International Journal of Engineering, Technology and Natural Sciences
E-ISSN : 2685-3191 | P-ISSN: 2775-7706
Vol 3 No 2 (2021)

Syncronize Data Using Map Reduce Model Programming

Murti Retnowo
Universitas Teknologi Yogyakarta,
murti.retnowo@uty.ac.id

Abstract

Research in the processing of the data shows that the larger data increasingly requires a longer time. Processing huge amounts of data on a single computer has limitations that can be overcome by parallel processing. This study utilized the MapReduce programming model data synchronization by duplicating the data from database client to database server. MapReduce is a programming model that was developed to speed up the processing of large data. MapReduce model application on the training process performed on data sharing that is adapted to a number of sub-process (thread) and data entry to database server and displays data from data synchronization. The experiments were performed using data of 1,000, 10,000, 100,000 and 1,000,000 of data, and use the thread as much as 1, 5, 10, 15, 20 and 25 threads. The results showed that the use of MapReduce programming model can result in a faster time, but time to create many threads requires a longer time. The results of the use of MapReduce programming model can provide time efficiency in synchronizing data both on a single database or a distributed database.

Keywords: *data synchronization, mapreduce, message passing interface multithread, static tasks assigment*

I. Introduction

The need for large computing capabilities at this time to process data on a large scale and in a short time is very basic to maximize work and achieve goals in a company. Some fields that require high level of computation are simulation, computation, and data processing. These problems often require repetitive calculations on large amounts of data to obtain valid results. In addition, the computing system must be able to solve the problem in a reasonable time. Traditional computing that has a single processor to carry out the tasks of a program is a way to improve performance in data processing. However, with a single processor it still takes a long time. This is because the computer will retrieve data one by one from each data server located in different locations. In fact, companies often demand that simulations, calculations and data processing to be completed in seconds or minutes. Simulations that are completed in a matter of days are usually not acceptable. The time required to perform simulations, calculations and data processing should be as short as possible so that information can be received in a short and accurate time. There are some problems that have a long lead time in their computational processing. An example is the processing of large amounts of data where the data server is located on several different computers so that the program takes a long time to synchronize or merge data from each computer [1].

The problems that will be discussed in this study have a fairly broad scope, so the problems are limited as follows:

1. The MapReduce process is carried out on a local network with different computer specifications (grid).
2. The computers used to form the grid or as nodes or workers are several computers that are connected in a network and are in one switch (local network).
3. The method used is Message Passing Interface and Static Task Assigment combined with MapReduce programming model.

Iqbal [2], conducted a study on how to analyze a unique identity that is owned by each resident on an ID card so that there is no duplicate ID card at the time of manufacture. The database stored is Very Large Database which requires a MapReduce Framework and a data warehouse in order to analyze the identity. Population is stored in the database. The data is exported into Text format and then the mapping process is carried out [3]. Kurniawan [4] conducted a study to find members of the Linux mailing list in Indonesia with Map Reduce that is used to search for member data, add, delete, view mailing list topics and run data nodes on a Linux-based operating system [4].

II. Method

The need for fast and accurate information is important at this time. Information is needed by management to make decisions in the production process and purchase raw materials for the next period. To get fast and accurate information, of course, a computer with hardware specifications that supports fast data processing and a high level performance is needed beside a client-server based program with a distributed database that can be accessed from all computers in a local network or a public network. It is a way to increase time efficiency in accessing data.

Replacing computer hardware is certainly very helpful in increasing data processing speed, but changing computer hardware costs a lot and programs that have been running well are not necessarily supported on new computer hardware. In addition, it is also necessary to change the settings of the new computer so that the program can run properly.

Changing an existing program at this time, of course, takes a long time because the source code of the running program is not known which is the latest and how the flow of the running program is. This is because the programmer who made the program has moved to a new workplace. Creating a new program surely takes longer because the new programmer must be able to understand the workflow of the current program and implement it in the new program. It takes a long time to learn new programs for users and test the program to find errors. Errors that exist will interfere the work processes that are already running well. In addition to computer hardware problems and running programs, transactions or data processing are carried out in many different locations and databases. This is because the program is still running on a stand-alone basis. Transactions that occur in several different places with programs that are still running on a standalone basis will cause problems in synchronizing data even though they are already using a distributed database, in this case using a MySQL database, but MySQL is still installed on each computer by using user settings that can only access from one place (local computer).

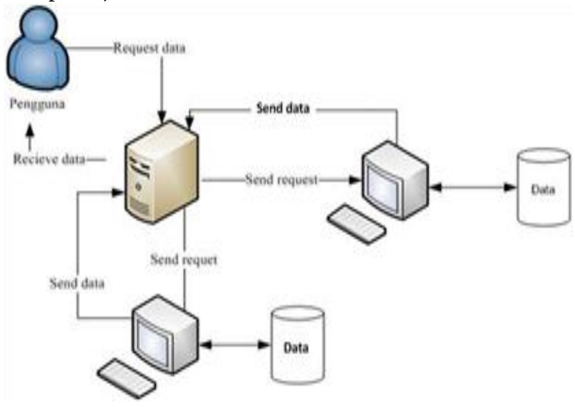


Figure 1 Systems Description

One of the current trends used by large IT companies in handling large and distributed amounts of data is the MapReduce programming model in which there are two processes namely the Map process where the data will be divided into several equal parts and the Reduce process where the data will be recombined and omitted if there are the same data from the previous mapping process. The MapReduce process will be run in a cluster or grid consisting of several independent computers by simultaneously or parallel processing (Multithread) which is used to synchronize data, import data or search data in a distributed database. The MapReduce programming model combined the Message-Passing Interface (MPI) and Static Tasks Assignment (STA) methods.

Delphi programming language is not only expected to improve system performance capabilities in handling large and distributed data but also to increase efficiency and time effectiveness, work in the process of synchronizing data from many client databases to database servers.

The description of the system in outline begins with the process of requesting data from the user. The process of requesting and sending data is as shown in Figure 1 where a user makes a data request; the data request will be sent to the server computer. The server computer that receives the data request will automatically distribute the data request to each client computer until the server computer has finished carrying out its work. The next process occurs on the client computer. After receiving the data request, the client computer will prepare the requested data according to the criteria that have been obtained from the server computer. The next process is to map data or collect data which will be synchronized to the database server in parallel (multithread). The data synchronization process that has been completed will be accessible to users who request the data while at the same time reducing or eliminating the same data with certain criteria.

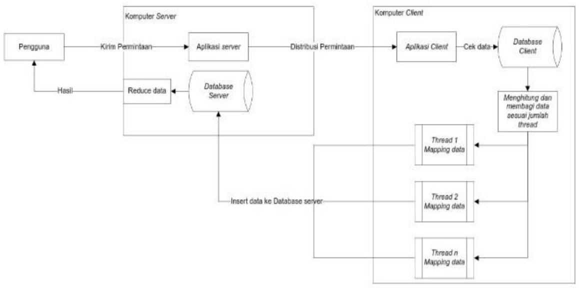


Figure 2 Data Analysis Systems

The stages of data analysis can be seen in Figure 2. The sending process begins with a data request made by the user by making criteria in the form of date limits and the number of threads to carry out the synchronization process. The data request criteria will then be sent to the server computer after receiving the criteria. The server computer will distribute the data request criteria to all client computers in one network. The client computer that has received the data criteria will collect data and share data according to the number of threads and carry out the data synchronization process to the server computer.

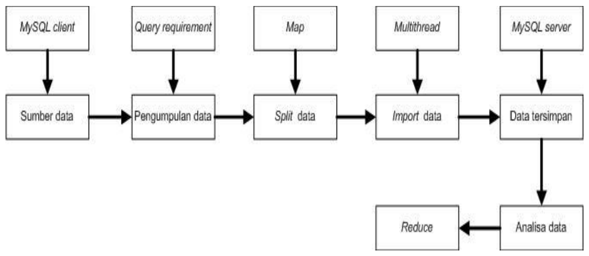


Figure 3 Data Flow Process

III. RESULT

The results of the training are used to determine the difference of speed of data synchronization carried out conventionally with a single process compared to the MapReduce programming model. Thus it will be known as the speed of time obtained from each program which is then compared to determine the time difference between each program. In addition to checking the data processing time, problems that occurred during system creation were also tested, such as checking connections, checking ports and restricting users who make transactions.

The testing process begins with testing to find out the time used to create a thread, processing 1000, 10,000, 100,000 and 1,000,000 data which will be synchronized with the number of processors 1, 5, 10, 15, 20 and 25 respectively.

A. *Make Thread*

Thread or lightweight process (LWP) is a basic unit of CPU Utilization that contains a program counter, a set of registers, and stack space. Threads will work together with other threads in terms of using the code section, data section, and resources of the operating system collectively (tasks). MapReduce is a distributed programming model that runs in parallel (multithread), which serves to speed up the processing of large amounts of data. The results of the average time used to create a thread of 25 threads can be seen in Table I. The time used to create a thread is influenced by the amount of data, the number of threads that have been created and the computer hardware used.

Table 1 Thread Creation Time

PC Name	Time Creation Thread (in Second) *			
	1	10	100	1.000
PC 1	1,52	1,36	2,8	15,96
PC 2	1,82	1,08	1,72	9,8
PC 3	1,32	1,96	1,68	20,53
PC 4	0,68	2,24	5,32	28,24

*In Thousands

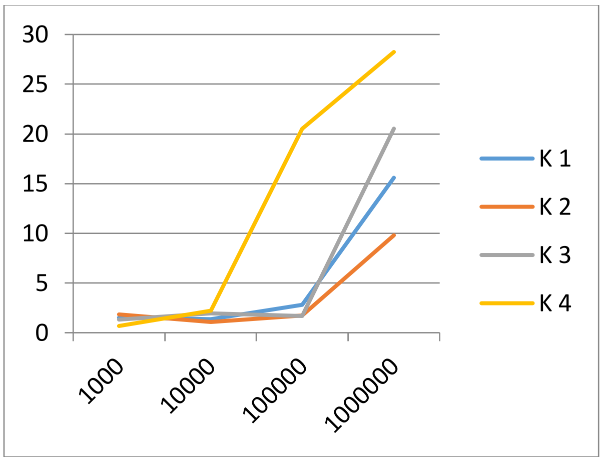


Figure 4. Chart Thread Creation Time

B. *Single Process*

Testing the data synchronization process is carried out using a single process on each computer. Data synchronization training with single process was conducted from 4 computers with different hardware specifications and using the same amount of data on each computer. Tests are carried out in order to determine the speed of data processing using a single process.

Table 2. Synchronization Process Results With Single Process

PC Name	Processing Time (in Second) *			
	1	10	100	1.000
PC 1	34	348	3.600	36.001
PC 2	36	365	3.694	36.940
PC 3	36	365	3.693	36.928
PC 4	36	364	3.693	36.932

*In Thousands

Testing were carried out on all computers with different hardware specifications. The results of testing the data synchronization process carried out on all computers using a single process with a small amount of data (1,000 data) indicate that the time used to perform the data synchronization process will not last long, ranging from 34 to 36 seconds, but when it is done with a lot of data such as 1,000,000 data, the data synchronization process takes a long time ranging from 36,001 to 36,940 seconds or 10 hours 1 second to 10 hours 15 minutes 40 seconds as shown in Table II. From the time it takes to synchronize data with 1,000,000 data to 10 hours, a program is needed to process data in a relatively shorter time.

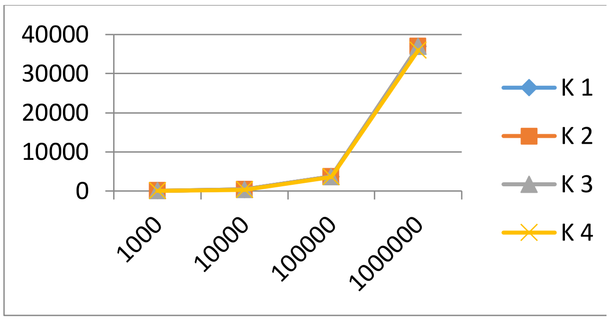


Figure 5. Chart Using Single Process

C. *Multithread (MapReduce)*

1) *Mapping Data*

In the second test using the MapReduce programming model combined with Message Passing and Static Task Assignment, it was able to reduce data processing time from 60% to 70%. The process begins with the distribution of data according to the number of threads that will carry out the data synchronization process, then the data mapping process is carried out according to the results of data sharing. The mapping process is carried out simultaneously with the thread creation process, each thread will get the same data. An example of data sharing can be seen in Figure 6 below:

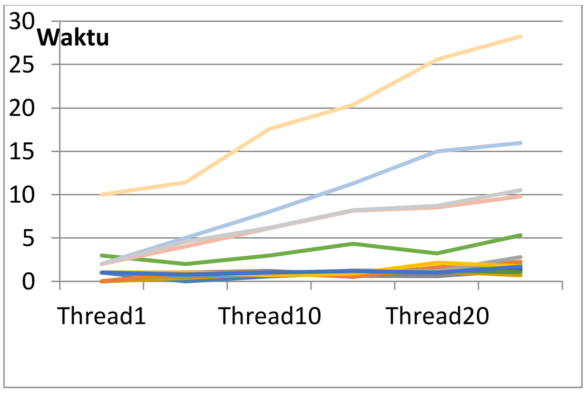
Table 3. Example of Data Calculation

1	Amount of Data	1.000.000
2	Amount of Thread	15
3	Remaining Quotient	1.000.000 mod 15 =10
4	Data for 1 Thread	(1.000.000-10)/15=66.666
5	Data Thread 1 -14	66.666 data
6	Data Thread 15	66.666 + 10 = 66.676 data

Table 4. Averacge Time to Process Mappind data (in Second)

PC Name	Data (*)	Jumlah Thread					
		25	20	15	10	5	1
PC 1	1	1,52	0,85	1	0,6	0	1
PC 2	1	1,32	0,7	0,73	1,2	1	1
PC 3	1	1,28	0,6	0,67	1,2	1	1
PC 4	1	0,68	1,15	1	0,9	0,4	0
PC 1	10	1,36	1,5	0,93	0,7	1	0
PC 2	10	1,08	1,25	1,07	0,9	0,6	1
PC 3	10	1,96	1,2	1	0,8	0,6	1
PC 4	10	2,24	1,6	0,53	1	1	0
PC 1	100	2,8	1,2	1,27	0,8	1	1
PC 2	100	1,72	2,15	0,87	0,7	1	1
PC 3	100	1,,68	1	1,2	1	0,8	1
PC 4	100	5,32	3,2	4,33	3	2	3
PC 1	1.000	15,96	15	11,27	8	5	2
PC 2	1.000	9,8	8,5	8,13	6,1	4	2
PC 3	1.000	10,52	8,7	8,2	6,2	4,6	2
PC 4	1.000	28,24	25,6	20,33	17,6	11,4	10

*** in Thousands**



Figur 6. Averacge Time to Process Mappind data (in Second)

The results of the average test time required to map data and create threads with the MapReduce programming model can be seen in Table IV where the table shows the results of testing thread creation or data mapping carried out on each computer with a different amount of data and the amount different threads. The more data that will be processed and the more threads created, the longer it will take to map the data, and the fewer threads created, the less time will be required.

2) *Reduce Data*

The reduce data process is used to combine data and eliminate data if there is a similarity of data or duplication of data resulting from the mapping process. The reduce process is carried out by each computer after the mapping and thread creation process have been completed. All threads that have been successfully created will carry out the process of reducing data by duplicating data from the client database to the server database.

Table IV shows the results of data synchronization carried out with the MapReduce programming model indicating a significant time change when using a single process for 1,000 data using a time of 68 seconds, when using the MapReduce programming model using 5 processors the average time required decreases about 6.8%, when using 25 listeners it drops to almost 90%, while using 20 listeners it drops by 95%. The time difference between 20 listeners and 25 listeners is due to the fewer number of threads created to perform processing. The more threads created will take time which is longer when compared to fewer threads.

The graph in Figure 8 shows the best average time for computers with the highest computer hardware specifications (PC 2) with no much time difference. The more threads created, the longer the data mapping process but the faster the data reduction process.

Table IV Average Data Reduce Time (In Seconds)

Name PC	**Data**	**Thread**					
		25	**20**	**15**	**10**	**5**	**1**
PC 1	1.000	6,00	4,00	4,60	6,00	11,00	68,00
PC **2**	1.000	4,96	4,00	5,00	5,00	11,00	74,00
PC 3	1.000	3,48	4,00	5,00	5,00	11,00	74,00
PC 4	1.000	5,32	4,00	4,07	6,00	11,00	75,00
PC 1	10.000	30,96	37,05	46,67	60,70	101,40	479,00
PC 2	10.000	31,00	38,00	48,00	65,20	108,60	520,00
PC 3	10.000	30,68	37,90	48,33	64,80	108,20	520,00
PC 4	10.000	30,48	38,00	48,40	63,10	108,40	522,00
PC 1	100.000	351,20	297,20	342,93	407,80	1.989,60	4.361,00
PC 2	100.000	361,24	311,65	357,93	433,50	2.055,60	4.461,00
PC 3	100.000	361,44	309,70	357,13	431,50	2.052,60	4.458,00
PC 4	100.000	356,88	307,70	353,60	433,80	2.051,60	4.460,00
PC 1	1.000.000	2.754,56	3.244,65	3.449,87	4.731,30	9.464,80	47.360,00
PC 2	1.000.000	2.809,80	3.338,00	3.611,07	5.011,50	10.027,60	50.200,00
PC 3	1.000.000	2.796,48	3.316,70	3.592,40	4.995,60	9.994,80	49.970,00
PC 4	1.000.000	2.765,84	3.694,85	3.552,80	5.613,40	11237.60	56.170,00

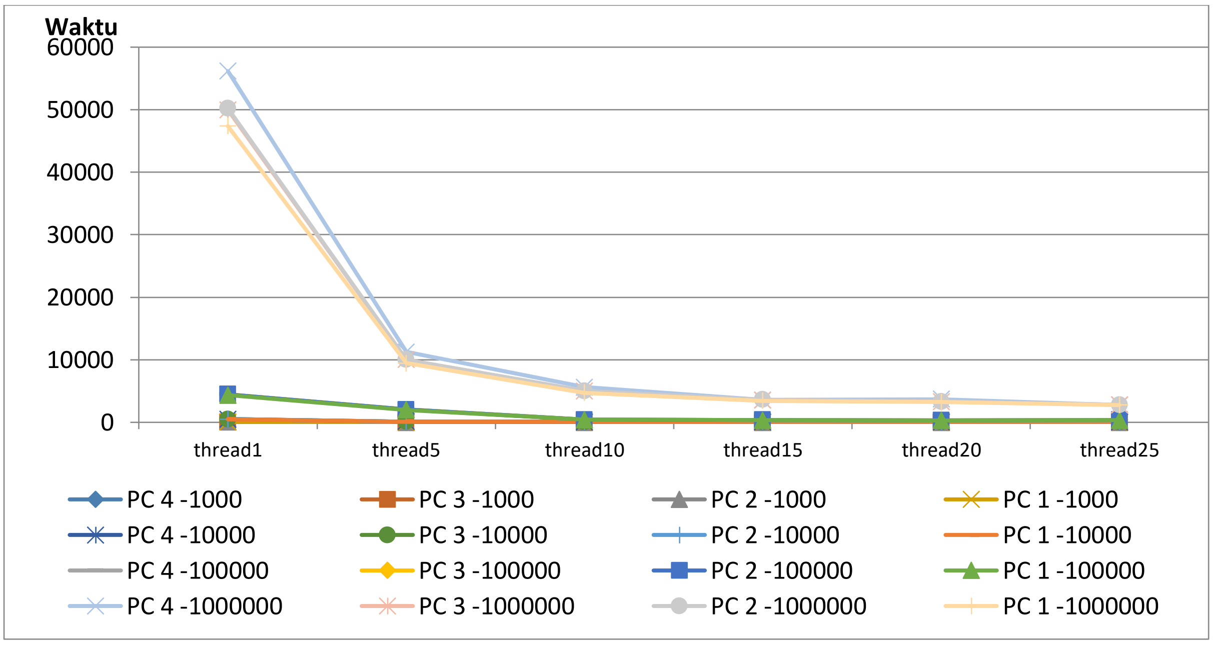


Figure 8 Average results of reduced data (synchronization)

Table V Comparison of Data Reduce Time

Name PC	**Amount of Data**	**Thread 25**			**Thread 20**			**Thread 15**			**Thread 10**			**Thread 5**			**Thread 1**		
		AVG	**MIN**	**MAX**	**AVG**	**MIN**	**MAX**	**AVG**	**MIN**	**MAX**	**AVG**	**MIN**	**MAX**	**AVG**	**MIN**	**MAX**	**AVG**	**MIN**	**MAX**
PC 1	1.000	6.00	6	6	4.00	4	4	4.60	4	6	6.00	6	6	11.00	11	11	68.00	68	68
PC 2	1.000	4.96	4	6	4.00	4	4	5.00	5	5	5.00	5	5	11.00	11	11	74.00	74	74
PC 3	1.000	3.48	3	4	4.00	4	4	5.00	5	5	5.00	5	5	11.00	11	11	74.00	74	74
PC 4	1.000	5.32	5	6	4.00	4	4	4.07	4	6	6.00	6	6	11.00	11	11	75.00	75	75
PC 1	10.000	30.96	30	31	37.05	36	38	46.67	46	62	60.70	59	62	101.40	101	102	479.00	479	479
PC 2	10.000	31.00	30	32	38.00	38	38	48.00	47	66	65.20	65	66	108.60	108	109	520.00	520	520
PC 3	10.000	30.68	30	31	37.90	37	38	48.33	47	66	64.80	64	66	108.20	108	109	520.00	520	520
PC 4	10.000	30.48	30	31	38.00	38	38	48.40	48	64	63.10	62	64	108.40	108	109	522.00	522	522
PC 1	100.000	351.20	347	356	297.20	296	300	342.93	337	410	407.80	405	410	1989.60	1984	1993	4361.00	4361	4361
PC 2	100.000	361.24	354	365	311.65	309	314	357.93	355	435	433.50	433	435	2055.60	2054	2057	4461.00	4461	4461
PC 3	100.000	361.44	357	366	309.70	304	312	357.13	354	433	431.50	430	433	2052.60	2051	2055	4458.00	4458	4458
PC 4	100.000	356.88	353	361	307.70	304	312	353.60	351	436	433.80	432	436	2051.60	2050	2055	4460.00	4460	4460
PC 1	1.000.000	2754.56	2618	2776	3244.65	3113	3263	3449.87	3434	4739	4731.30	4724	4739	9464.8	9448	9478	47360.0	47360	47360
PC 2	1.000.000	2809.80	2692	2825	3338.00	3208	3356	3611.07	3604	5020	5011.50	5007	5020	10027.6	10018	10040	50200.0	50200	50200
PC 3	1.000.000	2796.48	2684	2813	3316.70	3183	3348	3592.40	3580	5000	4995.60	4991	5000	9994.80	9990	10000	49970.0	49970	49970
PC 4	1.000.000	2765.84	2652	2782	3694.85	3580	3717	3552.80	3546	5620	5613.40	5604	5620	11237.6	11234	11240	56170.0	56170	56170

D. Conclution

The conclusions that can be drawn after conducting the research and experiments are as follows:

1. The use of the MapReduce programming model can speed up the time to synchronize data by duplicating data (partial synchronization) from the client database to the server database.
2. The data synchronization process uses 4 computer nodes when compared to using 1 computer node. The best results are shown by giving 25 threads to synchronize.
3. The more threads created and the larger the data, the more time it takes to map the data.
4. Based on testing the difference between the amount of processing time using 15 threads and 25 threads the difference in processing time is not significant.
5. Testing the MapReduce programming model using the Embarcadero RAD 10.6 programming language. The testing process uses data of 1.000, 10.000, 100.000 and 1.000.000 and the threads used are 5, 10, 15, 20 and 25 threads.

REFERENCES

[1] Anung, T. B., 2001 Komputasi Paralel Sebagai Alternatif Solusi Peningkatan Kinerja Komputasi INTEGRAL, vol. 6, no. 2.

[2] Dean, J., dan Ghemawat, S., 2004, MapReduce: Simplified Data Processing on Large Clusters, OSDI '04: 6th Symposium on Operating Systems Design and Implementation.

[3] Fayyad, U., dan Uthurusamy,R., 1996, "Data Mining and knowledge discovery in databases," Communications of the ACM, vol. 39,no. 11, pp. 24-26..

[4] Gaspersz, V, 1998, Statistical process control, Gramedia Pustaka Utama, Jakarta.

[5] Gusti, D. Z., 2012 Pemrograman terdistribusi MapReduce dengan framework Hadoop, Tugas Akhir, Institut Teknologi Sepuluh November, Surabaya

[6] Handly, M., 2003 3C03 Concurrency: Message Passing, © Wolfgang Emmerich, Mark Handley 1998 - 2003

[7] Ishikawa, K, 1976, Guide to Quality Control, Asian Productivity Organization, UNIPUB, ISBN 92-833-1036-5

[8] Ikrimach and Retnowo, M, 2021, Multi Criteria Decision Making Untuk Menentukan Program Studi Calon Mahasiswa, http://papersmai.mercubuana-yogya.ac.id/index.php/smai/article/view/86

[9] Khairul, F. A.,dan Suadi, W., 2014, Komputasi Pembobotan Dokumen Berbahasa IndonesiaMenggunakan MapReduce. Fakultas Teknologi Informasi Institut Teknologi Sepuluh Nopember

[10] Kurniawan, E. K., 2010, Penerapan Model Pemrograman MapReduce Pada Sistem Pencarian Milis Linux Di Indonesia, Jurusan Teknik Informatika, Fakultas Teknik dan Ilmu Komputer, Universitas Komputer Indonesia.

[11] Retnowo M., 2018, Optimize Production Based on Goods Using Supply Chain Management, https://senatik.itda.ac.id/index.php/senatik/article/download/288/pdf

[12] Meriem, M dan Belabbas, Y., 2011, Tasks assignment for cluster computing, International Journal of Web Services Volume 7 Issue 4, January 2011 Pages 427-443

[13] Nurwidyantoro, A., 2011, Paralelisasi Maximum Entropy Part Of Speech Tagging Untuk Bahasa Indonesia Dengan MapReduce, Tesis Program Studi S2 Ilmu Komputer, Fakultas Matematika Dan Ilmu Pengetahuan Alam, Niversitas Gadjah Mada, Yogyakarta.

[14] Saleh, M. A., Deldari, H., and Dorri, B. M., 2008,"Balancing Load in a Computational Cluster Applying Adaptive, Intelligent Colonies of Ants". Informatica.

[15] Setiyo, R. W.,Sakti, E.P.,dan Amron, K., 2014, Rancang Bangun Kuis Online dari Layanan E-learning pada Lingkungan Hadoop Program Teknologi Informasi dan Ilmu Komputer Malang

[16] Silberschatz, Galvin, P. and Gagne. G. 2005. Operating Systems Concepts. Seventh Edition.John Wiley & Sons.

[17] Utdirartatmo, F., 2003, Pemrograman Paraleldi Linx berbasis DSM (Distributed Shared Memory), Penerbit Andi, Yogyakarta.

[18] Wilkinson, B., and Allen, M., 2005, (Pararrel Programming Techniques and applications Using Networked Workstations and Computers~Second Edition, (diterjemahkan oleh Hidayat, s., Santosa, Y. B. Hery, A. P. dan Himamunanto, R 2010) Penerbit Andi Yogyakarta.