



Scalable energy optimization of resources for mobile cloud computing using sensor enabled cluster based system

Santosh Kumar Yadav¹ · Rakesh Kumar²

Accepted: 6 June 2024 / Published online: 18 June 2024

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

The rising craze of sensor enabled mobile devices promotes its usage in Mobile Cloud Computing (MCC), Mobile Edge Computing (MEC) and other distributed computing environments derived from the cloud computing environment. As the computing paradigm shifts from centralized to distributed computing and mobile devices are getting smarter and resource rich, it facilitates the user to do computation to its proximity. Hence, it is quite useful to incorporate the Wireless Sensor Networks (WSN) with distributed computing environment to better cater to the user needs. The proposed work enhances the MCC and MEC by incorporating sensor enabled computing along with the application of energy optimization techniques such as coyote optimization, Fuzzy Logic (FL), data redundancy and data compression. A new framework called Sensor Enabled-Scalable Key Parameter Yield of Resources (SE-SKYR) framework is proposed in this research work by integrating SKYR framework with cluster-based sensing mechanism. The proposed work uses SKYR framework which is a cloudlet based MCC framework and works well for MEC as well. Cloudlet is used as the main computing component available at the local level which suits both MEC and MCC. The existing system uses the concept of relay node to transmit data packets in transmission path from sensor nodes to server via edge cloud and hence causes delay in transmission of data. In the proposed work, we have introduced a Scalable Energy Optimization of Resource (SEOR) algorithm to optimize the energy consumption by various resources. SE-SKYR framework along with SEOR algorithm addresses the problems faced by the existing system. The complexity of the proposed SEOR algorithm is less as compared to its existing counterparts and is also comprehended from the results.

Keywords Mobile cloud computing · SKYR framework · Cloudlet based edge computing · SEOR algorithm · SE-SKYR framework · EAO algorithm · Sensor enabled mobile edge computing

1 Introduction

Cloud computing is the mother of all latest distributed computing's such as MCC, MEC, fog computing, transparent computing etc. These distributed computing are wireless enabled which can be used in different environments and domains. This research work integrates the distributed computing environment with WSN to facilitate the sensor enabled computing along with the application of energy

optimization techniques such as coyote optimization, fuzzy logic, data redundancy and data compression. This research work uses MCC and MEC as distributed computing which is combined with WSN to facilitate the computing at edge level or local to the user. Therefore, in this paper MEC and MCC are contemplated as same because cloudlet is used as main computing component at local level, which best suits the working of both the mentioned computing environments.

Distributed wireless computing which enables the local level computation are in demand as it saves energy and time but do possess some technical threat like privacy, security etc. The cloudlet can be used in mobile cloud computing and mobile edge computing directly. However, it requires some architectural change to fit in for fog computing and transparent computing. Mobile devices promote the ubiquitous computing which helps user in their routine work. According to CISCO visual networking index, global data generated by

✉ Rakesh Kumar
raakeshdhiman@gmail.com

Santosh Kumar Yadav
santoshkumaryadav85@gmail.com

¹ Department of CSE, NITTTR, Chandigarh (U.T), India

² Department of CSE, CUH, Mahendergarh, India

mobile devices has risen drastically and it is twice more than non-mobile devices data [1]. So, mobile users are demanding more functionality from these devices and hence making its application more complex and resource hungry. However, the resources of mobile devices are limited like computing capability, storage capacity, battery life and network bandwidth. Thus, due to wide gap between the demand of resources and available resources, it gives opportunity to local resource rich mobile devices like cloudlet to provide its services to thin mobile devices in mobile cloud computing architecture [2]. The cloudlet at local level leverages the computation capability of cloud in proximity to the user. Best example of such type of architecture is Amazon EC2 server via WAN. But this type of architecture causes problem to latency sensitive applications which may suffer end to end delay due to complicated network conditions of WAN. To overcome the problem of latency, components like cloudlet and Adhoc cloud are introduced [3].

Mobile edge computing is another environment derived from cloud computing after MCC which deploys cloud server in proximity to the user at base station to improve the computational capability [4]. It provides various advantages such as proximity, location awareness, real time radio network information, low latency, and high bandwidth. As per European 5G PPP (5G Infrastructure public private partnership), MEC is established as a technology for 5G network [5]. The authors have proposed a cloud architecture called as Edge Cloud, which is working in the backend of the central cloud wholly or partially and this cloud is managed by open stack cloud management framework [6]. The main goal of this edge cloud is to have a localized system with high accuracy, less latency and reliable video streaming & monitoring system decreases bandwidth consumption by pre-processing the maximum content at the network edge. In [7], authors have proposed two approaches for cloudlet-based computing keeping in mind the mobile nature of cloudlet. First approach is the one in which the cloudlet mobility and task execution is centrally managed by the cloud. The second approach is decentralized in nature in which the cloudlet itself is managing its mobility and task execution by keeping its record in its log or history file [8].

The major drawbacks of above discussed distributed computing's are: there is no efficient mechanism to manage the heterogeneous devices to impart services as cloudlet and access services as users. Secondly, these computing's do not facilitate the sensor-based task computation and energy efficient mechanism to manage it. Third important concern is that it lacks scalability and reusability in this type of computations. Fourth major problem relates to poor management of error free transmission of sensed and processed data and its storage for future use. These computing's also face difficulties in managing the data redundancy while facilitating the sensor-based tasks.

As per literature, there is no framework or mechanism available to facilitate sensor-based task in mobile cloud computing and mobile edge computing which optimizes the energy consumption. In this research, energy optimization is achieved by five ways. Firstly, potential and highly efficient cloudlet is used as cluster head and, the selection of cluster head is performed by using fuzzy logic and coyote optimization technique. Next, these cluster heads are arranged and placed in heterogeneous cluster network in such a way that relay node is not required which reduces the energy consumption. In third step, energy optimization is done at sensor node level where the data is sensed and processed at cluster head. There it removes the redundant data using sensing rate adjustment (SRA) and index bit encoding (IBE) technique. Fourth stage is the compression of data where data flow intensity is used to compress the data and last stage is error free transmission of data from sensor node to destination node which helps in reducing energy consumption. So, all these energy optimization methods are collectively embedded with SKYR framework which facilitates cloudlet-based computing for MCC and MEC.

This research paper also focusses on scalability and reusability of resources as demand for computing increases and decreases. If demand for computation increases then new mobile devices get chance to act as cloudlet and become part of heterogeneous cluster network else it gets discarded off from cluster network to reduce the complexity of network. Hence, this facilitates the scalability and reusability of same existing resource according to demand. Now, this SKYR framework is further improved to SE-SKYR framework which facilitates sensor-based task in MCC and MEC using cloudlet.

The rest of the paper is organized as follows. Section 2 provides a thorough literature review in the field of WSN and energy optimization techniques for MEC and MCC. Section 3 discusses ESDWSN framework and its EAO algorithm. The proposed framework (SE-SKYR) is presented in Sect. 4. Section 5 discusses the implementation details and observations. Finally, Sect. 6 concludes the paper with the directions for future research.

2 Literature review

This section explores the existing research work done in the field of WSN and energy optimization techniques in detail. Apart from this, it reviews the advanced version Software Defined Wireless Sensor Network (SDWSN) and its application in edge environment as Software Defined Wireless Sensor Network (ESDWSN). To the best of our knowledge there is no research work proposed in MCC or MEC with sensing mechanism. For this proposed work, ESDWSN is

taken as reference framework as ESDWSN is most similar to the current proposed work.

2.1 Wireless sensor network

This section reviews different aspects of wireless sensor network and how it can be integrated with the distributed computing architecture. Wireless sensor network is the main component in IoT enabled computing in which sensors are well connected with wireless media. There are many applications of IoT facilitated computing in the field of environmental monitoring, healthcare, smart building, smart cities, industry, weather, security, and military [9]. The WSN network have some limitations such as poor communication bandwidth, low memory size and limited power supply. This type of network generally works in hostile environment where it is difficult to replace or recharge the power of installed sensors [10]. Therefore, to counter this limitation, clustering technique is proved to be one of the best suited techniques to manage network energy consumption effectively and to prolong the network lifetime. The clustering based hierarchical technique is executed by partitioning sensor nodes into cluster in which each cluster is headed by one sensor node called as cluster head and other nodes act as members [11]. In such type of clusters, head is accountable for managing its sensor nodes and collecting the sensed data from them and submit it to the cloud or base station [12]. Selection of correct cluster head is the most important step as it helps to improve throughput, network lifetime, energy consumption and stability within complete system [13]. Clustering based approach to manage the wireless network has many limitations and hence researchers are working on metaheuristics-based clustering approach to enhance the lifetime of the network and to reduce the energy consumption [14, 15].

In [16], authors have proposed Profit Maximization Multi-Round Auction (PMMRA) algorithm which uses matching strategy for resource provider and user which calculates the payment for sharing resources between them. Therefore, SDWSN provides the balance allocation of energy among various stakeholders and reduces traffic load. Strategies discussed in [17] have improved the network performance to some extent, but have some limitations in data processing and network architecture. In these strategies, edge computing and novel software definition techniques are used separately. In [18, 19], authors have used the feature of edge computing in which the resource rich device can be assigned the role of edge cloud. The software defined technology and edge computing can be combined to create a new architecture called as ESDWSN which can exploit the advantages of both [20].

A heterogeneous wireless sensor network (HWSN) may have features like real time sensing, wireless transmission,

self-organization of network, collective processing etc. HWSN has many applications in the field of environmental monitoring, national defence, and hospitals to facilitate human being with better quality of services [21]. The wireless sensors are smaller in size and very low in price but due to this, it has little battery capacity and hence easily fails in harsh conditions. Therefore, with these limited resources, it is very hard to reduce energy consumption and improve the energy efficiency challenges of HWSN. In HWSN, some sensors acting as relay node to transmit data may drain out of battery due to this challenge. The energy consumption in relaying data takes more energy as compared to the sensing data and hence, it is one of the major challenges which attracts researchers to improve it [22]. The main advantages of wireless sensor are its small size and low price. However, it has a low battery capacity and may fail in critical situations. Therefore, major challenges involved with HWSN is to reduce energy consumption, improve energy efficiency to manage rare energy resources. Hence, energy optimization is the most important factor for the researchers to work for HWSN. To overcome the energy optimization problem, relay nodes are used to transmit the sensed data from source node to sink node. Relay node placement can work better till the network connectivity is good but when the connectivity gets lost, the relay node placement task becomes a NP hard problem [23, 24]. Therefore, relay node placement is more difficult to achieve for poor and complex connectivity and when the HWSN increases its size to accommodate increased sensor nodes.

2.2 Energy optimization techniques

This section explores the various energy optimization techniques and methods that can be incorporated with distributed computing's and WSN architecture. The energy optimization includes the optimization of sensed data, its processing, transmission, storage, reduction in computation, redundancy of data etc. There are various algorithms and techniques to optimize the energy in WSN and distributed computing's like Coyote Optimization Algorithm (COA) [25], Grey Wolf Optimization (GWO) [26], Particle Swarm Optimization (PSO) [27], Stable Election Protocol (SEP) [28] and Low-Energy Adaptive Clustering Hierarchy Protocol (LEACH) [29]. The major problem with these algorithms is unbalanced energy allocation among its member nodes which perform data computation and hence reduces the network lifetime. This problem can be addressed with the application of fuzzy logic in combination with COA.

Many authors have already used Fuzzy logic based clustering mechanism for cluster head selection [30–34]. In [20], Li et al. have proposed the working architecture and an algorithm called as Energy Allocation Optimization (EAO) which effectively manages the data for optimum energy

consumption. This EAO algorithm and its ESDWSN framework is used as reference for the proposed work. The challenges of this architecture are discussed in detail in Sect. 3.2.

In the proposed work, fuzzy logic is used to select the potential mobile device as cluster head. These mobile devices are transient in nature and are hence very unstable. Fuzzy logic is best suited where the system is uncertain and changing frequently and it is difficult to design any mathematical strategy [35]. Another factor which promotes the fuzzy logic over its counterparts is its complexity which is quite less as compared to the other decision-making methods [36]. FL is best suited for online applications and real time issues as it is quick in decision making. It rapidly adapts to various heterogeneous inputs [37]. Therefore, FL is best apt for the proposed work because it is very simple to implement, less complex, fast in execution, decision making is effective and works comfortably with unstable and uncertain conditions.

3 ESDWSN framework and its EAO algorithm

ESDWSN is a combination of SDN and EC in the domain of WSN. Hence, the researcher has named it as ESDWSN framework. This framework uses an algorithm mentioned as Energy Allocation Optimization (EAO) which solves the problem of energy consumption averaging and minimization (ECAM) [20]. This EAO algorithm uses an adaptive Levenberg–Marquardt algorithm which helps in energy minimization. So, the proposed work has improved this EAO algorithm and presented a framework which facilitates sensor enabled computing in mobile cloud computing and mobile edge computing. The working architecture of ESDWSN framework is shown in Fig. 1. It is divided into three layers: perceptual layer, edge layer and control layer. The perceptual layer consists of various data acquisition devices (DAD) whose task is to sense data and send it to the edge server [38]. The next layer is the edge layer which is hosting various edge servers in different perceptual areas of sensor

nodes. Main tasks of edge servers are analysis, processing, and data transmission of all gathered data in the current perceptual area [39]. The last layer is control layer which performs the task of decision making. It consists of SDN controller which has control logic that can be programmed and reconfigured according to the requirements which vary from time to time. The main purpose of this layer is to manage data processing and energy efficiency and its trade off.

In Fig. 1, the centralized SDN controller controls the set of edge servers in particular locations. These edge servers are also acting as sink nodes despite controlling the complete working of sensor nodes. Edge server connects, communicate, and controls the sensor nodes which are also known as source nodes and are denoted as N11, N12, etc. So, when sensor node N11 would like to communicate with edge server of region 1 then it needs to take support of sensor nodes N12 and N13 and, they act as relay nodes for former. But no relay node is required to sensor node N13 as it is in reach of edge server. Sensor nodes sense data in continuous mode or on demand mode as per request of edge server. Processed and synthesized data is kept at edge server and used as per demand. These edge servers can share their data among themselves. Therefore, in this architecture EAO algorithm is applied to optimize energy consumption in data processing and transmission [20].

ESDWSN framework divides the server nodes into edge region and these edge regions have radius R , which is the transmission range of each server. Therefore, the area of edge region can be denoted as $R * R = R^2$. This ESDWSN framework have proposed an energy system model which categorizes different data packets based on the energy consumption.

3.1 EAO algorithm

The EAO algorithm works in three stages and are as follows:

First Stage:—At this stage, total number of sensor nodes ‘ n ’ is divided into multiple edge region O_{opt} . The optimal number of edge region is represented as O_{opt} as shown in Eq. 1 [20]. So, the total number of sensor in each region is equal to n/O_{opt} .

$$O_{opt} = \sqrt{\frac{n\tau}{2\pi\phi}} \frac{D}{d^2} \quad (1)$$

Second stage:—This stage selects the non-redundant data elements. EAO algorithm has categorized the data packets based on the energy consumption level. So, if data packets are having different energy levels, it means that they are non-redundant data packets. Redundant data packets will have same energy level or belongs to same category. So, energy function ($f(x)$) and convex optimization functions are used to decide the non-redundant data packets.

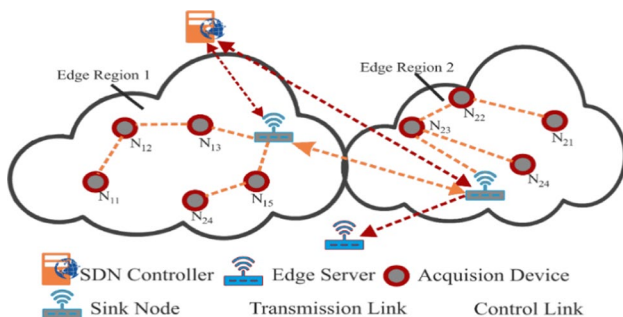


Fig. 1 Working architecture of ESDWSN framework [20]

Third Stage:—The third stage includes the transmission of non-redundant data packets from sensor node to edge server via various relay nodes. So, EAO algorithm chooses optimal transmission path with energy cost function ($f(x^*)$). The transmission path requires to estimate the energy cost function ($f(x^*)$) of accessible paths in each edge region. This energy cost function ($f(x^*)$) utilizes adaptive Levenberg–Marquardt (ALM) and Lagrangian dual technique in amalgamation with Karush–Kuhn–Tucker (KKT) condition to elucidate the ECAM drawback [20]. The overall complexity of this algorithm is less than or equal to $O\left(3n^2 + \frac{n}{O_{opt}}\right)$.

3.2 Drawbacks of EAO algorithm

Although EAO algorithm performs well, still it lags in few ways:

1. This algorithm uses relay node to transmit data packets in transmission path from sensor nodes to server via edge cloud and hence causes delay in data transmission.
2. The EAO algorithm is installed and executed at the centralized SDN (server) and it will delay the execution and data analysis process in the centralized sensor-based computing system. The control layer at center (at SDN) decides the transmission path and quality of data packets and its optimization. Due to this, it takes remote management of resources and hence lags in data processing and its manipulation.
3. During the relay process, each data packet is assigned to only one relay node and if the data packet is lost during the transmission process, then there is no procedure to retrieve the lost data packets. Hence data loss is major concern in EAO algorithm.
4. EAO algorithm does not use the data compression process to reduce the size of data. Moreover, the process used to remove the redundancy is not efficient for large data set.
5. Radius of each edge server is assumed to be same but in real time scenario it is very difficult to have so. To maintain the same radius, there must be some mechanism to address it. However, no much mechanism is provided in EAO algorithm or ESDWSN framework.
6. Its complexity can still be improved further.

4 Proposed work

SKYR framework [40–44] represents the cloudlet based MCC. This basic version is improved further to incorporate pricing module & utility module [45] which supports MCC and MEC. The later version of SKYR framework [45] is used in this proposed work to integrate with cluster-based sensing system. The proposed work considers MEC

and MCC as equivalent. The later version of SKYR framework [45] and cluster-based sensing system are discussed in Sect. 4.1.1 and 4.1.2 respectively. The integrated architecture serves the purpose of energy efficient sensing in MEC. To attain the objective of energy efficiency, the proposed work uses a mechanism (algorithm) to select the best set of cluster heads from the set of heterogeneous nodes, to compress and decompress data and remove data redundancy, to minimize the transmission delay and error in the transmitted data to the sink node. This work also suggests a scenario under which the energy consumption of complete system gets minimized which means it recommends the optimum number of clusters and radius range of sink node to effectively manage the clusters and sensor nodes. This is the brief system model of the proposed work. Detailed scenario description is discussed in later sections.

4.1 Cluster-based sensing SKYR framework for MCC

The proposed framework is named as Sensor enabled scalable key parameter yield of resources (SE-SKYR) framework which integrates cluster-based sensing system to the SKYR framework [45]. The SKYR framework facilitates MCC and MEC in which the main computing element is cloudlet, which is resource rich mobile device at the local level for thin mobile devices. Detailed explanation of SKYR framework [45] is given in Sect. 4.1.1. Cluster based sensing system has cluster head which is the intermediate node between the source sensor node and sink node. Sink node analyse the data received from sensor node via cluster head and relay node. Detailed working of cluster-based sensing system is explained in Sect. 4.1.2.

4.1.1 SKYR framework

SKYR framework is scalable key parameter yield of resources framework for cloudlet based mobile cloud computing. The working architecture of SKYR framework is shown in Fig. 2 [45]. Figure 2 has remote cloud at the top which regulates and controls the complete cloudlet based mobile cloud computing environment. Any mobile user who wants the services from cloud at local level may contact the remote cloud or may directly contact the primary cloudlet. Mobile user is the service seeker and will pay for the services used by it. Similarly, primary cloudlets and secondary cloudlets are resource or service providers, which share their resources and impart services and get paid for it. Primary cloudlet is basically acting as a cloud at local level which manages all the secondary cloudlets and handles the task requests of mobile users and, primary cloudlet receives payment in lieu of services. Then, the primary cloudlet keeps some money for its management and give rest to the secondary cloudlet. If any task which primary cloudlet is not able

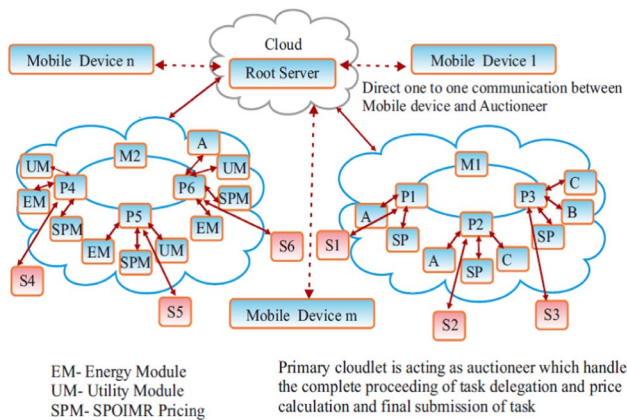


Fig. 2 Working architecture of SKYR framework [45]

to complete then it may contact the main remote cloud and get the task done for mobile users. Remote cloud manages the complete framework by creating master application virtual machine (MAVM) at respective primary cloudlet which is denoted as M1, M2 and M3. This MAVM helps cloud to communicate and track the working process of primary cloudlet and its subordinate secondary cloudlet to give best QoS to mobile users. Similarly, primary cloudlet installs the proxy VM for respective secondary cloudlet to interact with later and it is denoted as P1, P2, P3 etc. Within this proxy VM, there are various application VMs (of different applications) running on respective secondary cloudlet and are represented as UM (utility module), SPM (pricing module) and A & B represent any random application. Secondary cloudlet is denoted as S1, S2, S3 etc. which interacts with primary cloudlet via its respective proxy VM (P1, P2). Therefore, in this proposed work, SKYR framework is used and enhanced to make it more efficiently to handle the sensor enabled computing in mobile cloud computing and mobile edge computing. The proposed work incorporates the energy module which includes the SEOR algorithm with this SKYR framework [45] to make it sensor enable SKYR (SE-SKYR) framework. Complete SE-SKYR framework is explained later in this paper.

4.1.2 Cluster based sensing system

In a cluster-based sensing system there are four entities and these are destination sink node, source sensor node, cluster head and relay node. The task of source node is to generate the sensed data and relay node helps in transmitting the sensed data from source node to cluster head. Cluster head collects the sensed data from source node via relay node and then process & refine it before forwarding to the sink node, which is also called as destination node. Sink node analyses the data and acts based on the analysis or may store the analysed data for future use. The basic architecture of

cluster-based sensing system is shown in Fig. 3. In this figure, sink, cluster head and source node are shown but relay node is missing. This relay node can be assumed in between cluster head and source sensor node which is shown in Fig. 4. As shown in Fig. 3, there are three levels, at the bottom it has sensors, intermediate layer has cluster heads and the top most layer has sink or destination nodes. All cluster heads have direct one to one communication with the sink or destination node. Each cluster is a separate unit but the cluster heads can interact with each other based on the requirement. Therefore, the proposed work uses the combination of SKYR framework and cluster-based sensing system to make new framework called as SE-SKYR framework, which facilitates the sensor enabled computing task in mobile edge computing and mobile cloud computing environment.

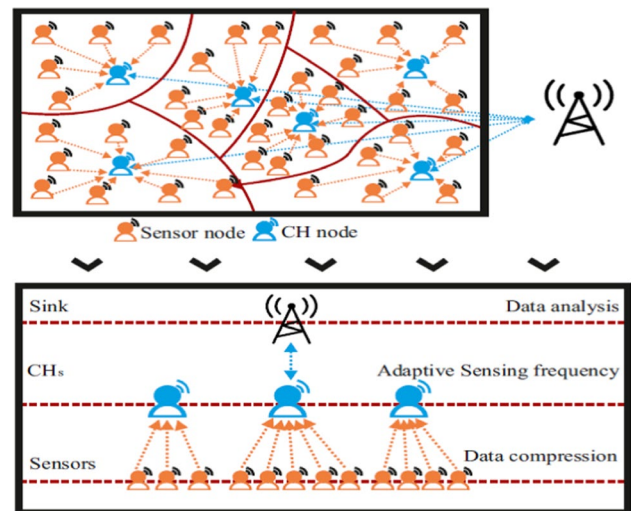


Fig. 3 Architecture of cluster-based sensing system

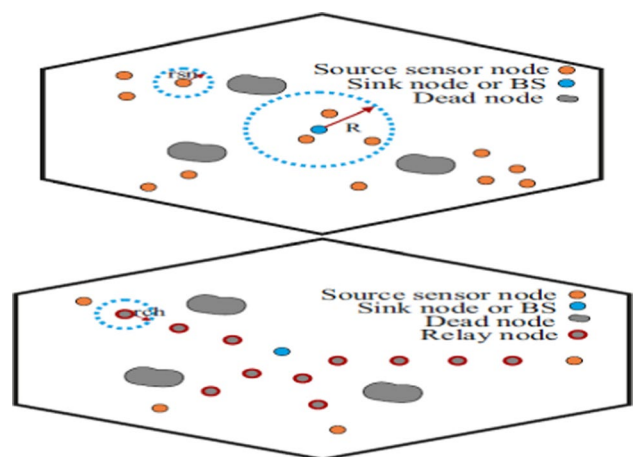


Fig. 4 Working of cluster-based sensing system

Figure 4 shows the communication among sensor node, cluster head (which also acts as relay node) and sink node. The upper half of the image shows the radius of sensor node denoted by r_{sn} and sink node radius as R . This radius denotes the communication range of the node. Gray area denotes the unreachable nodes or dead nodes which were part of the cluster-based sensing system but now not actively involved in the system. The lower half of the image depicts how relay nodes participate in the transfer of data from sensor node to the sink node. The ESDWSN framework and its EAO algorithm transfers the sensed data from sensor node to the sink node via multiple relay nodes. But the proposed work does not use multiple relay node rather it uses one cluster head which is so closely distributed as there is no need to have multiple relay nodes and this will save energy and avoid data loss in transmission.

4.1.3 Hexagonal layout for implementing the cluster-based scheme with SKYR framework

The proposed architecture includes the benefits of cloudlet based mobile edge computing, heterogeneous wireless sensor network and cluster-based sensing scheme for sensors. In this architecture, cloudlet is the main computing element and sensors are data collector or source node. Cloudlet are assigned two roles; primary cloudlet and secondary cloudlet. Secondary cloudlet is basically managing the sensor data which also acts as cluster head. Here, it will gather the data from source node (sensor), filter it, optimize, and transmit to the primary cloudlet which basically stores the qualitative data at local level and may send it to cloud for remote use and permanent storage.

A detailed hexagonal layout of SE-SKYR framework at cloudlet level is shown in Fig. 5. In this figure, primary cloudlet and its associated secondary cloudlets are arranged in hexagonal layout. Where primary cloudlet is located at the center of the hexagon marked with rectangular boxes in blue colour and its six vertices are assigned six secondary

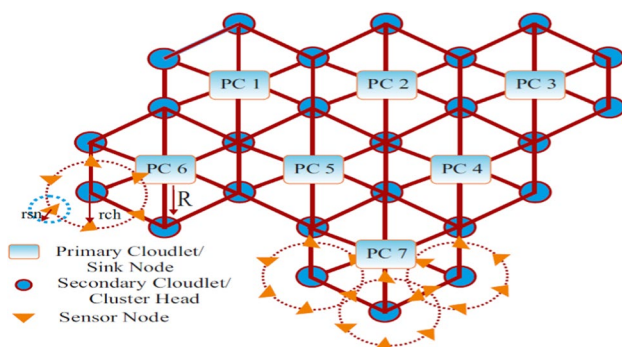


Fig. 5 Hexagonal layout of SE-SKYR framework with primary cloudlet, subordinate secondary cloudlet and affiliated sensors

cloudlets marked in blue circles. Now, these six vertices of hexagon are acting as a center for circle at which secondary cloudlet lies and sensor nodes lie on the circumference of these circles and these sensor nodes are marked in orange colour. These hexagon and circle follow the radius as given in Fig. 4 and communicate according to their radius range.

Therefore, in this hexagonal layout three-layer communication is followed, first layer is sensor node, intermediate layer is of secondary cloudlet which acts as cluster head and top layer is for primary cloudlet which is acting as a sink or destination node. Secondary cloudlet is transient device which can go online and offline as per its convenience. So, primary cloudlet is assigned to the newly available secondary cloudlet to replace the former and manage the sensing of data without any disruption. There is no relay node to send the transmitted data to the cloud and primary cloudlet and thus will substantially reduce the transmission time. In this proposed architecture, relay node does not fit to find its place as secondary cloudlet are very close to sensor node and it will manage its working and activity very closely. Figure 5 shows the data flow which originates from sensor nodes (acting as source) and goes to primary cloudlet (acting as sink node at local level). In between these two, data flows via secondary cloudlet which refines and optimizes the data and forwards it to the primary cloudlet for qualitative analysis and storage at local level. Also, it may be forwarded to cloud for permanent storage if required by the computing system. In this proposed work, primary cloudlet is considered as sink node because it can also act based on the sensed data and can store it as well for future use. Hence, in this architecture relay node is removed which improves the performance of the system on energy and efficiency parameters. Complete working architecture of SE-SKYR framework is shown in Fig. 6.

Figure 6 shows an improved version of SKYR framework [45]. In this improved version, primary cloudlet, secondary cloudlet and sensor nodes are arranged as per the hexagonal layout as shown in Fig. 5 which integrates the cluster-based scheme with SKYR framework. Notation of various stakeholders are mentioned in Fig. 6 are same as already explained in Figs. 2, 3, 4 and 5. Besides these stakeholders, there is one more entity marked in green circle, which represents the member node which has the potential to become cluster head if the existing cluster leaves the computing system. So, in that case, primary cloudlet in particular location looks for secondary cloudlet to replace the left one. In the proposed framework, sensor nodes and member nodes are added as compared to SKYR framework [45]. The proposed work includes the energy module and sensing module which improves the energy optimization in sensing task. These modules are added as application VM of respective secondary cloudlet under its proxy VM in primary cloudlet. SEOR algorithm facilitates these sensing and energy module functionalities. Rest of the

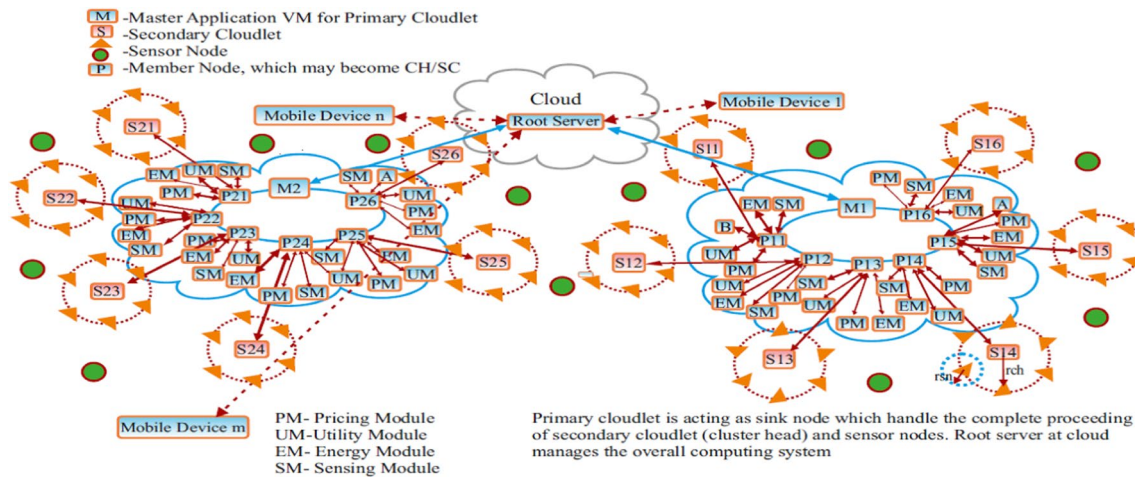


Fig. 6 Complete layout of SE-SKYR framework for cloudlet-based sensing enabled mobile cloud computing

working is same as SKYR framework [45]. The logical working flow of SE-SKYR framework is shown in Fig. 7. This proposed SE-SKYR architecture resolves the problem 1, 2 and 5 (mentioned in Sect. 3.2) of ESDWSN framework and its EAO algorithm incorporation with SEOR algorithm is discussed in next section.

In Fig. 7, green marked item represents the proposed work, rest of the working of SE-SKYR framework is same as of SKYR framework. In this framework, task is divided into two categories, one is user requested task and other is sensor based primary cloudlet-initiated task. This proposed work focused on sensor-based task. In the sensor-based task, SEOR algorithm and its related energy and sensing modules get integrated with the existing work flow. The detail flow of SEOR algorithm is shown in Fig. 8.

4.2 Energy optimization using SEOR algorithm

The main objective of the proposed work is to reduce the energy consumption and to improve the efficiency of the complete system (specifically sensing of data). So, to accomplish this goal, proposed work introduces scalable energy optimization of resources (SEOR) algorithm. This algorithm comprises of five procedures which are discussed in later sections. It also reduces the complexity in whole sensing process in comparison to EAO algorithm of ESDWSN framework despite achieving energy efficiency. SEOR algorithm comprises of four steps which are shown in Fig. 8. These steps are discussed in detail in the next section.

4.2.1 Selection of cluster head from set of heterogeneous nodes

The mobile cloud computing and mobile edge computing use heterogeneous mobile devices for computing. When

these devices support sensing-based computing then it makes a heterogeneous wireless sensor network. A system model of partitioned heterogeneous wireless sensor network is shown in Fig. 4 [46]. In this, sink node lies at center of the circle having radius ' R ' and source node signal strength radius is represented by ' r_{sn} '. Therefore, to have communication between them, ' r_{sn} ' must lie within the range of ' R ' as shown in Fig. 4. In this figure, the unreachable area is marked with gray colour. Cluster head node manages the data collection from sensor node or source node and transmits to sink node via relay node. These cluster heads are resource rich sensor enable devices which facilitate the task of management and controlling of sensor nodes and relay nodes. This type of sensor network can be used for noise, humidity, temperature, and light intensity etc. Selection of relay node is based on the bit error rate (BER) threshold factor. Nodes with low BER are chosen as relay nodes. If any relay node is getting off then new node is placed such that partitioned HWSN will remain connected. In [46], rectangular area represents the sink node and its associated cluster head and sensor node. In the proposed work, it is changed to hexagonal layout to make it more suitable for SKYR framework. The sink node receives, analyses, and processes the data as per the requirement. Sink node detects the flow of data traffic and hence if any sensor node fails or data flow is lower than the threshold level then it detects this as failure. Therefore, this sink node considers the hexagonal area to sense the data from sensor node because in hexagonal area the sensor node and cluster head are well within the range and nearly equidistant from each other. All cluster heads are equidistant from their primary cloudlet by ' R ' and sensor node will be equidistant from its respective cluster head (CH) by distance of r_{ch} .

This same concept of sink node and sensor nodes are applied with hexagonal area in the proposed work and

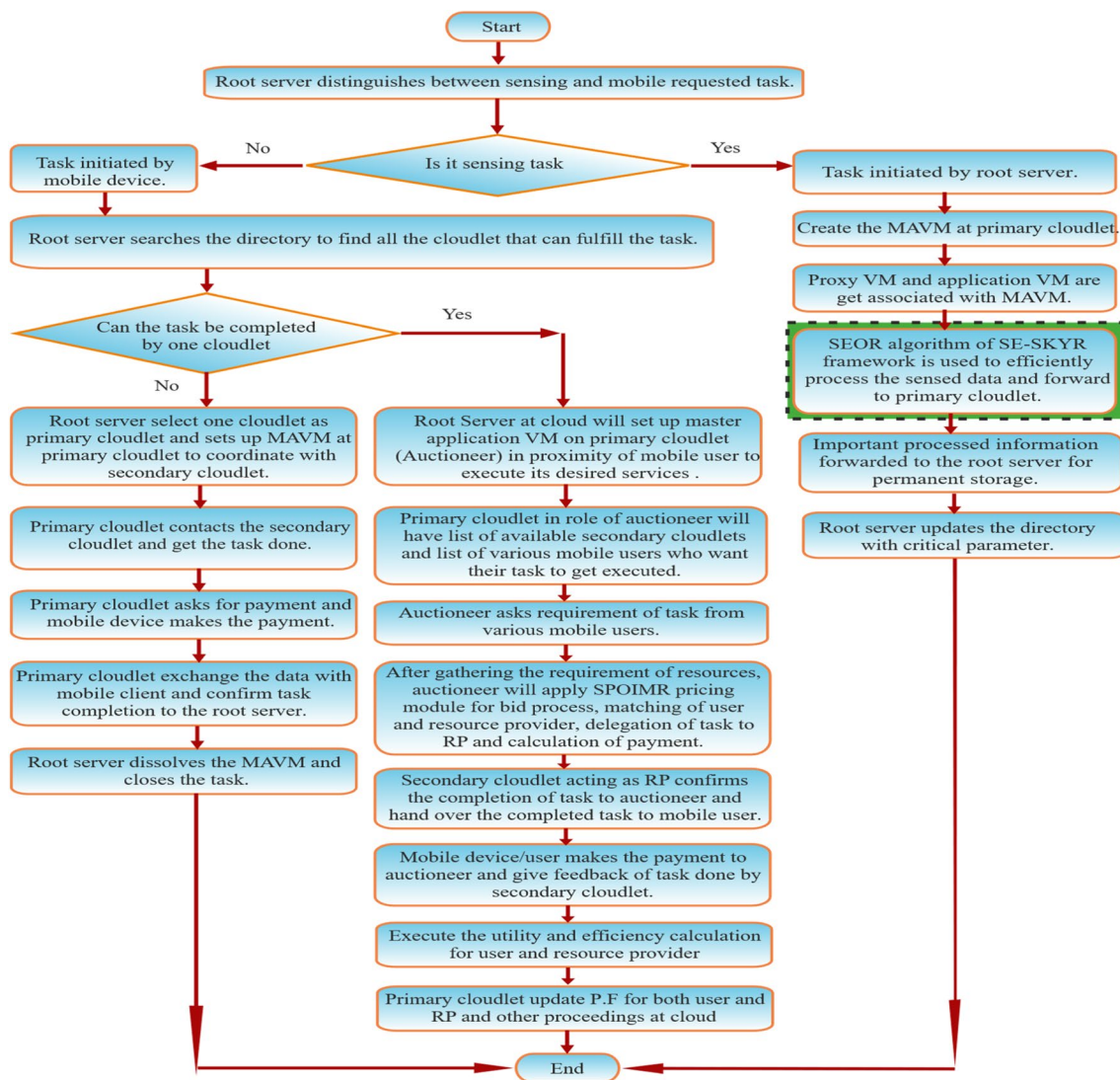


Fig. 7 Complete working architecture of SE-SKYR framework

multiple relay nodes are avoided to reduce the transmission cost. Only one hop communication is used to make the sense data transmission error free and time efficient. A detailed cluster scheme for wireless sensor network is discussed by the authors in [47]. Cluster based network architecture has various advantages such as:—it provides high network scalability irrespective of the number of sensor nodes deployed, it reduces communication overhead among nodes, it provides mechanism to handle the node failures and last but most important feature is that it reduces the number of hops from source node to sink node and hence saves the energy consumption. In this type of cluster scheme, all nodes are divided into various clusters and each cluster is assigned its head. This head is denoted as CH which receives data from each node within its cluster area and forwards the sensed data to the sink node

after some requisite processing. These cluster heads are same sensor node with better resources to manage some extra function and computation. There are two methods to choose the cluster head and these are static mode and dynamic mode. In static or fixed mode, cluster head is prefixed and it will remain same. On the other side, the dynamic mode of cluster selection is in which the selection of cluster node is based on consideration of several metrics like remaining energy of the node, distance from sink node, power backup, processing capability, signal strength etc. and these metrics are considered in the proposed work.

The proposed work uses coyote optimization algorithm and fuzzy logic system to optimize the energy consumption at various levels such as data processing, transmission, and storage at sink node. Coyote optimization is a meta heuristic algorithm which helps to optimize the CH selection and

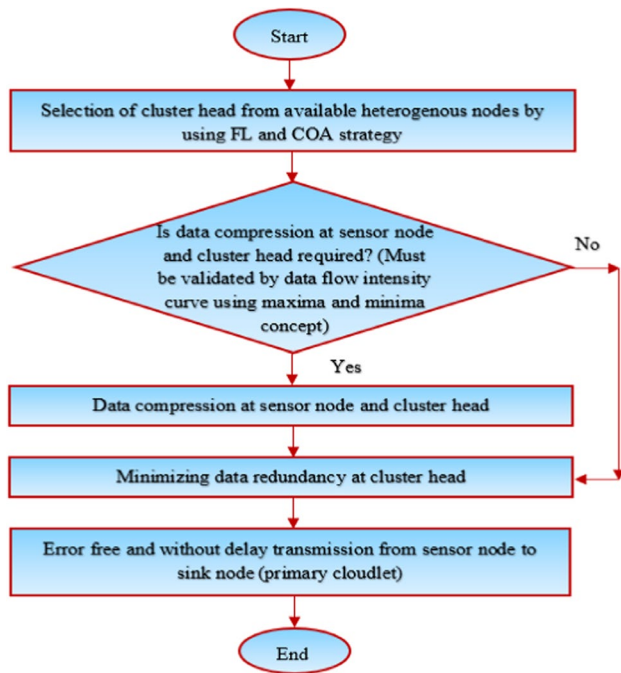


Fig. 8 Working of SEOR algorithm in SE-SKYR framework

cluster formation to prolong the network's lifetime, achieves energy efficiency and increases the network throughput.

The main benefits of Coyote optimization algorithm (COA) are its speed, smoothness and stable convergence as compared to other algorithms [25]. Fuzzy logic technique is very effective in machine learning, data mining and rain fall prediction [48]. In the proposed work, the FL system is used to balance the clustering process for maximizing the wireless network lifetime and reducing energy consumption. The combination of FL and coyote optimization are utilized in the proposed work and collectively it is named as fuzzy logic analysis and coyote optimization (FLACO) scheme. This FLACO procedure is executed in two phases. First phase is application of fuzzy logic and the second phase is Coyote optimization application.

A. Fuzzy logic phase: In this phase, the initial set of cluster heads are selected based on the parameters of *RE*, *DPC*, *ND* and ' ω '. These parameters are dynamic in nature and

collected in real time scenario. These parameter values are used as input to the system and generate the output as probability values. Probability value of these four parameters of CH's are shown in Table 1. In this fuzzy logic system, if-then rules are applied on the calculated probability value and those nodes are selected as CH's which have higher probability to act as cluster head. This if-then rule is summarized in Table 2. These probability values of Table 2 help to pick the right potential cluster head.

4.2.2 Consideration of residual energy (RE) and distance between source to cluster head and cluster head to sink (DBS)

In Fig. 9, source node is denoted by S, destination node or sink node is denoted as D and cluster head which also acts as relay node between sink and source node is denoted as CH. Therefore, the distance between S & D is denoted as d_{sd} , distance between S and CH is denoted as d_{sch} and distance between cluster head and sink node is represented as d_{chd} . These distances follow the mentioned Eqs. 2, 3 and 4 which are correlation with Fig. 4 and Fig. 9.

$$d_{sch} \leq r_{sn} + r_{ch} \quad (2)$$

$$d_{chd} \leq r_{ch} + R \quad (3)$$

$$d_{sd} \leq R + 2r_{ch} + r_{sn} \quad (4)$$

At any time t , the data signal received by CH is:

$$y_{sch}(t) = \sqrt{P_A} h_{sch} x(t) + n_{sch} \quad (5)$$

In Eq. 5, P_A denotes transmission power of S, h_{sch} represents channel coefficient from S to CH and n_{sch} is gaussian white noise. Similarly, when D receives the data signal, it is given by Eq. 6.

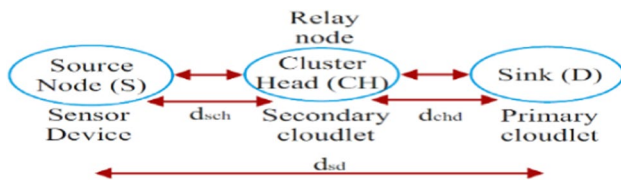
$$y_{sd}(t) = \sqrt{P_A} h_{sd} x(t) + n_{sd} \quad (6)$$

Table 1 Input variables and their linguistic values

Variables	Linguistics values		
	Less	Average	More
<i>Residual Energy</i>			
<i>Distance to PC</i>	Close Range	Medium Range	Long Range
<i>Node Density</i>	Scarce	Moderate	Heavy
<i>Bandwidth (ω)</i>	Low	Medium	High
<i>Probability of CHs</i>	Very Less (VL)	Slightly Moderate (SM)	Little Strong (LS)
	Less (L)	Moderate (M)	Strong (S)
	Slightly Less (SL)	Very Moderate (VM)	Very Strong (VS)

Table 2 Fuzzy logic rule table

Rule	ω	RE	ND	DPC			Probability		
				1	2	3	1	2	3
1	Low	Less	Scarce	Long Range	Medium Range	Close Range	VL	L	SL
2	Low	Less	Moderate	Long Range	Medium Range	Close Range	L	SL	SM
3	Low	Less	Heavy	Long Range	Medium Range	Close Range	SL	SM	M
4	Low	Average	Scarce	Long Range	Medium Range	Close Range	L	SL	SM
5	Low	Average	Moderate	Long Range	Medium Range	Close Range	SL	SM	M
6	Low	Average	Heavy	Long Range	Medium Range	Close Range	SM	M	VM
7	Low	More	Scarce	Long Range	Medium Range	Close Range	SL	SM	M
8	Low	More	Moderate	Long Range	Medium Range	Close Range	SM	M	VM
9	Low	More	Heavy	Long Range	Medium Range	Close Range	M	VM	LS
10	Medium	Less	Scarce	Long Range	Medium Range	Close Range	L	SL	SM
11	Medium	Less	Moderate	Long Range	Medium Range	Close Range	SL	SM	M
12	Medium	Less	Heavy	Long Range	Medium Range	Close Range	SM	M	VM
13	Medium	Average	Scarce	Long Range	Medium Range	Close Range	SL	SM	M
14	Medium	Average	Moderate	Long Range	Medium Range	Close Range	SM	M	VM
15	Medium	Average	Heavy	Long Range	Medium Range	Close Range	M	VM	LS
16	Medium	More	Scarce	Long Range	Medium Range	Close Range	SM	M	VM
17	Medium	More	Moderate	Long Range	Medium Range	Close Range	M	VM	LS
18	Medium	More	Heavy	Long Range	Medium Range	Close Range	VM	LS	S
19	High	Less	Scarce	Long Range	Medium Range	Close Range	SL	SM	M
20	High	Less	Moderate	Long Range	Medium Range	Close Range	SM	M	VM
21	High	Less	Heavy	Long Range	Medium Range	Close Range	M	VM	LS
22	High	Average	Scarce	Long Range	Medium Range	Close Range	SM	M	VM
23	High	Average	Moderate	Long Range	Medium Range	Close Range	M	VM	LS
24	High	Average	Heavy	Long Range	Medium Range	Close Range	VM	LS	S
25	High	More	Scarce	Long Range	Medium Range	Close Range	M	VM	LS
26	High	More	Moderate	Long Range	Medium Range	Close Range	VM	LS	S
27	High	More	Heavy	Long Range	Medium Range	Close Range	LS	S	VS

**Fig. 9** Consideration of distance and communication among source node, cluster head and sink

Here, h_{sd} represents the channel coefficient from S to D and n_{sd} denotes the gaussian white noise. Similarly, data signal from CH to D may be represented by Eq. 7.

$$y_{chd}(t) = \sqrt{P_{CH}} h_{chd} x(t) + n_{chd} \quad (7)$$

where P_{CH} represents the transmission power of CH, h_{chd} denotes channel coefficient from CH to D and n_{chd} depicts Gaussian white noise. So, in the proposed system N_s denotes the number of source sensor devices, N_{CH} represents the

number of cluster node which is acting as relay node as well, C_{CH} depicts the cost of relay of data by cluster head, E_{SCH} denotes the energy consumption from sensor node to cluster head and E_{CHD} represents the energy consumption from cluster head to destination node (sink node). In the proposed work, it is assumed that all clusters have the same cost. So, E_{SCH} and E_{CHD} is dependent on distance between nodes and data transmission load. Therefore, system model for energy consumption can be represented by Eq. 8 [47].

$$\begin{aligned} & \text{Min } C_{CH} N_{CH} + E_{SCH} + E_{CHD} \\ & \text{s.t } CH \in \text{reachable \& live node,} \\ & d_{sch} \leq r_{sn} \& d_{chd} \leq R \end{aligned} \quad (8)$$

Therefore, the system complies with Eq. 8 and hence the cost of relay nodes and network energy consumption in the system will remain minimum and fulfils the objective. Minimization of cluster head can be achieved by minimizing the cost of cluster head as each relay has same cost. Therefore, assuming energy consumption same for a unit distance, system model Eq. 8 becomes Eq. 9 [47].

$$\begin{aligned}
 &N_{CH} + d_{sch} + d_{chd} \\
 &s.t \ CH \in \text{reachable \& live node}, \\
 &dsch \leq r_{sn} \& d_{chd} \leq R
 \end{aligned} \quad (9)$$

Therefore, with the application of Eq. 8 and 9, the number of cluster heads and distance between source node to cluster head and cluster head to destination node can be minimized. So, with the application of Eq. 8 and Eq. 9 and the proposed concept, the $R.E$ value gets calculated for all member nodes which may get selected as cluster head to manage the sensor nodes. The calculated value gets updated which helps fuzzy logic and coyote optimization to select the best potential cluster head.

b. Coyote optimization phase: Next phase in implementation is Coyote optimization. The initial set of chosen CHs are used here and acting as input for this phase. While applying the coyote optimization concept, following representations are used:—clusters are represented as pack, a cluster head as an alpha and a normal node is represented as coyote. Remaining normal nodes are acting as candidate nodes. If any cluster head leaves in between the computation due to any reason, next cluster head will be chosen from the set of normal nodes. This fuzzy logic and coyote optimization is executed in combination to select the cluster head (for a cluster) and helps in the communication between CH and sensor node within the cluster and also facilitates communication among different clusters. Therefore, this helps inter and intra cluster communication. The combination of FL and coyote optimization concept is named here in the proposed work as FLACO sub procedure. This FLACO procedure enables the effective and energy efficient selection of cluster head and its complete management during the entire data transmission process. In the proposed work, one hop or direct transmission is adopted, it means there is a direct communication between sensor node and the cluster head (that is secondary cloudlet).

The distance between cluster head and sensor node is kept as small as possible so that D^2 model or free space transmission model can be applied. If distance is large or multi hop communication is used then multipath model or D^4 model is used. Therefore, energy consumption is one of the parameters in the selection of cluster head. Hence, energy consumption between cluster head and sensor node can be computed by transmitting I bits/packages as defined by Eq. 10 [49]. Here D is distance between any two transmitting and receiving nodes.

$$E_{tx}(I, D) = \begin{cases} I * E_{elec} + I * e_{fs} * D^2, & \text{if } D < D_0 \\ I * E_{elec} + I * e_{amp} * D^4, & D \geq D_0 \end{cases} \quad (10)$$

Similarly receiving energy can be defined by Eq. 11

$$E_{RX}(I, D) = I * E_{elec} \quad (11)$$

In Eq. 10 and 11, I represent the size of the data packets while E_{elec} represents energy used per bit. D_0 denotes the threshold distance which determines whether to use e_{amp} or e_{fs} , where e_{fs} is the free space energy model and e_{amp} is multipath energy model. The threshold distance D_0 is represented by Eq. 12.

$$D_0 = \sqrt{\frac{e_{fs}}{e_{amp}}} \quad (12)$$

The prime objective of FLACO procedure is to increase the network lifetime by balancing the energy consumption in the network. The proposed FLACO procedure utilizes coyote optimization with an adequate preliminary population set to connect to the ideal solution in an appropriate time period. In the process of optimization, COA strategy facilitates balance between exploitation and exploration in the procedure. In this procedure coyotes chase their prey in the packs. In the hunting process, each pack is guided by an alpha male which uses infiltration method. In the proposed strategy, multiplication of N_p (number of packs) and N_c (number of coyotes) defines the population size, which acts as a significant potential solution for optimization problem. While starting the coyote optimization process, coyotes are randomly assigned to the packs. Each coyote represents a single solution to the problem $U = (U_1, U_2, U_3, \dots, U_{PD})$ in which problem dimension is denoted by P_D . Each Coyote is started with a random position solution at the start of the algorithm and this can be represented by Eq. 13 [25].

$$U_{c,j}^{p,t} = lb_j + r_j(ub_j - lb_j) \quad (13)$$

In Eq. 13, r_j is a random number in a range $[0,1]$ and lb_j and ub_j represent lower and upper bound for the search space and $j \in (P_1, P_2, \dots, P_D)$.

In the next step, the coyote adopts to the environmental settings and fitness function can be represented as Eq. 14 [25].

$$ff_c^{p,t} = f(U_c^{p,t}) \quad (14)$$

As the problem is a minimization problem, the alpha coyote of each pack can be defined by Eq. 15 [25].

$$\alpha^{p,t} = \left\{ U_c^{p,t} \mid \arg_{c=\{1,2,\dots,N_c\}} \min(f(U_c^{p,t})) \right\} \quad (15)$$

The new social condition of the coyote is updated by Eq. 16 [25].

$$U_{c,j}^{p,t} = U_c^{p,t} + r_{j1} \delta_{j1} + r_{j2} \delta_{j2} \quad (16)$$

In Eq. 16, the distance between any random coyote in the pack is alpha (denoted by δ_{j1}) and the distance between a single coyote and the average location of all coyote in a pack

is represented by δ_{j_2} . The random numbers are denoted by r_{j_1} and r_{j_2} which lie within a range $[0,1]$. Now, examine the potential of the new solution and assess the fitness function of this solution by using Eq. 17 [25].

$$newff_c^{p,t} = f(newU_c^{p,t}) \quad (17)$$

Now, the coyote decides whether the new social condition is superior than the previous one to retain or reject by Eq. 18 [25].

$$U_{c,j}^{p,t+1} = \begin{cases} newU_c^{p,t}, & newff_c^{p,t} < ff_c^{p,t} \\ U_{c,j}^{p,t}, & otherwise \end{cases} \quad (18)$$

Further, coyote optimization strategy considers the birth and death of coyote and its computation is dependent on the amalgamation of the social conditions of two packets and an environmental component as denoted by Eq. 19 [25].

$$pup_j^{p,t} = \begin{cases} U_{r1,j}^{p,t}, & rn_j < P_s \text{ or } j = j_1 \\ U_{r2,j}^{p,t}, & rn_j \geq P_s + P_a \text{ or } j = j_2 \\ R_j, & otherwise \end{cases} \quad (19)$$

where, $P_s = 1/D$ represents the scatter probability and $P_a = \left(\frac{1-P_s}{2}\right)$ denotes the association probability. In the p^{th} pack, r_{j_1} and r_{j_2} denote the random coyotes and, j_1 and j_2 denotes two random dimensions of the problem. R_j is a random number inside the decision variable bound of the j^{th} dimension and random number in the range of $[0,1]$ is denoted by rn_j . Therefore, by following the above condition, the pup will live if the fitness value is less than the previous one or else it will die.

The FLACO procedure uses the above discussed coyote optimization strategy and its equations to select the best cluster head which will enhance the lifetime of the network.

As discussed, the FLACO procedure is based on two optimizations, firstly it applies fuzzy logic system and second it uses coyote optimization strategy. The former chooses the CH based on the FL system and then later is used to optimize it using the clustering options. The threshold function of the FLACO procedure which defines the minimum requirement to be fulfilled by any node to become the cluster head is given by Eq. 20 [50].

$$T.f() = A * f()_{intra-opt} + B * f()_{inter-opt} \quad (20)$$

where A and B are constants and $A + B = 1$. $f()_{intra-opt}$ represents the function which optimizes the distance between the cluster head and its members within the cluster to improve the intra-cluster communication is given by Eq. 21 [50].

$$f()_{intra-opt} = \frac{1}{N_s} \sum_{k=1}^{N_s} d(S_k, CH_m) \quad (21)$$

In Eq. 21, N_s represents the number of neighbour nodes of cluster m and $d(S_k, CH_m)$ denotes the distance between all the nodes in the cluster and its cluster head CH_m . Intra-cluster communication function $f()_{intra-opt}$ also depends on the transfer rate (T_r) with which sensor node sends data or communicate with the cluster head. T_r depends on the bandwidth and the number of nodes available in the cluster. This transfer rate can be represented by Eq. 22.

$$T_r = \left(\frac{d * \omega}{S_{n+1}} \right) \quad (22)$$

Here d is distance between CH and node. (S_{n+1}) denotes the number of nodes and ω denotes the bandwidth available with the system.

Similarly, inter-cluster communication can be improved by minimizing the distance between the primary cloudlet

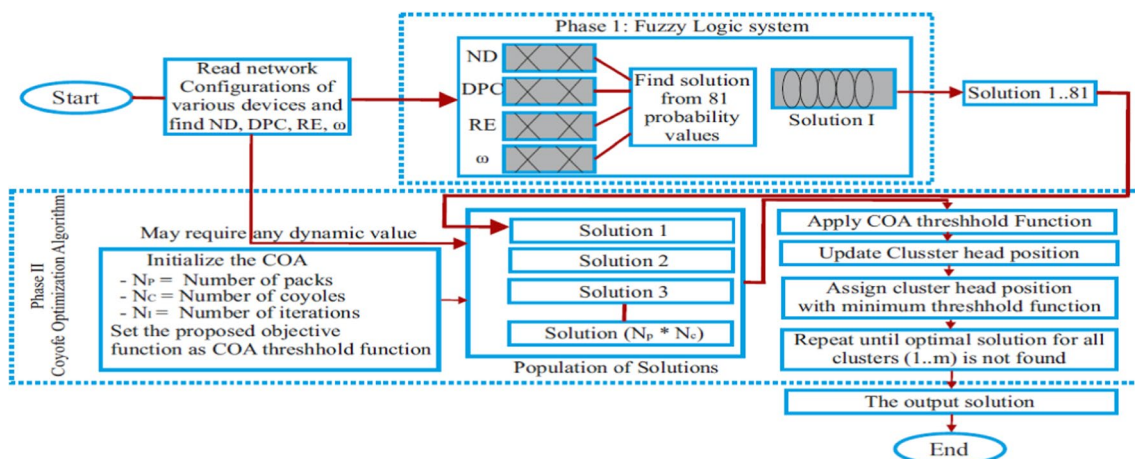


Fig. 10 Logical flow of FLACO procedure in SEOR algorithm

and the cluster head and, the average distance between each cluster head to enlarge the effective communication area of the sensor network. This function can be represented by $f()$ $_{inter_opt}$ as given in Eq. 23 [50].

$$f()_{inter_opt} = \frac{D}{\frac{1}{C} \sum_{m=1}^C \sum_{n=1}^C d(CH_m, CH_n) \quad n \neq m} \quad (23)$$

where C is the number of clusters, D is the average distance between the primary cloudlet and secondary cloudlet (cluster head) and $d(CH_m, CH_n)$ is the distance between secondary cloudlet m and n . So, based on the threshold requirement, coyote optimization method selects the best CH in each iteration from the set of population of CH's. Inter-cluster communication depends on the speed with which data communication occurs between CH's and, CH & Primary Cloudlet. $f()_{inter_opt}()$ depends on transfer rate T_r and this can be denoted by Eq. 24.

$$T_r = \frac{(D_{ch-ch}) * \omega}{C} + \frac{(D_{sk-ch}) * \omega}{SK + C} \quad (24)$$

Here, D_{ch-ch} represents the distance between two secondary cloudlet which is acting as cluster head, ω denotes the bandwidth, C represents the number of cluster heads within the coverage of sink node or primary cloudlet, D_{sk-ch} depicts the distance between sink node and its associated cluster heads and SK represents the number of sink nodes. So, after application of fuzzy logic and coyote optimization strategy, suitable cluster heads are selected for various clusters ($C_1, C_2, \dots, C_m$). This step of SEOR algorithm solves the problem 1, 2 and 3 of EAO algorithm of ESDWSN framework. Logical flow of FLACO procedure is shown in Fig. 10.

Selection of cluster head by using FLACO procedure
FLACO procedure ()

Requirements: List of nodes and its parameters.

Objectives: Find the best suited CH's.

```

Step
1: Begin
2:  $\forall CH(1,2,\dots,m)$ , P.C nodes,  $\rightarrow$  read its parameter
3: Phase 1: Apply fuzzy logic, objective is to find the probable list of CH's
4: Set FL parameters R.E, DBS, ND,  $\omega$ 
5:  $\forall$  nodes apply fuzzy logic based on table 1 and if-then rule of table 2
6: Return probable CH's ( $CH_1, CH_2, CH_3, \dots, CH_m$ ) selected from set of nodes.
7: Phase 2: Apply coyote optimization method to find list of best suited CH's
8: Set  $N_p$  pack,  $N_c$  coyote and max iteration (Max), number of iteration ( $N_i$ )
9: Set coyote adaption as defined in equation 14
10: While  $N_i \neq \text{Max}$ 
11: {
12:   For  $N_p = 1$  to  $n$ 
13:   {
14:      $\alpha^{p,t} = \{U_c^{p,t} \mid \arg_{c=\{1,2,\dots,N_c\}} \min f(U_c^{p,t})\}$ 
15:     {
16:       For  $N_c = 1$  to  $m$ 
17:       {
18:         Coyote position  $= U_c^{p,t} + r_{j1} \cdot \delta_{j1} + r_{j2} \cdot \delta_{j2}$ 
19:         Threshold function ( $T.f()$ )
            $= A * f()_{intra\_opt} + B * f()_{inter\_opt}$ 
20:         New coyote condition
            $U_{c,j}^{p,t+1} = \begin{cases} \text{new } U_c^{p,t}, & \text{new } f f_c^{p,t} < f f_c^{p,t} \\ U_{c,j}^{p,t}, & \text{otherwise} \end{cases}$ 
21:       }
22:     }
23:     Updated birth of new coyote and death of old coyotes
            $pup_j^{p,t} = \begin{cases} U_{r1,j}^{p,t}, & rn_j < P_s \text{ or } j = j_1 \\ U_{r2,j}^{p,t}, & rn_j \geq P_s + P_a \text{ or } j = j_2 \\ R_j, & \text{otherwise} \end{cases}$ 
24:   }
25:   End For
26:   Set  $P_e = 0.005 (N_c)^2$ 
27:   Update coyote age
28: }
29: End While
30: Return list of best CH's

```

4.2.3 Data compression and data redundancy

This cluster-based scheme uses period collection to send data from sensor to cluster head and sends data from cluster head to the sink using index bit encoding (IBE) compression scheme. This compression reduces the size of the data to be transmitted. At the end of each period, the cluster head receives the sensed data of all sensor nodes. The CH applies sensor level algorithm to process the data before sending it to the sink node for storage and analysis for future use [47]. Therefore, the proposed work uses modified IBE and sensing rate adjustment (SRA) at the cluster level which will save energy and reduces the size of the data.

The cluster-based scheme is used in the proposed work. Sensor node collects data periodically and transmit it to the

cluster head. But before sending it to cluster head, compress it using IBE method. Therefore, after receiving data from every sensor node after each periodic cycle it is processed by SRA method and then forward it to the primary cloudlet which is acting as a sink node at local level. The proposed work focuses especially on energy consumption and communication overhead.

4.2.3.1 a. Data compression strategy Network of sensor node can be represented using organized connected graph $G=(N,A)$, where $N=\{N1, N2, \dots, Nn\}$ is a collection of sensor nodes and ‘A’ represents the arc which connects the sensor nodes. The data collection by sensor node can be mathematically represented by Eq. 25.

$$Q_i^t = \{q_i\} \quad (25)$$

where sensor node S_i collects a set of data reading i during the time interval t and forms a vector Q_i^t . Here, r_1^t represents the sensor reading of sensor node i at time interval 1.

The proposed work focusses on considering the similarity of collected data. It focuses on the similarity of collected data which is inversely correlated to the rate of variation in the condition of monitored target. It means that bigger the rate of variation in the condition is, the lower the level of similarity and vice versa. It can also be considered that, when the condition is stable, the rate of variation would be very small which implies high level of similarity and hence there will be high level of redundancy in the sensed data. SRA method is used in removal of redundancy which is discussed later in this section. The proposed work focuses on compression of raw data using IBE, which does not comprise the temporal information of collected data of sensor node. Similarity function (denoted as Δ) is used to represent the similarity between two collected data readings. Similarity function $\Delta_{q_j^i, q_k^i}$, where q_j^i, q_k^i are two readings collected by sensor node S_i and these two data readings are considered similar if and only if the difference between them is less than ‘ ϵ ’ and it can be represented using Eq. 26. Figure 11 shows the energy level of different types of data packets collected by various sensor during sensing process. These energy (e) levels help to distinguish the redundancy in collected data and its compression. These energy levels are marked in Fig. 11 as $e(1), e(2), e(3), e(4), \dots, e(l)$.

$$\Delta_{q_j^i, q_k^i} = |q_j^i - q_k^i| \leq \epsilon \quad (26)$$

where ϵ is determined by various dynamic parameters. Similarity function Δ , if applied leading to Q_i^t of i^{th} sensor node for a time interval t . This Q_i^t set of readings will have unique readings denoted as Unq_i^t in which every reading of former will match to only one reading of Unq_i^t and number of iterations and its place of occurrence will make the weight of

reading of Q_i^t . Therefore, the combination of unique readings and their weights form the compressed set of readings Q_i^t which is forwarded to the cluster head for processing. Working of this compression and decompression is shown in Fig. 12 [47].

The data of sensor nodes are collected as a sequence of multiples of 8 as it would corresponds to one-byte unique index as represented by Eq. 27.

$$unq_k^i \epsilon (Unq_i^t, b_k^i) \quad (27)$$

where b_k^i is a binary code with a size equals to the number of readings in Q_i^t . Therefore, reading unq_k^i in Unq_i^t for each q_j^i in Q_i^t is set to 1, if unq_k^i and q_j^i are similar corresponding bit in b_k^i , otherwise it is set to 0. Weighted code can be represented as $wgt(b_k^i)$. The code weight of b_k^i is the integer value corresponding to the code b_k^i . So, Q_i^t set of readings after the weighted code conversion can be represented as $Q_i^t = [(unq_1^i, wgt(b_1^i)), (unq_2^i, wgt(b_2^i)), (unq_3^i, wgt(b_3^i)), \dots, (unq_k^i, wgt(b_k^i))]$. The size of set of reading is set to be multiple of 8.

This sub procedure besides converting the set of readings to index bit encoding, also assures no data loss by acknowledging data transfer. Therefore, for safe transfer of data, it uses XOR operation in the 8-bit string of count number and checksum by adding the equation. These operations are also performed by the cluster heads to check whether the received data is correct or not. If the XOR operation gives all ‘1’ and checksum calculated after retrieving the data matches with the checksum sent along with the data, confirms that there is no data loss.

Data compression procedure ()

Requirements: Time duration:— T , similarity threshold: ϵ , set of readings of sensor in period: Q_i^t

Objective: Compressed reading sets of sensors in time duration ‘ T ’- Q_i^t and generate the checksum.

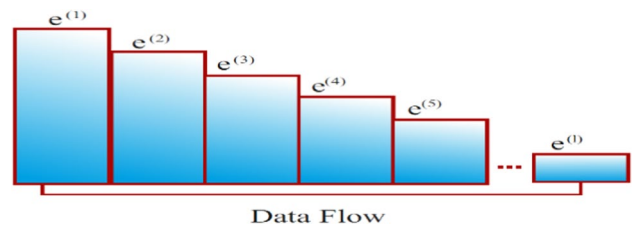


Fig. 11 Energy levels of various types of data packets during sensing process [20]

```

Step
1:   $Unq_i^t \leftarrow \phi$ 
2:  For  $k=1$  to  $T$ , read  $q_k^t \in Q_i^t$ 
3:  {
4:     $Sim=1 \leftarrow$  if, there exist  $unq_j^t \in Unq_i^t$  where
       $\Delta_{q_k^t, unq_j^t} \leq \varepsilon$ 
5:    If  $Sim \neq 1$ , then
6:    {
7:       $b_j^t = \{0, 0, \dots, k-1\}$ 
8:       $Unq_i^t = Unq_i^t \cup \{q_k^t, b_j^t\}$ 
9:    }
10:   End if
11:    $unq_j^t \in Unq_i^t$  where  $j \in [1, Count(Unq_i^t)]$ 
12:   For  $i=1$  to  $n$ 
13:   {do
14:     If  $sim=1$ 
15:     {
16:        $b_j^t = b_j^t \cup \{1\}$ 
17:     }
18:   }
19:   End for
20:    $Q_i^t = \phi$ 
21:   For  $i=1$  to  $n$ 
22:   {
23:      $unq_j^t \in Unq_i^t$  where  $j \in [1, Count(Unq_i^t)]$ 
24:      $wgt(b_j^t) \leftarrow Calculated\_Code\_Weight(b_j^t)$ 
25:      $Q_i^t = Q_i^t \cup \{unq_j^t, wgt(b_j^t)\}$ 
26:      $Chk.Sum^T = \sum_{i=1}^n (Unq_i^t, b_j^t)$ 
27:   }
28: End For
29: }
30: End For

```

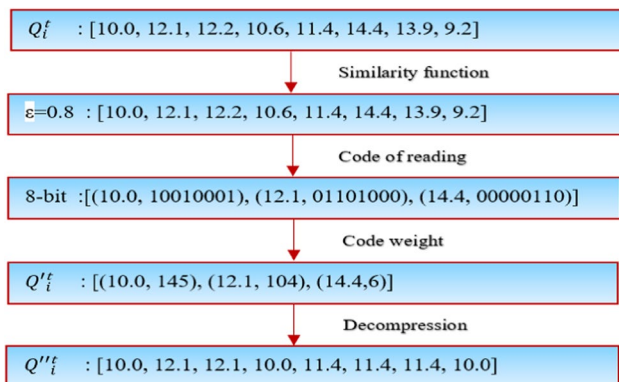


Fig. 12 Working of compression and decompression process through IBE using similarity threshold as $\varepsilon=0.8$

4.2.3.2 b. Compression using data flow intensity The proposed work considers a continuous generation of data by sensor node i for the mentioned time duration “ T ”. The data flow is continuous and hence its intensity flow follows some equation like sine wave or cosine wave. The same can be depicted from Fig. 13. The figure shows the continuous graph of sine wave equation, which depicts the intensity of data gathering (which varies with time and has maxima and minima). So, data intensity can be represented by using sine equation as.

$$f(x) = y(x, t) = A \sin(kx - wt + \phi) \quad (28)$$

Similarly, it can be represented with cosine equation as

$$f(x) = y(x, t) = A \cos(kx - wt + \phi) \quad (29)$$

Sine and cosine equations can represent any continuous data intensity behaviour. Here, in the proposed work, sine equation is considered. Keeping it as a reference, other factors are calculated and used. In the mentioned equation, A denotes the amplitude of the wave which is shown in Fig. 13. Other parameters are w , k and ϕ representing angular frequency, wave number and phase of sine wave respectively. Only amplitude is used for calculation in the proposed work and remaining parameters are just part of equation to represent the sine equation.

As we have generated the equation for the representation of intensity of data flow. Now, next parameter to find is maxima and minima for the curve is shown in Fig. 13. Therefore, to calculate the maxima and minima for any function $f(x)$, find the first and second derivatives of $f(x)$ and these are represented as $f'(x)$ and $f''(x)$ respectively. Now, perform the 1st derivative test and equating it with zero to get the critical point (x) where the continuous function $f(x)$ may have minimum or maximum value. The confirmation of minimum and maximum for function $f(x)$ at critical point ‘ x ’ can be assured by 2nd derivative. Conditional value for maxima and minima is shown in Table 3.

If the condition shown in Table 3 satisfies for maxima and minima, confirms that the function $f(x)$ has either maximum or minimum value at critical point. These maxima and minima points are marked in sine curve as shown in Fig. 13. Significance of these maxima and minima points is that, at maxima the intensity of data flow is maximum and at minima it is minimum. Therefore, after calculating the maxima and minima, we find the distance between maxima and minima and that is represented by Eq. 30.

$$h = \sqrt{A^2 + t^2} \quad (30)$$

where ‘ A ’ is amplitude and ‘ t ’ is time gap between maxima and minima. Now calculate the Q as cross over between $fn(x)$ and has marked in Fig. 13. Calculate the work load factor

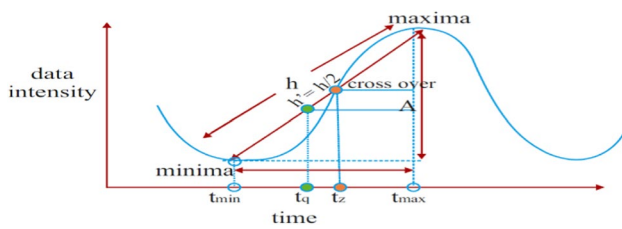


Fig. 13 Shows maxima and minima for a data intensity function

Table 3 Condition for maxima and minima for any function

Maxima	Minima
$F''(x)=0$	$f''(x)=0$
$F''(x)<0$	$f''(x)>0$

which considers the number of data source generator (sensor node) as given by Eq. 31.

$$Z = (A(1 - 1/n)) \quad (31)$$

where A is amplitude and n represent the number of sensor nodes. If n is very large, the work load is maximum and hence all generated data must be compressed. Now, calculate the t_q and t_z from Eq. 28 and put $x=Q$ and $x=x_{max}-Z$. This will give time t_q and t_z as shown in Fig. 13. Consider the minimum $f(t_q, t_z)$, to select the time line which is minimum in this range for data compression. If t_q is minimum, the data compression starts at t_q in $f(x)$, otherwise it starts at t_z on timeline. With this compression and decompression condition, time is saved. If there are very few numbers of data source generator, then compression is not required. But, if there are large number of data source generator then compression and decompression is needed. So, this dynamic approach saves time and manages the data properly.

In the proposed work, data compression occurs between sensor node to secondary cloudlet and secondary cloudlet to primary cloudlet. It means sensor nodes and CH apply this procedure to compress and decompress data. We can dynamically manage the rate of intensity of data and accordingly take the decision whether to compress (or decompress) the data or not. Therefore, this process can be applied at sensor to cluster head communication and cluster head to sink communication.

If IBE method applies to large period size, then it will be difficult to compress the data (or compression ratio will get affected). Therefore, to overcome this problem IBE can use two strategies: scaling down and period dividing [47]. In the first strategy, the collected readings are very large in number and hence these readings are reduced by averaging the successive readings till the number of readings are reduced to the desired level. Similarly, period dividing is

done into smaller blocks, so that compression ratio can be maintained. This data compression step of SEOR algorithm resolves the problem 4 of EAO algorithm.

Data flow intensity check procedure ()

Requirements: Check the flow of data and associate it by the equation to represent it.

Objective: Find the point of compression, t_z and t_q represents cross over point and half way point respectively.

Step		
1:	For $i=1$ to n and m of S_n & CH_m	/* n and m represent sensor node and cluster head respectively
2:	{	
3:	If $\frac{PF_i}{T} \geq PF \forall i \in (1, 2, \dots, n) \& (1, 2, \dots, m)$	/* PF is a potential factor for respective type of curve and it is predefined and assumed value.
4:	{	
5:	$\forall n$ and m	
6:	Find $f(x)$ and $f''(x)$ and its respective maxima and minima point.	
7:	Find the cross over point on curve and median of h line (slope).	
8:	Calculate its respective t_z and t_q Point in the time line	
9:	If $(t_z > t_q)$	
10:	{	
11:	t_q is selected as compression point	
12:	}	
13:	else	
14:	{	
15:	t_z is selected as compression point	
16:	}	
17:	}	
18:	}	

4.2.3.3 c. Data redundancy methodology Secondary cloudlet acting as a cluster head will perform the task of data redundancy on the data sent by various sensors. A cluster head is managing multiple sensor nodes, so there may be chances that the multiple sensing devices sense the same set of data and send it to the secondary cloudlet. Now, it's the responsibility of secondary cloudlet to remove the redundant data and process the unique data of various sensor node devices.

Let us assume that a period ‘ T ’ is divided into t equal time slots and represented by Eq. 32

$$T = [s_1, s_2, s_3, \dots, s_t] \quad (32)$$

So, after the end of every period, the cluster head is receiving data from the sensor node devices and these data may have specific level of redundancy due to spatial temporal correlation between them. Therefore, in order to lower the transmission of these redundant data from member node devices, the CH would adapt the sensing frequencies of the sensors in order to curtail the redundancy. Sensor node devices send the data in the form of compressed set of readings. The reading can be represented as Q_i^t which is sent by node n_i . So, these set of readings need to be decompressed using reverse IBE (or index bit decoding). Therefore, Q_i^t when decoded will be represented as $Q_i'^t$ which is equivalent to Q_i^t as given in Eq. (33)

$$Q_i''^t \approx Q_i^t \quad (33)$$

Then this decompressed set of readings can be vertically stacked to form a matrix M^T at each period T as given in Eq. 34. This means matrix will be generated after every period.

$$\begin{pmatrix} Q_1^T \\ Q_2^T \\ \vdots \\ Q_t^T \end{pmatrix} = \begin{pmatrix} s_1 & s_2 & \dots & s_t \\ q_1^1 & q_2^1 & \dots & q_n^1 \\ q_1^2 & q_2^2 & \dots & q_n^2 \\ \vdots & \vdots & \dots & \vdots \\ q_1^t & q_2^t & \dots & q_n^t \end{pmatrix} \quad (34)$$

Each row Q_k^T for $k \in t$ is a set of readings of n nodes at any instant of time t . Therefore, the prime objective is to learn the redundancy existing in M^T and based on it, transform the sensing frequency of all nodes in the cluster for the upcoming period $T+1$ in order to reduce the assembly of redundant data. Therefore, to control the redundancy level of node, cluster head applies the T -test statistics and adapts to the sensing frequencies of the sensors. While performing the redundancy test, the cluster head collects the set of readings for time slot t^i and t^{i+1} , matches to check the similarity and based on this similarity check, it calculates the redundancy rate for the period T as RR_T . This test is called as paired T -test as it is applied on the successive set of readings of paired observation by statistical definition. The true mean difference between paired observation is zero, considered as null hypothesis of this T -test and its alternative hypothesis states that there is a significant difference.

Therefore, for T -test, assume the set of readings of all nodes during a time slot ‘ t ’ as $Q^t = [q_1^t, q_2^t, q_3^t, \dots, q_n^t]$

where q_i^t is a reading collected by node N_i and there are N number of nodes available to collect the readings. Similarly, set of readings for time slot $t+1$ is also collected and represented as Q^{t+1} . Now, these successive set of readings form the pair sets for T -test, which can be represented by Eq. 35 [47].

$$T - test (Q^t, Q^{t+1}) = \frac{\bar{x}^{t,t+1} - \mu}{\frac{\sigma^{t,t+1}}{\sqrt{n}}} \quad (35)$$

where $\bar{x}^{t,t+1}$ is the mean of difference set $\{Q^{t+1} - Q^t\}$, μ is the population mean complete matrix M^T . $\sigma^{t,t+1}$ is the standard deviation for difference set $\{Q^{t+1} - Q^t\}$. Thus, the pair of reading Q^t and Q^{t+1} are considered to have significant difference if the value of the T test is less than a threshold value defined as ‘ θ_t ’ and it can be represented as Eq. 36.

$$T - test (Q^t, Q^{t+1}) \leq \theta_t \quad (36)$$

where θ_t is the T -test value for false rejection probability risk $\alpha \in [0,1]$. So, based on this threshold value and T test, the set of readings which do not have significant difference are taken as similar readings. After calculating the similarity between two set of readings, finally redundancy rate for the complete period ‘ T ’ is calculated by Eq. 37 as:

$$RR_p = \frac{\sum_{i=1}^{t-1} T - test (Q^t, Q^{t+1}) \leq \theta_t}{t-1} \quad (37)$$

Significance of this RR_p is that, it lies between 0 and 1. ‘0’ signifies no redundancy and ‘1’ represents maximum redundancy. The value of α impacts the redundancy rate to significant level, the higher it is, the greater the tolerance is to the difference between sets. Changes in the redundancy rate is dynamically managed by the sink node which is primary cloudlet in the proposed work and the same must be followed by the cluster head. This data redundancy procedure of SEOR algorithm elucidate the problem 3 of EAO algorithm.

Data redundancy procedure ()

Requirement: $Q_k''^T \approx Q_k^T \forall t \in \forall T$

Objective: $RR_p \approx 0 \forall T \forall N$

```

Step
1: Calculate  $Q_k^T \approx Q_k^T \forall t > \forall T$ 
2: For  $T = 1$  to  $Z$ 
3: {
4:   For  $n=1$  to  $N$ 
5:   {
6:     For  $t=1$  to  $t_n$ 
7:     {
8:       Find  $Q^t, Q^{t+1} \forall$  set of reading of  $Q_k^T$ ,
9:       Where  $t \in T$ ,
10       $Q^t = [q_1^t, q_2^t, \dots, q_n^t]$ 
11       $Q^{t+1} = [q_1^{t+1}, q_2^{t+1}, \dots, q_n^{t+1}]$ 
12      Calculate similarity test for  $Q^t, Q^{t+1}$ 
13       $T\text{-test}(Q^t, Q^{t+1}) = \frac{\bar{x}^{t,t+1} - \mu}{\frac{\sigma^{t,t+1}}{\sqrt{n}}}$ 
14      If  $(T\text{-test}(Q^t, Q^{t+1}) \leq \theta_t)$ 
15:       Not similar
16:     else
17:       Similar  $Q^t$  and  $Q^{t+1}$ 
18:     }
19:   End For
20: }
21: End For
22: Calculate redundancy rate  $RR_p$  for  $\forall T$ 
23:  $RR_p = \frac{\sum_{i=1}^{t-1} T\text{-test}(Q^i, Q^{i+1}) \leq \theta_t}{t-1}$ 
24: If  $(RR_p \geq 0.6)$ 
25: {
26:   Redundancy Rate is High;
27:   Reduce the distance between nodes
   (member) and adjust frequency, so that it
   does not overlies with others
28: }
29: Elseif  $(0.3 \leq RR_p \leq 0.6)$ 
30: {
31:   Redundancy rate is manageable, but must
   be reduced
32: }
33: Else
34: {
35:   Very low redundancy, its working fine
36: }
37: }
38: End for

```

4.2.4 Error free and reduced delay transmission from sensor node to sink node

To transfer sensed data from source node to sink, there is a need to have cluster head which removes the redundant data and processes it and, then transmit it to the sink node for

storage and future use. Although the proposed work uses fuzzy logic and coyote optimization to select a best device to act as cluster head but there may be a situation when any cluster head may fail due to some unforeseen reason be it a battery failure or any. So, in that case there must be some mechanism and strategy available to replace the failed cluster head with new cluster head for edge area as shown in hexagonal format in Fig. 5. Consequently, it selects the new cluster head which also acts as one hop connection between the source node and sink node. Therefore, there must be some devices to be kept as backup that can act as cluster head in case if any existing cluster head fails to facilitate its services.

Therefore, to select the cluster head to act as relay node in one hop connection between source node and sink node, whale optimization (WO) method is used in this proposed work [46]. The social behaviour of hump back whales is mimicked by swarm intelligence (SI) method. In the same way, bubble net hunting strategy is used by WO method. The target prey strategy is considered the best solution under WO method. Therefore, using this hunting strategy, other search agents update their location to discover the best search agent. This hunting strategy can be represented by Eq. 38 and 39.

$$D = \|CX * (t) - X(t)\| \quad (38)$$

$$X(t+1) = X * (t) + AD \quad (39)$$

where D represents moving direction towards target prey, A and C are coefficients, $x^*(t)$ denotes the target prey, where t is iteration counter. Vectors A and C are calculated by Eq. 40 and 41[46]:

$$A = 2ar - a \quad (40)$$

$$C = 2r \quad (41)$$

These vectors A and C (represented by equation) can hold positive or negative value. Here, r_i denotes a randomized vector in range $[0,1]$ and factor ' a ' is linear in nature and can reduce from 2 to 0 after several iterations. To execute the bubble net behaviour of hump back whales, two methods are used and these are shrinkage bracketing and spiral updating position method. In shrinkage bracketing method, value of ' a ' is linearly reduced and spiral updating position method is executed by managing the complete trajectory of the shape which is represented by Eq. 42 [46].

$$X(t+1) = De^{bi} \cos(2\pi l) + X * (t) \quad (42)$$

where D is the distance between Whale i and target prey, b is a constant to define the shape of logarithmic spiral, l is a random number between $[-1,1]$.

So, Eq. 39 and 42 can be used with equal probability. This whale optimization method is implemented with

Brachistochrone curve. This curve is shown in Fig. 14, which depicts that the fastest rolling curve is not linear (and straight) denoted as C1 however, it is C3 which is downward convex curve. Therefore, whale optimization considers three solutions C1, C2 and C3 and finds the best among them. Figure 15 shows that the number of nodes required to choose the best cluster head from them is very less in C3 as compared to the other solutions C1 and C2. Therefore, this strategy helps the proposed work to select the cluster head as relay node in few available cluster heads and hence suits the SEOR algorithm of SE-SKYR framework. This modified whale optimization process addresses the problem 3 of EAO algorithm. Sub procedure to choose cluster head as relay node is case when any failure occurs during the ongoing process.

Modified whale optimization (MWO) procedure ()

Input: AWO and environmental setting parameters.

Objective: Optimal solution to choose new cluster head in case of failure.

Step		/*Number of available nodes
1:	For $K = 1$ to n	
2:	{	
3:	Use modified WO method to search candidate	
4:	Solution of N_m and position of cluster head as relay node	/*Number of cluster head candidates
5:	Compute distance and energy consumption by equation 8 and 9 for each N_m	
6:	Compare distance and energy consumption for N_m nodes	
7:	Return (selected nodes as CH _m)	
8:	}	
9:	End For	

4.2.5 Computing energy consumption and finding the critical point for minimized usage

When the cluster node processes the sensed data and transfer it to the sink node then it consumes some energy [51] and this energy is denoted by Eq. 43.

$$e_h = \frac{n}{o} \omega \bar{e} + \omega \bar{e} + \omega \varphi d^4 \quad (43)$$

where n represents the number of nodes, ω depicts traffic flow (bandwidth), $\bar{e}$ denotes the energy consumed in the electronic circuit, φ indicates the power dissipation factor

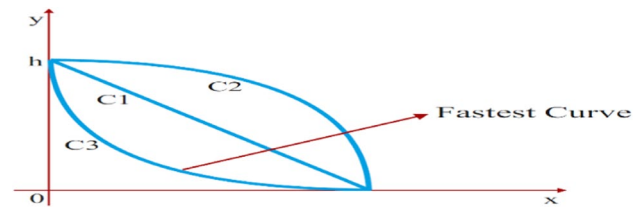


Fig. 14 Partitioned and simplified system model of heterogeneous WSN [46]

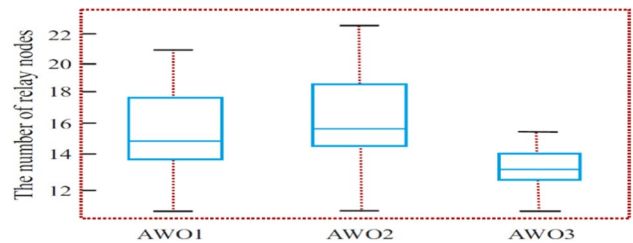


Fig. 15 Display of three adaptive strategies w.r.t number of relay nodes [46]

of the power amplifier circuit of the channel propagation model and O denotes the optimum number of edge regions (clusters). The energy transmission from source node to cluster node can be represented by Eq. 44.

$$E = \omega \bar{e} + \omega \tau d^2 \quad (44)$$

where d^2 denotes the distance between source node to the cluster head. The energy consumption of all nodes within the cluster can be represented as Eq. 45 [20].

$$\begin{aligned} e(d^2) &= \iint (x^2 + y^2) \rho(x, y) dx dy \\ &= r^2 \rho(r, \theta) dr d\theta \end{aligned} \quad (45)$$

where area of each cluster is approximately $\frac{D^2}{O}$ and distribution density of source node in cluster is $\rho(x, y)$.

Assume the radius of each cluster or edge region is r_{ch} . However, for simplicity it is considered as $r = \frac{D}{\sqrt{\pi O}} \rho = \frac{1}{\frac{D^2}{O}}$

Then, the above equation can be simplified as Eq. 46.

$$e(d^2) = \rho \int_{\theta=0}^{2\pi} \int_{r=0}^{\frac{D}{\sqrt{\pi O}}} r^2 dr d\theta = \frac{D^2}{2\pi O} \quad (46)$$

This equation can be further simplified to Eq. 47.

$$e = \omega \bar{e} + \omega \tau e d^2 \quad (47)$$

Total energy consumption by the complete network can be represented by Eq. 48.

$$E_{total} = \omega \left(n\bar{e} + n\varphi d^4 + n\bar{e} + \omega\tau \frac{D^2}{2\pi O} \right) \quad (48)$$

E_{total} will be minimum at some critical point and this can be confirmed by taking partial derivative and 2nd derivative test with respect to D .

Take partial derivative of Eq. 48 w.r.t D .

$$\frac{d(E_{Total})}{d(D)} = 0 + 0 + 0 + \frac{2n\tau D}{2\pi O} = 0 \quad (49)$$

$$D = \frac{2\pi O}{2n\tau} \quad (50)$$

Taking first derivative with respect to the distance D gives Eq. 49 and equating it to zero gives the value of D as given by Eq. 50 which is considered as critical value where energy consumption will be minimum. Take second derivative of Eq. 49 confirms the minimum energy consumption condition. Second derivative of Eq. 49 is given by Eq. 51.

$$\frac{d''(E_{Total})}{d(D)} = \frac{n\tau}{\pi O} \quad (51)$$

Second derivative is a positive value which confirms the condition mentioned in Table 3 that the critical point mentioned in Eq. 50 consumes minimum energy. Therefore, this first derivative and second derivative tests confirm the minimization of energy at $D = \frac{2\pi O}{2n\tau}$. Similarly, taking the partial derivative of Eq. 48 will give the optimum number of clusters required in this SE-SKYR computing framework and can be represented as $Cluster_{opt} = \sqrt{\frac{n\tau}{2\pi\phi} \frac{R}{r_{ch}^2}}$, where R is equivalent to D and r_{ch} is equivalent to d of Eq. 48. The above-mentioned calculation for SEOR algorithm shows that it resolves the problem 5 of EAO algorithm.

SEOR algorithm.

Step	
1:	SEOR algorithm ()
2:	{
3:	$\forall$ nodes
4:	FLACO procedure ()
5:	Data flow intensity check procedure ()
6:	Data compression procedure ()
7:	Data redundancy procedure ()
8:	Modified whale optimization procedure ()
9:	}
10:	Repeat SEOR algorithm ()

4.3 Complexity analysis

Complexity of complete SEOR algorithm depends on the complexity of five procedures discussed earlier. The complexity of FLACO procedure is composed of complexity of fuzzy logic and coyote optimization scheme. For implementing the fuzzy logic, the complexity is $O(n)$ and all member nodes are assigned with specific parameters based on which the tentative list of potential cloudlets who can act as cluster head is prepared. Now the new set of member nodes is denoted as N_p and from this N_c will be selected. Therefore, the complexity of this sub part of FLACO procedure is $O(n)$ as $N_p > N_c$. The process of fuzzy logic and coyote optimization are executed independently one after another. Hence, complete complexity of FLACO procedure is $O(n+n)$ that is $O(2n)$.

Complexity of implementing the data compression procedure is dependent on the condition of data flow intensity curve under which it is to be executed and the time period (T). If this procedure applies frequently with smaller time period (T) then the complexity is $O(n \times n)$ which of the order of $O(n^2)$. Complexity for executing the procedure 3 of data flow intensity condition check is $O(n+m)$ which is of the order $O(n)$. Procedure 4 executes the data redundancy functionality and its complexity is $O(n)$ as T is very small and can be neglected as $p > n$. Complexity of procedure 5 for implementing the MWO method for error free and without delay transmission is $O(n)$.

So, the overall complexity of SEOR algorithm is $O(2n + n^2 + n + n + n)$ that is $O(n^2 + 5n)$ and if it is compared to EAO algorithm which has complexity of $O\left(3n^2 + \frac{n}{O_{opt}}\right)$, if O_{opt} is considered as 1 as it is subset of n , which denotes the optimum number of cluster heads and relay nodes which makes means $O_{opt} \approx 1$. Hence, its complexity is of the order of $O(3n^2 + n)$. So, if we compare SEOR algorithm complexity of $O(n^2 + 5n)$ with $O(3n^2 + n)$ of EAO algorithm then former performs better for large value of n with provision of data compression, error free transmission and energy efficient sensor-based task execution. Therefore, complexity analysis suggests that the proposed SEOR algorithm is efficient than the EAO algorithm.

5 Implementation and observations

This section discusses about the implementation details and comparison of the proposed SEOR algorithm of SE-SKYR framework with some of the other existing algorithms/frameworks. The proposed SE-SKYR framework improves the sensing capability and data quality of the sensed data within the system. The proposed framework uses SEOR algorithm which works on four aspects and these are:

selection of best suited cloudlet as cluster head under the primary cloudlet, improving the quality of data by applying data compression and redundancy technique, ensuring the error free transmission of data from source node to destination node via cluster head and optimizing the energy consumption in data capture and transmission process. The proposed work is implemented in Cloudsim simulator as the earlier framework was also implemented in the same tool. For the execution of above mentioned SEOR algorithm and its comparison with other available algorithms/frameworks the simulated application parameters taken into consideration are given in Table 4. For executing the redundancy procedure another set of simulation parameters are followed and that are shown in Table 5 and 6. Therefore, the result comparison considers six reference algorithms and techniques and these are EAO, FSO, PMMRA, COA, ORC and GWO.

5.1 Network throughput (Gb/s) vs. simulation time

In this result section, network throughput is compared for various algorithm for different radii with respect to the varying simulation time. Here, three different radii 50 m, 100 m & 150 m are considered. Simulation time is considered in minutes. The result shown in Fig. 16 depicts the behaviour of different algorithms in three radii with dotted lines. The graph depicts that throughput for all the algorithms is maximum when the radii of cluster head is 50 m. This is because increase in radius requires a greater number of relay nodes and it is getting difficult to find the appropriate relay nodes. Therefore, it increases the energy consumption and packet loss during transmission of data. But the proposed SEOR algorithm does not use the relay nodes. Here, cluster heads play the role of relay nodes as well, which improves the energy consumption and packet loss. The mentioned result also depicts that the network throughput rises drastically initially but stabilizes gradually with the passage of simulation time.

The proposed SEOR algorithm pays more emphasis on bandwidth while selecting a cloudlet as cluster head and that helps it to perform better than the other algorithms and hence improves the energy consumption and data loss in transmission. The same can be comprehended from the result, where SEOR algorithm has highest network throughput and its curve stabilizes faster than the other algorithms as shown in Fig. 16. Another factor which makes the SEOR algorithm better than others is the evenly distribution of sensor nodes as per hexagonal layout proposed in SE-SKYR framework. This helps in having nicely connected cluster elements with head and efficiently scheduling the data packets by considering data flow intensity discussed earlier. This helps to avoid the network congestion and reduces the data packet loss. These factors assist in managing the energy distribution and time to all sensor nodes by the cluster head and

primary cloudlet. Hence, proposed SEOR algorithm is best suited for cloudlet-based sensing in mobile edge computing or mobile cloud computing.

5.2 Average delay vs. simulation time

Average delay is the complete time delay between source sensor node to destination sink node while sending sensed data. The result in Fig. 17 depicts that the average delay for SEOR algorithm is least and stable as compared to the other algorithms /techniques. The FSO curve shows jitter (delay variance) with respect to the simulation time as it reorganizes the split data flows according to the real time status of SDWSN. EAO, PMMRA, COA, ORC and GWO algorithms use sequential relay node for transmission of data from source to destination which increases the time delay. These algorithms do not use an efficient mechanism for compression of data and removing the redundancy in data. Due to these factors, proposed SEOR algorithm performs better than the other existing algorithms and has stable minimized curve as shown in Fig. 17.

5.3 Energy consumption vs. number of nodes

This section shows the energy consumption with the varying number of nodes. As the number of nodes increases, it will also increase the energy consumption. This behaviour is also depicted in Fig. 18. As discussed earlier, the main factor that consumes energy are involvement of relay nodes and using uncompressed and redundant data in the system. So, both these factors are well taken care in the proposed work. Besides these factors, another important parameter is selection of best suited cluster head which has sufficient resource and energy to manage the data and its related processing.

Table 4 Simulated application parameters

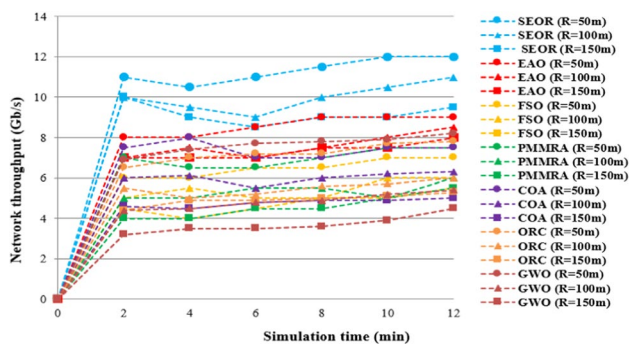
Description	Parameters	Value
Network area	$N*N$	$100 \times 100 \text{ m}^2$
Total number of nodes	N	200 nodes
Number of cluster nodes	$0.1xn$	20 nodes
Sensor nodes/devices	$0.9xn$	180 nodes
Initial energy of a normal node	E_o	0.5 J
Energy factor of cluster nodes	a	1
Initial energy of a cluster node	E_a	$E_o (1 + a)$
Packet size of data	PS	512 bytes
Dissipated energy	E_{elect}	50 nJ/bit
Energy of aggregation by CH	EDA	5 nJ/bit/message
Radius of sensor nodes	r	150 m
Radius of cluster head	R	300 m
Data transmission rate	ω	250 kbit/s
Simulation time	T	10 min

Table 5 Simulated application parameters for checking the redundancy

Description	Parameter	Values
Period size	T	32 and 64
Similarity threshold	ε	Any value based on sensed data like light 10, 15, 25
Criticality level application	C_r	Low and High
False-rejection probability risk value	t	0.2

Table 6 Redundancy rate table (RRT)

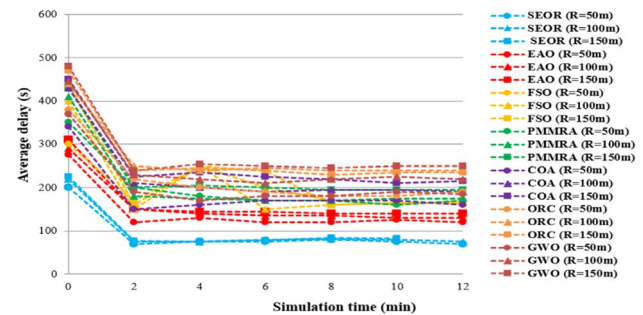
Redundancy rate, RRp	Description
$0 < RRp < 0.3$	Little redundancy rate
$0.3 < RRp < 0.6$	Moderate redundancy rate
$0.6 < RRp < 1$	Excessive redundancy rate

**Fig. 16** Network throughput vs. simulation time

This will also improve the energy consumption to great extent. Therefore, result depicts that the energy consumption increases initially for SEOR algorithm, stabilizes in between and then decreases marginally as number of nodes increases further. But other existing algorithms consume more energy as compared to the SEOR algorithm. There is a great fluctuation in energy consumption for EAO, FSO, PMMRA algorithms because they use uneven energy source sensor nodes.

5.4 Number of cluster heads vs. number of nodes

The proposed work has avoided the use of relay node and cluster head performs the same role here. As the number of sensor nodes increases in the system, the requirement for cluster head also increase to cater the generated data from sensor nodes. So, primary cloudlet assigns new cloudlet as cluster head to manage the greater number of nodes. Large number of cluster heads avoid data congestion, data loss and better utilization of energy and time. Hence the proposed SE-SKYR framework and its SEOR algorithm

**Fig. 17** Average delay vs. simulation time

efficiently manages the increasing number of sensor nodes. The result shown in Fig. 19 confirms that as the number of nodes increases in the system, it also increases the number of cluster heads. But the increase is more in SEOR curve as compared to its counterparts. This is because SE-SKYR framework uses cluster head as relay node.

5.5 Network lifetime vs. simulation time

This result shows the comparison of network lifetime of complete sensor network with varying simulation time for SEOR and other algorithms. In this simulation, factors like radius, number of sensor nodes, cluster heads etc. are kept fixed and network lifetime is analysed with the varying simulation time. Performance of SEOR algorithm is better than the other algorithms as its fuzzy logic and coyote optimization strategy selects best set of potential secondary cloudlets as cluster heads. This increases the sustainability of the complete sensor network for longer time. Other factors which improve the network lifetime is usage of data flow intensity curve with maxima and minima concept. This reduces the irrelevant time in compressing every set of data and SRA method removes the data redundancy which improves the network's lifetime. So, due to these factors, SEOR algorithm performs well and same can be seen from the Fig. 20.

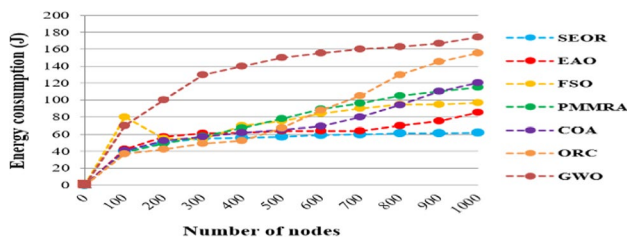


Fig. 18 Energy consumption vs. number of nodes

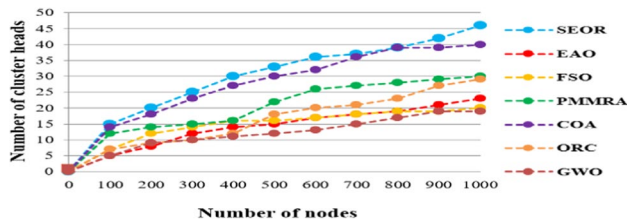


Fig. 19 Cluster head vs. number of nodes

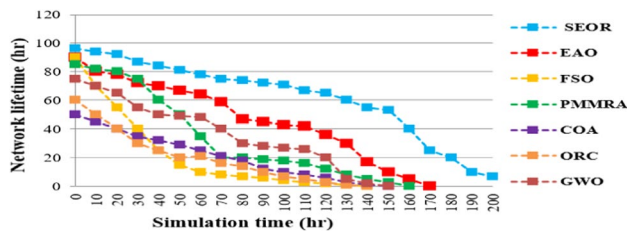


Fig. 20 Network lifetime vs. simulation time

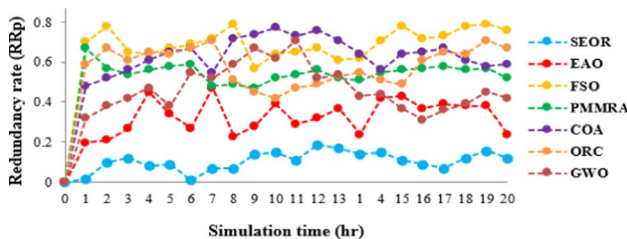


Fig. 21 Redundancy rate vs. simulation time

5.6 Redundancy rate vs. simulation time

The proposed work has used the redundancy rate based T -test and sensing rate adjustment technique which keep the redundancy in the proposed work within the limit of 0 to 0.2 but it is very high in other existing algorithms. The result shown in Fig. 21 depicts the same.

6 Conclusion and future work

The proposed work has thoroughly discussed the various distributed computing's, energy optimization techniques, fuzzy logic, coyote optimization, cluster-based sensor network and data optimization methods. The proposed work has considered the ESDWSN framework for sensing data through sensor node and EAO algorithm as a reference work and proposes improvement on four factors. The proposed work does not use the relay node (as it causes the transmission delay) which the reference work uses. The proposed work selects the best set of cloudlets to act as cluster head using fuzzy logic and coyote optimization technique which fulfil the need of relay nodes. It also improves the compression factor and reduces data redundancy. For compression, data flow intensity is applied with 1st derivative and 2nd derivative test which suggest the condition to perform the data compression. As applying data compression on large sets of packets consumes more energy in compression and decompression process. So, it's not worth to apply compression on every data sets. The proposed work has improved the error rate and energy efficiency in data transmission using modified whale optimization method. This makes the SE-SKYR framework and its SEOR algorithm best suited for cloudlet-based sensing task in mobile cloud computing and mobile edge computing. It is also more efficient as the time complexity of SEOR algorithm is less as compared to the EAO algorithm. It also suggests the critical point and number of optimized clusters where the energy consumption is minimized in the system. The result section depicts the same and proves that the SE-SKYR framework with SEOR algorithm suits best for cloudlet-based sensing task.

In future, we will explore pricing model for sensor-based tasks in SE-SKYR framework which will motivate the sensor devices to collect data and cloudlets to process data as cluster heads. We will facilitate SE-SKYR framework with scheduling algorithm to efficiently schedule the sensed data. Future work will also emphasize on security and reliability of sensed data in SE-SKYR framework. In other words, various important aspects of sensed data for SE-SKYR framework may be considered for future research analysis.

References

1. C. V. N. Index. (2015). Global mobile data traffic forecast update, 2014–2019. *White Paper*.
2. Dinh, H. T., Lee, C., Niyato, D., & Wang, P. (2013). A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18), 1587–1611.

3. Satyanarayanan, M., Bahl, P., Caceres, R., & Davies, N. (2009). The case for VM-based cloudlets in mobile computing. *Journal of IEEE pervasive computing*, 8(4), 14–23.
4. Hu, Y. C., Patel, M., Sabella, D., Sprecher, N., & Young, V. (2015). Mobile edge computing: A key technology towards 5G. *Eur. Telecommun. Standards Inst., Sophia Antipolis, France, white paper 11*.
5. 5G Vision: The 5G Infrastructure Public Private Partnership: The next generation of communication networks and services. (2016). [Online]. Available: <https://5gpp.eu/wpcontent/uploads/2015/02/5G-Vision-Brochure-v1.pdf>
6. Satyanarayanan, M., Chen, Z., Ha, K., Hu, W., Richter, W., & Pillai, P. (2014). Cloudlets: at the leading edge of mobile-cloud convergence. In *Proceedings of 6th International Conference on Mobile Computing, Applications and Services (MobiCASE)*, IEEE, pp. 1–9.
7. Nagy, M., & Kotuliak, I. (2014). Utilizing Open Flow, SDN and NFV in GPRS Core Network. In *Proceedings of International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*, pp. 184–193.
8. Balbas, J. J. P., Rommer, S., & Stenfelt, J. (2009). Policy and charging control in the evolved packet system. *Journal of IEEE Communications Magazine*, 47(2), 68–74.
9. Zieliski, Z., Chudzikiewicz, J., & Furtak, J. (2018). An approach to integrating security and fault tolerance mechanisms into the military IoT. In *Proceeding of International Conference on Security and Fault Tolerance in Internet of Things*, Springer, pp. 111–128.
10. Mehta, K., & Pal, R. (2017). Energy efficient routing protocols for wireless sensor networks: A survey. *International Journal of Computer Application*, 165(3), 398–406.
11. Sarkar, A., & Murugan, T. S. (2017). Cluster head selection for energy efficient and delay-less routing in wireless sensor network. *Journal of Wireless Network, Springer*, 25(1), 303–320.
12. Wang, Q., Lin, D., Yang, P., & Zhang, Z. (2019). An energy-efficient compressive sensing-based clustering routing protocol for WSNs. *Journal of IEEE Sensors*, 19(10), 3950–3960.
13. Shivappa, N., & Manvi, S. S. (2019). Fuzzy-based cluster head selection and cluster formation in wireless sensor networks. *Journal of Institution of Engineering and Technology*, 8(6), 390–397.
14. Kaur, S., & Mahajan, R. (2018). Hybrid meta-heuristic optimization-based energy efficient protocol for wireless sensor networks. *Egyptian Informatics Journal*, 19(3), 145–150.
15. Nandakumar, A. N. (2018). Novel bacteria foraging optimization for energy efficient communication in wireless sensor network. *International Journal of Electrical and Computer Engineering*, 8(6), 4755–4762.
16. Wang, Q., Guo, S., Liu, J., Pan, C., & Yang, L. (2022). Profit maximization incentive mechanism for resource providers in mobile edge computing. *Journal of IEEE Transactions on Services Computing*, 15(1), 138–149.
17. Li, G., Guo, S., Yang, Y., & Yang, Y. (2018). Traffic load minimization in software defined wireless sensor networks. *Journal of IEEE Internet of Things*, 5(3), 1370–1378.
18. Zheng, Q., Zheng, K., Zhang, H., & Leung, V. C. M. (2016). Delay-optimal virtualized radio resource scheduling in software-defined vehicular networks via stochastic learning. *Journal of IEEE Transaction on Vehicular Technology*, 65(10), 7857–7867.
19. Ma, X., Wang, S., Zhang, S., Yang, P., Lin, C., & Shen, X. S. (2021). Cost-efficient resource provisioning for dynamic requests in cloud assisted mobile edge computing. *Journal of IEEE Transaction on Cloud Computing*, 9(3), 968–980.
20. Li, G., & Xu, Y. (2019). Energy consumption averaging and minimization for the software defined wireless sensor networks with edge computing. *Journal of IEEE Access*, 7(1), 173086–173097.
21. Han, X., Cao, X., Lloyd, E. L., & Shen, C. C. (2010). Fault-tolerant relay node placement in heterogeneous wireless sensor networks. *Journal of IEEE Transaction on Mobile Computing*, 9(5), 643–656.
22. Guo, J., & Jafarkhani, H. (2016). Sensor deployment with limited communication range in homogeneous and heterogeneous wireless sensor networks. *Journal of IEEE Transactions on Wireless Communications*, 15(10), 6771–6784.
23. Zhang, X., & Zhang, X. (2017). A binary artificial bee colony algorithm for constructing spanning trees in vehicular ad hoc networks. *Journal of Ad Hoc Network*, 58(1), 198–204.
24. Lloyd, E. L., & Xue, G. (2007). Relay node placement in wireless sensor networks. *Journal of IEEE Transaction on Computers*, 56(1), 134–138.
25. Pierzan, J., & Dos Santos Coelho, L. (2018). Coyote optimization algorithm: A new Metaheuristic for global optimization problems. In *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8.
26. Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Journal of Advances in Engineering Software*, 69(1), 46–61.
27. Kulkarni, R. V., & Venayagamoorthy, G. K. (2011). Particle swarm optimization in wireless-sensor networks: A brief survey. *Journal of IEEE Transactions on Systems, Man, and Cybernetics, part C, Application & Review*, 41(2), 262–267.
28. Smaragdakis, G. (2004). *SEP: A stable election protocol for clustered heterogeneous wireless sensor network*. Boston University, Boston, MA, USA, Technical report.
29. Heinzelman, W. R., Chandrakasan, A., & Balakrishnan, H. (2000). Energy efficient communication protocol for wireless microsensor networks. In *Proceedings of 33rd Annual Hawaii International Conference on System Science*, pp. 1–10.
30. Sert, S. A., Bagci, H., & Yazici, A. (2015). MOFCA: Multi-objective fuzzy clustering algorithm for wireless sensor networks. *Applied Soft Computing*, 30, 151–165.
31. Singh, M., Soni, S., & Kumar, V. (2016). Clustering using fuzzy logic in wireless sensor network. In *Proceeding of International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 1669–1674.
32. El Alami, H., & Najid, A. (2020). Fuzzy logic-based clustering algorithm for wireless sensor networks. *Journal of Sensor Technology: Concepts, Methodologies, Tools, and Applications*, 3(1), 351–371.
33. Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020). Applications of wireless sensor networks: An up-to-date survey. *Journal of Applied System Innovation*, 3(1), 1–14.
34. El-Sayed, H., Sankar, S., Prasad, M., Puthal, D., Gupta, A., Mohanty, M., & Lin, C. T. (2017). Edge of things: The big picture on the integration of edge, IoT and the cloud in a distributed computing environment. *Journal of IEEE Access*, 6(1), 1706–1717.
35. Zhou, D., Chao, F., Lin, C. M., Yang, L., Shi, M., & Zhou, C. (2017). Integration of fuzzy cmac and belc networks for uncertain nonlinear system control. In *Proceedings of 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–6.
36. Salmani, V., Ensafi, R., Astaneh, K. N., & Naghibzadeh, M. (2007). A fuzzy-based multi-criteria scheduler for uniform multiprocessor realtime systems. In *Proceedings of 10th International Conference on Information Technology (ICIT 2007)*, pp. 179–184.
37. He, T., & Ukil, A. (2016). Design of fuzzy logic-based controller for energy efficient operation in building industrial electronics society. In *proceedings of 42nd Annual Conference of the IEEE (IECON)*, pp. 90–95.

38. Kaur, K., Garg, S., Aujla, G. S., Kumar, N., Rodrigues, J. J. P. C., & Guizani, M. (2018). Edge computing in the industrial Internet of Things environment: Software-defined-networks-based edge-cloud interplay. *Journal of IEEE Communication Magazine*, 56(2), 44–51.
39. Zhang, Y., Du, P., Wang, J., & Xin, N. (2019). Resource scheduling for delay minimization in multi-server cellular edge computing systems. *Journal of IEEE Access*, 7(1), 86265–86273.
40. Kumar, R., & Yadav, S. K. (2017). Scalable key parameter yields of resources model for performance enhancement in mobile cloud computing. *Springer Journal of Wireless Personal Communications*, 95(4), 3969–4000.
41. Yadav, S. K., & Kumar, R. (2021). A mobile cloud computing framework for execution of data as a service using cloudlet. *Kuwait Journal of Science*, 48(3), 1–12.
42. Yadav, S. K., & Kumar, R. (2022). Scalable profit optimized incentive mechanism for resources in cloudlet based mobile edge computing framework. *Journal of Wireless Personal Communications*, Springer, 125(1), 159–207.
43. Yadav, S. K., & Kumar, R. (2023). EVACON-rainsnow computing: An amalgamation of cloud and its inherited computing to encourage end user for both localized and globalized remote computing. *Journal of Wireless Personal Communications*, Springer, 132(1), 2737–2792.
44. Yadav, S. K., & Kumar, R. (2023). ASME-SKYR framework: A comprehensive task scheduling framework for mobile cloud computing. *Journal of Wireless Networks*, Springer, 29(8), 1–24.
45. Yadav, S.K. & Kumar, R. (2021). A scalable and utility driven profit maximized auction of resources model for cloudlet based mobile edge computing. *Springer Journal of Wireless Personal Communications*, [online] <https://link.springer.com/article/https://doi.org/10.1007/s11277-021-08223-7>
46. Xie, J., Zhang, B., & Zhang, C. (2020). A novel relay node placement and energy efficient routing method for heterogeneous wireless sensor networks. *Journal of IEEE Access*, 8(1), 202439–202444.
47. Sayyida, A. A., Harbb, H., Ruiza, M., & Velasco, L. (2020). ZIZO: A zoom-in zoom-out mechanism for minimizing redundancy and saving energy in wireless sensor networks. *Journal of IEEE Sensors*, 21(3), 3452–3462.
48. Janarthanan, R., Balamurali, R., Annapoorani, A., & Vimala, V. (2020). Prediction of rainfall using fuzzy logic. *Journal of Materials Today*, 37(2), 959–963.
49. Ni, Q., Pan, Q., Du, H., Cao, C., & Zhai, Y. (2017). A novel cluster head selection algorithm based on fuzzy clustering and particle swarm optimization. *Journal of IEEE/ACM Transaction of Computational Biology and Bioinformatics*, 14(1), 76–84.
50. Mohamed, A., Saber, W., Elnahry, I., & Hassanien, A. E. (2020). Coyote optimization based on a fuzzy logic algorithm for energy-efficiency in wireless sensor networks. *Journal of IEEE Access*, 8(1), 185816–185829.
51. Nakagaki, T., Tero, A., Kobayashi, R., Onishi, I., & Miyaji, T. (2008). Computational ability of cells based on cell dynamics and adaptability. *Journal of New Generation Computing*, 27(1), 57–81.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



Santosh Kumar Yadav received the B.E. degree in Computer Science and Engineering from Panjab University, Chandigarh, India in 2007. He completed his M.E in Computer Science and Engineering from NITTTR, Chandigarh, India. He completed his PhD from Panjab University Chandigarh, India. He has 15 years of teaching experience and currently working as lecturer in Chandigarh College of Engineering and Technology, Chandigarh, India.



Rakesh Kumar received the B.Tech degree in Computer Science and Engg. from IKGPTU Jalandhar, India and M. Tech degree from GGSIPU, Delhi, India. He did his doctorate in Computer Engineering from NIT, Kurukshetra, India. He has 19 years of teaching experience and published more than 50 international and national papers in various reputed journals/conferences. He is currently working as Professor and Head at Central University of Haryana, Mahendergarh, India.