Auto-Tuned Hadoop MapReduce for ECG Analysis

Kerk Chin Wee* and Mohd Soperi Mohd Zahid
Faculty of Computing,
Universiti Technology Malaysia,
Skudai, Johor, Malaysia.
*e-mail: kerkchinwee@hotmail.co.uk, e-mail: soperi@utm.my

Abstract— **Electrocardiograph (ECG) analysis brings a lot of technical concerns because ECG is one of the tools frequently used in the diagnosis of cardiovascular disease. According to World Health Organization (WHO) statistic in 2012, cardiovascular disease constitutes about 48% of non-communicable deaths worldwide. Although there are many ECG related researches, there is not much efforts in big data computing for ECG analysis which involves dataset more than one gigabyte. ECG files contain graphical data and the size grows as period of data recording gets longer. Big data computing for ECG analysis is critical when many patients are involved. Recently, the implementation of Hadoop MapReduce in cloud computing becomes a new trend due to its parallel computing characteristic which is preferable in big data computing. Since large ECG dataset consume much time in analysis processes, this project will construct a cloud computing approach for ECG analysis using MapReduce in order to investigate the effect of MapReduce in enhancing ECG analysis efficiency in cloud computing. However, the performance of existing MapReduce approach is limited to its configuration based on many factors such as behaviors of cluster and nature of computing processes. Hence, this research proposes MapReduce Auto-Tuning approach using Genetic Algorithm (GA) to enhance MapReduce performance in cloud computing for ECG analysis. The project is expected to reduce ECG analysis process time for large ECG dataset compared to default Hadoop MapReduce.**

Keywords— **ECG, Hadoop MapReduce, Auto-Tuning.**

I. INTRODUCTION

Electrocardiograph (ECG) analysis brings a lot of technical concerns because it is the most effective tool in cardiovascular disease diagnosis. According to World Health Organization (WHO) statistic 2012, cardiovascular disease caused about 48% non-communicable deaths worldwide which is the major cause of non-communicable death. In order to save more lives from cardiovascular disease deaths, the efficiency of ECG analysis process should be improved. Traditionally, an ECG report will be printed after the completion of an ECG recording and doctors need to manually analyze printed ECG in order to gather patient heart information. It will suffer doctors and drag down process efficiency if the number of ECG reports is large. For example, an ECG record for 24 hours monitoring of a patient may size up to 100 MB or more. Depending to different needs of ECG analysis, there can be up to hundreds ECG records for different uses which may generate around hundreds MB to 1GB ECG dataset per day. Due to different uses, an ECG record can be a 15 minutes, 30 minutes, one or more hours, or even 24 hours record. Therefore, ECG dataset can be not only big in size, but complex to be analyzed. Due to short time constraint of getting ECG analysis result, an efficient computing paradigm is needed to save more lives at stake. The recent ECG devices are equipped with wireless communication technologies which enable ECG data transmission from ECG devices to other computational devices or network terminals. With this feature, it makes the computerization of ECG analysis process possible. Nevertheless, there is lack of researches in big data computing for ECG analysis which involves dataset more than one gigabyte although there are many researches for ECG analysis algorithm, ECG personal monitoring and other ECG cloud solutions,.

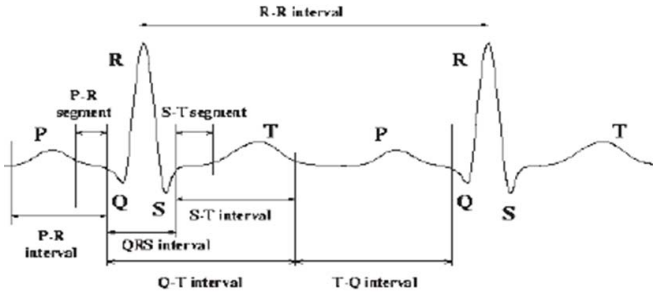
Recently, MapReduce becomes popular paradigm for big data computing due to its parallel computing ability. However, it is supported by some researches that the default MapReduce system has much lower efficiency than its desired condition. In order to enhance the efficiency of MapReduce in computing large ECG dataset over 10GB which is mixed with different ECG data file types including header file, data file and annotation file, this research introduces a GA Auto-Tuning approach into default Hadoop MapReduce system custom ECG analysis process into cloud computing process using MapReduce paradigm and investigate its performance. Notice that the ECG dataset mixes with different file types because the annotation files record the time of occurrence for events, data files record ECG signals and header files record details of ECG signals. Although more works must be done to obtain the required information based on different ECG files, the parallel computing characteristic of MapReduce framework is capable to grouping the files efficiently and work on them in parallel in order to shorten the information extraction period.

978-1-4673-9572-4/15/$31.00 ©2015 IEEE

II. ECG

ECG is the record of the bio-electric potential variation detected via the electrodes throughout the time. The bio-electric potential variation is heart signal with regular circles as a heartbeat. Each heartbeat cycle has a P-wave, QRS complex and T-wave as shown in Fig. 1.

Fig. 1. ECG Signal



The ECG features represents identical heart activities respectively such as P-wave is the signal generated by atria depolarization meanwhile QRS complex is the signal generated by depolarization of ventricles. Hence, any abnormal of ECG signal features address heart functionality problem specifically. Notice that QRS complex is critical to identify heart rate from ECG signal because it is significant to identify the number of heart cycle per minute. Therefore, this research focus on extract QRS complex from ECG signal.

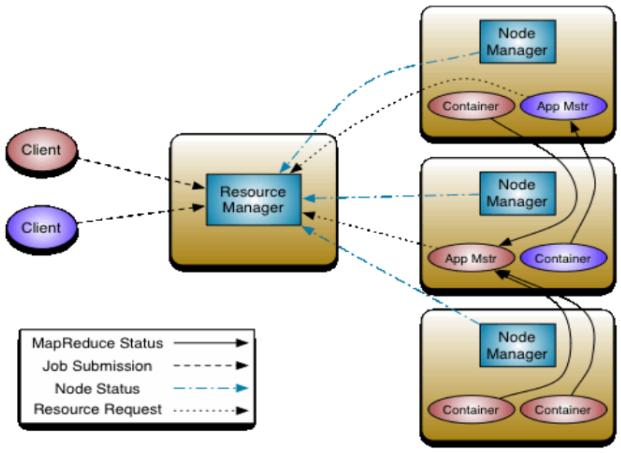
III. CLOUD PLATFORM – GOOGLE CLOUD PLATFORMS

Cloud is the optimum solution for platform of computation because it provides flexibility in scaling resources based on the computational needs. There are several cloud platforms provided by big IT companies such as Amazon, Google and Microsoft for users to purchase cloud resources desirably. Among the cloud platforms, this research selected Google Compute Engine (GCE), the IAAS (Infrastructure As A Service) type platform of Google Cloud Platform because it is the most cost efficient platform compared to other IAAS platform such as Amazon Web Services (AWS). According to [10] and [11], at the time being, purchasing a virtual machines (VM) for 2CPU and 6~8GB RAM specification costs $0.104~$0.126 per hour in AWS but only costs $0.07~$0.1 per hour in GCE. Notice that the cost for purchasing a VM in AWS is fixed price but GCE vary with the usage where $0.07 per hour is the price of holding a VM without using it and $0.1 per hour is the price of holding a VM with full usage. In order to purchase GCE services, users are required to have a Google account. The detail of using GCE in this research will be illustrated in section VI part B.

IV. HADOOP MAPREDUCE

The Hadoop framework used for MapReduce in this research is Hadoop-2.6.0 version. It has default capacity scheduler for task scheduling and YARN architecture for NextGen MapReduce or MapReduce v2. In YARN architecture, there are two important components to able the parallel computation work in MapReduce cluster which are Resource Manager (RM) and Node Manager (NM). There will be only one RM works on master node of MapReduce cluster to schedule the MapReduce tasks according to the conditions of slave nodes in the cluster meanwhile there will be a NM works on each slave node to manage task execution and update the task status in the slave node. The details of YARN architecture is shown in Fig. 2. Since NextGen MapReduce is expected to superior MapReduce v1 provided in Hadoop 0.x or 1.x versions in term efficiency due to framework structuring, this research decides to make it as the research target in Auto-Tuning because still not many MapReduce research involve NextGen MapReduce in their researches.

Fig. 2. Apache YARN architecture



In view of MapReduce paradigm, there is almost no different between MapReduce v1 and NextGen MapReduce which processes as shown in Fig. 3. However, YARN architecture make the MapReduce framework changes by changing the software architecture which are:

i. from jobtracker to resource manager
ii. from tasktracker to node manager

These changes are due to the demands of enhancing the capacity of Hadoop framework in better schedule, keep tracking and fast react to MapReduce tasks. Notice that MapReduce job is a client request for program such as a request of ECG analysis meanwhile MapReduce tasks are the distributed works from a MapReduce job to make it process in parallel such as reading ECG file to get QRS complex data. The number of MapReduce tasks depends on the number of splits per data in a MapReduce job.

V. GENETIC ALGORITHM (GA) DESIGN FOR HADOOP AUTO-TUNING

The GA design for the Auto-Tuning system makes use of default GA algorithm for tournament selection of best Hadoop MapReduce configuration. Since there are over 200 Hadoop parameters in its configuration which affect MapReduce efficiency, this research selected the following Hadoop parameters as tuning targets to enhance Hadoop MapReduce configuration. The reason of selecting important Hadoop parameters only but not all Hadoop parameters in configuration tuning is because too many Hadoop parameters will detract the focus of Auto-Tuning system. The selected Hadoop parameters are shown in TABLE I.

TABLE I. Hadoop Parameters for Auto-Tuning

Hadoop Parameter	Description
mapreduce.task.io.sort.factor	Maximum streams merged at once while sorting files.
mapreduce.task.io.sort.mb	Memory-limit while sorting data for efficiency
mapreduce.reduce.shuffle.parallelcopies	Limited number of parallel copies run by reduces to fetch outputs from very large number of maps
mapreduce.map.memory.mb	Map task resource limit
mapreduce.reduce.memory.mb	Reduce task resource limit
mapreduce.map.java.opts	heap-size limit for child jvms of mapper nodes
mapreduce.reduce.java.opts	heap-size limit for child jvms of reducer nodes
yarn.resourcemanager.scheduler.class	RM scheduler algorithm

In the GA design, each Hadoop parameter is a gene of the solution which is the configuration. Since different Hadoop parameters have different range of configuration values, decimal value instead of binary value is used to represent the value of Hadoop parameters as shown in Fig. 4. This research decomposed the GA algorithm in Auto-Tuning into four processes, selection, evaluation, crossover and mutation as shown in Fig. 5. The GA algorithm will start from evaluation by fitness equation so that selection can be conducted to select a group of solutions A with better fitness value from the pool of solutions, followed by crossover which random select any pairs of parent solutions from solutions A to generate new solutions B which are produced by inherit genes from parent solutions, before mutation take place in randomly change values of genes to produce flexibility to search best solution in Hadoop MapReduce configuration.

Fig. 4. GA concept of Hadoop configuration

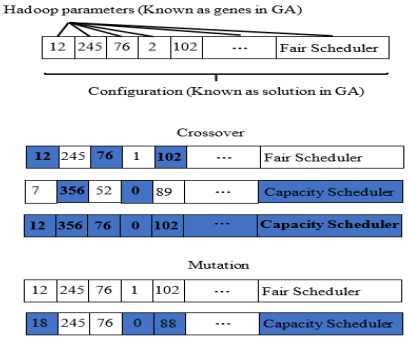
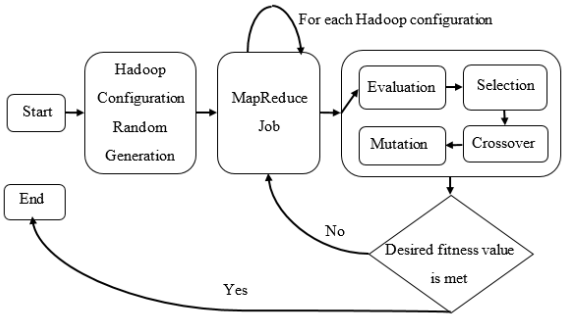


Fig. 5. GA Auto-Tuning overview



Notice that GA algorithm only starts to work after running MapReduce jobs for all random generated initial Hadoop MapReduce configurations to get job performance before starts new jobs again once new configurations are generated after mutation process, then, repeat the GA algorithm processes before new jobs are started for new configurations. This process is the tuning process to fine-tune Hadoop parameters in its configuration and only stop when the Hadoop MapReduce configuration reached desired fitness value.

Considering the number of initial random generated Hadoop configurations as C and number of Hadoop parameters to be fine-tuned as H, the complexity of GA Auto-Tuning process is

$$O(n) = CH+L(CJ+(C+C+0.5CH+0.5CH)) \quad (1)$$

whereas L is the repeating time of GA processes and J is average MapReduce job complete time. This is because the program takes CH time to generate C number of Hadoop MapReduce configurations or solutions with H number of Hadoop parameters per solution. Each time before GA algorithm start, Hadoop take CJ time to run C times of MapReduce jobs to get job complete time for each solution. Since the fitness value of each configuration will be obtained from its job complete time directly, evaluation process take C to compute fitness value each solution and selection process also take C time to select suitable solutions. Let the maximum number of selected solutions to

be C which means all solutions are selected, the maximum number of crossover pairs should be 0.5C and crossover should take 0.5CH take to generate 0.5C new solutions. Since mutation process only involves new solutions, then, the maximum time consume should be also 0.5CH. Since the process repeat from running MapReduce jobs to GA mutation, (CJ+(C+C+0.5CH+0.5CH) time will repeat L times until the desired fitness value is met. This is why the complexity of Auto-Tuning process should be as in Eq. 1. Notice that the number of solutions selected each time should not exceed C so that the complexity will not go beyond Eq. 1.

VI. Related Works

Wang et al. (2014) proposed a solution for ECG mobile computing that using smartphone as ECG analysis tool and cloud as the training agent for ECG analysis model. The research gives impressive result which speedup the analysis processing averagely 37 times and save around 88% of mobile device energy compared to only use mobile devices for analysis. However, this research also reveals the truth that the limitation of computational resources of mobile device in term of processing power, energy resource such as battery life and data storage gives a great challenge to mobile based healthcare system. Therefore, instead of using mobile computing, cloud computing is more suitable for ECG analysis.

Sahoo et al. (2014) first proposed distributed computing approach for ECG analysis using MapReduce for ECG feature extraction. The research shows the execution time for ECG analysis greatly reduced compared to using desktop and the execution time also reduce with the increase of processing nodes in cloud which in turn prove the capability of MapReduce in enhancing cloud computing efficiency. In the research, 3.2GB ECG signal for 4 ECG channels requires 33 minutes to be processed by desktop but only need 1.57 minutes to be processed by MapReduce using 4 nodes. Although Sahoo et al. (2014) shows the capability of MapReduce in speedup ECG analysis for signals with same channel number, the research do not investigate the capability of MapReduce in ECG analysis with mixed ECG signals which provided by this research.

VII. Methodology

A. Material and Tools

The ECG dataset for this research is a mixed collection of ECG dataset from three ECG databanks provided by Physionet, the online ECG dataset provider. The ECG databanks are European ST Databank which consists of 90 records of 30 minutes ECG data with total size 488MB, Long term AF Database which consists of 84 records of 21 hours ECG data with total size 3.7GB and Long term ST Databank which consists of 89 records of 24 hours ECG data with total size 6GB. The details of ECG dataset are stated in TABLE. II.

TABLE II. ECG Dataset Details

ECG data size	10.2GB
ECG record number	263
ECG record files	.hea, .dat and .ann files
ECG dataset composition	1. 90 records of 30 mins ECG data 2. 84 records of 21 hours ECG data 3. 89 records of 24 hours ECG data

As mentioned in section III, the tool for cloud computing in this research is GCE platform. This research only uses Virtual Machines (VM) provided by GCE platform for MapReduce master and slave nodes. Meanwhile, the file system for Hadoop framework is Hadoop Distributed File System (HDFS) which locates data among VMs in MapReduce cluster and enable the data to be shared among cluster. The details of GCE cluster setup will be illustrated in part B.

B. MapReduce Cluster Setup

GCE setup starts from creating a Google account if do not have it. Once having a Google account, the research starts from setup a MapReduce cluster by purchasing VMs with specifications listed in TABLE III. Next, Oracle Java 8 Java Virtual Machine (JVM) is installed in every VMs. Then, Hadoop-2.6.0.tar.gz file is downloaded into master node and extracted into home directory. After that, adding Hadoop environment variable in .bashrc file under home directory, and edit core-site.xml, hdfs-site.xml and mapred-site.xml under Hadoop-2.6.0 folder before creating folders for datanode and namenode in order to setup default Hadoop framework. Finally, copy the Hadoop 2.6.0 folder in master node to other slaves by gcloud feature and add Hadoop environment variables in the .bashrc file of slave nodes.

TABLE II. VM specification for MapReduce cluster

Data disk type (Master and worker nodes)	standard persistent disk
Data disk size in GB (Master and worker nodes)	50
Machine type (Master and worker nodes)	n1-standard-2 (2vCPUs, 7.5GB RAM)
Machine OS	Ubuntu 14.04 LTS
Enable request between cluster nodes	Yes
Enable http request	No
Enable https request	No

Notice that the number of slave nodes is vary throughout the experiment in the research to investigate the effect of it

on Auto-Tuning performance. The network of cluster setup by Apache Hadoop is closed which only allow the VMs to be controlled remotely by Security Shell (SSH) with Google account. This ensure the cluster security.

C. ECG Analysis Approach

The ECG analysis approach for this research is ECG feature extraction for QRS detection. The approach first read ECG information from header files of EDF format ECG by wfdb library function, followed by using high-pass and low-pass filters to obtain QRS complex from ECG signal (.dat file), before save it into QRS file (.qrs file). The filters work to remove unwanted components such as P-wave, T-wave and noises from ECG signals in order to get the QRS complex as residual signal. Since ECG analysis program should perform the QRS detection for each record in ECG dataset, the QRS detection algorithmshould be performed recursively throughout the ECG dataset. MapReduce Algorithm for ECG analysis

Typically, MapReduce algorithm is divided into map and reduce functions. Map function is usually the main computation needed to be executed in parallel on multiple mappers as map tasks meanwhile reduce function is the result reduction process running on one or more reducers as reduce tasks. In this research, the map function is QRS detection and reduce function is QRS detection result writing.

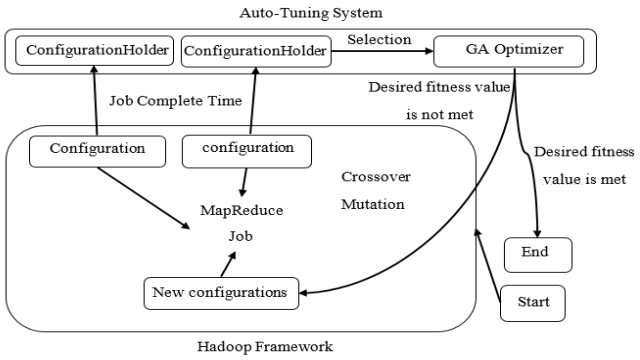
MapReduce process works by first breakdown the large ECG dataset around 10.2GB into small workloads of size around 64MB by resource manager, before the resource manager schedules and assigns workloads to mappers as map tasks. After all map tasks completed, the immediate results released by node managers are scheduled and assigned to reducers as reduce tasks. After all reduce tasks are finished, the QRS detection results are collected as .qrs files in HDFS.

Notice that slave nodes in MapReduce cluster are assigned as mappers and reducers in MapReduce process above. In case of ECG analysis, the dataset is the collection of small ECG data files which size from 5KB to 100MB. In order to make Hadoop to perform MapReduce for multiple small ECG files, the input format for Hadoop must be set on CombinedFileInputFormat and the split operation for single file should be disabled.

D. Auto-Tuning System Development

Since the GA algorithm of Hadoop Auto-Tuning system is discussed in section V. This section is to illustrate the system overview of Auto-Tuning system architecture which is the software development view of Fig. 5. The Auto-Tuning system consists of an object of GA optimizer and C objects of ConfigurationHolders as shown in Fig. 10. The GA optimizer has GA functions of selection, crossover and mutation besides findBestConfiguration function to get the Hadoop MapReduce configuration with highest fitness value meanwhile ConfigurationHolder is the extended class of Configuration to store information of Hadoop MapReduce configuration or solution parameter values, job complete time and configuration fitness value. The process starts from getting job complete time and compute fitness value for each solution before storing the values to ConfigurationHolder. Next, C number of solutions are selected by GA optimizer for tournament of best solution and other solutions are discarded. Then, GA optimizer will perform crossover and mutation if desired fitness value is not met before new jobs are launched for getting job complete time for each of new solutions. After that, the process will be repeated until the desired fitness value is met and GA optimizer will get the best solution from the objects of ConfigurationHolder by judging the fitness value of solutions.

Fig. 6. Auto-Tuning System Architecture



VIII. RESULT

The result of Hadoop MapReduce for ECG analysis with GA Auto-Tuning is shown in TABLE III.

TABLE III. Job Complete time of MapReduce for ECG Analysis in seconds

	Job Complete time (sec)		
No. node in cluster	Default MapReduce	Auto-Tuned MapReduce	Enhancement (sec)
Node 1	488.7	444.5	44.2
Node 2	259.3	232.6	26.7
Node 3	185.1	166.6	18.5
Node 5	131.1	122.8	8.3

The result in TABLE III shows that Auto-Tuning works better if the number of slave nodes in MapReduce cluster is less. The decrement of job complete time reduces from 44.2 secs in MapReduce with single node cluster to 8.3 secs in MapReduce with five node clusters.

IX. Discussion

The reduction of job complete time decrement with increment of slave node number in the cluster might be due to the scheduling effect among the nodes in cluster drag down the MapReduce efficiency. In single node cluster, the effect of scheduling the tasks among cluster is minimized because only one slave work for MapReduce and no smooth scheduling is needed to schedule the resources and tasks among cluster. In this case, an optimum Hadoop MapReduce configuration is critical because fine-tuned parameters allows the slave node to work at optimum condition. However, once the number of slave nodes increases, the scheduling of resources and tasks among slave nodes in MapReduce cluster become more and more important because it responds to the speed of MapReduce task assignment and policy of handling failure tasks. In this case, although an optimum Hadoop MapReduce configuration is still important for optimizing the working condition of slave nodes, a scheduling algorithm may drag down the enhancement in efficiency of MapReduce process once it do not schedule the resources and tasks perfectly. Since the scheduling algorithm for Hadoop MapReduce of this research is default Hadoop schedulers which are fair scheduler and capacity scheduler, the scheduling ability of scheduler is still not optimum yet to ensure good Hadoop MapReduce efficiency.

X. Conclusion And Future Work

In conclusion, it is proven that MapReduce is proven to be able to work on large ECG datasets with 263 records which totally sizes 10.2GB. It extracts QRS complex from these 263 records with around 6 mins in single node cluster and getting faster with large MapReduce cluster. This is encouraging because the same dataset needs around 1 hour to extract these QRS complexes by custom computational method using VM of same specification. The Hadoop MapReduce Auto-Tuning also works successfully as it able to reduce the job complete time of ECG analysis in single node cluster from 4 mins to 3 mins 20 secs, although its effect is getting lower with the increment of slave node number in MapReduce cluster. In order to further enhancement the efficiency of NextGen MapReduce process, the researches on Hadoop MapReduce schedulers in NextGen MapReduce should be encouraged so that the efficiency of Hadoop MapReduce can be optimized in multimode clusters.

Acknowledgment

The authors would like to thank Ministry of Education Malaysia and Universiti Teknologi Malaysia (UTM) for supporting this research under Vote no 4F375.

References

[1] SAHOO, S. S., JAYAPANDIAN, C., GARG, G., KAFFASHI, F., CHUNG, S., BOZORGI, A., CHEN, C. H., LOPARO, K., LHATOO, S. D. & ZHANG, G. Q. (2014). Heart beats in the cloud: distributed analysis of electrophysiological 'Big Data' using cloud computing for epilepsy clinical research. Journal of the American Medical Informatics Association, 21, 263-271

[2] SEENA, V. & YOMAS, J. (Year) A review on feature extraction and denoising of ECG signal using wavelet transform. Devices, Circuits and Systems (ICDCS), 2014 2nd International Conference on, 6-8 March 2014 2014. 1-6.

[3] VELIC, M., PADAVIC, I. & CAR, S. (Year) Computer aided ECG analysis - State of the art and upcoming challenges EUROCON, 2013 IEEE, 1-4 July 2013 2013 Zagreb, Croatia. 1778-1784.

[4] VANEGHI, F. M., OLADAZIMI, M., SHIMAN, F., KORDI, A., SAFARI, M. J. & IBRAHIM, F. (Year) A Comparative Approach to ECG Feature Extraction Methods. Intelligent Systems, Modelling and Simulation (ISMS), 2012 Third International Conference on, 8-10 Feb. 2012 2012. 252-256.

[5] MAZOMENOS, E. B., BISWAS, D., ACHARYYA, A., TAIHAI, C., MAHARATNA, K., ROSENGARTEN, J., MORGAN, J. & CURZEN, N. (2013). A Low-Complexity ECG Feature Extraction Algorithm for Mobile Healthcare Applications. Biomedical and Health Informatics, IEEE Journal of, 17, 459-469.

[6] XIAOJUN, Z., XIULI, M. & YANG, L. (Year) An adaptive threshold algorithm based on wavelet in QRS detection. Audio, Language and Image Processing (ICALIP), 2014 International Conference on, 7-9 July 2014 2014. 858-862.

[7] GUNARATHNE, T., TAK-LON, W., QIU, J. & FOX, G. (Year) MapReduce in the Clouds for Science. Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, Nov. 30 2010-Dec. 3 2010.

[8] LI, M., XU, G.-H., WU, L.-F. & JI, Y. (Year) Performance Research on MapReduce Programming Model. Instrumentation, Measurement, Computer, Communication and Control, 2011 First International Conference on, 21-23 Oct. 2011 2011b. 204-207.

[9] GAIZHEN, Y. (Year) The Application of MapReduce in the Cloud Computing. Intelligence Information Processing and Trusted Computing (IPTC), 2011 2nd International Symposium on, 22-23 Oct. 2011 2011 Wuhan, Hebei, China. 154-156.

[10] CHANGLONG, L., HANG, Z., KUN, L., MINGMING, S., JINHONG, Z., DONG, D. & XUEHAI, Z. (Year) An Adaptive Auto-configuration Tool for Hadoop. Engineering of Complex Computer Systems (ICECCS), 2014 19th International Conference on, 4-7 Aug. 2014 2014. 69-72.

[11] DILI, W. & GOKHALE, A. (Year) A self-tuning system based on application Profiling and Performance Analysis for optimizing Hadoop MapReduce cluster configuration. High Performance Computing (HiPC), 2013 20th International Conference on, 18-21 Dec. 2013 2013. 89-98.

[12] KEWEN, W., XUELIAN, L. & WENZHONG, T. (Year) Predator - An experience guided configuration optimizer for Hadoop MapReduce. Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, 3-6 Dec. 2012 2012. 419-426.

[13] YIGITBASI, N., WILLKE, T. L., GUANGDENG, L. & EPEMA, D. (Year) Towards Machine Learning-Based Auto-tuning of MapReduce. Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2013 IEEE 21st International Symposium on, 14-16 Aug. 2013 2013 San Francisco, California, United States. 11-20.

[14] https://cloud.google.com/compute/?utm_source=google&utm_medium=cpc&utm_campaign=2015-q1-cloud-japac-my-gce-bkws-freetrial&utm_content=en&gclid=CN_n0aGv88cCFVQpjgodLUoLvA#pricing

[15] https://aws.amazon.com/ec2/pricing/