AgriFog- A Fog Computing based IoT for Smart Agriculture

Sucharitha. V, Prakash. P , Ganesh Neelakanta Iyer

Abstract*: *The smart agriculture or precision agriculture is contemplated to play a vital role in augmenting the various farming activities. IoT based farm management systems have emanated from the rapid expansion of connectivity. The smart agriculture or precision agriculture is contemplated to play a vital role in augmenting the various farming activities. The existing systems which are based on traditional cloud models are inadequate to handle the large amounts and variety of data generated by the IoT devices connected. In order to decrease the latency in aiding the real time decisions based on the data produced, it is essential to bring the data processing closer to the source of its production. This can be addressed by adopting the fog based models. An IoT-Fog based farm management system can be more competent in terms of optimal bandwidth utilization and low latency for real time decision making. The architecture of the proposed approach has been presented and elucidated. The AgriFog application has been modelled and simulated using iFogSim. The results substantiate the postulate that the fog based model of the farm management system is more efficient and preferable for adoption because of its support for effective scalability with better response time and reduced latency.

Index Terms*: *Fog Computing, Farm Management System, IoT, Smart Agriculture

I. INTRODUCTION

Agriculture, being the largest source of food production is a mainstay of a country's economy. The evolution of automation has diffused into the various realizations of farm management systems. Automated sections of agricultural systems with high monetary outcomes focus on one single area. The Hands-Free Hectare [1] is an initiative in United Kingdom, where a hectare of barley crop was tended, grown and harvested completely without direct human intervention. This was made possible by the sensors and actuators embedded in the drones and farming machinery like tractors, rovers and harvesters. FarmBot [2] is an open source automated precision farming machine which can be used to grow food in constrained areas like backyards and rooftops with minimum intervention. The ecological factors like carbon footprint have been taken into consideration and minimized during various processes. Sowing, soil monitoring, watering, weeding and harvest alert are the functionalities included. The mobile and web interfaces have been built for seamless remote management of the farm.

Cloud paradigm helps the applications to be up and running with its agility, resource pooling and sharing. The specifications need not be elaborated because of the pay-as you-go Cloud Computing model. The lack of elaborate description induces delays in latency-sensitive applications which are specific about the placement of nodes in the same proximity. The need for a paradigm to support data management, low-latency and analytics efficiently lead to the introduction of Fog Computing [3] which is an extension to Cloud Computing. Some of the expounding attributes of the Fog are low latency, location awareness, distributed geography, mobility, pre-dominant role of wireless access and the strong presence of streaming and real-time applications with heterogeneity. In the subsequent sections, we discuss the related work followed by the system design and implementation. The results obtained are delineated with the corresponding conclusions. Finally, possible future extensions are suggested.

II. RELATED WORK

In [4], the description of a farm management system is concerned with the organization and operation of a farm with the objective of making agriculture more profitable and efficient through automation. In other words, a farm management system is a system for collecting, processing, storing and disseminating data in the form of information required to execute the functionalities that exist within the farm. Evolution of these systems is observed as the coupling with the farming equipment to facilitate the automatic implementation of decisions as designed according to the requirements specified by the farmers. A smart farm includes automation of the operations that are part of the farm. Some of the essential operations that are automated are water level management, soil profile data collection and environmental data collection. A significant number of use cases were produced in [4] based on the predefined conditions. A list of generic enablers which are a set of domain neutral software tools like cloud hosting, data/context management services, service delivery framework, IoT services Enablement and security which are common to all internet applications were introduced. Upon extensive analysis of the use cases, functional architecture of an FMS system is described with all essential modules to facilitate seamless support and extension through the integration of autonomic and cognitive elements.

Revised Manuscript Received on March 20, 2019.

Sucharitha.V, Department of Computer Science and Engineering, Amrita School of Engineering, Amrita University, Coimbatore, India.

Prakash.P, Department of Computer Science and Engineering, Amrita School of Engineering, Amrita University, Coimbatore, India.

Ganesh Neelakanta Iyer, Department of Computer Science and Engineering, Amrita School of Engineering, Amrita University, Coimbatore, India.

An implementation of an open architecture that is used to corroborate numerous futuristic concepts for the agricultural sector is presented in [5].

The application is based on Generic Enablers. Cloud Edge is one of the six categories from which a number of generic enablers were integrated into the model. The issue of open and standardized interfaces for the Cloud FMS is identified as the most crucial issue while converting the concept of a market place of services developed by independent providers into reality. A substantial case study of designing IoT enabled applications in precision agriculture and ecological monitoring is discussed in [6]. A multiple and concurrent views based approach that has been widely adopted by the software development community and aligned with the latest international standard ISO 42011 is used to define the architecture of the application. Weather data including air temperature, air pressure, humidity and wind speed are collected as a part of the ecological monitoring module. An illustration of an implementation of the proposed architecture along with its initial evaluation has been included in [6].

Some peculiar challenges while designing a sensor-cloud system for agricultural applications have been addressed in [7]. The virtualization technique underlying the proposed sensor-cloud framework has also been mathematically characterized. A theoretical prototype that is suitable for multi-user, multi-organization and multi-application scenario has been depicted in this work. A reference to the ZeroConf protocol that allows the applications programmers to interact with the sensors without the necessity for physical configuration has been included in their design. The proposed system model was analyzed on the basis of its energy consumption, duty and network lifetime. An Indian scenario of remote agricultural monitoring using IoT has been discussed in [8]. The major modules of the proposed system are farm side, server side and client side. The Farm Side module consists of the Ubi-Sense motes, each of which is a generic sensor board having Temperature and Relative Humidity, Light Intensity, Barometric Pressure, Proximity sensing and Buzzer. Zigbee technology is used for supporting the fast access to the equipment within short ranges. An automated monitoring system which can be mounted on a multi-rotor UAV or carried by human operators for handheld use was proposed in [9]. The onboard sensing modalities include LiDARs, spectrometer and a thermal camera. They were selected to monitor a range of plant physiological and morphological properties. Methods to estimate the plant morphology, plant vigor, leaf area and fruit counts. Machine learning techniques employed to estimate properties of interest such as leaf area, and fruit count from the data acquired by the sensor suite along with the state of the art tools to visualize the data generated by the system. A comprehensive field monitoring and automation using IoT in Agriculture Domain have been presented in [10]. The work proposes an e-Agriculture application based on KM Knowledge base and monitoring modules. Plant growth monitor, irrigation planner, Crop Profit Calculator and Water need calculator are some of the major modules described. An algorithm is framed in this work for estimating the water need using the concept of evapotranspiration. A comparative study of the existing systems based on factors like knowledge base, monitoring modules, efficiency and reliability. The Cloud Computing based Smart Agriculture IoT systems have been surveyed in [11] to understand the diverse technologies in order to come up with sustainable smart agricultural applications. A multilayered IoT Agriculture architecture is proposed along with the description of a mathematical explanation for the High Yield Process. In [3], the initial definition of the Fog Computing and its characteristics are mentioned. The work aims at establishing why the Fog layer is appropriate for a large number of applications. The striking features of Fog are: Low latency and location awareness, widespread geographical distribution, Mobility, Very large number of nodes, Predominant role of wireless access, Strong presence of streaming and real-time applications with Heterogeneity. Cognition and Agility as mentioned in [12], are the factors that can be taken for advantage while adopting Fog Computing models. Smart Agriculture and Precision Agriculture are the applications of improved IoT and Cloud-based solutions which can be extended to be implemented in the fog computing paradigm. Farm management system overlaps certain functionalities from both smart agriculture and precision agriculture. Its requirements of low latency, heterogeneity, and support for a large number of nodes, mobility, widespread geographical distribution and location awareness can incur a lot of cost if the implementation is a cloud-based one. This is because the farms are usually a huge source of data and all the data if sent to the cloud for processing could cause huge network traffic and induce large delays and increase latency. The Fog based paradigm does a low-level processing on data and does not require a consistently alive network connection unless the remote monitoring is done live. The local decision making can take place in the fog nodes which have considerable storage and computing capabilities. This reduces the latency and network costs to a large extent. The work on Fog Computing[13] describes major components in the Fog Architecture. Smart Traffic Light System and Wind Farm are the two use cases discussed in this work. The objective of achieving distributed policy-based orchestration resulting in the scalable management of individual subsystems and the overall service is also delineated. This work[14] exposes the various fog computing software systems and explores the various fog computing applications along with the modeling and simulation platforms. From this work, it can be discerned that the vast potential of IoT can be reached through the advancements in Fog Computing. In [15,18], the various solutions that Fog Computing presents in order to address the IoT challenges. A few open questions and research challenges pertinent to fog computing as an emerging area of research have been included in the paper.

Mobile Fog [16] is a high-level programming model for IoT based applications that are geographically distributed, large in scale and latency sensitive. This is one of the first fog based programming models that can be easily adopted for the development of any Fog Based IoT application without any complications. [17] WM-FOG is a computing framework for the fog domain whose design embraces a flexible software architecture,
which can incorporate different design choices and user-specified polices. More specifically, it provides a flexible way to define workflows that can be easily deployed and executed on fog-based systems.

A discussion on fog computing, its applications and issues have been presented in [18]. Possible IoT applications that can use the Fog have been included in this work along with a customized Fog Computing Architecture. A comparison of the parameters of the Fog and Cloud Computing is contained in this paper.

III. PROBLEM STATEMENT

The farm management system, when implemented using IoT and cloud based mechanism can have setbacks in energy consumed, data processing, network communication and infrastructure for large scale extension. Our areas of concentration include streamlined energy and network utilization. Low latency data processing to power real time decision making, cost efficient system design and deployment are the other key aspects where we channel our efforts into. The insights gained from the survey of related work, the following issues have been considered for our solution:

(1)Data processing closer to its source of generation

(2)Scaling the system with diverse modules across the distributed infrastructure with low additional cost

(3)Providing efficient network bandwidth and energy utilization

(4)Low latency data processing to enable faster real time decision making.

The existing farm management based on the cloud and IoT solutions have been proposed in order to deal with some of the above mentioned challenges. However, these approaches do not provide comprehensive and flawless designs. Our research and analysis helped to discern the weaknesses and understand the lacking design requirements that need to be met in order to build a robust system. However, most of the current systems are focused on either precision agriculture or animal husbandry. A full-fledged farm management system needs to be an aggregation of smart agriculture modules coupled with animal management modules. The proposed design can support the inclusion of diverse modules in a customized manner without a significant rise in network utilization and increase in response time.

IV. APPLICATION DESIGN

In a generic perspective, Farm Management can be perceived as a skill to efficiently utilize the resources and achieve higher profits through best practices of optimization. By observing major parameters of the farm, scarce and surplus resources can be discerned. Based on this knowledge, the precise allocation of resources can be accomplished. To come up with the model of a farm management system from square one, the initial step was to complete the design of a basic application catering to the basic requirements. As part of the software development life cycle, the requirements gathering phase yielded in an extensive list of use cases among which the most essential high level use cases as listed in Table 1 and have been chosen for the implementation of the first version of AgriFog application.

Water and nutrients are the most essential resources that are necessary for the crops to grow in good health. Supplying the right amount of water to the plants is of prime importance because overwatering as well as under watering both have adverse effects on the plant growth. By monitoring the water levels, farmers can understand the exact water requirement of the field and water it accordingly. The nutrients like nitrogen, phosphorus and potassium are required in variable amounts as per the crop type and growth. Also, different stages of the crops will have different nutrient consumptions. To cater to the nutrient demands, farmers need to know precise numbers about the deficiency. The mismanagement of water and its over utilization when not required can have detrimental repercussions in the future. Whereas in the current scenario where the issue of water shortage has surfaced, the water pump can run out of water at the source which results in the dry run condition. When the motor pump dry runs, then it gets damaged and demands the replacement of expensive spare parts. Thus the water level at source needs to be monitored so that when the water level goes below a certain level, the motor pump is signaled to be turned off automatically. Assuming that a farm landowner can have multiple farms that are geographically distributed, it becomes tedious for the owner to observe the conditions in each of the farms simultaneously.

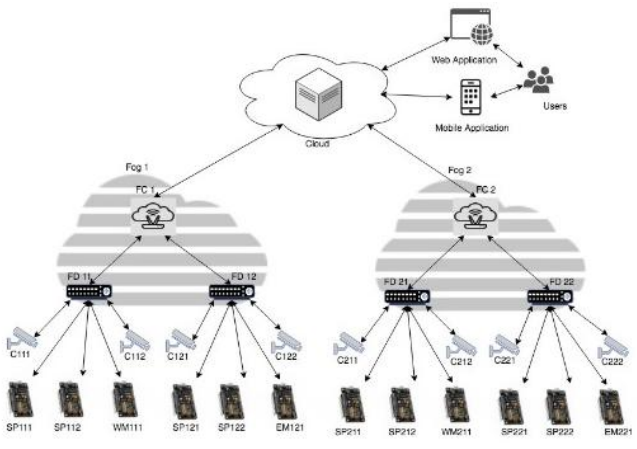

Figure 1: Physical Network Hierarchy

The AgriFog application supports remote access of the Farm Management Data with complete access for the owner. From a bigger perspective, the consolidated data of the soil profile and ecological monitoring is required by the Government to populate Agricultural census which are the major source of information on cropping patterns, irrigation status and soil information. Since the farm data is updated in real time, it is made available for the Government adhering to the owners preferred privacy policies.

Table 1: Actors and their corresponding Use Cases

Actor / Role	Use Case
Farmer	Monitoring soil profile and supplying the deficient nutrients
Farmer	Observing the soil moisture and dispense the adequate amount of water
Helper	Monitor the water level in the source and turn off before the pump is run in dry conditions
Owner	Data Analysis and monitoring of all the vital information like ecological conditions, soil profile and security concerns in the farms located at geographically distinct locations
Government	Agricultural census

After fixing upon the requirements by defining high level use cases, the next step is to come up with a model for the AgriFog application. The process of system modelling is used to conceptualize and construct the application from a realistic perspective. The functional modelling of an application delineates the functions which compromise the activities, actions, processes and operations within the specified constraints. Detailed discussion about each functionality is discussed in the System Design and Implementation section where the various modules are incorporated into the application model represented as a directed acyclic graph(DAG) in order to support the implementation of the system in iFogSim[19]. The Farm Management Systems require location and hierarchy-aware processing to handle data streams from the extensively distributed edge devices. An AgriFog application instance consists of distributed processes that are programmed onto the distributed computing instances in the fog, cloud and the edge devices. The sensing and aggregation which are application specific tasks are carried out by the running process in its corresponding location and level in the network hierarchy. The physical hierarchy of devices through a logical order of application processes is described in the Figures [1,2]. The hierarchy of physical devices is clearly depicted in the Figure [1]. As visible in the illustration, a process on an edge device becomes a leaf node whereas the processes which run in the cloud become the root nodes in the hierarchy. The intermediate nodes are the processes which run on the fog layer nodes. A connection between two processes indicated a path between any two nodes in the hierarchy. The path may not be a direct physical connection but can be a logical connection. The communication among the root nodes is within the corresponding data center, thus it is relatively less expensive when compared to the interconnectivity between the processes in the other layers. The Fog Device in AgriFog handles the workload from the geospatial region which it belongs to. Figure[1] shows the processes c111, c112, sp111,sp112,wm111 running in the edge nodes in region 1 mapped to the Fog device FD11. Similarly, the fog devices fd11 and fd12 in one location are connected to the gateway FC1 which in turn connects to the core cloud processes covering 2 geographically separated farms. Once the system is understood in terms of its concept and the context, it is necessary to understand and plan for the deployment of the system. The application is built upon the basis of the plug and use type of design as discussed in MOSDEN [21]. Since the use cases and the basic functional model is available, the number of nodes that need to be deployed in a test bed must be systematized.

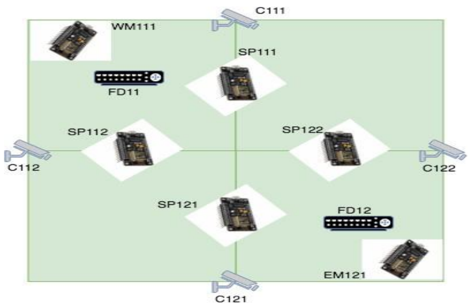

Figure 2: Physical Network Hierarchy

Table 2: AgriFog : System Establishment Cost Estimation

Node/ Device	Count	Cost in
Fog Nodes (WiFi Routers)	2	10000
Surveillance Cameras	4	4000
Soil Profile Nodes	4	4000
Water Controller Node	1	1000
Environment Monitor Node	1	1000
Total		20000

In our case, we assume that the area of our testbed is close to 4 acres and the area has a square boundary.

An acre of land approximately has 4047 square meters of area. The Fog Nodes are wireless and each node can cover an area with a radius which is close to 50m. By optimizing the placement of the fog nodes and the edge nodes as depicted in Figure[2], we can reduce the number of fog nodes required for the infrastructure. Under each Fog node, there are 5 edge devices among which 2 are surveillance cameras and 2 are soil probes. The last node is either an environment monitor or motor controller. If the node is close to the motor, then it is a motor controller. In our model, we assume that the farm is a square shaped one. Therefore, this basic prototype of a farm management system can be established for an area of 4 acres with the following configuration of nodes and costs as tabulated in Table[2].

V. SIMULATION MODELLING

The application was modeled to evaluate the performance and resources consumed. Firstly to simulate an application in iFogSim, we need an overview of the architecture adhered by the farm management system. An exhaustive list of devices along with their features is made. This list helps us to specify the physical nodes that need to be created along with their corresponding capacities and configurations. The nodes include devices like sensors, actuators, gateways and cloud virtual machines. The topology of our farm management system architecture is created using the GUI which is a module in the iFogSim tool.

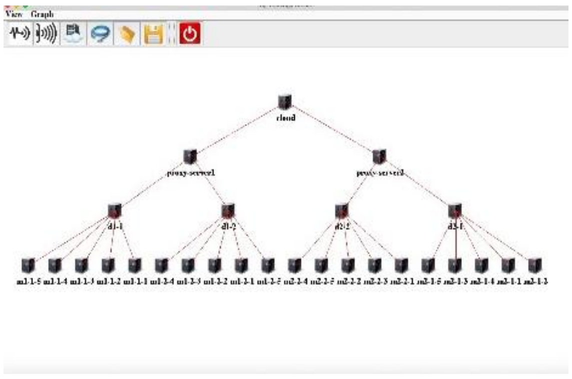

Figure 3: Topology in iFogSim GUI

Firstly to simulate an application in iFogSim, we need an overview of the architecture adhered by the farm management

system. An exhaustive list of devices along with their features is made. This list helps us to specify the physical nodes that need to be created along with their corresponding capacities and configurations. The nodes include devices like sensors, actuators, gateways and cloud virtual machines. The topology of our farm management system architecture is created using the GUI which is a module in the iFogSim tool. The Figure [3] shows an illustration screen of the Agrifog topology created using the GUI tool. The key parameters while creating and modeling the resources consumed by the nodes are tuple transmit rates, CPU and RAM.

In our model of farm management system, we have considered that every acre of land is called a region. A farm can have many regions and each region has 5 edge devices which support inputs from sensors and send signals to actuators. The region has a network gateway which is connected to the ISP (Internet Service Provider) gateway. The ISP gateway is further connected to the cloud. The link latency for a link connecting any two nodes is specified for latency computations for various operations.

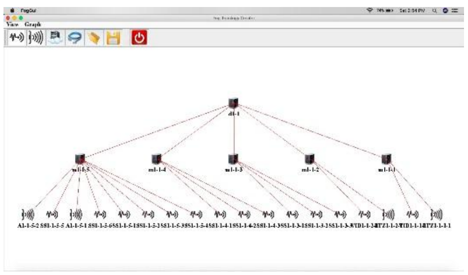

Figure 4: Extended Topology in iFogSim GUI

Five edge devices that constitute a region include two surveillance and monitoring cameras, two soil profile monitors and one environment monitor. The video cameras have a camera sensor and a PTZ(Pan Tilt Zoom) actuator. The soil profile monitors have streaming only type of sensors for attributes like soil moisture, soil temperature and soil NPK (Nitrogen Phosphorus Potassium). The environment monitoring module has sensing devices like rainfall, water level and GPS along with stream- sensing devices including PH, air quality, humidity and temperature. The actuator with this module is a motor switch controller. All these devices are included in the AgriFog topology considering a generic example with 2 areas with 2 regions in each area. The topology from the point of view of each region is visualized and created in Figure [4] using the GUI in iFogSim. After creating the topology containing the attributes and links of every node, the subsequent task is to construct the application using three classes of iFogSim namely AppModule, AppEdge and AppLoop. The Agri-Fog application is modeled as a DAG (directed acyclic graph) as illustrated in figure[5] for mapping various modules to three classes.

Instances of AppModule class represent processing elements of fog applications and realize the vertices of the DAG in DDF model. AppModule is implemented by extending the class PowerVm in CloudSim. For each incoming tuple, an AppModule instance processes it and generates output tuples that are sent to next modules in the DAG. The number of output tuples per input tuple is decided using a selectivity model which can be based on a fractional selectivity or a busty model.

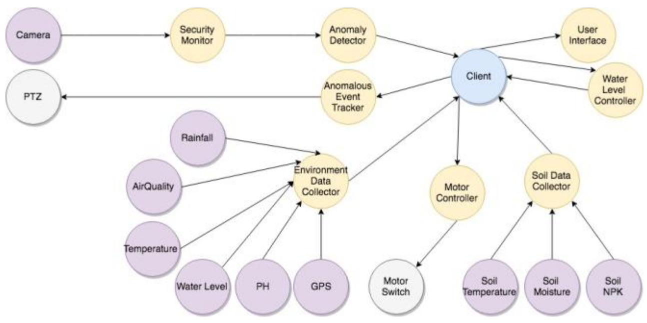

Figure 5: Application Model Graph

The various modules are modeled using the AppModule class while the edges are created using the AppEdge class. The control loop is created using AppLoop class for monitoring resource management and latency.

The functionalities of the mentioned modules are as following:

- *Security Monitor*: The security monitor is a module that records the sequence of events, processes the video data by extracting the keyframes. This video data after initial processing is sent to the cloud for periodic processing.
- *Anomaly Detector*: This module is an extension to the Security Monitor. The Anomalous activities are detected in real-time from the live stream of video data and triggered for notification and tracking of the anomalous event.

• *Anomalous Event Tracker*: Upon detecting an anomalous event, the anomaly detector module triggers the tracking of the aberrant event. This module tracks the anomalous events by its motion and direction.

• *Client*: The client module is a fog node module which connects all the other edge modules to the cloud.

• *Environment Data Collector*: The environment data collection module gathers the environment details like ph, air quality, location, temperature, humidity, pressure and rainfall from the corresponding sensors.

• *Water Level Monitor*: To automate the motor pump functioning, it is essential that we know the level of water. This module senses the water level and notifies if the water level becomes low and triggers the motor pump turn off module.

• *Soil Data Collector*: To monitor the deficiency in water and minerals, this module collects the data from soil temperature, soil moisture, potassium, phosphorus and nitrogen sensors and shares it to the client for analysis.

• *User Interface*: The client assimilates data and does preliminary processing before the complete transmission to the cloud. This user interface module avails the data from the client as well as the historical data that is stored in the cloud for querying and visualization.

• *Motor Controller*: The motor pump needs to be turned on and off based on the water levels at the source. When the water level in the water source is low the water level monitor module signals the motor controller module to turn off the motor pump.

The camera, rainfall, air quality, temperature, water level, ph, GPS, soil temperature, soil moisture and soil npk are modeled as sensors in the application. PTZ and Motor Switch are included in the system as actuators.

VI. RESULTS

In this section, the Fog computing and Cloud computing environments were simulated to study the application and evaluate the efficiency of cloud-only and cloud-fog placement strategies. Network usage, average latency of control loop and tuple CPU execution delay are the major parameters considered in the results for our comparative study. The AgriFog application was modeled as a class named AgriFogSim.java in iFogSim. The sensors and actuators could be added to the application according to the System Model and Architecture described in the previous sections. The application is geographically distributed into locations and each location is further split into regions. The location in our context is synonymous to a Farm and the Region is analogous to a unit area of 4 acres. To comprehend the efficiency of the system from multiple perspectives, the experimentation has been done on various configurations of regions and farms. Different configurations are listed in Table[3]. The time taken for cloud simulation of the application was very less compared to the amount of time consumed for each simulation in the Fog configuration. The outcomes of the simulation demonstrate how different workloads impact the various parameters taken for consideration in our comparative study.

Table 3: Configuration details

	Configuration 1	Configuration 2	Configuration 3	Configuration 4
Farms	1	1	2	2
Regions per Farm	1	2	1	2

In order to test the feasibility of our application architecture and system model, the number of regions in each farm were varied keeping the inherent devices in each region constant. It has been observed that the performance of the application is dependent on the latencies of the connecting links between the fog devices. The energy consumed for running the application depends on the configuration of the devices running the application along with the amount of data generated and the processing requirements.

1. Network usage

The network usage of the AgriFog application in the various configurations of the Clod based and Fog based topologies is depicted in Figure [7]. The rise in the number of devices connected within the application significantly increase the load on the network in both the cases of Fog and Cloud models. As there is a growth in scale, the network consumption also increases but at a slower pace in fog based models than in the cloud-based models. This result can be expounded as the efficient scalability of fog-based applications. The rapid growth in network use in the case of cloud-based execution can impede the performance of the application by giving rise to network congestions. In order to avoid such situations, the fog-based execution models are chosen because the data is preprocessed at the source and there is no requirement to transmit the data very frequently. The decision making modules when present in the fog devices can make the real time applications faster with lower latency and reduced response time.

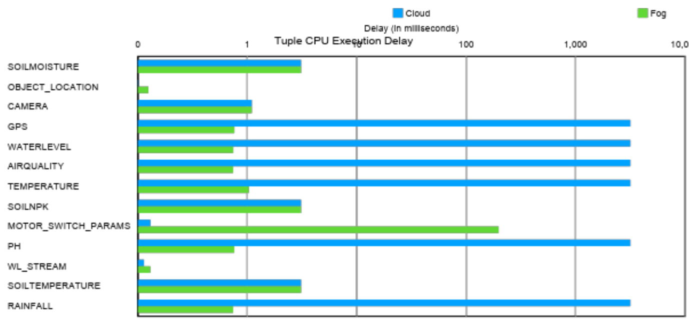

Figure 6: Sensor Tuple Execution Delay Plot

2. Tuple CPU Execution Delay

The data is carried in tuples within the connection link. The transmission of each tuple involves a latency.

This latency can in turn cause the delay in CPU executions which require the tuple. The delay directly depends upon the tuple length, link latency and number of links that are connected to reach the CPU for the execution. The fog based deployment model promotes the preprocessing of the data at the source or the nearest fog node before being sent to the cloud for the further processing. The preprocessing eliminates redundancies and makes the data rich enough to capture the scenario with minimum description. The reduction in delay is well observed in the cases of the fog devices under observation as depicted in Figure [6]. The fog topology mitigates the scope for congestion in the network and thus reduces the negative impacts on the response time due to the increase in the size of the topology due to the addition of more devices. It is observed that the cloud-based deployment brings about large delays making the application prone to higher response times. The high response time makes the application unfit for real-time scenarios. Therefore, for real-time or near real-time very low delays are expected even when the system is scaled high which is the motivation to employ the Fog Based Architectures.

3. Average latency of control loop

The significant delays in the application can be due to the high latency between the links which are a part of the control loop. Thus the control loops latency can be directly mapped to the application's latency. Reduction in the latency in the control loop can help in making the application more responsive. For the analysis of the average latency of control loop in AgriFog, the loop between the camera, client and PTZ is taken into consideration. The simulation results yielded similar values of latency for all the configurations for each type of topology.

The latency was about 1.1 milliseconds in the fog based model as compared to 105.18 milliseconds in the cloud-based model. From these values, it can be clearly inferred that the fog based model minimizes the response time. Another extrapolation to this conclusion is that the increase in scale which means larger network links and higher data generation rates, does not affect the response time of the application when the fog based architecture is adopted. Based upon the comparative study of the fog based and edge-based deployments in various configurations, it can be discerned that the cloud-based topology deteriorates with increase in the application scale whereas the fog based topology is able to handle the scalability issues efficiently by maintaining very low latency and response times. Also, the network utilization is reduced considerably in the fog based results due to the control on network congestion.

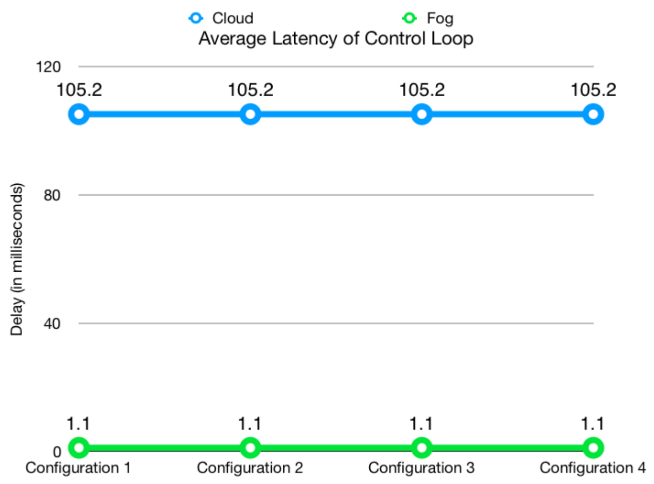

Figure 7: Total Network Usage Plot

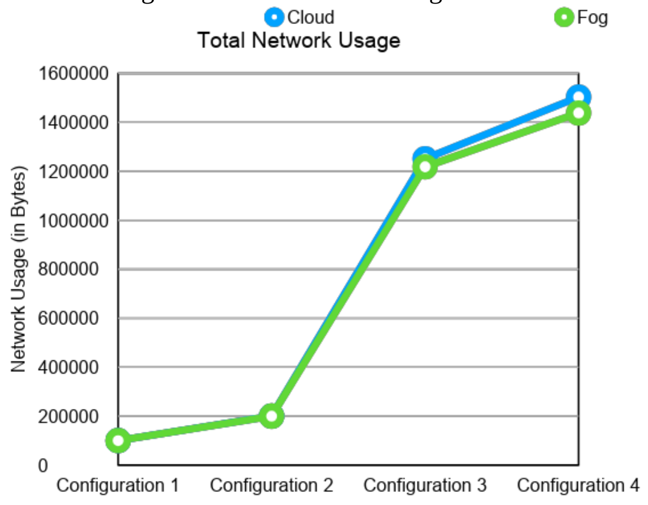

Figure 8: Average Latency of Control Loop

VII. CONCLUSION

The key contributions of our work include the following. Firstly, we try to delineate an efficient overall architecture of a Farm Management System through inspirations of designs from many of the previous state of art models. Secondly, we adopt a fog-based paradigm for the realization of the FMS and propose AgriFog which is a cost-efficient fog based architecture of a Farm Management System. Then, the AgriFog application is designed and simulated in iFogSim to perform a comparative study on the performance of the Cloud based and Fog Based implementations of the application in various configurations. The outcomes of the simulations are assimilated to discern the performance highlights of the fog based deployment over the cloud-based deployment. The results show that the latency for a sensor to actuator loop is reduced significantly while the Tuple Execution Delay is lesser in a Fog Based System. The network consumption in a fog based architecture is lower than the cloud based architecture. The scalability is a key issue in which the latency and response time are maintained in a fog-based architecture.

Since the application is aimed at serving geographically distributed clients with low latency and realtime responses, it is ideal to select a fog-based architecture for the AgriFog application.

VIII. FUTURE WORK

Similar to Mobile crowd sensing [20] where a fog-based data analytics scheme is introduced with cost-efficient resource provisioning for IoT crowd sensing applications, a data analytics module can be introduced within the fog devices to enable local real-time analytics more efficiently. A middleware architecture for the fog devices to enable uniform interfacing is also a possible extension to the Fog based IoT systems. This will enable the diverse range of IoT devices to be included in the applications in a much simpler manner. The various policies and load balancing methods could be deployed within the middleware for enhancement in the performance of our application.

ACKNOWLEDGMENT

We would like to convey our heartfelt regards to the Department of Computer Science and Engineering, Amrita University for providing us with the required motivation and resources to accomplish the research activities in a fruitful manner.

REFERENCES

1. H. F. Hectare, Hands free hectare - home. URL http://www.handsfreehectare.com
2. Farmbot, The farmbot project - home. URL http://wiki.farmbot.org/thefarmbot-project
3. F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role In the internet of things, in: Proceedings of the first edition of the MCC workshop on Mobile cloud computing, ACM, 2012, pp. 13-16.
4. A. Kaloxylos, R. Eigenmann, F. Teye, Z. Politopoulou, S. Wolfert, C. Shrank, M. Dillinger, I. Lam-propoulou, E. Antoniou, L. Pesonen, et al., Farm management systems and the future internet era, Computers and electronics in agriculture 89 (2012) 130{144.
5. A. Kaloxylos, A. Groumas, V. Sarris, L. Katsikas, P. Magdalinos, E. Antoniou, Z. Politopoulou, S. Wolfert, C. Brewster, R. Eigenmann, et al., A cloud-based farm management system: Architec-ture and implementation, Computers and Electronics in Agriculture 100 (2014) 168{179.
6. T. Popovic, N. Latinovic, A. Pesic, Z. Zecevic, B. Krstajic, S. Djukanovic, Architecting an iot-enabled platform for precision agriculture and ecological monitoring: A case study, Computers and Electronics in Agriculture 140 (2017) 255{265.
7. T. Ojha, S. Misra, N. S. Raghuwanshi, Sensing-cloud: Leveraging the bene ts for agricultural applica-tions, Computers and electronics in agriculture 135 (2017) 96{107.
8. K. Patil, N. Kale, A model for smart agriculture using iot, in: Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), 2016 International Conference on, IEEE, 2016, pp. 543{545.
9. J. Das, G. Cross, C. Qu, A. Makineni, P. Tokekar, Y. Mulgaonkar, V. Kumar, Devices, systems, and methods for automated monitoring enabling precision agriculture, in: Automation Science and Engineering (CASE), 2015 IEEE International Conference on, IEEE, 2015, pp. 462{469.
10. I. Mohanraj, K. Ashokumar, J. Naren, Field monitoring and automation using iot in agriculture domain, Procedia Computer Science 93 (2016) 931{939.
11. M. S. Mekala, P. Viswanathan, A survey: Smart agriculture iot with cloud computing, in: Microelec-tronic Devices, Circuits and Systems (ICMDCS), 2017 International conference on, IEEE, 2017, pp. 1{7.
12. M. Chiang, S. Ha, I. Chih-Lin, F. Risso, T. Zhang, Clarifying fog computing and networking: 10 questions and answers, IEEE Communications Magazine 55 (4) (2017) 18{20.
13. F. Bonomi, R. Milito, P. Natarajan, J. Zhu, Fog computing: A platform for internet of things and analytics, in: Big data and internet of things: A roadmap for smart environments, Springer, 2014, pp. 169{186.
14. A. V. Dastjerdi, R. Buyya, Fog computing: Helping the internet of things realize its potential, Computer 49 (8) (2016) 112{116.
15. M. Chiang, T. Zhang, Fog and iot: An overview of research opportunities, IEEE Internet of Things Journal 3 (6) (2016) 854{864.
16. K. Hong, D. Lillethun, U. Ramachandran, B. Ottenw•alder, B. Koldehofe, Mobile fog: A program-ming model for large-scale applications on the internet of things, in: Proceedings of the second ACM SIGCOMM workshop on Mobile cloud computing, ACM, 2013, pp. 15{20.
17. Z. Hao, E. Novak, S. Yi, Q. Li, Challenges and software architecture for fog computing, IEEE Internet Computing 21 (2) (2017) 44{53.
18. P. Prakash, K. Darshaun, M. V. Ganesh, B. Vasudha, et al., Fog computing: Issues, challenges and future directions, International Journal of Electrical and Computer Engineering (IJECE) 7 (6) (2017) 3669{3673.
19. H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, R. Buyya, ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments, Software: Practice and Experience 47 (9) (2017) 1275{1296.
20. Perera, C., Jayaraman, P.P., Zaslavsky, A., Christen, P. and Georgakopoulos, D., 2014, January. Mosden: An internet of things middleware for resource constrained mobile devices. In 2014 47th Hawaii International Conference on System Sciences (pp. 1053-1062).
21. Perera, Charith, et al. "Mosden: An internet of things middleware for resource constrained mobile devices." *2014 47th Hawaii International Conference on System Sciences*. IEEE, 2014.

AUTHORS PROFILE

Sucharitha V is pursuing her final semester of Masters in Technology in Computer Science and Engineering at Department of Computer Science & Engineering, School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore. She is a Technology Enthusiast eager to learn, explore and teach. She has handled roles including Software Developer, Product Design Intern and Product Architect in corporate as well as being a Research Associate and Junior Research Fellow in academic institutions for 3+ years. Visualization of data intrigues her the most while her other areas of interest include Machine Learning, Big Data, Cloud/Fog Computing and Sustainable Technologies.

Prakash P currently serves as Assistant Professor at the department of Computer Science and Engineering, Amrita School of Engineering, Coimbatore Campus. His areas of research include Cloud Computing, Automata Theory and Analysis of Algorithms. Dr. Prakash's broad areas of research interest are Cloud computing and Big data analytics. More specifically, his work involves in developing the algorithms or frameworks for the energy efficient in high performance computing. He is also exploring the integration and data analysis of Internet of Things (IoT) with cloud computing.

Ganesh Neelakanta Iyer currently is an Associate Professor in the Department of Computer Science & Engineering, School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore. He brings in a decade of industry experience in various companies including Sasken Communication Technologies and NXP semiconductors. He has handled several roles in the software industry including QA Architect, Technical Support Manager, Engineering development and Technology Evangelist. His technical knowledge and experience are in various areas including Cloud/Edge/Fog Computing, Computer Networks, Software Engineering practices and Quality Analysis, Economic models and current day practices on cloud-based enterprise architectures, Internet of Things (IoT) based systems, Machine Learning and technology for traditional Indian dance popularization.