicate lines from a text. It proved to work with high efficiency and it especially works well for a relatively large set of data which is unable to processed by a single host. After designing and implementing the required system, the evaluation of the system is performed and it also provides a further analysis by comparing with real Map-Reduce.

Introduction

A distributed system consists of a collection of computing components. It enables computers to coordinate their behavior and functions and share their computing resources to provide a transparent service to the system users[1]. Distributed system takes the advantages of the connected hosts and owns stronger computing ability than a single host. At the same time, Map-Reduce as a model that processing large sets of data with a parallel, distributed algorithm on a cluster, it has been widely used during the past decades[2]. This project aims to implement a Map-Reduce based distributed system and the system is proposed to remove duplicate lines from a text file. When the processing data comes to a large amount, a distributed system is expected to works better than individual host. A series of experiment is performed and further analysis is also given to provide a better understanding of the performance of the system. Since it based on Map-reduce, the construction of this system would then be compared with real Map-Reduce model.

This project involves both design and implementation of the required distributed system. The designing process would be given first. Following with the construction of the system. The evaluation of the system is performed according to the size of file that be processed. Execution time are recorded and they help to illustrate the degree of how the system improved the efficiency. An analysis of the results would be given including a comparison with the real Map-reduce model.

System Design

The host machine that reads the given file and assigns the work to other machines can be viewed as the client side in the system. It reads every line of the given file and number the lines(words) which would help the host to order the words at the final step. Assigning the words by using hash code make sure the same word would send to the same machine which avoid any extra efforts to check or remove duplicate lines after the files come back to the host machine. And do the mod computation of hash code helps to balance the load of the clusters to some degree. However, still there is the possibility that some of the machines handle more jobs than the others. At the server side, after opening a server socket that keep listening for connections and receive a set of words and then write them into a file. The file would then entering the reducing process. In this case, it will first sorting

the words by alphabetic order and remove the duplicate lines and keep the words appeared earlier. By keep the words in alphabetic order could ensure the duplicate lines moving closed. After removing the duplicate line, the working machines would sort the words again but following the number order and send it back to the host. Order by number is helpful when sorting the words according to the original arrangement. Finally the host machine would receive multiple files(depend on the reducer number) and all files following the number order. Sorting them together by using the merge sort algorithm. The final output file then generated after these steps. The flow chart of this system processing is shown as Figure 1.

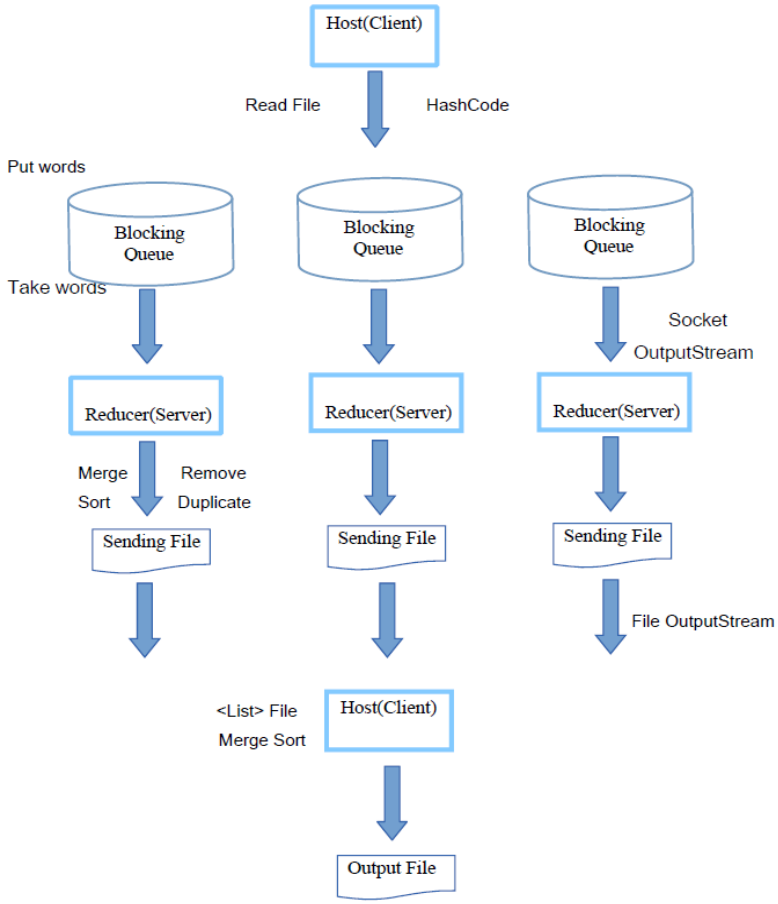


Figure 1. The flow chart of the system process

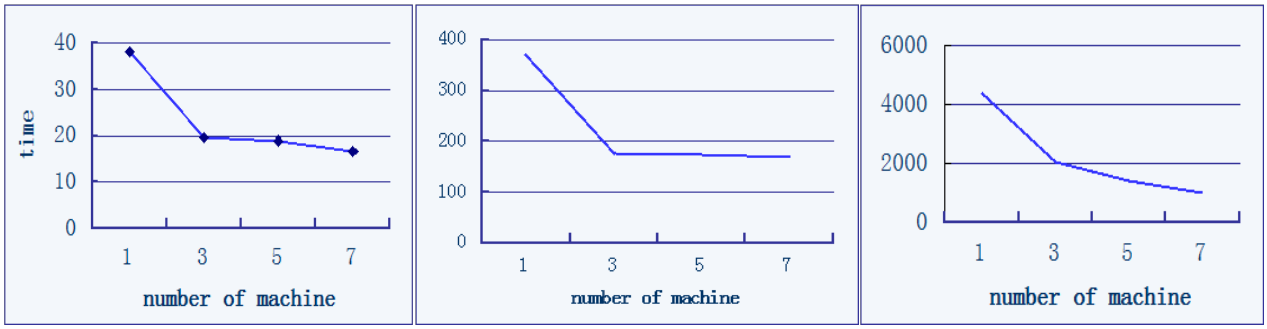
Performance Result

A series of experiment is performed, they are differ at the size of given file, the number of machines that involved and the duplicate rate of the given file. There are three different size of files are executed with size of 10MB, 100MB, 1GB separately. For each size of file, it executed with number of machines 1, 3, 7, 10. And for each size of file, three kinds of files with duplicate rate at 0.2, 0.5 and 0.8 are involved. The test results are shown as table1 below.

From the table1, we could easily found that with the increasing number of machines involved, the execution time experiences a decrease trend. It also illustrated on figure2. Especially when the file comes to size 1GB, the more machines involved in the system, the less time consumed for the processing. And the higher duplicate rate always has a relatively shorter execution result. The size of file effected the execution time as expected as well.

Table 1. Test results

Number of machine	10MB (0.2)	10MB (0.5)	10MB (0.8)	100MB (0.2)	100MB (0.5)	100MB (0.8)	1GB (0.2)	1GB (0.5)	1GB (0.8)
1	41.90	38.07	35.15	405.70	369.54	338.03	4533.80	4386.81	4250.03
3	21.51	19.67	17.83	200.55	176.74	157.60	2317.29	2039.16	1846.50
7	20.47	18.90	17.01	189.03	173.90	134.50	1570.92	1398.27	1201.75
10	21.46	16.66	16.62	170.39	168.41	126.97	1151.60	1001.83	983.75



(a) **(b)** **(c)**

Figure2. Test analysis on file with size 10MB, 100MB, 1GB

Evaluation and comparison with Map-Reduce model

In the real Map-Reduce model, it first assigns the work which is similar with the reading file stage on this system. Without any other efforts, the mapper will work on its given parts of file and then shuffle all the information he got with others to conduct the overall computation. However, in this case, no shuffle stage involved because of the words already be sent according to the hash code of the mod computation and it helps to avoid the conflict among machines. For the files with relatively small size, this system may has a high efficiency than the real Map-Reduce model. But for the larger ones, the initial stage consumes a lot of resources and it is time consuming.

Conclusion

This system works well with the given files. Although when the file comes to the size of 1GB, it is not as stable as working with smaller files. The construction is expected to improve when working with large sets of data. Also since it based on the real Map-Reduce model, it provides decent performance in general and it is seemed as a real distributed system. Moreover, it is suitable for other types of tasks, for instance words counting, by simply overwrite the server and client and keep the rest parts and the entire construction.

Acknowledgements

This work was financially supported by Beijing College Student Research Training Program (2014,2015), Open Project of Digital and Interactive Media Key Laboratory (KF2013-01，KF2013-13), Beijing Institute of Fashion Technology Teaching reform and innovation team project（JGTD-1404）, PHR(IHLB)-Innovative Research Team PTTBIFT_TD_002, Beijing education reform project（2013-ms145）.

References

[1] Wang,Y., & Anane,R. (2014, September). Load Balancing in Distributed System, University of Birmingham

[2] Wikipedia. http://en.wikipedia.org/wiki/Mapreduce

[3] Anane, R., & Anthony, R. J. (2003, March). Implementation of a proactive load sharing scheme. In Proceedings of the 2003 ACM symposium on Applied computing (pp. 1038-1045). ACM.

[4] Sran, N., & Kaur, N. (2013). Comparative analysis of existing load balancing techniques in cloud computing. International Journal of Engineering Science Invention, ISSN (Online): 2319–6734, ISSN (Print): 2319, 6726, 60-63.

[5] Mobile Peer to Peer File Dispersal and Encryption System

Khalid Ashraf ,2009