Scientia Iranica D (2023) 30(5), 1714–1730

Sharif University of Technology
Scientia Iranica
Transactions D: Computer Science & Engineering and Electrical Engineering
http://scientiairanica.sharif.edu

Improving the performance of heterogeneous IoT networks through multi-stage and parallel computing systems

K. Venkata Sowmya and J. Kodanda Rama Sastry*

Department of Electronics and Computer Science, Koneru Lakshmaiah Education Foundation University, Vaddeswaram, Guntur District, Andhra Pradesh, India 522502.

Received 30 March 2021; received in revised form 26 October 2021; accepted 18 April 2022

KEYWORDS
Device level clustering;
Layered networking;
Parallel architectures;
Performance optimization;
Topology binding.

Abstract. Multiple networking layers in the IoT network are in a state of heterogeneity that must be addressed to facilitate proper communication to prevent any degradation in the performance of an IoT network. The performance of the IoT networks depends on the networking topologies used in different layers, clustering algorithms, protocols used for handling heterogeneity, communication speeds, and data packet sizes. In this research, the performance of the IoT networks is improved by adding device clustering through a multi-stage network and an efficient SOJK clustering algorithm in the device layer which consequently minimizes power depletion and enhances the quantity of both data and data packets transmitted at zero power. A mechanism to handle heterogeneity resulting from Wi-Fi, CDMA, and USB communication protocols is also presented while optimizing the communication speeds and data transmission size. According to the findings, at the cellular speed of 170 Mbps and data size of 468 bytes, the optimum response time of an IoT network will be obtained. The time taken to transport information from the device to the storage layer is reduced by 57% compared to the time taken to transport the data using the prototype network.

1. Introduction

1.1. Terminology

Cluster: A cluster is a set of sensors/actuators connected in a network using a topology, the outputs of which are connected to a set of cluster heads.

Cluster head: A cluster head is a communicating device with a base station.

Sensor: A sensor is a device that senses an operating physical phenomenon such as temperature, light, etc.

Switch box: A switch box is a device that receives and transmits the signals in different paths.

Base station: A base station is a communicating device for communication using mobile communication protocols such as CDMA. It carries remote communication.

Controller: A microcontroller is a microcomputer that receives and stores the sensing data, triggers a control action required, and forwards the data to the service server when needed.

Services server: The server is a computing unit which services the requests received from the users

*. *Corresponding author.*
E-mail addresses: sowmyakhambhampati@kluniversity.in (K. Venkata Sowmya); drsastry@kluniversity.in (J. Kodanda Rama Sastry)

doi: 10.24200/sci.2022.58006.5514

or the controller for effecting local processing and transmission of the data/information.

Gateway: Gateway is a device that connects the server of the service to the cloud through the Internet.

Cloud: The clous is an infrastructure that provides services demanded by the users.

The objectives of the present research are briefly listed in the following:

1. Determination of the networking topology that connects the device layer of an IoT network to the controller layer through a base station to achieve the parallel processing and availability of the alternate paths to affect the communication;
2. Invention of a clustering algorithm in the device layer that decreases power depletion rate, enhances the life span of the devices, and determines the cluster heads and alternate paths for communication;
3. Calculation of the optimum Wi-Fi and cellular communication speeds to achieve the highest performance level at the cluster head level that addresses heterogeneity;
4. To invent a method for computing the performance of an IoT network as the parameters of the network change.

The performance of the IoT networks is an essential factor to calculate given that many networking layers must be incorporated and that networking must be done using many heterogeneous devices. In this respect, several criteria should be taken into account when assessing and improving the performance of the IoT networks [1]. Keeping in view the minimum power dissipation, we can claim that the multiplicity of the alternate paths for communication is one of the most important criteria.

Many layers exist in the IoT networks that are created as a result of the interconnection between small devices that can communicate with low bandwidth and high devices that can also communicate with high bandwidths. Performance optimization needs to be done considering the network topology used in each layer during the interconnections between the networks in different layers [2].

Performance computing in each layer and performance logging in a remote server are equally important. It should be noted that determining a specific topology in each layer, considering different types of things in each layer, and achieving the most appropriate interconnection between different networks are also quite challenging. That the main focus is put only on the mere power dissipation in order to extend the service life of all devices in a network is insufficient [3].

The existence of heterogeneity is one of other critical issues to be considered to keep the time delays caused by frequent protocol conversions at its minimum value. The problems here are how to enhance the performance of the IoT networks with significant heterogeneity and explore the different layers in an IoT network by choosing appropriate networking topologies and internetworking the networks contained in different layers of an IoT network.

The overall performance of the network is computed as shown in Eq. (1):

$$P_{IoT} = P_{dl} + P_{cl} + P_{sl} + P_{gl} + P_{cll}, \tag{1}$$

where P_{IoT} is the performance of an IoT network; P_{dl} the performance of an IoT network at the device level; P_{cl} the performance of an IoT network at the controller level; P_{sl} the performance of an IoT network at the services level; P_{gl} the performance of an IoT network at the gateway level; P_{cll} the performance of an IoT network at the cloud level.

The performance of each layer in an IoT network needs to be analyzed independently and in conjunction with its superseding and underfeeding layers.

The performance of an IoT network can be improved in each layer by using different networking topologies and software to make communication intelligent.

As the first layer, the device layer must be prioritized to obtain the solutions to enhancing its performance. Some of the implemented techniques are described in [4]. Fixing the communication speeds, keeping in view both transmission and reception, and considering the protocols used for interfacing with other devices are the other critical issues that must be addressed.

Failure of the communication channels is most frequently observed in the IoT networks; therefore, selecting an appropriate networking topology can facilitate the availability of many alternate channels for effecting fail-free communication.

In case a device is supposed to support heterogeneous protocols, additional processes should be run which in turn would lead to delays in data transmission. In addition, latency is another undesired outcome caused by the varying speeds supported by the heterogeneous protocols within the device. How to determine specific speeds and data packet size is also another critical issue that must be discussed [5].

This study discusses the applications of networking topologies to enable the availability of many alternate paths for communication. In addition, it highlights the necessity of fixing the communication speeds and data sizes that leads to the excellent IoT performance within the cluster heads that handle heterogeneous communication using Wi-Fi and CDMA communication protocols. The latency resulting from the variations in speeds must also be reduced by zero by choosing appropriate communication speeds and data sizes.

Performance is a bottleneck when it comes to different layers of the IoT network. It was proved in our earlier work that introduction of a crossbar networking and use of a separate device for affecting communication with the base station would improve the performance of an IoT network. Similarly, it is highly possible to improve the performance of each layer by using different networking topologies. However, the problem here is to integrate different networking topologies used in different layers in order to improve the overall performance of an IoT network.

The most important issue here is how to integrate a topology to elevate the number of transmission paths between the base stations and controllers in the control layer interconnected with a multi-stage network implemented in the device layer. The integration approach must prioritize increasing the number of alternate paths that help improve the data transmission speeds of the network.

The other main problem is to reduce the latency caused by the protocol conversions (Handling Heterogeneity) in different devices. Determination of the communication speeds and data sizes to be transmitted is the most important problem that must be taken into consideration.

In addition, identification of a clustering algorithm that incorporates the power dissipation, elongation of service life, and reduction of latency caused by variable communication speeds and data transmission sizes considering different protocols is the most challenging problem.

2. Literature survey

Literature review in this very field was done to explore the contributions made to find the suitable networking topologies, clustering algorithms used in the device layer, and heterogeneity handling methods considering different communication protocols.

Sowmya and Sastry [6] posed several issues be taken into account to improve the performance of the IoT networks.

Several authors presented some algorithms to select a device to act as a cluster head to communicate the sensed data and receive the data so as to affect the actuating function. They mainly focused on the minimization of power dissipation. Some of the contributions to the development of the LEACH-C protocols were made by some researchers [7–21]. Puschmann, et al. [22] suggested how to determine the optimum number of the clusters to reduce the gadgets needed for data transmission.

Some contributions that focus on how data storage and is ensured during the transmission from the devices to the cloud are the studies carried out by Tao and Ji [23] and Kwon and Park [24].

Geethika Reddy et al. [25] and Sai Rama Krishna and Sastry [26] considered different networking topologies to improve the fault tolerance of the IoT networks. Rajasekhar and Sastry [27,28] discussed how the networking topology could establish hybrid embedded comprehensive networks. Sastry et al. [29] and Sasi Bhanu et al. [30] investigated the application of networking topologies to enhance the performance of the IoT networks considering the services and gateway layers.

Vishnu Priya and Sastry [31] presented SDN, keeping network management based on the data flow from the devices.

Pavithra and Sastry [32] and Sastry et al. [33] studied the challenges of handling the heterogeneity in different layers of IoT networks in the devices and determining the type of such challenges that crop when communication should be established.

3. Research GAP

To the best of the authors' knowledge, no studies have been conducted on determining the suitable networking topologies implemented in the device and controller layers that provide several communication paths and reduce the response time. At the same time, some authors suggested use of crossbar and butterfly networks for determining the fault tolerance of the IoT networks. To our knowledge, use of communication delays as well as non-participating cluster heads, especially for carrying communication, and handling heterogeneity caused by some protocols like Wi-Fi and CDMA that facilitate communication between the device layer and base station, en route to the controller layer have not been addressed yet. The OFDM protocol for communicating with the base station has not been used so far due to the distance limitations. Base stations are situated at distances much farther than those supported by the OFDM. Similarly, to date, Narrow Band IoT(NB-IoT)-based protocol has not been explored since the devices in the device layer are expected to be dispersed geographically, thus posing a distance limitation to an effective communication through switches.

Computing the response time of IoT networks: Every IoT network comprises many layers and in each layer, several devices are used and networked for better realization of its specific function. All the devices interact in each layer and across the layers. The time consumed by every device and within every layer for sensing, processing, protocol conversion, receiving, and transmitting must be computed and added to calculate the overall response time. Computing the response time is considered a real challenge to improving the performance of an IoT network.

A process is added into every layer to log the data

Table 1. Time components that affect the performance of an IoT network.

Serial no.	Time component symbol	Time component description
1	TRL_i	Time taken to receive the data in the ith layer
2	$TUPL_i$	Time taken to unpack and pack data in the ith layer
3	TTL_i	Time taken to transmit the data in the ith layer
4	$TTLL_i$	Time taken to log the data
5	$TSUM_i$	Total time taken to process the data $\sum_{i=1}^{N} TRLi + TUPLi + TTLi + TTLLi$

while receiving, unpacking, packing, and transmitting the data. Table 1 shows the time components that must be considered in each of these layers. The summation of the time taken in each layer is the total time demanded by an IoT network to complete the transactions. The transactions start from a low-end device to data storage in the cloud or the time taken to receive the message from the end-user to the device where the user's request is processed.

4. Prototype network for experimentation

Figure 1 presents a typical IoT network developed for the experiments. The IoT network was built considering all the typical and comprehensive IoT network layers including the device, controller, services, gateway, and computing layers. The devices in the device layer are connected as a cluster. The clusters are connected to the controller en route to the base station. The controller is then connected to the services layer connected to the cloud via the Internet.

The gateway is developed using several devices and servers. All the devices and servers are connected according to the networking diagram shown in Figure 1. Further details of the devices used for the development of the IoT network are given in Table 2.

This model is used as an experimental model for implementing the changes made and showing how the performance of the IoT network could be improved. Figure 1 lists different techniques used within the IoT network including sensing, controlling, service rendering, and heterogeneous communication among

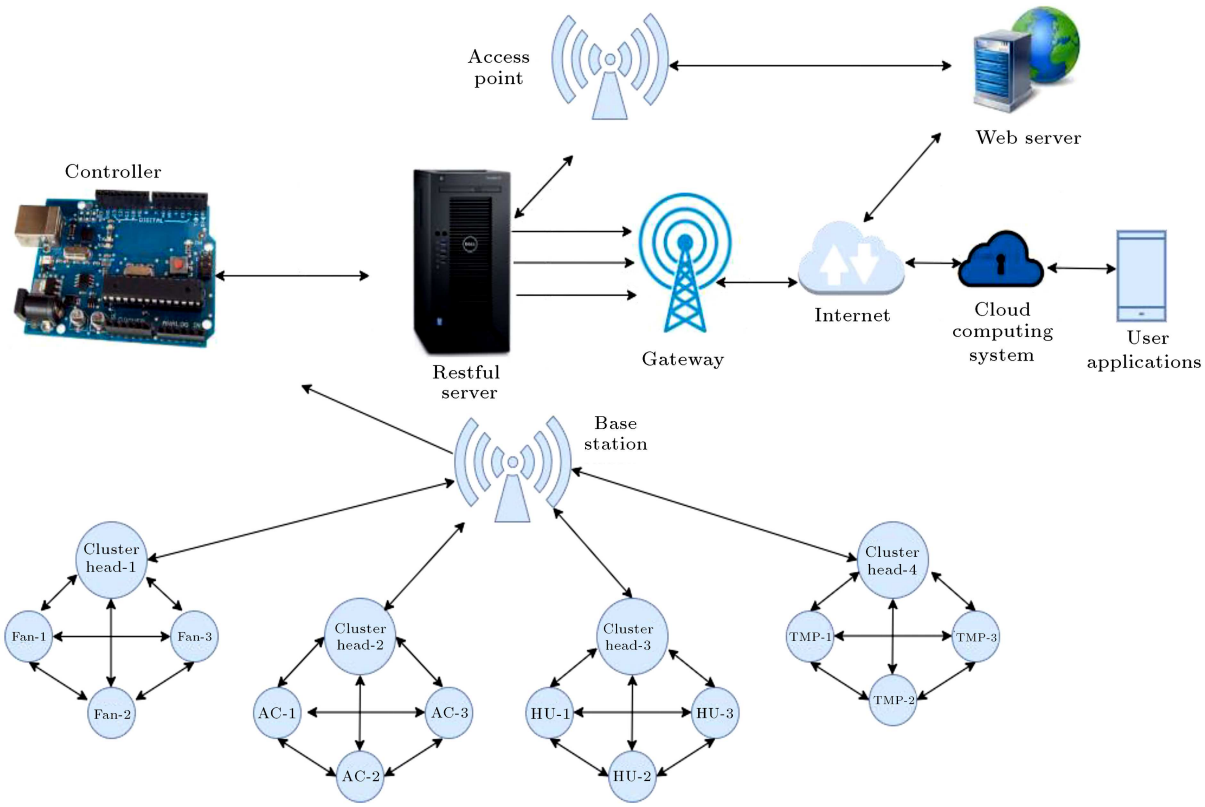

Figure 1. Typical prototype network.

Table 2. Devices and servers used for the development of the prototype IoT network.

Serial no.	Device/server	Model number
1	Sensors and actuators	C-Mote, Centre for Development of Advanced Computing (CDAC) connected with temperature, Humidity, Wi-Fi, and relays
2	Cluster-heads	C-Mote CDAC
3	Base stations	LTE-A PRO/LTE-A/LTE
4	Controllers	Arduino and Raspberry PI
5	RESTful server	HP-ProLiant-Django
6	WEB server	HP-ProLiant-JavaBeans Open Source Software (JBOSS)
7	Access point	CISCO Wi-Fi (802-11ax) catalyst 9105
8	Gateway	CDAC Wireless Network Development Kits (WSNDK)
9	Cloud computing system	OpenStack, SQL server

Table 3. Performance improving methods.

Serial no.	Performance improvements	Purpose
1	Introducing the multi-stage network (crossbar topology) in the device layer	To increase the number of paths available for communication
2	Implementing an efficient algorithm that considers power dissipation, prolonging the service life of the devices at the cluster level where protocol conversion occurs	To relieve the actual devices from the burden of communicating with the base stations and increase the service life of the devices
3	Determining the optimum speed and data size to communicate using Wi-Fi and CDMA protocols to communicate between the device layer and base station	To reduce the latency time and handle heterogeneity
4	Introducing more base stations such that there are at least two base stations to communicate from each of the communicating devices	To make available at least two alternate paths for communication
5	Introducing more controllers to handle more data traffic	To implement parallel communication for data services
6	Merging data from four controllers into a single conduit using the multi-USB system to channel the output to the server of the restful service	For piping the data into the restful server

different devices and layers in an IoT network, storing and retrieving the data from databases for building an IoT network.

5. Materials and techniques

5.1. Performance improvement mechanisms

Table 3 shows the performance improvement mechanisms proposed to improve the performance of the prototype IoT network.

Among the important design issues are how to establish an interplay among the communication systems, reduce the protocol conversion and latency time, fix the data size so as to reduce the total performance time, and provide the alternate paths for communication.

5.2. Revised IoT network-catering to performance improvements

The revised IoT network is shown in Figure 2. In the revised network, a crossbar network is introduced to the device layer to connect the inputs to the base station

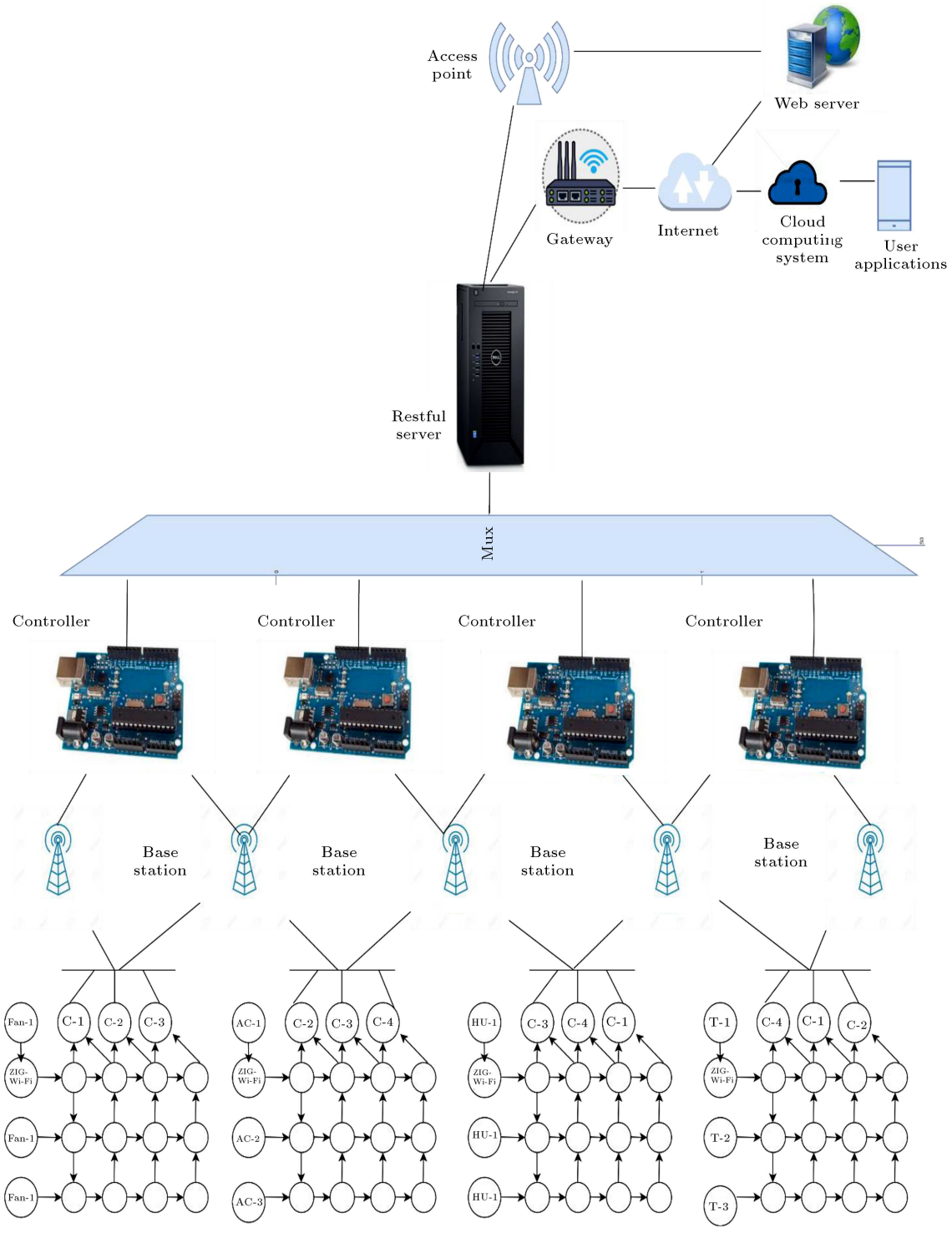

Figure 2. Modified IoT network.

through independent cluster heads that do not serve any purpose of sensing or actuating. Dual base stations are used, and each one is connected to all the cluster heads providing dual paths for communication that cater to failure conditions and accelerate the communication speed. To the controller layer, four controllers are added, each communicating with two base stations, thus catering for dual paths for communication. The controller failure rates are high, necessitating more controllers. The controllers' output is piped to the service server via a USB hub. The rest of the network structure is the same as that built into the prototype.

Establishing a crossbar-based network connecting the remotely situated devices or actuators is, to some extent, complicated. To be specific, the Wi-Fi-based interfaces must be properly configured, and the con-

nections should be established properly. The devices used to establish the revised IoT network are listed in Table 2. In addition, the HP switches are used for establishing the clusters.

5.3. *Performance improvement through the availability of alternative paths for communications*

The IoT devices are fragile with quite high failure rates. There should be alternative paths for communication to transmit the data from a device to a destination point. Since the devices fail to cater to such a situation, the redundant devices should be introduced into an IoT-based system.

Networking of the devices in a cluster is the key to improving the performance of every cluster. The network used for connecting the devices in a cluster can be established using many topologies such as the crossbar, butterfly, multi-stage, etc. Multi-cluster-based communication helps improve the response time as the number of paths existing in the network increases.

According to the revised IoT diagram in the device layer, a crossbar network is used for connecting the inputs from the devices to the cluster heads. Several communication paths are established through one of the cluster heads to facilitate communication with the base station. The network topology is shown in Figure 3. Each of the cluster heads communicates with two different microcontrollers via any of the two base stations.

A crossbar network topology deals with N inputs (fault rates of the incoming devices) and M outputs (fault rates of the outgoing devices). One switch box is associated with each input and output, an additional hardware element is added to the network. The box, as such, is called the ij box. The switch box in the row i and column j connects the network input on the row i to the network output on the row j.

Each switch box forwards the data from its left link to the right link, which means propagating the data horizontally and forwarding the data flow through its bottom link to its top link. The switch box also is capable of moving data from its left link to the top link.

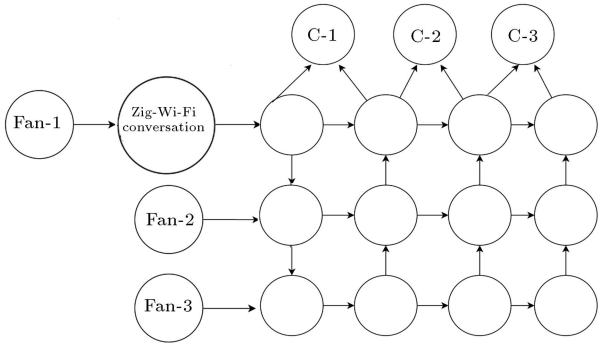

Figure 3. Crossbar-based networking topology.

Every link at most can carry only one data element, and each switch box will be able to process two data elements simultaneously. A switch box can forward data from its left link to its right link while forwarding it from its bottom link to the top link.

The routing strategy is rather obvious. For example, if we want to send a message from input 3 to output 5, we should follow the given procedure. The input will first arrive at the switch box (3,1), which will forward it to (3,2), and so on until it reaches the switch box (3,5). This switch box will turn the message into column 5 and forward it to the box (2,5), which will send it to the box (1,5), which will consequently send it to its destination. If input 3 is to be sent to output 5, the data will be received by the switch box (3,1), which will forward it to (3,2) and so on until it reaches (3,5), which will then forward it to (2,5) and then to (1,5) and ultimately to the connected device.

Any input-output combination in this network can be realized without collision at the output (No two inputs compete for the same output line). This network is thus quite suitable when the process runs faster and involves transmission, especially the data transmission from a sensor. The connectivity of the crossbar is analyzed to assess the failure rates of the individual components.

q_l is the probability that a link is faulty; p_l is the 1-ql = probability that a link and the switch box is not faulty.

Counting from 1, for input i to be connectable to output j, we must go through a total of $i + j$ links. The probability where all of them are fault free is P_l^{i+j}. The probability where a network is fault free can be computed using Eq. (2):

$$Q = \sum_{i=1}^{N} \sum_{j=1}^{M} P_l^{i+j} = {P_l}^2 \frac{1 - {P_l}^N}{1 - P_l} \frac{1 - {P_l}^M}{1 - P_l}. \tag{2}$$

The reliability of the device-based clusters proved to be tremendously enhanced by creating networks like crossbar, butterfly, etc. The number of the communication paths from any node can be computed using Eq. (2) where n = Number of inputs and if $n = 3$, then the number of paths available for communication from any node equals 31. The number of the communication paths increases with an increase in the number of inputs and network topologies like crossbar.

For input 1, a total of 16 paths are traverse, i.e., six paths in row 1, five paths in row 2, and five paths in row 3. For input 2, a total of 10 paths are traversed, i.e., five paths in row 2 and five paths in row 3. For input 3, only 5 paths are traversed. The total number of the traversed paths is 31, as shown in Eq. (3):

$$np = 2^{n+2} - 1, \tag{3}$$

where np is the number of paths.

Table 4. SOJK algorithmic steps.

Step	Type of execution done
1	Deploy the switches
2	Deploy cluster heads
3	Maintain a repository containing the details of the cluster heads (CRP = Cluster Rated Power, CLP= Cluster Leftover Power, CNOP = Number of Packets dispatched, CSDATA = Size of the data dispatched to the base station
4	Maintain connectivity of cluster heads to the topmost switches
5	Deploy the sensors and switches that connect the sensors to the clusters
6	Maintain a repository containing the details of the sensors (SRP=Sensor Rated Power, SLP=Sensor Leftover Power, SNOP=Number of Packets dispatched to a chosen cluster head, SSDATA=Size of the data dispatched to cluster head from the sensor
7	Keep track through a repository of paths from the sensor to the cluster head and find the path with the least power
8	Select the paths in an ascending order of the power needed for data transmission in parallel channels. The number of channels required to transmit the data depends on the size of the data to be sent
9	Break the data into the number of channels selected and form packets as per the protocol supported by the device
10	Transmit the packets through the selected channels
11	Every device updates its status after completion of data transmission
12	Repeat Steps 8-11 every time; then, a need is felt to transmit the data to the remote cloud or when data is to be received from the cloud

5.4. Improving the performance through implementing effective clustering algorithm

Section 2 reviews a number of algorithms as well as the parameters used for selecting the cluster head. Most of these algorithms concentrated on the depreciation of the power consumed for communication through selecting a cluster head. Every device is configured as a cluster head, and the same process is used not only for sensing but also for communicating the data to the cluster head. The sensors thus have a load of not only sensing but also transmitting when selected as the cluster heads. The number of paths that can be used for data routing to a cluster head is fixed. There will be a complete setback when the designated cluster head fails for any reason. To avoid this bottleneck, consideration of the availability of several cluster heads and selection of one of the cluster heads based on its working status help drastically improve the performance of the cluster head. This also frees the sensing devices from the overhead of communication with the base station. There can be several clusters in a network, and all the clusters can share a set of cluster heads, thereby reducing the number of cluster heads required for communication.

As shown in Table 4, an algorithm called SOJK is developed and implemented within each cluster head that stores the information about the rated power, size of the data transmitted, left overpower, several data packets dispatched, etc. The algorithm also maintains the way the devices, switches, and cluster heads are connected to each other. A repository of

sensors is also maintained that stores the data related to the rated power, used power, data transmitted, and transmitted packets. Another repository is also maintained that stores the information about the paths starting from a sensor to a cluster head and each path status. The algorithm selects the least number of used paths, meaning that the combined power consumption is the least. The data to be transmitted from a sensor is then broken into several data segments based on the user and available channels and then, the packets are dispatched. Since many channels are used simultaneously, the response time required to transmit the data will be the least. The complexity of the SOJK algorithm can be computed using Eq. (4).

Complexity calculations of SOJK

NCH	Number of Cluster Heads
NSW	Number of Switches
NSE	Number of Sensors
NPA	Number of Paths
nCH	Number of operations to be carried to maintain cluster repository = NCH
nSW	Number of operations to be carried for maintaining connectivity of cluster heads with topmost switches
nSE	Number of operations to be carried for maintaining and updating the sensors * 2
nPA	Number of operations to be carried to enumerate paths and maintain path repository
nPC	Number of operations to be carried to break a packet into several channels
nTR	Number of operations to be carried for transmitting data into a selected number of channels

$$SOJK_{Complexity} = O(nCH + nSW + nSE + nPA + nPC + nTR). \quad (4)$$

5.5. Handling heterogeneity

In the device layer, all the devices can communicate using Wi-Fi protocol. The devices must communicate over long distances through the base stations as the devices are remotely situated.

The device-level cluster heads that connect the devices using the Wi-Fi protocol must communicate with the base stations using cellular communication. For this reason, a strong interface must be built into the cluster heads to facilitate the interplay between Wi-Fi and cellular communication.

The base stations communicate with the microcontroller, which is responsible for keeping the track of individual devices and triggering actions required for controlling the devices. The controllers, on the one hand, must be able to communicate with the base stations using cellular communication and, on the other hand, must transmit the data digitally to a multi-USB port with very high speed, thus requiring a buffering that matches the speeds of cellular communication and direct digital transmission to be incorporated into the controllers. This is the case when a single server is used for implementing restful services. The connectivity, however, could differ when a greater number of servers are used.

The IoT networks consist of heterogeneous devices in terms of the implemented protocols, nature of the devices, bandwidth requirements, etc. Hence, a growing need is felt for frequent protocol conversion required to establish the data flow within the IoT networks. The communication between the device and base station requires protocol conversion from Wi-Fi to cellular slow- to high-speed conversion. Similarly, the conversion from the cellular to USB affected by the controllers also requires protocol conversion.

In the cluster heads, the protocol conversion from the Wi-Fi to cellular is required to communicate with the base stations. The conversion should be done in such a way that the latency time is minimized.

Individual devices communicate with the cluster head using the Wi-Fi protocol, and the cluster head converts the Wi-Fi packet to a Cellular packet. Cellular communication speeds are much higher than the Wi-Fi speeds. The communication speeds must be selected intelligently to prevent the occurrence of delay between the Wi-Fi and cellular transmission.

The cluster head needs to communicate with the base station using cellular communication only. One must be built into the controller board a cellular chip to efficiently establish communication with the base station to maintain the speed levels. There could be a communication delay due to the protocol conversion at the cluster head level.

Figure 4 illustrates the internal architecture of working between the cluster head, base station, and microcontroller.

Each cluster (c_1, c_2, c_3) converts the Wi-Fi into cellular and sends the result via the base station to the controller. The CDMA packets are then converted to the USB packets and transmitted to the service server.

6. Results and discussions

6.1. Computing the performance of the prototype network

The performance computations of the prototype model was carried out considering all the four clusters of sensing and actuating devices, each headed by a cluster head that receives the data using the Wi-Fi protocol

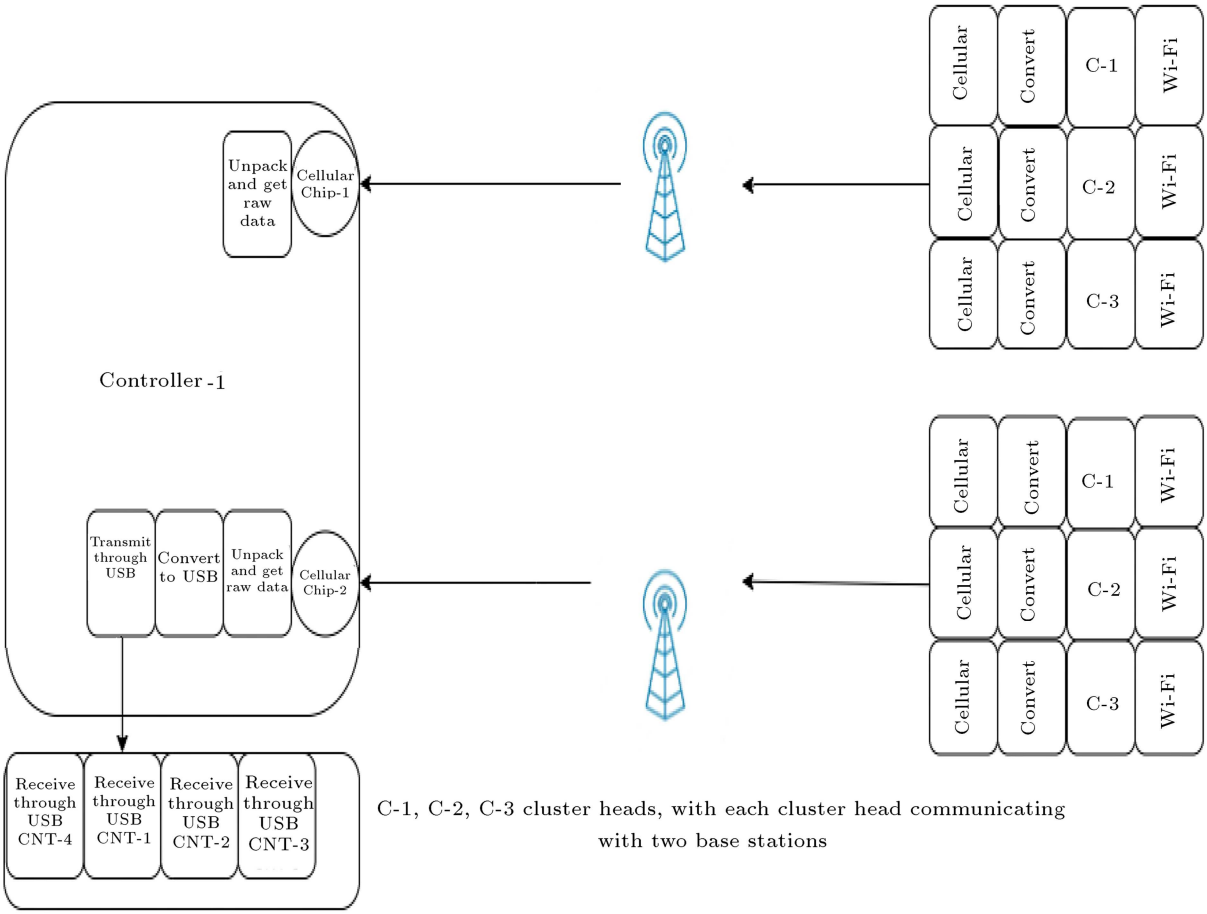

Figure 4. Internal architecture for effective communication between the cluster heads, base stations, and microcontrollers.

fixed at 11 Mbps with the data size of 13 bytes and CDMA communication speed fixed at 110 Mbps. The Ethernet speed is fixed at 110 Mbps for data transmission to the cloud through the gateway. The response time calculations for data transmission are represented in Table 5.

Table 5 shows the details of all the devices connected to the network. The performance computations of each device are done considering the reception, conversion, and transmission. The reception-related details of each device include some information about the size of the data received, protocol used for receiving the data, data packet size, Wi-Fi speed, time taken to receive the data, number of communication channels used to receive the data, and response time. The protocol used for conversion and the time taken for conversion are also included in the conversion details. On the transmission side, computations include the data size, conversion protocol, data packet size, transmission speed, transmission time, number of available channels, and single-channel transmission time. The total time taken to receive, convert, and then transmit is shown in the last column of the mentioned table. The time computations are done considering all the devices. The sum of time taken to undertake processing in each device is the response time of the IoT network.

Wi-Fi communication takes at the minimum speed of 11 Mbps which must be taken into the design as the devices have less power. The cluster head data should be moved to the base station using cellular communication varying from 100 Mbps onwards, and the size of the data packet can be fixed at its minimum value, i.e., 64 bytes. The latency in the communication overhead depends on the cellular speed chosen for communication.

According to the table, the transmission of data to a cloud takes 1152 microseconds to complete.

6.2. Computing the response time of revised IoT network

More detailed performance computations of the revised IoT network are shown in Table 6. According to this table, the Wi-Fi speed is set at 11 Mbps so as to facilitate the Wi-Fi data size of 48 bytes per packet using the cellular speed fixed at 170 Mbps and data size of 64 bytes/packet using zero latency time of data transmission at the cluster head. However, 0.001 microseconds of time is needed for repacking and packaging into cellular data packets.

Table 5. Time computations of prototype IoT network.

		Reception						Conversion time			Transmission							
Transmitting device code	Device description	Total data in bytes received	Protocol used for receiving the data	Incoming data packet size in bytes	Incoming speed used for receiving the data in Mbps	Total time spent for reception in microseconds	Number of paths available for reception	Time spent for receiving per path in microseconds	Protocol used for conversion	Protocol conversion time in microseconds	Latency time in microseconds	Total data in bytes to be transmitted	Protocol used for transmission	Outgoing data packet size in bytes	Transmission speed in Mbps	Time taken to transmit in microseconds	Number of paths available for transmission	Micro secs of time spent for transmission per path	Total time spent within the device for receiving and transmission in microseconds
N0010	Fan-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0011	Fan-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0012	Fan-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
CLUS1	Cluster head-1	13	Wi-Fi	48	11	34.91	1	34.91	CDMA	0.001	0.000	13	CDMA	68	110	4.95	1	4.95	39.856
N0020	AC-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0021	AC-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0022	AC-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
CLUS2	Cluster head-2	13	Wi-Fi	48	11	34.91	1	34.91	CDMA	0.001	0.000	13	CDMA	68	110	4.95	1	4.95	39.856
N0030	HU-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0031	HU-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0032	HU-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
CLUS3	Cluster head-3	13	Wi-Fi	48	11	34.91	1	34.91	CDMA	0.001	0.000	13	CDMA	68	110	4.95	1	4.95	39.856
N0040	TEMP-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0. 002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0041	TEMP-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0. 002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
N0042	TEMP-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	1	34.91	34.913
CLUS4	Cluster head-4	13	Wi-Fi	48	11	34.91	1	34.91	CDMA	0.001	0.000	13	CDMA	68	110	4.95	1	4.95	39.856
BASI	Base station-1	13	CDMA	68	170	3.200	1	3.200	NA	0.000	0.000	13	CDMA	68	110	4.95	1	4.95	8.145
N005A	Controller-1	156	CDMA	211	170	9.929	1	9.930	USB	0.001	0.003	156	USB	215	120	14.33	1	14.33	24.267
N006	Restful server	156	USB	215	480	3.583	1	3.583	NA	0.000	0.000	156	Wi-Fi	191	11	138.91	1	138.91	142.492
N007	Gateway	156	Wi-Fi,	211	11	153.455	1	153.455	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	167.384
N008	Access point	156	Wi-Fi	211	11	153.455	1	153.455	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	167.384
N009	WEB server	156	Ethernet	211	100	16.880	1	16.880	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	30.809
N00A	Cloud	156	Ethernet	174	100	13.920	1	13.920	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	27.849
N00B	User device	020	Ethernet	38	100	3.040	1	3.040	Ethernet	0.004	0.005	020	Ethernet	38	100	3.04	1	3.04	6.089
Total response time							24	497.110									24	655.60	1152.798

Table 6. Performance computations-revised IoT network.

		Reception							Conversion time			Transmission							
Transmitting device code	Device description	Total data in bytes received	Protocol used for receiving the data	Incoming data packet size in bytes	Incoming speed used for receiving the data in Mbps	Total time spent for reception in microseconds	Number of paths available for reception	Time spent for receiving per path in microseconds	Protocol used for conversion	Protocol conversion time in microseconds	Latency time in microseconds	Total data in bytes to be transmitted	Protocol used for transmission	Outgoing data packet size in bytes	Transmission speed in Mbps	Time taken to transmit in microseconds	Number of paths available for transmission	Micro secs of time spent for transmission per path	Total time spent within the device for receiving and transmission in microseconds
N0010	Fan-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0011	Fan-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0012	Fan-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
CLUS1	Cluster head-1	13	Wi-Fi	48	11	34.91	2	17.450	CDMA	0.001	0.000	13	CDMA	68	170	3.20	2	1.60	19.056
N0020	AC-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0021	AC-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0022	AC-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
CLUS2	Cluster head-2	13	Wi-Fi	48	11	34.91	2	17.450	CDMA	0.001	0.000	13	CDMA	68	170	3.20	2	1.60	19.056
N0030	HU-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0031	HU-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0032	HU-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
CLUS3	Cluster head-3	13	Wi-Fi	48	11	34.91	2	17.45	CDMA	0.001	0.000	13	CDMA	68	170	3.20	2	1.60	19.056
N0040	TEMP-1	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0041	TEMP-2	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
N0042	TEMP-3	13	NA	13	NA	0.001	1	0.001	Wi-Fi	0.001	0.002	13	Wi-Fi	48	11	34.91	2	17.45	17.459
CLUS4	Cluster head-4	13	Wi-Fi	48	11	34.91	2	17.450	CDMA	0.001	0.000	13	CDMA	68	170	3.20	2	1.60	19.056
BASI	Base station-1	13	CDMA	68	170	3.200	1	3.200	NA	0.000	0.000	13	CDMA	68	170	3.20	1	3.20	6.400
BAS2	Base station-2	13	CDMA	68	170	3.200	1	3.200	NA	0.000	0.000	13	CDMA	68	170	3.20	2	1.60	4.800
BAS3	Base station-3	13	CDMA	68	170	3.200	1	3.200	NA	0.000	0.000	13	CDMA	68	170	3.20	2	1.60	4.800
BAS4	Base station-4	13	CDMA	68	170	3.200	1	3.200	NA	0.000	0.000	13	CDMA	68	170	3.20	2	1.60	4.800
1BAS5	Base station-5	13	CDMA	68	170	3.200	1	3.200	NA	0.000	0.000	13	CDMA	68	170	3.20	1	3.20	6.400
N005A	Controller-1	13	CDMA	68	170	3.200	2	1.600	USB	0.001	0.003	13	USB	72	120	4.80	1	4.80	6.404
N005B	Controller-2	13	CDMA	68	170	3.200	2	1.600	USB	0.001	0.003	13	USB	72	120	4.80	1	4.80	17.459
N005C	Controller-3	13	CDMA	68	170	3.200	2	1.600	USB	0.001	0.003	13	USB	72	120	4.80	1	4.80	6.404
N005B	Controller-4	13	CDMA	68	170	3.200	2	1.600	USB	0.001	0.003	13	USB	72	120	4.80	1	4.80	6.404
MUSB	Multi-USB port	156	USB	215	480	3.583	4	0.896	NA	0.000	0.000	156	USB	72	480	1.20	1	1.20	2.096
N006	Restful server	156	USB	215	480	3.583	1	3.583	NA	0.000	0.000	156	Wi-Fi	191	11	138.91	2	69.45	73.038
N007	Gateway	156	Wi-Fi,	191	11	138.909	1	138.909	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	152.838
N008	Access point	156	Wi-Fi	191	11	138.900	1	138.909	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	152.838
N009	WEB server	156	Ethernet	174	100	13.920	1	13.920	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	27.849
N00A	Cloud	156	Ethernet	174	100	13.92	1	13.92	Ethernet	0.004	0.005	156	Ethernet	174	100	13.92	1	13.92	27.849
N00B	User device	20	Ethernet	38	100	3.04	1	3.04	Ethernet	0.004	0.005	020	Ethernet	38	100	3.04	1	3.04	6.089
Total response time		–	–	–	–	–	43	405.41	–	–	–	–	–	–	–	–	52	375.63	792.192

As observed in Table 6, it takes only 792 microseconds to transmit 174 bytes of payload as against 1152 microseconds taken to transmit the size of payload which in turn yields 57% timesaving.

6.3. *Performance improvement due to alternative paths in the IoT network (Paths versus response time)*

The response time of an IoT network depends on the number of communications built into the IoT network. The topologies used in different layers also play a critical role in determining the number of paths. The number of clusters keeping the number of inputs also has a great bearing on the performance of the IoT network. It is shown in the revised network how a crossbar network can be used in the device layer to increase the number of available paths to affect the communication. Table 7 shows the number of clusters considered for catering to 12 inputs per cluster, total paths contained in the network, response time in Microseconds, and a % increase in the response time due to a decrease in the number of paths. In this table, the response time can be improved up to 30% in the case of using three clusters and three inputs to each cluster, which yields 256 paths leading to a response time of 431 microseconds. Figure 5 shows the behavior of the IoT network containing several paths as well as the decrease in the response time.

6.4. *Performance improvement due to SOJK clustering algorithm (Power dissipation, number of packets, max data transmitted, response time)*

The performance of the revised IoT network was evaluated followed by implementing the SOJK algorithm by transmitting three data packets one after the other and recording the extent of power dissipation within the sensing devices.

Table 8 lists the number of sent packets, total amount of consumed power, power generated from cluster head rotation, actual amount of consumed power, power depletion rate, and response time.

Data analysis mainly focuses on the power depletion, number of packets transmitted before reaching the zero-power stage, and total size of the transmitted data. Figure 6 presents an analysis of the number of packets transmitted in the zero-power state. Figure 7 exhibits the behavior of power depletion during data transmission. Figure 8 gives an estimation of the total number of the transmitted packets concerning the total power consumed.

The response time increases with an increase in the number of the transmitted packets. On the contrary, the total power consumed decreases as the number of packets transmitted decreases.

Table 9 presents a comparative analysis of the SOJK algorithm in comparison with other algorithms. From the comparison, it can be concluded that the power depletion rate is the least, more packets are transmitted, and more paths are used for transmission.

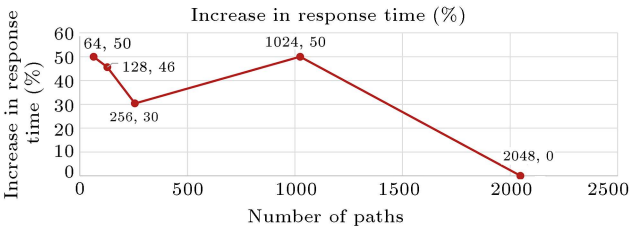

Figure 5. Number of paths vs. % improvement in the response time.

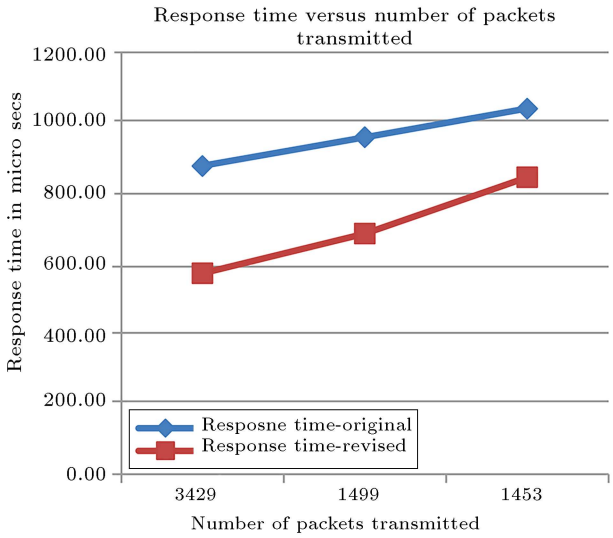

Figure 6. Comparative analysis of the response time vs. number of packets transmitted at zero power state.

Table 7. Performance improvement due to increase in the number of paths.

Revised IoT network					
Number of clusters	**Total inputs**	**Total inputs per cluster**	**Number of paths**	**Response time in microseconds**	**Increase in response time (%)**
1	12	12	2048	150	0
2	12	6	1024	300	50
3	12	4	256	431	30
4	12	3	128	792	46
6	12	2	64	1584	50

Table 8. Data summarization of revised IoT model.

Packet number	Original total power of the devices in the clusters in watts	Size of the data transmitted in bytes	Total power consumed in watts	Power gained due to cluster Head rotation in watts	Actual power consumed in watts	Power depletion in watts	Estimated number of packets transmitted before the power goes to 0 stage	Response time in microseconds
1	48.000	1279	47.971	0.015	47.986	0.014	3429	582.089
2	47.971	1387	47.954	0.015	47.939	0.032	1499	690.877
3	47.954	1480	47.936	0.015	47.921	0.033	1453	844.914

Table 9. Comparative analysis of clustering models.

Parameter used	LEACH	ILEACH	RLEACH	SOJK LEACH
The average size of the data in bytes transmitted in three packets	1312	1312	1312	1312
Average power Depletion for three packets of data Transmission in watts	0.432	0.322	0.271	0.021
The average number of the estimated number of packets transmitted at zero power state	1123	1275	1279	2127
The average number of paths used for transmission	15	15	15	372
Number of conversions used	0	0	0	4
Average response time in microseconds	954.328	901.77	895.38	705.96

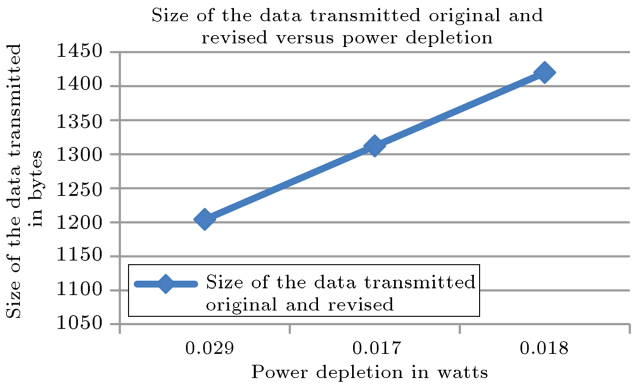

Figure 7. Comparative analysis of power depletion vs. size of the transmitted data.

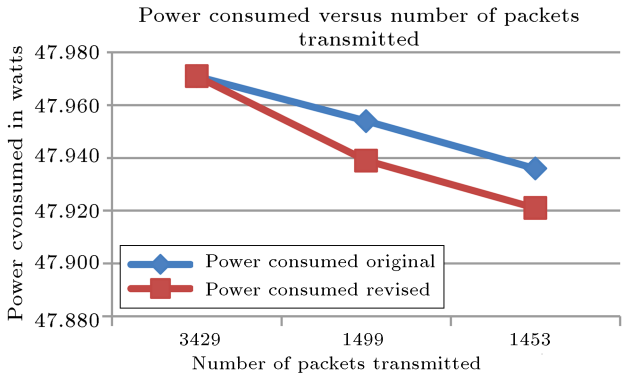

Figure 8. Comparative analysis of total amount of consumed power vs. number of transmitted packets.

6.5. Results and discussions-handling heterogeneity

The Wi-Fi and cellular speeds must be appropriately determined to obtain zero latency. Table 10 shows the time delay resulting from varying cellular communication speeds and number of data packets chosen for transmission. Only one packet of data transmitted in a single transmission from the device is chosen, fixing Wi-Fi communication speed at its minimum value because of the low energy levels of the device. As observed in Table 10, the latency is zero when a single cellular packet is chosen, and the same is transmitted at the communication speed of 170 Mbps.

Figure 9 confirms that at the cellular speed of 170 Mbps, the latency value is zero. In the case of two communication channels, the data transmission from the devices to the controller will be faster than usual, and the performance of the IoT network will be improved quite rapidly.

Table 11 presents the performance computations in the varying sizes of the data transmitted from the device and rate of decrease in the time taken to complete the data transmission.

Table 11 shows the data transmission considering all the devices @ 1560 bytes that gives the optimum response, as depicted in Figure 10.

Table 10. Wi-Fi and cellular speed communication.

Wi-Fi							Cellular							
Speed (Mbps)	Overhead (Bytes)	Payload (Bytes)	Packet size (Bytes)	Number of packets	Total bytes to be transmitted	Time taken to transmit in microseconds	Speed (Mbps)	Overhead (Bytes)	Payload (Bytes)	Packet size (Bytes)	Number of packets	Total bytes to be transmitted	Time taken to transmit in microseconds	Delay time (microseconds)
11	35	13	48	1	48	34.91	110	51	13	64	7	448	32.58	209.45
11	35	13	48	1	48	34.91	120	51	13	64	6	384	25.60	174.54
11	35	13	48	1	48	34.91	130	51	13	64	5	320	19.69	139.63
11	35	13	48	1	48	34.91	140	51	13	64	4	256	14.63	104.72
11	35	13	48	1	48	34.91	150	51	13	64	3	192	10.24	069.81
11	35	13	48	1	48	34.91	160	51	13	64	2	128	06.40	034.90
11	**35**	**13**	**48**	**1**	**48**	**34.91**	**170**	**51**	**13**	**64**	**1**	**64**	**3.01**	**0.00**

Table 11. Comparisons of performance considering the prototype model and the revised model with change in data size.

Type model	Total data transmitted in bytes	Response time in microseconds	Time taken in microseconds per byte transferred	Decrease in time (%)
Prototype model	156	1152.80	7.39	0.00
Device and controller modified model	156	792.19	5.08	31.28
	312	1244.48	3.99	21.45
	468	**1696.768**	**3.63**	**9.10**
	1560	4862.783	3.12	14.02

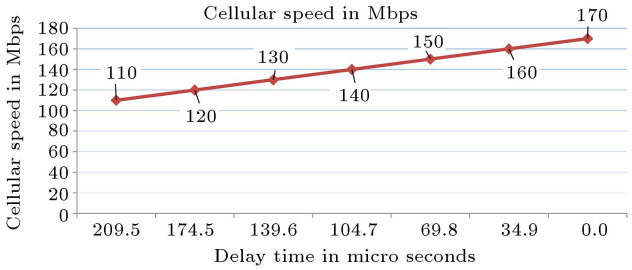

Figure 9. Cellular communication speed vs. delay time (keeping several Wi-Fi packets and the communication speed fixed).

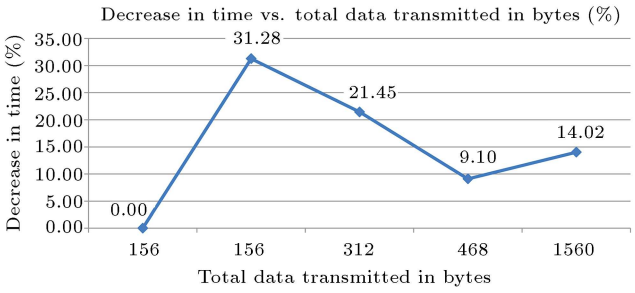

Figure 10. Evaluating optimum performances in terms of % reduction in time to complete data transmission.

7. Conclusions

- The performance of the IoT networks is the most crucial issue that must be addressed, especially when they are used in the defense, automobile, aerospace, and other sectors;
- Compared to the performance achieved through the prototype model, the performance of the IoT networks in terms of the time taken to transmit the data from device to the storage layer was improved up to 57%. Grouping devices achieve this as clusters of devices connected into a multi-stage network within the device level. The performance improvement is primarily due to the availability of more communication paths;
- Use of multiple base stations and microcontrollers using dual paths for communication increased the number of available paths between different layers of the IoT networks, hence an increase in the response time;
- Heterogeneity happens in the case of using two protocols for communication. Of note, it must be efficiently handled to eliminate latency by choosing proper communication speeds and data packet sizes. At the cellular speed of 170 Mbps and Wi-Fi speed

of 11 Mbps, zero latency was obtained when the data size was fixed at 468 bytes;

- Proper coupling was required in the case of using different topologies in different layers to make several paths available for communication. In each layer of an IoT network, different performance-related issues must be addressed;
- Based on the parameters affecting the performance of the service layers, the gateway layer could be addressed as a future scope;
- The power depletion rate and total amount of consumed power decreased while the total data and packets transmitted increased through the SOJK algorithm. A satisfactory number of packets were transmitted considering the zero-power state of the devices;
- The SOJK algorithm proved to be quite more efficient than other algorithms in terms of its power depletion rate, response time, number of packets dispatched, and total data transmitted.

References

1. Rahimi, P. and Chrysostomou, C. "Improving the network lifetime and performance of wireless sensor networks for IoT applications based on fuzzy logic", *15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Santorini, Greece, pp. 667–674 (2019).
2. Behzad, M., Abdullah, M., Hassan, M.T., et al. "Performance optimization in IoT-based next-generation wireless sensor networks", *Transactions on Computational Collective Intelligence XXXIII. Lecture Notes in Computer Science*, 11610, Springer, Berlin, Heidelberg, pp. 1–31 (2019).
3. Khan, Imran., Haque, Md.Ershad, Asikuzzaman, Md and Shah, Syed Bilal. "Comparative study of IoT-based topology maintenance protocol in a wireless sensor network for structural health monitoring", *Remote Sensing*, **12**(15), p. 2358 (2020).
4. Ogudo, A., Kingsley Muwawa, Dahj and Jean Nestor, Dahj Khalaf, Osamah Daei Kasmaei, H. "A device performance and data analytics concept for smartphones' IoT services and machine-type communication in cellular networks", *Symmetry*, **11**(4), p. 593 (2019).
5. Metongnon, L., Ezin, E.C., and Sadre, R. "Efficient probing of heterogeneous IoT networks", *IFIP/IEEE Symposium on Integrated Network and Service Management (IM).*, Lisbon, Portugal, pp. 1052–1058 (2017).
6. Sowmya, K.V. and Sastry, J.K.R. "Performance evaluation of IoT systems - basic issues", *Int. J. Eng. Technol*, **7**(2–7), pp. 131–137 (2018).
7. Geetha, V. Kallapur, P.V., and Tellajeera, S. "Clustering in wireless sensor networks: Performance comparison of LEACH and LEACH-C Protocols Using NS2", *Procedia Technol*, **4**, pp. 163–170 (2012).
8. Vijitha Ananthi, J., Chinnalagi, V., Murugeshwari, R., et al. "An effective performance of smart sensor network using IOT", *International Journal of Advance Research, Ideas and Innovations in Technology*, **3**(2), pp. 638–646 (2017).
9. Bhandari, S., Sharma, S.K., and Wang, X. "Cloud-assisted device clustering for lifetime prolongation in wireless IoT networks", *Can. Conf. Electr. Comput. Eng.*, Windsor, ON, Canada, pp. 8–11 (2017).
10. Ashwini, M. and Rakesh, N. "Enhancement and performance analysis of the LEACH algorithm in IoT", *Proc. Int. Conf. Inven. Syst. Control. ICISC.*, Coimbatore, India, pp. 1–5 (2017).
11. Choi, D.K., Jung, J.H., Kang, H.W., et al. "Cluster-based CoAP for message queueing in Internet-of-Things networks", *Int. Conf. Adv. Commun. Technol. ICACT., PyeongChang*, Korea (South), pp. 584–588 (2017).
12. Khedira, S.E.L., Nasri, N.A., Wei, A., et al. "A new approach for clustering in wireless sensors networks based on LEACH", *Procedia Comput. Sci.*, **32**, pp. 1180–1185 (2014).
13. Choi, D.K., Jung, J.H., and Koh, S.J. "Enhanced cluster-based CoAP in Internet-of-Things networks", *Int. Conf. Inf. Netw., Chiang Mai.*, Thailand, pp. 652–656 (2018).
14. Liu, Y. and Zhou, Y. "Development of distributed cache strategy for the analytic cluster in an Internet of Things system", *ICNSC 2018 - 15th IEEE Int. Conf. Networking, Sens. Control.*, Zhuhai, China, pp. 1–6 (2018).
15. Behera, T.M., Samal, U.C., and Mohapatra, S.K. "Energy-efficient modified LEACH protocol for IoT application", *IET Wirel. Sens. Syst*, **8**(6), pp. 223–228 (2018).
16. Behera, T.M., Mohapatra, S.K., Samal, U.C., et al. "Residual energy-based cluster-head selection in WSNs for IoT application", *IEEE Internet Things J.*, **6**(3), pp. 5132–5139 (2019).
17. Sowmya, K.V. and Sastry, J.K.R. "Performance optimization within the device layer of IoT networks", *Journal of Engineering and Technology*, **13**(6), pp. 1338–1346 (2020).
18. Sowmya, K.V., Chandu, A., Nagasai, A., et al. "Smart home system using clustering based on internet of things", *J. Comput. Theor. Nanosci*, **17**(5), pp. 1–6 (2020).
19. Ju, Q. and Zhang, Y. "Adaptive clustering for the Internet of Battery-less Things", *IEEE Wirel. Commun. Netw. Conf. WCNC.*, San Francisco, CA, USA, pp. 1–5 (2017).
20. Kumar, P. "Data stream clustering in the Internet of Things", *SSRG Int. J. Comput. Sci. Eng.*, **3**, pp. 14–19 (2016).
21. Zhao, S., Yu, L., Cheng, B., et al. "IoT service clustering for dynamic service matchmaking", *Sensors* (Switzerland), **17**(8), pp. 1–17 (2017).

22. Puschmann, D. Barnaghi, P., and Tafazolli, R. "Adaptive clustering for dynamic IoT data streams", *IEEE Internet Things J.*, **4**(1), pp. 64–74 (2017).
23. Tao, X. and Ji, C. "Clustering massive, small data for IoT", *2nd Int. Conf. Syst. Informatics, (ICSAI 2014)*. Shanghai, China, pp. 974–978 (2015).
24. Kwon, M. and Park, H. "The cluster formation strategies for approximate decoding in IoT networks", *Int. Conf. Inf. Netw.*, Kota Kinabalu, Malaysia, pp. 366–368 (2016).
25. Geethika Reddy, A., Upendra, Y., Sastry, J.K.R., et al. "An approach to compute fault tolerance of an IoT network having clustered devices using crossbar networks", *Int. J. Emerg. Trends Eng. Res.*, **8**(4), pp. 987–1004 (2020).
26. Sai Rama Krishna, J.S.B.M. and Sastry, J.K.R. "Building fault tolerance within wireless sensor networks: A butterfly model", *Res. J. Appl. Sci.*, **12**(2), pp. 139–147 (2017).
27. Rajasekhar, J. and Sastry, J. "Building composite embedded systems based networks through hybridization and bridging i2c and can", *J. Eng. Sci. Technol*, **15**(2), pp. 858–881 (2020).
28. Rajasekhar, J. and Sastry, J. "On developing high-speed heterogeneous and composite ES network through multi-master interface", *International Journal of Advanced Computer Science and Applications*, **11**(12), pp. 320–333 (2020).
29. Sastry, J., Ramya, G.S., Niharika, V.M., et al. "Performance optimization of IoT networks within gateway layer", *Recent Adv. Comput. Sci. Commun*, **13**(6), pp. 1338–1346 (2019).
30. Sasi Bhanu, J., Sastry, JKR., Venkata Sunil Kumar, P., et al. "Enhancing performance of IoT networks through high-performance computing", *International Journal of Advanced Trends in Computer Science*, **8**(3), pp. 432–442 (2019).
31. Vishnu Priya, B. and Sastry, J.K.R. "A comparative analysis of the methods used to build information/content-centric networks over software-defined networks", *J. Eng. Technol*, **7**(2), pp. 997–1003 (2018).
32. Pavithra, T. and Sastry, J.K.R. "Strategies to handle heterogeneity prevalent within an IOT based network", *Int. J. Eng. Technol*, **7**(2–7), pp. 77–83 (2018).
33. Sastry, J.K.R., Naga Sai Tejasvi, T., and Aparna, J. "Dynamic scheduling of message flow within a distributed embedded system connected through an RS485 network", *ARPN J. Eng. Appl. Sci*, **12**(9), pp. 2809–2817 (2017).

Biographies

Kambhampati Venkata Sowmya is a Master of Technology holder in Electronics and Communication Engineering and presently working @ KLEF University, Vaddeswram, India. She published 10 Scopus and SCI indexed papers as on date. She is presently working as a scholar in the final stage of graduating a PhD degree.

Jammalamadaka Kodanda Rama Sastry holds double PhDs in the field of Computer Science and Engineering and Management and presently working @ KLEF University, Vaddeswram, India. Has published 247 publications in reputed journals indexed in SCI and SCOPUS. He specializes in the fields of IoT, embedded systems, cloud computing, AI, ML, data science, WEB technologies, and software engineering. He has 28 years of IT experience and 15 years of Academic and Research Experience throughout his professional career.