ICIC Express Letters
Part B: Applications
Volume **13**, Number **8**, August **2022**

ICIC International ©2022 ISSN 2185-2766
pp. **819–826**

GRAPHICAL MULTICONFIGURATION OF A SMART MINING FOG SIMULATION USING IFOGSIM TOOLKIT

Pujianto Yugopuspito* and Nicholas Prawiro Wei Hung Chen

Department of Informatics
Universitas Pelita Harapan
Jalan MH Thamrin Boulevard 1100, Lippo Village Karawaci, Tangerang 15811, Indonesia
nh70021@student.uph.edu
*Corresponding author: yugopuspito@uph.edu

Received December 2021; accepted February 2022

Abstract. *Mining is a very dangerous activity which causes numerous amounts of accidents every year; therefore, Smart Mining is a more automated solution for that. This work emphasizes the importance of easy interpretation in configuring Fog Simulations, about Smart Mining Fog Simulation in this context, by providing users with a multi-configured GUI to aid the learning process. The results of this work demonstrate an easy-to-use multi-configuration GUI that allows users to not only visualize the values in graphic format from custom Fog Device configurations, but also to save those configurations for later comparison and contrast situations between Cloudward and Edgeward module deployments. The sample configurations have also been tested and showed very significant differences in values especially for the performance measurement metrics, namely application loop delay and tuple CPU execution delay. For latency-sensitive topic like Smart Mining, easier interpretation of these differences can reduce the number of unexpected accidents in hazardous environments.*
Keywords: Fog Computing, iFogSim, Smart Mining, Multi-configuration

1. **Introduction.** CISCO introduced the term Fog Computing around the year 2012. In essence, Fog Computing is a decentralized computing structure that extends cloud-based services to the edge of the network by using devices that reside between IoT devices and the Cloud [1], with the goal to support real-time data processing and latency sensitive applications. The idea in this context is that Fog Devices such as gateways, switches, routers, can store application modules before they are being sent over to the Cloud [2].

So, in short, rather than having the collected data from insurmountable amounts of IoT devices around the world sent through a single network channel that links straight to the cloud, the data can be distributed to Fog Devices that reside closer to the source of information, resulting in lower latency and more real-time control.

According to research done by IBM and [3], every individual requires approximately 3.11 million pounds of fuel, minerals, and metals in his/her life. Hence, based on the information alone, we can deduce that Smart Mining requires loads of data analysis, and would pose a lot of risk too. During mineral and coal mining, for example, chemical reactions, hazardous gas emission, suffocation and rock sliding are among the many probably risks that could happen to the lives of the mining personnel [4]. Therefore, it is of utmost importance, that IoT devices relevant to the scope of mining are deployed to gain better productivity, reduce operational costs but gain enhanced safety.

Modelling the interconnected infrastructure of IoT devices to their Fog Devices to a central Cloud, would be extremely time-consuming, expensive, and too complex [5]. An alternative for this is made possible by the advent of Fog Computing simulators. This

DOI: 10.24507/icicelb.13.08.819

paper will use the iFogSim Fog Simulator toolkit to model the fog environment along with custom resource configurations.

A close related work discusses about the advent of agricultural initiatives like Hands-Free Hectare and [6] without direct human intervention leveraged the power of cloud computing in its operations related to sensors and actuators embedded in the drones and farming machinery like tractors; hence AgriFog [7], was a case study done on the extension of cloud computing to Fog Computing in the agricultural industry to further elaborate on dealing with complications in latency-sensitive applications, location awareness, distributed geography, mobility and the predominance of wireless access of real-time applications.

Another related work [8] shows several case studies following the implementation of the iFogSim toolkit. The Smart Mining industry case study provided in the aforementioned related work lacked necessary elaboration on many levels of Smart Mining, since it was more of a sample implementation idea on further use with the iFogSim toolkit.

Hence, the aim of this paper will be to elaborate on Smart Mining, together with the creation of a multi-configuration GUI that provides comprehensible graphical outputs in a charted format, and varying input configurations (e.g., Cloudward and Edgeward configurations). It hopes to make understanding and visualizing easier for anyone from business owners to even new aspiring learners.

The following sections of the paper will be organized as follows. In Section 2, the paper touches on essential theoretical grounding about the Smart Mining Fog Simulation using the iFogSim toolkit. Section 3 elaborates on the flow of methods and procedures that will be used to produce the multi-configuration GUI. Section 4 shows the result of the multi-configuration GUI, and discussions of various sample configurations and the comparison of their outcomes. Finally, conclusions and future works are presented in Section 5.

2. Theoretical Foundation.

Cloud computing delivers infrastructure, platform, and software (application) as services, which are made available as subscription-based services in a pay-as-you-go model to consumers [10]. These services are referred to as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Cloud [11] computing aims to power the next generation data centers by architecting them as a network of virtual services (hardware, database, user-interface, application logic) so that users can access and deploy applications from anywhere in the world on demand at competitive costs depending on users QoS (Quality of Service) requirements [12].

Fog Computing is a term coined by Cisco and defined as a distributed computing paradigm that extends the services provided by the cloud to the edge of the network [13]. It enables the seamless convergence of infrastructure stretching from the cloud datacenter to devices on the network edge (including intermediate devices like ISP gateways, cellular base stations, and private cloud deployments) into a continuum of resources, to be provisioned to multiple tenants for hosting applications.

iFogSim is a Fog and IoT environments simulator dedicated to managing IoT services in a Fog infrastructure [14] as an alternative to a high-cost, unrepeatable and uncontrollable real IoT environment, as illustrated in Figure 1 [5].

Within iFogSim, several examples have already been deployed for learning purposes under the package name *org.fog.test.perfeval*. This is where the user will have to define the physical, logical and management components before the simulation can be run.

3. Methodology.

The Smart Mining system proposed by [9] follows a straightforward master-worker application model with four modules that include

1) Master Module: This module will collect the data from all sensors, categorize the data and send the specific data to the respective modules for further processing. This

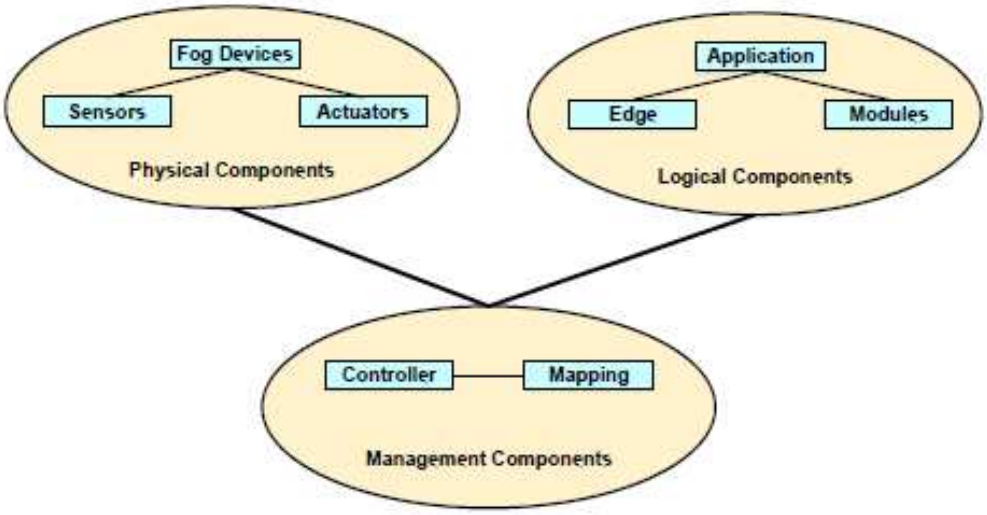

FIGURE 1. High-level overview of the interactions among the iFogSim components

FIGURE 2. Improved master-worker application model lifecycle of the Smart Mining environment

module will also collect the processed data (response) from the modules and send it to the respective actuators. This will be situated within the master node.

2) Gas Info Module, Chemical Info Module, Surrounding Info Module: These modules are responsible for processing the raw data collected by the sensors. The raw values will be analyzed, and the outputs will be generated into a response. These modules will also be the situation within the Master Node, as shown in Figure 2.

The previous Smart Mining source code already had a "vague" fog topology in place, but the absence of a fog medium in the 2nd tier felt a little "lonely"; hence a revamp of the fog topology was done. A router was added to fill the spot in the 2nd tier, so that

TABLE 1. Smart Mining fog hierarchy comparison

Tier	Fog Device [9]	Proposed Fog Device
0	Cloud	Cloud
1	Proxy Server	Proxy Server
2	–	Router
3	Fog Node	Master Node Microcontroller
4	Sensor + Actuator Devices	Sensor + Actuator Microcontroller

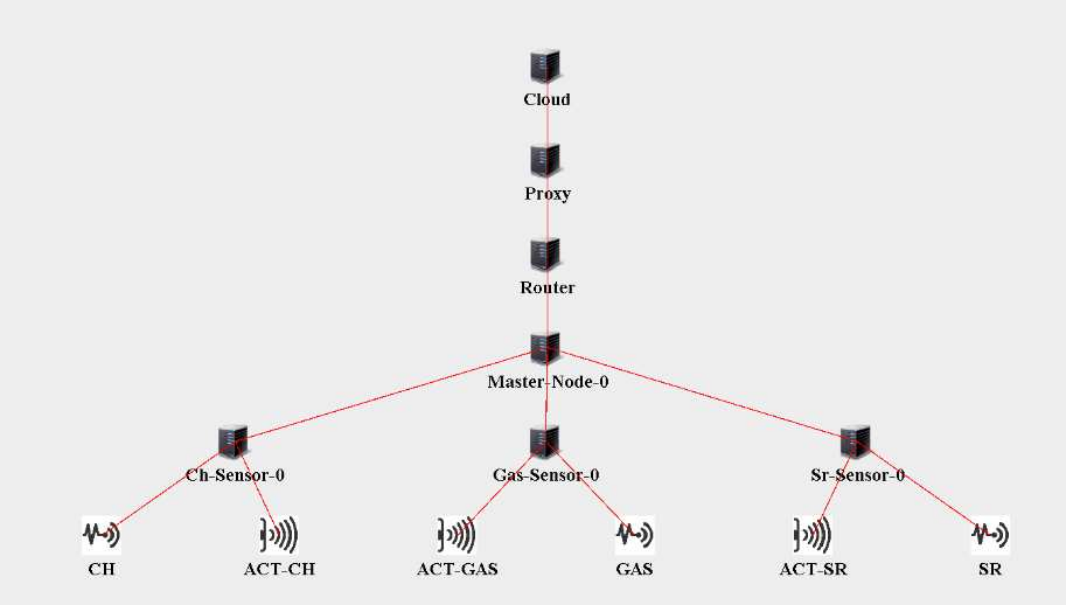

FIGURE 3. Smart Mining visual topology

TABLE 2. Fog Device specifications

Fog level	Fog Device	MIPS	RAM (MB)	Uplink (Kbps)	Downlink (Kbps)	Rate per MIPS	Busy power	Idle power
0	Cloud	44800	40000	100	10000	0.01	$16 * 103$	$16 * 83.25$
1	Proxy Server	7000	4000	10000	10000	0.0	107.339	83.433
2	Router	2800	4000	10000	10000	0.0	107.339	83.433
3	Master Node Microcontroller	4744	1000	10000	10000	0.0	107.339	83.433
4	Sensor + Actuator Microcontroller	80	0.16 (min 1)	10000	10000	0.0	107.339	83.433

the transmission of data streams within the network would feel more all-encompassing in this paper, as tabulated in Table 1, illustrated in Figure 3.

Given that there are not any rudimentary options to follow by, for the purpose of variance – the higher-tier (1st and 2nd tier) fog specifications will follow the nodes in the pre-included DCNSFog sample. The lower-tier (3rd and 4th tier) fog specifications will emulate the real-life brands, Raspberry Pi 2 and Climastick v1.1 single core, respectively. It will provide a more ideal simulation experience, as shown in Table 2.

JavaFX with Scenebuilder will be the prime driver in the creation of all the graphics seen in this paper. The console outputs will be referenced to create the visual bar charts. The *Controller* class in the package *org.fog.placement* from the iFogSim toolkit, is linked

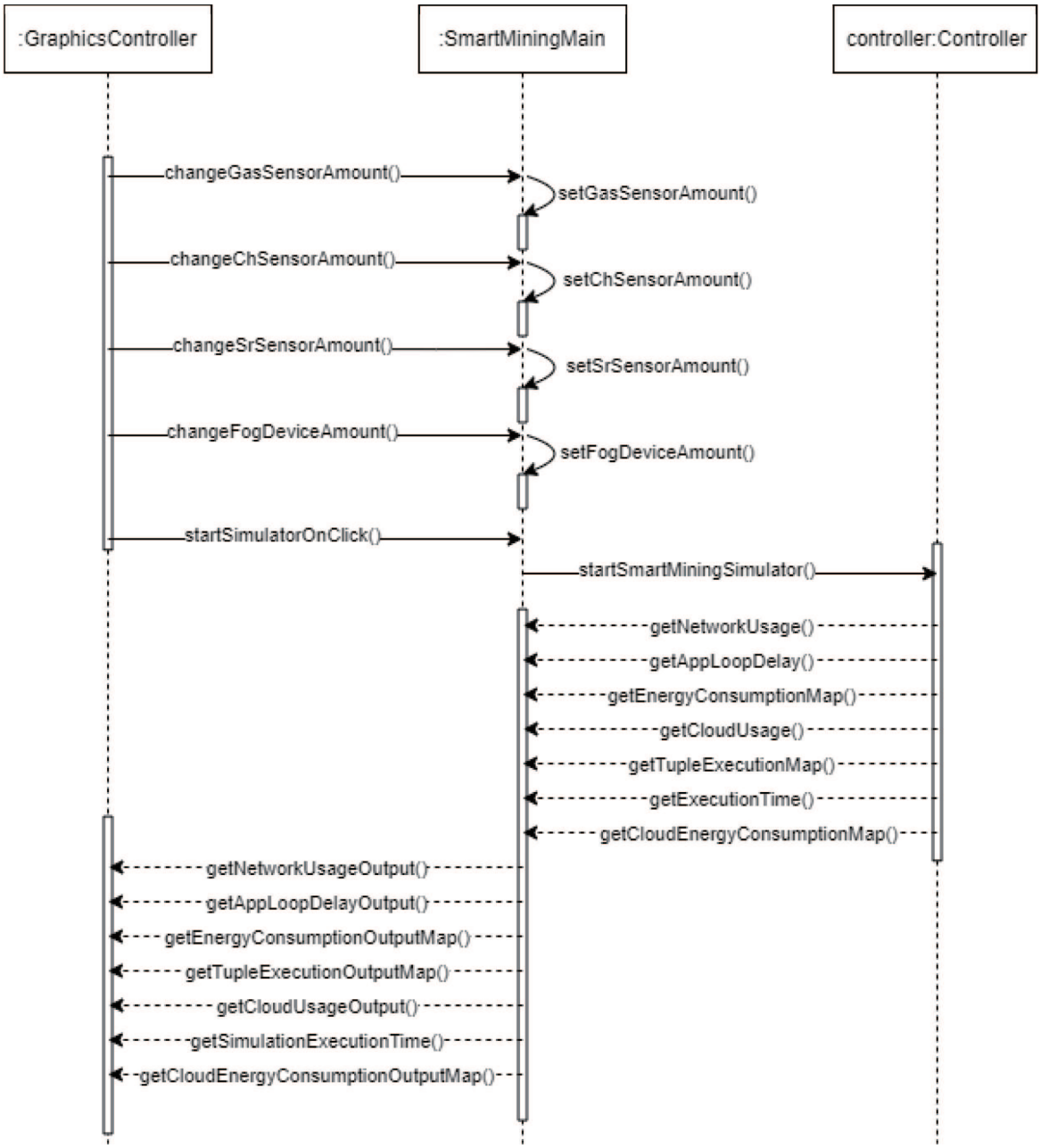

FIGURE 4. Sequence diagram of obtaining the console outputs and getting them into *GraphicsController*

to all the components within the class relationship diagram; hence, this class will also contain the methods that are responsible for the output of each and every performance measurement metric. Getters and Setters will need to be made in a logical order so that the outputs can then be retrieved from the *Controller* class to the main GUI controller class, which we will call the *GraphicsController*. This is planned out in the sequence diagram shown in Figure 4. After the classes have been created and linked with each other, the planned full lifecycle of the application will then be actualized in Figure 5.

4. **Results and Discussion.** In the initial attempt to run the source code, it contained loads of errors. Much debugging, re-designing and perfecting had to be done before the simulation produced any meaningful results. Once the desired performance measurement metric outputs were shown in the console output, it was only a matter of the correct placement of the getter and setter methods in the relevant classes (*Controller, SmartMiningMain, GraphicsController*) as can be seen in Figure 3, to get the multi-configuration

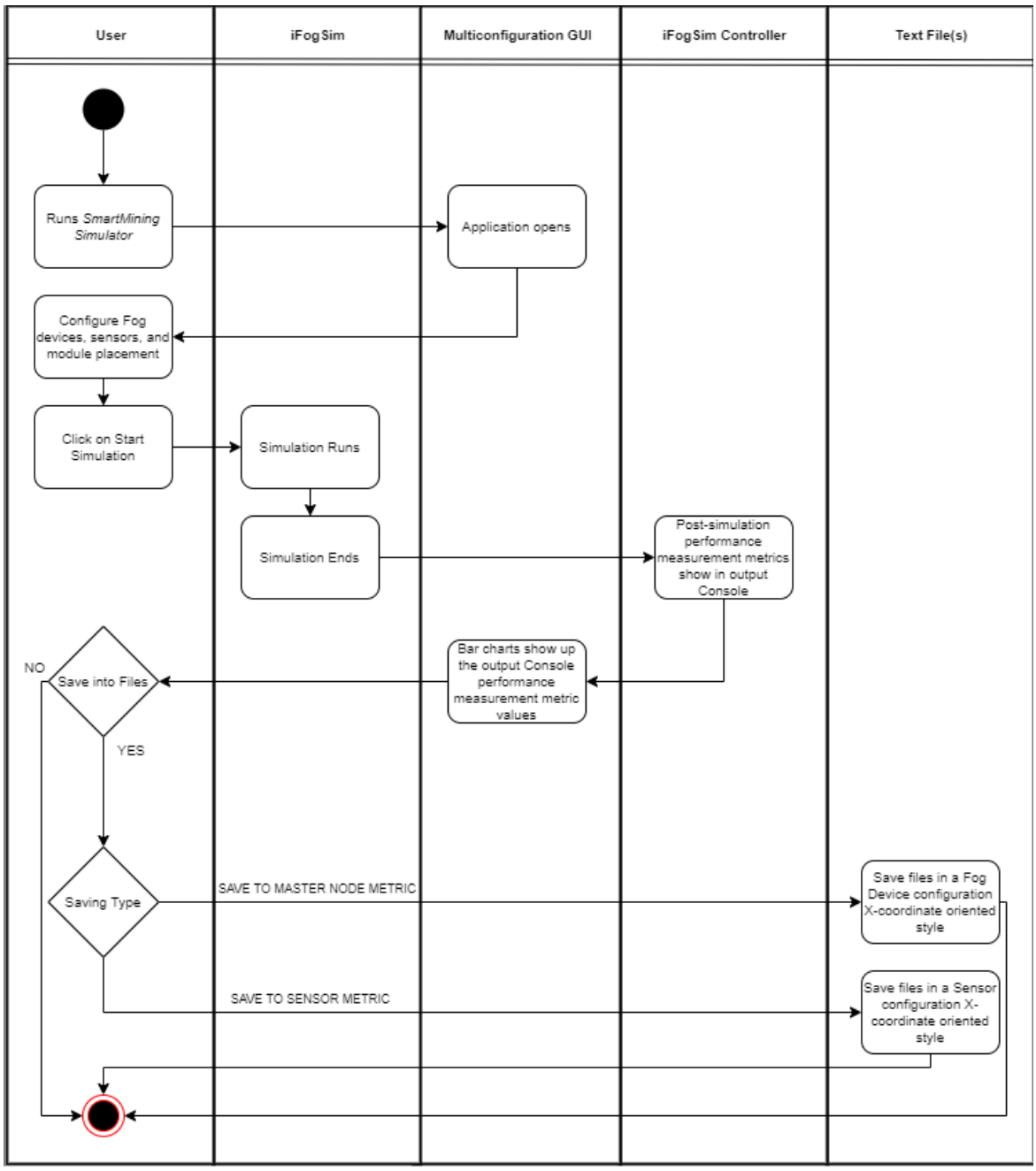

FIGURE 5. Activity diagram lifecycle of the Smart Mining Fog Simulation

GUI up and running. The result of the completed GUI is shown in Figure 6. The left figure portrays the outlook of the multi-configuration selection panel, whilst the right figure portrays the results of the simulated configuration.

Following the completion of the GUI, several configurations were tested and compared with each other. Performance metrics like network usage (both for the main Fog network usage and Cloud network usage), Fog Device energy consumption and Cloud energy consumption showed close constants despite the variation in Fog Device and sensor configuration. However, application loop delay and tuple execution delay showed significant differences between Cloudward module placement (application modules are placed in the Cloud) and Edgeward module placement (application modules were placed closer to the sensors and actuators, within the Raspberry Pi 2 microcontrollers themselves).

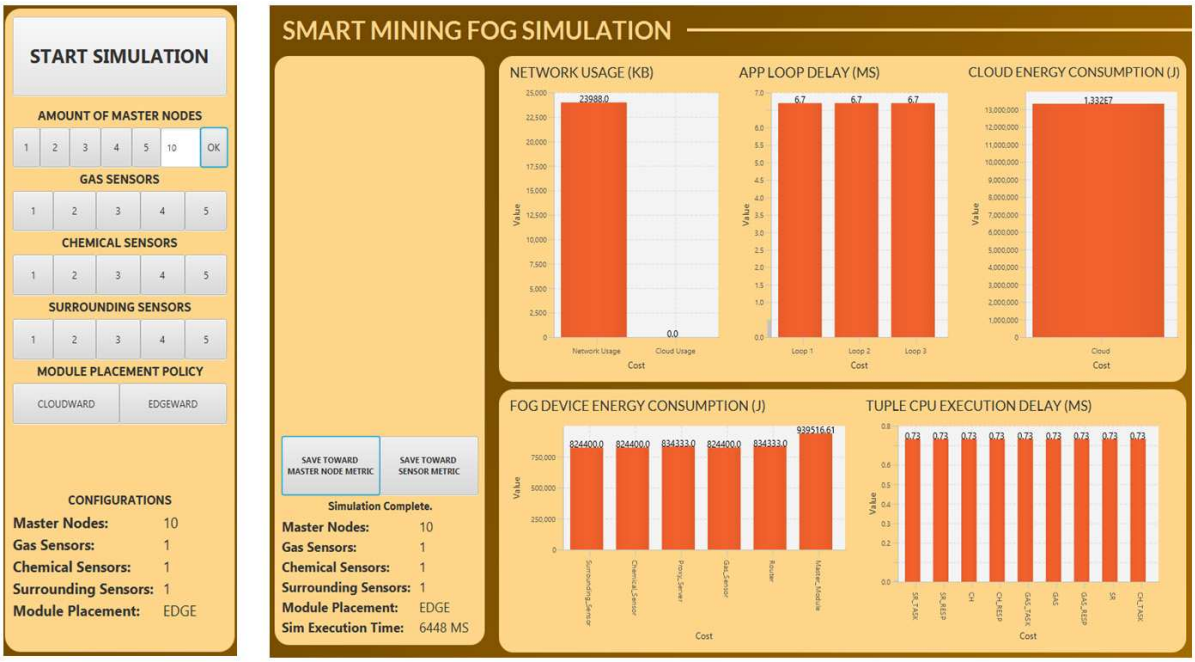

FIGURE 6. Completed result of multi-configuration GUI

TABLE 3. Cloudward application loop delay for Chemical Sensors – Actuators

# Raspberry Pi Fog Device	Application loop delay time (ms)	Tuple CPU execution delay time (ms)		
	ACT_CONTROLCH	CH	CH_TASK	CH_RESP
1	608.3	0.17	0.10	0.04
3	957.9	0.11	0.12	0.14
5	3,740.1	0.10	0.16	0.12
10	4,910.6	0.15	0.10	0.09
50	5,090.6	0.10	0.16	0.12
100	5,943.1	0.10	0.11	0.09

In this paper, we showed the differences in application loop delay and tuple execution delay values for the chemical sensors and actuators. The values for chemical sensor-actuator application loop delay through Edgeward module placement showed a constant of 6.7 ms, despite the changes in the Raspberry Pi microcontroller Fog Device amount, and the values for chemical sensor-actuator tuple CPU execution delay through Edgeward module placement showed a constant of 0.73 ms despite the changes in the Raspberry Pi microcontroller Fog Device amount. However, in Cloudward module placement, these values differ significantly and can be seen in Table 3 as an example for Chemical Sensors – Actuator. This example shows the loop delay time and tuple CPU execution delay time of Chemical Actuator (ACT_CONTROLCH), and Chemical, i.e., sensor (CH), task (CH_TASK), and response (CH_RESP), respectively.

5. **Conclusions and Future Works.** With the creation of a multi-configuration GUI to aid new learners in Fog Computing, understanding the Fog Computing concepts has become much easier, thus providing the learners a better perspective on what kinds of configurations would better suit their use case and budget with respect to Smart Mining. However, this paper did not thoroughly cover the value anomalies found within the configuration comparisons, specifically in application loop delay and tuple execution delay. Future work can be done to better explore the anomalies, so that the comparison results

can be better interpreted, therefore, better predictions and decision-making can also be rendered through the Smart Mining Fog Simulation.

REFERENCES

[1] R. Mahmud, R. Kotagiri and R. Buyya, Fog computing: A taxonomy, survey and future directions, *J. Big Data*, vol.6, no.111, pp.103-130, 2019.

[2] D. Silva, G. Asaamoning, H. Orrillo, R. Sofia and P. Mendes, An analysis of fog computing data placement algorithms, *Proc. of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pp.527-534, 2019.

[3] E. I. Gaura, J. Bursey, M. Allen, R. Wilkins, D. Goldsmith and R. Rednic, Edge mining the Internet of Things, *IEEE Sensors Journal*, vol.13, no.10, pp.3816-3825, 2013.

[4] M. Afrin, J. Jin, A. Rahman, Y. Tian and A. Kulkarni, Multi-objective resource allocation for Edge Cloud based robotic workflow in smart factory, *Future Generation Computer Systems*, vol.97, pp.119-130, 2019.

[5] R. Mahmud and R. Buyya, Modelling and simulation of fog and edge computing environments using iFogSim toolkit, *Fog and Edge Computing*, pp.433-465, 2019.

[6] W. Bai, H. Zhang and C. Chen, A new cooperative cache optimization algorithm for Internet of Vehicles based on edge cloud network, *International Journal of Innovative Computing, Information and Control*, vol.16, no.3, pp.1059-1075, 2020.

[7] V. Sucharitha, P. Prakash and G. N. Iyer, AgriFog – A fog computing based IoT for smart agriculture, *International Journal of Recent Technology and Engineering*, vol.7, no.6, pp.210-217, 2019.

[8] K. S. Awaisi et al., Toward a fog enable efficient car parking architecture, *IEEE Access*, vol.7, pp.159100-159111, 2019.

[9] K. S. Awaisi, A. Abbas, S. U. Khan, R. Mahmud and R. Buyya, Simulating fog computing applications using iFogSim toolkit, in *Mobile Edge Computing*, A. Mukherjee, D. De, S. K. Ghosh and R. Buyya (eds.), Cham, Springer, 2020.

[10] R. Calheiros, R. Ranjan, C. de Rose and R. Buyya, CloudSim: A novel framework for modeling and simulation of cloud computing infrastructures and services, *arXiv.org*, arXiv: 0903.2525, 2009.

[11] A. Weiss, Computing in the clouds, *netWorker*, vol.11, no.4, pp.16-25, 2007.

[12] R. Buyya, C. S. Yeo and S. Venugopal, Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities, *Proc. of IEEE International Conference on High Performance Computing and Communications*, 2008.

[13] F. Bonomi, R. Milito, P. Natarajan and J. Zhu, Fog computing: A platform for Internet of Things and analytics, *Big Data and Internet of Things: A Roadmap for Smart Environments*, pp.169-186, 2014.

[14] M. I. Naas, J. Boukhobza, P. R. Parvedy and L. Lemarchand, An extension to iFogSim to enable the design of data placement strategies, *Proc. of IEEE 2nd International Conference on Fog and Edge Computing (ICFEC)*, pp.1-8, 2018.