MODELING OF SYSTEMS UNDER CLOUD ENVIRONMENT

Sharafadeen Muhammad, Ibrahim Kabiru Dahiru, Ahmad Abubakar, and Muhammad Sanusi Ibrahim

Department of Electrical & Electronics Engineering, Jigawa State Polytechnic, Dutse, Nigeria, Tel: +2348068286220, e-mail: sharafadeen@jigpoly.edu.ng

Department of Computer Engineering, Kano State Polytechnic, Kano, Nigeria, Tel: +2348036559060, e-mail: kabirudahiruibrahim@kanopoly.edu.ng,

Department Electronique, Energie Électronique, Automatique, Université Grenoble Alpes, Grenoble, France

Department of Computer Science, Faculty of Science, Kaduna State University, Kaduna, Nigeria

Received Date: April 7, 2020; Revised Date: January 12, 2021; Acceptance Date: May 6, 2021

Abstract

The emergence of large amount of data requires an efficient means of processing and storage facilities. Cloud computing provides an effective solution; MapReduce programming paradigm has the ability to handle such data by implementing Hadoop, but came up with some conflicting challenges in terms of Service Level Agreement (SLA) between major stakeholders. This paper focuses on coming up with a MapReduce model through system identification in order to address the requirement of the service time to meet-up the SLA within the limit of defined threshold in the presence of uncertainties in the system. A second order nonlinear model was obtained, which shows a good representation of the real system and could be used to develop control laws on the real system.

Keywords: Cloud computing, Hadoop, MapReduce, System identification

Introduction

The world is rapidly moving towards information technological age, as a result, there exists a vast increase in the amount of data to be managed. The generated data on a daily basis amount to a number of Petabytes (2^{50} bytes) ranging to Exabyte (1000 petabytes), therefore, there is great challenge in handling the processing and storage of such Big Data [1]. To address the above stated challenges, the idea of cloud computing was coined. The cloud computing environment is a distributed system [2],[3],[4],[5] which is open and large in which the resources provided as services to the computing infrastructures are made available via the internet [6]. Cloud computing provide organizations the ability to scale-up their IT infrastructures which include hardware, software and services by remotely giving them the ability to optimize the utility of these IT infrastructures through the provision of a virtualize data center. The MapReduce is a programing model for processing large data sets [3]. MapReduce handles the need for parallelization of a very large amount of data over several machines that are in a large cluster, it is a massive processing paradigm [3],[4] for the parallel processing of data which is distributed over a commodity cluster (due to its scalability, it runs on clusters). In contrast to serial data processing, MapReduce achieved the processing ability of vast amount of data due to parallelisms over shorter period of time [1]. It works based on two functions, a map function and a reduced function [3]. The idea of cloud

computing offer developers a means to transparently handle data partitioning, replication, task scheduling and fault tolerance on a cluster of commodity computers [5].

The core functionality of MapReduce is made up of two basic functions with two distinctive steps; the map step in which the master node takes a large volume of input data and fragments it into smaller pieces, then distributes them to worker nodes. Worker nodes on the other hand redistribute them into further smaller pieces, process them and send the processed output back to the master [3]. The reduced step takes the processed output from the master as intermediate values and further reduce them to a smaller solution. The flow pattern is such that the input key/value pairs split into different segments that are sent into different machines in parallel, each of the machine runs the data mapped to it [3]. Therefore, two scripts are involved which are the map script and the reduced script [4]. Figure 1 shows the architectural representation and execution overview of a MapReduce. The MapReduce architecture has three principal phases which are Mapper, Reducer, and Shuffler [7].

Khezr and Navimipour have used MapReduce in the optimization algorithms. The paper presents some MapReduce applications in the optimization algorithms like particle swarm optimization (PSO), cuckoo search, and ant colony algorithm [12]. Berekmeri in his work was able to realized first order dynamic model of a mapreduce using system identification [8].

In this paper, the basic conflicting requirement of cloud services based on availability and fast response time to guarantee the performance of MapReduce was studied. The first order identified model of MapReduce [8],[9],[11] which was a simple model that is less robust to disturbance was adopted and treated as a black box model to finally proposed a second order model of MapReduce that represents more the dynamics of the system.

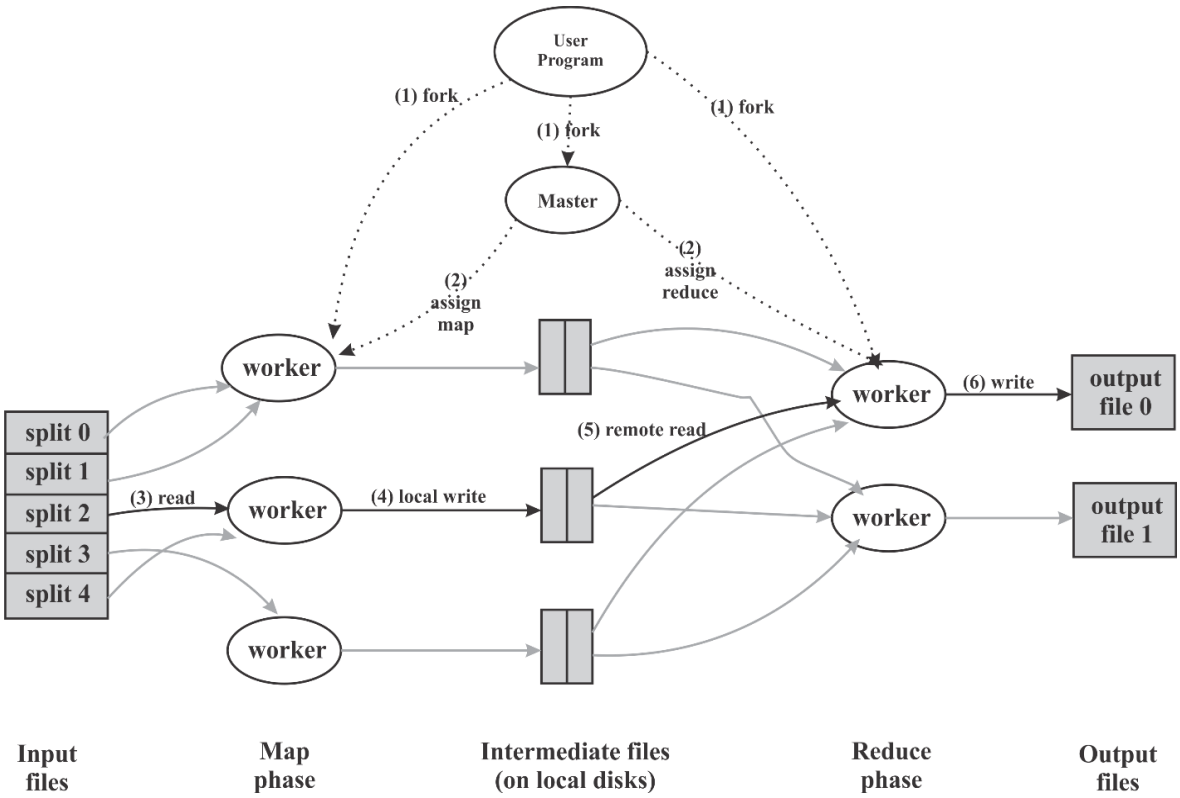


Figure 1. Execution overview [3]

Experiment Layout

The experiment was carried out in a remote server on Grid5000 cluster accessed from local machine using the internet protocol Secure Socket Shall (SSH) as a command line interface. The experimental set-up is as shown in Figure 2. The SSH connects the local computer to the remote server via the master node of the MapReduce which executes the MapReduce BenchMark Suite (MRBS) tool.

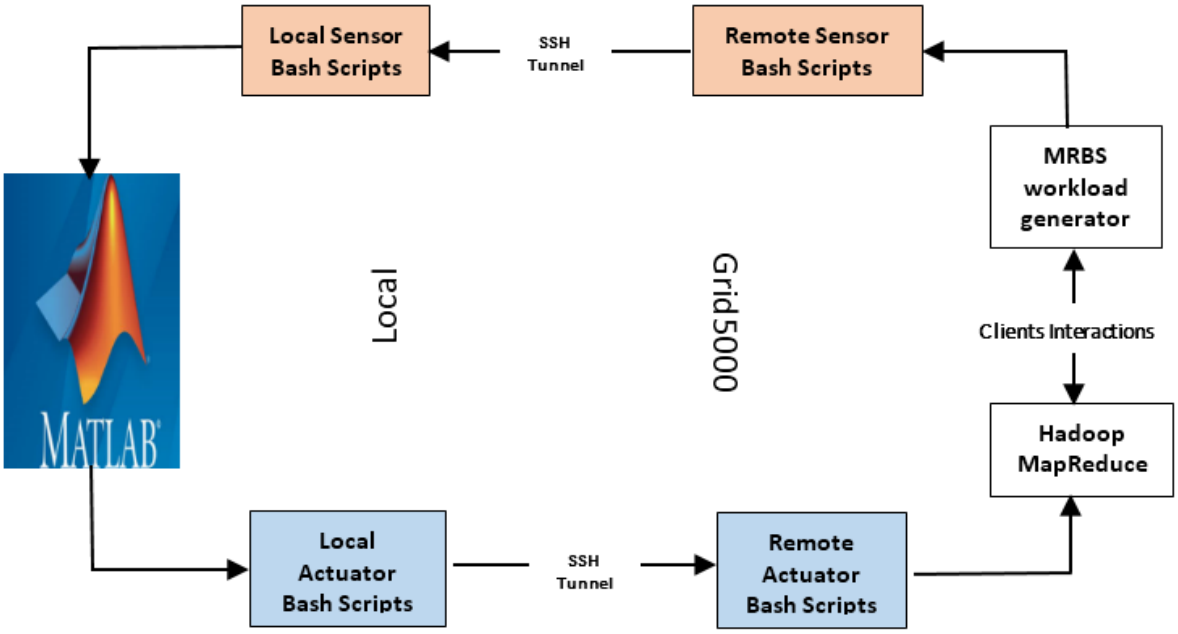


Figure 2. Experimental setup

Experimental Cloud Environment

The experiment was deployed on-line using Grid5000 platform. The clients run over the reserved nodes of the remote server with specifications; Quad core CPU 2.53GHz, a RAM of 15GB with disk space of 298GB, the network has an infinite Band of 20GB. The MBRS of the implemented MapReduce client's interactions runs on Apache Hadoop, the Apache Hive connects the Structured Query Language (SQL) into MapReduce jobs. Request over the cloud are BI types (Business Intelligence) with amount of data as large as 10GB. The response time of submitted MapReduce jobs were measured in the course of running the experiment using sensors script, both local and remote sensors that retrieve the performance metrics of the MRBS. The control inputs were targeted to be the number of nodes deployed and the number of clients running the MapReduce jobs in the Hadoop execution files, hence these rises to the provision of two sensors and actuators scripts written and implemented in Linux Bash Scripts. In the course of running the experiments, a sampling time T is assigned to be the period of the control loop, after any given instance of discrete time interval k, all files from the last time window $(k - T)$ are processed to obtain the performance.

MapReduce Modelling

The system identification modelling approach for dynamic systems was used. Due to the complex nature of MapReduce (multiple parameters up to 170), it was assumed that just a single job runs at a given time defined by the period T in the cluster size [9]. It is assumed

that only the steady state response of the system is captured by the MapReduce due to its high complexity. Software is subjected to updates constantly with time, these lead to a strong assumption that our captured dynamics (Mappers and Reducers) are high level performance metrics and are not affected during these changes, hence an agnostic model to such changes. This work was built as a black box model obtained based on data collected through experiments by varying the input (Number of clients) to measure the output (service time) and parameter estimation algorithm was run to obtain the model.

Model Structure

The base line assumption dealt with an agnostic model taking the high-level dynamics of the MapReduce as the map function (mappers) and reduce function (reducers). The proposed dynamic model is as shown in Figure 3.

The average service time *y(k)* is considered the output (at the *kth* interval of time) which has effect on both the mappers and reducers. The input *u(k),* is defined by the number of nodes under consideration in the cluster. The variable *d(k),* represent the number of clients, which changes in the course of running the experiments on the MapReduce and considered the measured disturbances. Within the limit of the operating points, the system is assumed to have a linear behavior and applying the superposition theorem, with output represented as in Equation 1.

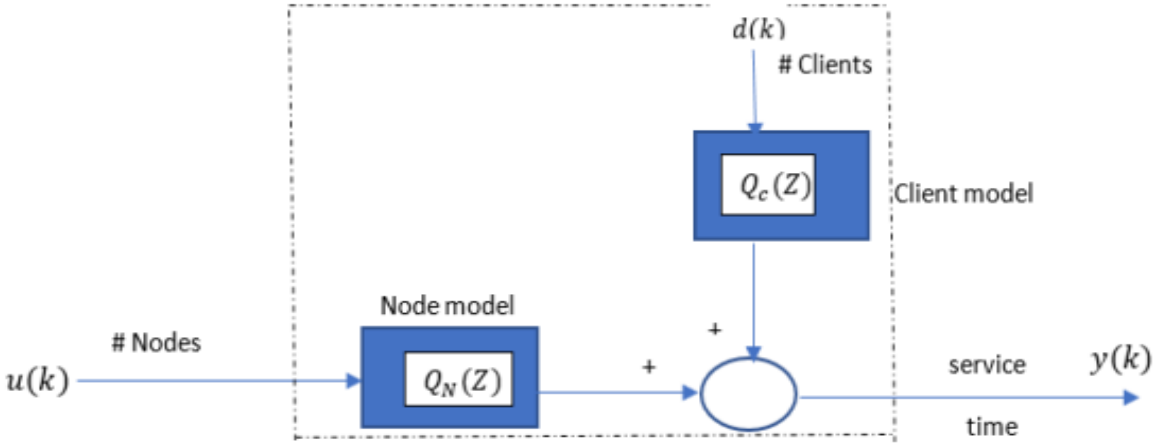


Figure 3. MapReduce dynamic model

$$y(k) = Q_N(z)u(k) + Q_C(z)d(k) \quad (1)$$

where Q_N (z) and Q_C (z) represent the discrete time model between service time and the number of nodes and clients respectively.

At any instant k, Equation 2 shows the weighted sum of the previous response time, and cluster size.

$$y(k+1) = -\sum_{i=1}^{m} x_i . y(k+1-i) + \sum_{i=0}^{n} y_i . u(k-T-i) \quad (2)$$

where T is the size of sliding window that assigns measurable dynamics.

In vector notation, with the unknown parameters within the vector θ, Equation 2 is represented as;

$$\theta = [x_1, x_2, \dots, x_m \quad y_1, y_2, \dots, y_n] \tag{3}$$

$$\emptyset(K) = [-y(k), \dots, y(k+1-n) \; u(k-T), \dots, u(k-T-m)] \tag{4}$$

$$y(k+1) = \theta^T.\emptyset \tag{5}$$

Applying the prediction error algorithms (PEM), the objective function is minimized using numeric optimization technique (Quasi-Newton method) with cost function;

$$J = \min_{\theta} \sum_{k=1}^{N} e^2(k) \tag{6}$$

The error (e) of the predicted output $y_n(k)$ to that of the measured output $y(k)$ for N size of data is given by;

$$e = y_n(k) - \theta^T.\emptyset(k-1) \tag{7}$$

For a steady state response in the compensatory path varying the number of nodes (control input) to measure the service time, the identified model with coefficients x_i, y_j and the delay T which were found by identification, have the form as given in Equation 8.

$$y_N(k) = -\sum_{i=1}^{N} x_i \, y_N(k-i) + \sum_{j=0}^{N} y_j \, u(k-T-j) \tag{8}$$

The direct path which introduces the required disturbance by varying the number of clients to measure the service time, the identified model coefficients x_q, y_r and a delay T which were found by identification is given by Equation 9:

$$y_C(k) = -\sum_{q=1}^{N} x_q \, y_C(k-q) + \sum_{r=0}^{N} y_r \, d(k-T-r) \tag{9}$$

Results and Discussion

Clients Variation Model

The nonlinear ARX model was used to obtain the clients' model. Figure 4(a) shows the result of the self-validation model, the model fits to 75.79% showing a good tracking of the real system.

From Figure. 4(b), after cross validation with a different data set, the nonlinear ARX model track the real system with a fit of 68.84%. The output unit, input unit and output delay were taken as [2 1 5] respectively. The final prediction error was estimated as 0.0779 which shows a high accuracy, with a loss function of 0.06998 and fit to working data of 85.8%. Based on the validation criteria [10], the model can be concluded to be a good model. The nonlinear ARX model obtained was tested and validated on a different data set 'clientmodelMC20', and a reasonable tracking of the real system was achieved.

Model regressor for the ARX model is as given in the Equation 10 as:

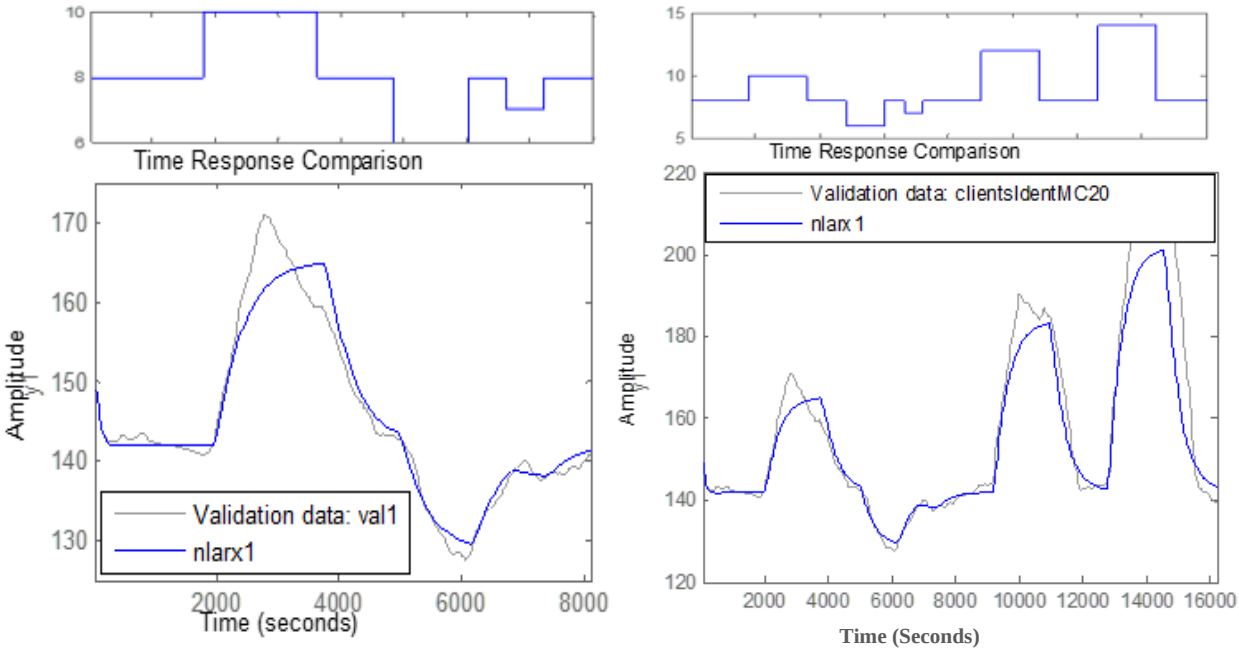
$$y1(k-1), y1(k-2), y1(k-3), y1(k-4), y1(k-5), u1(k-2) \quad (10)$$

where $y1(k-1), y1(k-2)$ ... shows that the next discrete output is related to the previous output.

The coefficient of the ARX model is:

$$y_C(z) = \frac{B(z)}{A(z)} u(z) + e(z) \quad (11)$$

where; $A(z) = 1 - 1.629Z^{-1} + 0.65Z^{-2}$ and $B(z) = 0.1927Z^{-5}$



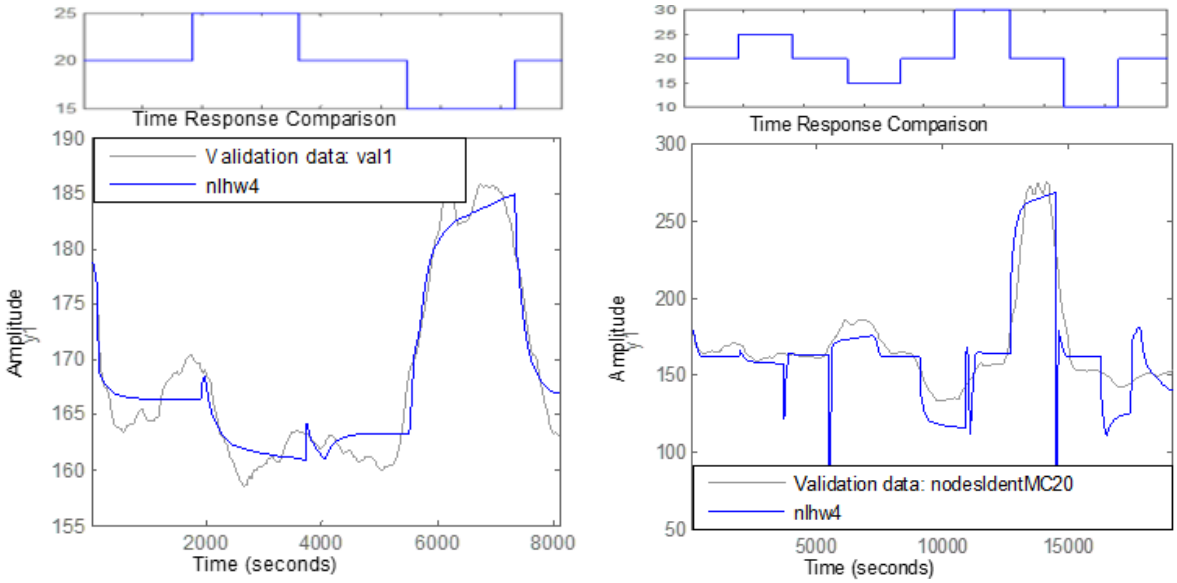
(a) **(b)**

Figure 4. (a) Clients self validation (b) Clients cross validation.

Nodes Variation Model

In the nodes model identification, the Hammerstein Wiener non-linear model with input dead zone and output piecewise forms of nonlinearities captured the dynamics of the system. As seen from Figure 5(a), for the nodes self-validation, the model fits to 75.48% and showing a reasonable representation of the real system.

From Figure 5(b), after cross validation, the model successfully tract the real system having a fit of 50.24%. The output unit, input unit and output delay were taken as [7 2 0] respectively.



(a) **(b)**

Figure 5: (a) Nodes self validation, (b) Nodes cross validation

The discrete-time output-Error model for the node is:

$$y_N(z) = \frac{C(z)}{F(z)}u(z) + e(z) \tag{12}$$

where; $C(z) = -0.5648Z^{-4} + Z^{-5} - 0.4353Z^{-6}$ and $F(z) = 1 - 1.836Z^{-1} + 0.3262Z^{-2}$

The nonlinear model obtained was linearized using the Matlab function *(getlinmod)* to apply superposition principle. The continuous transfer function is discretized by taking a sampling period of 30 seconds using Tustin bilinear transformation. The identified algorithms for the desired model structure of the MapReduce (second order linear difference model with delay) are given in Equation 13.

$$y(k) = z^{-4}\frac{0.1927Z^{-1}}{1-1.629Z^{-1}+0.629Z^{-2}}d(k) + z^{-4}\frac{-0.5641+Z^{-1}-0.4353Z^{-2}}{1-1.836Z^{-1}+0.3262Z^{-2}}u(k) \tag{13}$$

Finally, the linearized forms of the transfer functions obtained for both the clients and the nodes were tested on the real system with the nonlinear model to shows how approximate within the limit of operating points defined by the number of nodes and clients, the discretized model captured the dynamics of the real system.

Models Comparison

A comparison was made between the first order model (G) in [8],[9],[11] with the second order model (G2) obtained for both the clients and the nodes models. The linearized forms of the transfer functions for the client models are represented as;

$$G = \frac{1.07(Z^{-1}+1)}{1-0.79Z^{-1}}Z^{-8} \tag{14}$$

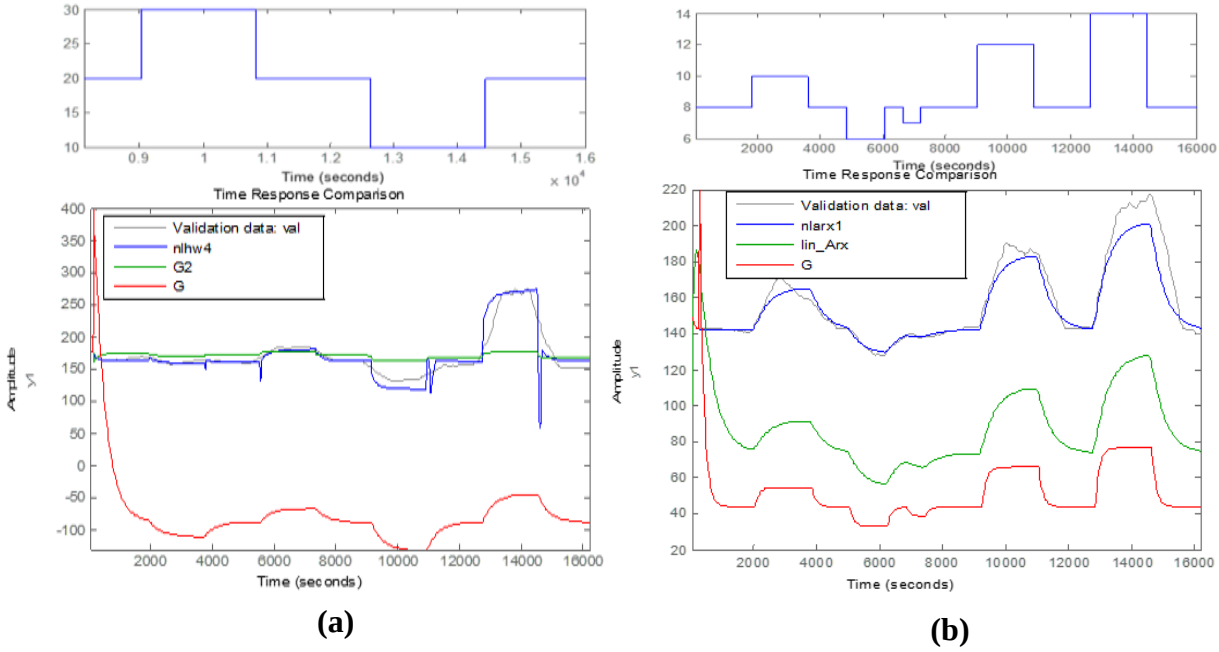
$$G2 = \frac{0.1927Z^{-1}}{1-1.629Z^{-1}+0.629Z^{-2}}Z^{-4} \tag{15}$$

From Figure 6 (a), the green plot (G2) is the second order linearized model which can be seen to have represented more precisely the dynamics of the system and closer to the real system compared to the plot (G) the first order model in [8],[9],[11].

Similarly, the linearized forms of the transfer functions for the nodes models are represented as;

$$G = \frac{-0.18(Z^{-1}+1)}{1-0.92Z^{-1}} Z^{-5} \tag{16}$$

$$G2 = Z^{-4} \frac{-0.5648+Z^{-1}-0.4353Z^{-2}}{1-1.836Z^{-1}+0.3262Z^{-2}} \tag{17}$$



(a) **(b)**

Figure 6: (a) Clients model comparison, (b) Nodes model comparison

For the node model in figure 6(b), it is clear that the first order model(G) has a wide variation from the real system which implies that the second order model(G2) represents more closely the real system and hence a better performance is expected.

Conclusions

This paper presents a model of MapReduce programming paradigm used for large scale distributed computing applied to cloud services. First, interactions were made with the Grid5000 cluster, an experimental test-bed for large data. On the cluster side, experiments were performed using Linux, an open-source Unix operating system. Data obtained was used to perform system identification using MATLAB toolbox. A second order nonlinear model was proposed.

The model was linearized and discretized to obtain the transfer function. Based on the performances obtained, it can be concluded that the second order model will give a better performance when implemented on the real system. From the perspective of future work, the error between two curves can be computed so as to clearly identified the best model. The proposed model can be chosen for developing control laws (such as adaptive feedback and feedforward, optimal feedforward etc.) on the real system.

Acknowledgement

Experiments presented in this work were carried out using the Grid5000 test-bed, supported by a scientific interest group hosted by INRIA and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

References

[1] H.S. Bhosale, and D.P. Gadekar, "A review paper on big data and hadoop," *International Journal of Scientific and Research Publications*, Vol. 4, No. 10, pp. 1-7, 2014.

[2] A. Iosup, X. Zhu, A. Merchant, E. Kalyvianaki, M. Maggio, S. Spinner, T. Abdelzaher, O. Mengshoel, and S. Bouchenak, *Self-Awareness of Cloud Applications: Self-Aware Computing Systems*, pp. 575-610, Springer, 2017.

[3] J. Dean, and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," In: *The 6th Conference on Symposium on Operating Systems Design & Implementation,* USENIX Association, Berkeley, United States, 2004, doi: 10.5555/1251254.1251264

[4] D. MacLean, "A very brief introduction to MapReduce," 2011. Retrieved from htpps://hci.stanford.edu/courses/cs448g/a2/files/map_reduce_tutorial.pdf

[5] A. Sangroya, D. Serrano, and S. Bouchenak, *MRBS: A Comprehensive MapReduce Benchmark Suite,* Research Report RR-LIG-024, LIG [Online], Grenoble, France, 2012 [Online]. Available: http://rr.liglab.fr/research_report/RR-LIG-024_orig.pdf

[6] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and Research Challenges," *Journal of Internet Services and Applications*, Vol. 1, No. 1, pp. 7-18, 2010, doi: https://doi.org/10.1007/s13174-010-0007-6

[7] S.N. Khezr, and N.J. Navimipour, "MapReduce and its applications, challenges, and architecture: A comprehensive review and directions for future research," *Journal of Grid Computing*, Vol. 15, No. 3, pp. 295-321, 2017, doi: 10.1007/s10723-017-9408-0

[8] M. Berekmeri, *Modeling and Control of Cloud Services: Application to MapReduce Performance and Dependability*, Thesis (PhD), Universit´e Grenoble Alpes, France 2015.

[9] M. Berekmeri, D. Serrano, S. Bouchenak, N. Marchand, and B. Robu, "A control approach for performance of big data systems," *IFAC Proceedings Volume*, Vol. 47, No. 3, pp. 152-127, 2014.

[10] S. Rachad, B. Nsiri, and B. Bensassi, "System identification of inventory system using ARX and ARMAX models," *International Journal of Control and Automation*, Vol. 8, No. 12, pp. 283-294, 2015. doi: 10.14257/ijca.2015.8.12.26

[11] S. Cerf, M. Berekmeri, B. Robu, N. Marchand, S. Bouchenak, and I.D. Landau, "Adaptive feedforward and feedback control for cloud services," *IFAC-PapersOnLine*, Vol. 50, No. 1, pp. 5504-5509, 2017. doi: 10.1016/j.ifacol.2017.08.1090.

[12] S.N. Khezr, and N.J. Navimipour, "MapReduce and its application in Optimization Algorithms: A comprehensive Study," *Majlesi Journal of Multimedia Processing*, Vol. 4, No. 3, pp. 1-33, 2015.