Cost-effective routing as a service in sensor-cloud

Biplab Kanti Sen*

Department of Computer Science and Engineering,
University of Calcutta,
Kolkata, 700 098, India
Email: bksen.cu@gmail.com
*Corresponding author

Anupam Sarkar

Indian Statistical Institute,
Kolkata, 700108, India
Email: anupam_r@isical.ac.in

Sunirmal Khatua and Rajib K. Das

Department of Computer Science and Engineering,
University of Calcutta,
Kolkata, 700 098, India
Email: skhatuacomp@caluniv.ac.in
Email: rajib.k.das@ieee.org

Abstract: Sensor-cloud is a collaborative platform which allows multiple wireless sensor networks (WSN) to pull their resources together for better utilisation and efficiency. In this work, we consider a set of applications, each characterised by the set of targets it intends to cover and our aim is to provide routing as a service (RaaS) to all these applications in the sensor-cloud. We should be able to run data gathering for each application in parallel with the minimum use of sensor-cloud resources. This is achieved by creating a collection of data gathering trees. We have proposed an algorithm which selects a minimal set of sensors to cover the required targets and obtains a spanning forest connecting them so that, the total energy cost of all the trees in the forest is minimised. Experimental results demonstrate the advantages of serving the applications by a sensor-cloud over standalone WSNs in terms of the usage of resources.

Keywords: sensor-cloud; WSN; wireless sensor network; virtualisation; data gathering; RaaS; routing as a service; VSN; virtual sensor network; data aggregation.

Reference to this paper should be made as follows: Sen, B.K., Sarkar, A., Khatua, S. and Das, R.K. (2020) 'Cost-effective routing as a service in sensor-cloud', *Int. J. Sensor Networks*, Vol. 32, No. 1, pp.42–53.

Biographical notes: Biplab Kanti Sen is an SRA in the Department of Computer Science and Engineering, University Of Calcutta. His research interests include wireless sensor network, computer vision, sensor-cloud, IoT.

Anupam Sarkar is a Junior Research Fellow at the Indian Statistical Institute. His research interests include containerisation, docker, software security, fuzzing.

Sunirmal Khatua is an Assistant Professor of Computer Science and Engineering Department, University of Calcutta. His area of research interests include wireless sensor network, parallel and distributed computing, sensor-cloud, IoT.

Rajib K. Das is an Associate Professor of Computer Science and Engineering Department, University Of Calcutta. His area of research interests include wireless sensor network, parallel and distributed computing, NLP.

This paper is a revised and expanded version of a paper entitled 'Target coverage using a collaborative platform for sensor cloud' presented at the *9th IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Indian Statistical Institute, Kolkata, India, 15–18 December, 2015.

1 Introduction

Wireless sensor networks (WSN) are deployed to serve for numerous real-life applications such as target tracking, battlefield monitoring, telemonitoring, ubiquitous monitoring, and several other applications (Haenggi, 2004; Leontiadis et al., 2012; Misra et al., 2017; Chen et al., 2015; Chang and Tassiulas, 2004; Fazio and Puliafito, 2015). Sensor-cloud, the new paradigm of WSN, has been recognised as a potential substitution of typical WSNs which are deployed by an organisation to serve their own applications. But this dedicated approach of executing sensor based application has proven to be inefficient in terms of use of resources (Leontiadis et al., 2012). Sensor-cloud is a collaborative platform for WSN where multiple users' applications co-exist on the same shared physical resources provided by different organisations. Such platform virtualises the collaborated physical resources giving the impression of a homogeneous network where multiple virtual sensor network (VSN) can be launched for multiple applications (Leontiadis et al., 2012; Sen et al., 2015). An application on such a platform uses virtual sensors deployed on physical sensors and may span multiple WSNs.

In a WSN, deployed sensors are equipped with a limited energy source and in general there is no facility for recharging. In order to prolong the lifetime of the network under these limitations, we try to regulate different networking operations in an energy efficient way. Data gathering in a WSN is an indispensable network operation that causes major energy drainage during the reception and transmission of data packets. Hence, in a standalone WSN, we aim to construct a data gathering tree which maximises the lifetime of the network. However, in sensor-cloud like multi-application collaborative platform, we face a different situation. Here, several users request the VSN Broker to deploy their applications in a Sensor-cloud environment specifying their parameters. The VSN broker then provisions some physical sensors to serve the applications and then designs routing paths to enable collection of the sensed data to one or more BS and finally to the respective users (Dinh and Kim, 2018). The first part can be termed as *Sensors-as-a-Service* (Madria, 2018) and the next as *Routing-as-a-Service*. Here, the situation is dynamic as some applications may terminate and new applications may be deployed in the future. So, our objective here should be to provide RaaS using the minimum amount of resources (physical sensor nodes and battery energy of sensors) from the Sensor-cloud.

The task of provisioning sensors and then providing routing paths in a collaboration is different from the task of constructing data gathering tree in a stand-alone WSN in the following aspects

i In a standalone WSN, there is a single BS and all sensors belong to a single provider. In contrast, we have multiple BSs each provided by a WSN vendor and the sensors come from different providers through collaboration.

ii In a standalone WSN, there is a single data gathering tree. In a Sensor-cloud, we create multiple data gathering trees each rooted at a BS, and the data for a particular application may be collected at multiple BSs. In this way, the coverage area for an application can be scattered over a wide geographical region, which is not possible in standalone WSN.

In this paper, we solve the problem of providing cost-effective RaaS in Sensor-cloud by solving the following subproblems:

i selecting a minimum subset of sensors needed to serve all applications taken together (Sen et al., 2015)

ii constructing a data gathering forest consisting of only the selected sensors.

The advantages of collaboration to reduce the cost (sensors as well as energy) is fairly obvious. A sensor covering a target belonging to multiple applications needs to transmit only one copy of the sensed data to its BS. From the BS, separate copies of that data are sent to the interested applications. This reduces the total number of sensors needed compared to the case where each application has its own WSN. The total number of transmitted messages are also reduced, thereby reducing the transmission energy cost.

Our contributions in this paper are the following:

- Deriving the energy cost of a data gathering tree for different data aggregation models such as

  i complete aggregation model

  ii raw data model

  iii partial aggregation model (packet size is in between the previous two models).

- Showing that for the second model the shortest path tree (SPT) is optimal in terms of energy cost.
- Providing an algorithm to construct energy-efficient data gathering forest in a Sensor-cloud.
- Designing an iterative method to further reduce the cost of a data gathering forest.

The rest of this paper is organised as follows. Section 2 describes our system model and formally defines the problem. The algorithms for sensor node selection and tree construction are presented in Sections 3 and 4 respectively. Results and analysis are presented in Section 5. Section 6 reviews the related works and Section 7 concludes the paper.

2 System model and problem definition

In this work, we have adopted the Sensor-cloud model as presented in Sen et al. (2015) where sensor-cloud is defined as follows:

Definition 1: Sensor-cloud is a model for enabling ubiquitous, convenient, and on-demand network access to a shared pool of configurable sensors that can be rapidly provisioned and released with minimal management effort or WSN provider interaction.

Sensor virtualisation is a key technology in Sensor-cloud which is defined as:

Definition 2: Sensor Virtualisation is the partitioning of a physical sensor node into smaller virtual sensor nodes (VN) which sense and forward a physical parameter along with other collocated VNs.

Sensor nodes are virtualised to provide virtual sensor nodes which act as resources for the Sensor-cloud.

A sensor-cloud application A is characterised by the tuple $\{A_{id}, \{P\}, \{T\}\}$ where A_{id} is the application id, P is one or more parameters (temperature, humidity etc.) to be sensed and $\{T\}$ is the set of locations of interests (targets) for the application.

Virtual sensor network (VSN) is defined as:

Definition 3: The virtual sensor network is a network of VNs over a dispersed geographical area, deployed on the physical sensor nodes of multiple WSN providers, for a particular Sensor-cloud application.

The sensor-cloud architecture proposed in Sen et al. (2015) uses the definitions and notations presented in Table 1.

Table 1 Definitions for sensor cloud architecture

A	*Sensor cloud application*	$(A_{id}, \{P\}, \{T\})$
SN	Physical sensor node	$(SN_{id}, \{P\}, Loc, R)$
VN	Virtual sensor node	$(VN_{id}, SN_{id}, \{P\}, Type, Parent)$
$Parent$	Parent of a virtual sensor node	$VN_{id} : SN_{id}$
WSN	Physical sensor network	$(WSN_{id}, \{SN\}, Adj, b_j)$
VSN	Virtual sensor network	$(A_{id}, VSN_{id}, \{VN\})$

Unlike conventional WSN, the sensors needed for an application may consist of multiple components each with its own BS. The sensor cloud user (SCU) deploys her application in the Sensor-cloud environment through the VSN Broker. A sensor-cloud framework having 4 WSNs serving n applications are shown in Figure 1. The sensor nodes belonging to the WSN's are coloured green, red, white and grey. It is seen that Application 1 uses one sensor each from 3 different WSN's.

The VSN broker, on getting requests from a set of SCU's with their requirements, sets on creating VSN's in order to provide the service. This involves solving two sub-problems stated as follows:

- selecting a minimum set of SN to provide the service requested by all applications
- construct a set of data gathering trees each rooted at a BS such that total energy spent per round of data collection is minimised.

The first subproblem, when solved, identifies the set of sensors needed to cover the targets specified by the applications. Also, some additional sensors may be needed as relay nodes to deliver the sensed data to the nearest BS.

After solving the first subproblem we are left with a set of connected components, each having some sensor nodes and a BS. Our objective for the second subproblem is to construct a data gathering tree for each component such that total energy spent per round is minimised. This problem may be trivial or non-trivial based on the data aggregation model assumed.

3 Sensor node selection

We consider a sensor-cloud network consisting of sensors and BSs as the set $SC = \{s_1, s_2 \ldots s_n\}$ out of which the first k are BSs. We denote the set of BSs by $\mathcal{B} = \{s_1, s_2 \ldots, s_k\}$, the set of sensor nodes by $S = \{s_{k+1}, s_{k+2}, \ldots, s_n\}$ and set of all targets by $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$.

Definition 4 (Sensor node selection problem)**:** Given a set of applications, $A_i = (A_{id}, \{P\}, \{T_i\}),\ 1 \leq i \leq q,$ where $T_i \subset \mathcal{T}$ is the set of targets to be covered for application A_i, find the smallest set of sensors $S' \subset S$ such that

i for each target $t \in \cup(A_i.T)$, there exists at least one sensor $s \in S'$ which covers t

ii for each sensor $s \in S'$, there is a path via other sensors in S' to one of the BSs.

The first constraint ensures coverage and the second ensures connectivity.

Finding minimum set of sensors for coverage and connectivity: In order to find a minimum set of sensors for coverage and connectivity, we have adopted a two-phase algorithm (Sen et al., 2015). The first phase uses an integer linear programming (ILP) based solution to find a minimum number of sensors to ensure coverage and the second phase employs a heuristic to find an additional set of sensors to ensure connectivity. We assume that each sensor has a coverage radius r_s and a communication radius r_c. We say a sensor s covers a target t, if $d(s,t)$, the Euclidean distance between s and t, is less than or equal to r_s. All the definitions used for sensor selection are listed in Table 2.

Table 2 Definitions for sensor selection

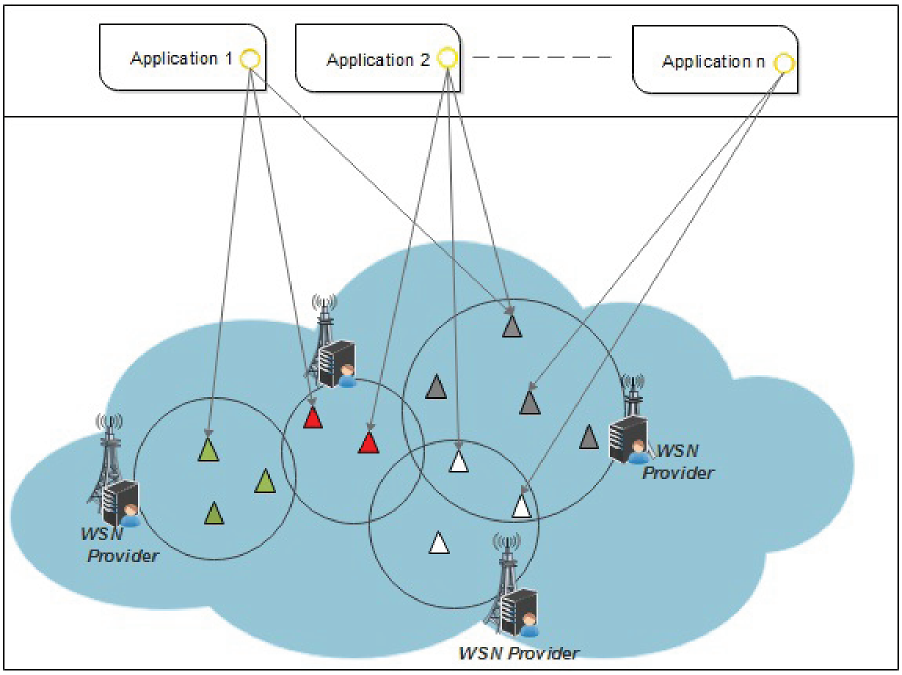
r_c	Communication range
r_s	Sensing radius or coverage radius
$d(s,t)$	Euclidean distance between s and t
BS	Base station
S	set of all sensor nodes
$\mathcal{T}$	Set of all targets

3.1 Ensure coverage

We formulate an ILP to select a minimum set of sensors needed to satisfy all the applications $A_i, 1 \leq i \leq q$.

Let $T_{all} = \cup(A_i.T)$ denote the set of all targets which need to be covered.

Figure 1 Sensor-cloud consisting of 4 WSNs (see online version for colours)



We define an array y of size m, where:

$$y_j = \begin{cases} 1, & \text{if } t_j \in T_{all} \\ 0, & \text{otherwise} \end{cases}$$

Here, $y_j = 1$, implies that target t_j needs to be covered.

We define an $n \times m$ matrix Cov where

$$Cov_{i,j} = \begin{cases} 1, & d(s_i, t_j) \le r_s \\ 0, & \text{otherwise} \end{cases}$$

First k rows of Cov are all zero's as $s_1, s_2, \ldots, s_k$ are Base Stations.

The matrix Cov and array y are parameters of ILP and the binary variables to be solved are x_i, $k+1 \le i \le n$, where

$$x_i = \begin{cases} 1, & \text{if } s_i \text{ is selected} \\ 0, & \text{otherwise} \end{cases}$$

Now we can state the ILP formally as:

$$\text{minimise:} \;\; Z = \sum_{i=k+1}^{n} (x_i)$$

Subject to:

$$i) \;\; \sum_{i=k+1}^{n} (Cov_{i,j} \times x_i) \ge y_j, \; \forall j = 1 \cdots m$$

The constraint is to ensure that each target $t_j \in T_{all}$ is covered.

The sensors selected by ILP (as indicated by the solution x_i) are termed as *coverage nodes*.

3.2 Ensure connectivity

Even though ILP will produce a minimum set of sensors to cover all targets T_{all}, there has to be a path from each sensor in the set to at least one BS. The second phase of the algorithm (Component Connectivity) finds an additional set of sensors needed to ensure connectivity.

We call this additional set of sensors the *connecting nodes*.

Algorithm component connectivity: It starts with the adjacency matrix Adj, list of all base station $\mathcal{B}$ and set of coverage nodes cv obtained from ILP. Adj is an $n \times n$ matrix which is based on the communication radius r_c of sensors. Adj is defined as follows:

$$Adj_{i,j} = \begin{cases} 1, & \text{if } d(s_i, s_j) \le r_c \\ 0, & \text{otherwise} \end{cases}$$

The algorithm calls a procedure *findComponents* which returns the set of connected components consisting of only the coverage nodes. Here, C_{BS} denotes the set of components with at least one BS and C_{WBS} denotes the set of components without any BS. Now an iterative procedure is used to determine the closest pair of components (c_1, c_2) where $c_1 \in C_{WBS}$ and $c_2 \in C_{BS} \cup C_{WBS}$. To achieve this, we take a component $c \in C_{WBS}$ and enlarge it by adding all its one-hop neighbours. We continuing doing this until we find another component or no further enlargement is possible. If no further enlargement is possible and we have not reached another component, it means that component in C_{WBS} cannot be connected. If we reach another component, we note the distance between these two components. We do this for each component in C_{WBS} and the pair which corresponds to the minimum distance is selected for merging. The pair (c_1, c_2) and the set of intermediate nodes in the shortest path between them together constitute the merged component m_c. Component c_2 is now replaced by m_c and c_1 is removed from C_{WBS}. The process terminates when C_{WBS} becomes empty or there are components in C_{WBS} for which no closest component c_2 can be found. In the former case, components in C_{BS} contain all coverage and connecting nodes and list (M) contains all connecting nodes; in the latter case, the network is disconnected and D contains the set of nodes which cannot be connected.

The reasons we want a minimum subset of sensors for coverage and connectivity are twofold. Smaller the number of sensors in a tree, less is the total energy cost of the tree. Secondly, the cost of providing the service to an application would also be less if less number of sensors are employed which in turn will make it more attractive to the users.

3.3 ILP for both coverage and connectivity

It is possible to find the set of coverage and connecting nodes together as done in the following ILP. Even though this ILP takes prohibitively large computation time for standard size sensor-cloud, we can use the solution for a small sized problem to measure the performance of the heuristic presented in the previous subsection.

In this ILP along with the parameters Cov and y, we use the adjacency matrix Adj defined as before. We run breadth first search (BFS) starting from all the BSs and find the maximum hop distance of all the sensors from the nearest BS. We call it K.

The ILP solves binary variables $x_{i,p},\ \ k+1 \leq i \leq n, 1 \leq p \leq K$, where,

$$x_{i,p} = \begin{cases} 1, & \text{if } s_i \text{ is selected and } s_i\ is\ p \text{ hops away} \\ & \text{from a BS} \\ 0, & \text{otherwise} \end{cases}$$

We can now state the ILP as

$$\text{minimise} \quad Z = \sum_{i=k+1}^{n} \sum_{p=1}^{K} x_{i,p}$$

subject to :

i) $\displaystyle\sum_{i=k+1}^{n} \sum_{p=1}^{K} (Cov_{i,j} \times x_{i,p}) \geq y_j$, for $j = 1 \cdots m$

ii) $\displaystyle\sum_{p=1}^{K} x_{i,p} \leq 1$, for $i = k+1 \cdots n$

iii) $\sum_{j=1}^{n} (A_{i,j} \times x_{j,p-1}) \leq x_{i,p}$, for $i = 1 \cdots n$, and $p = 1 \ldots K$

iv) $x_{i,0} = 1$, for $i = 1 \cdots k$

v) $x_{i,0} = 0$, for $i = k+1 \cdots n$

The first constraint is to ensure that each target $t_j \in T_{all}$ are covered. The second constraint implies that $x_{i,p}$ can be 1 for only one value of p. The third constraint ensures that a sensor, if selected and at p hops from a BS must have adjacent at least one selected sensor at $p-1$ hops from a BS. The fourth constraint implies that all BSs are selected and the fifth that any sensor must be at least one hop away from a BS.

4 Power efficient data gathering forest

After the sensor selection is completed, we have to construct a set of trees using those sensors so that the total energy cost of all of them combined is minimised. In this section we present the energy consumption model adopted, the cost of a data gathering tree based on that model, algorithm to construct data gathering forest and finally an iterative method to reduce the energy cost.

4.1 Energy consumption model for communication

The energy of a node is drained out with time for sensing, computing, and communication. However, communication is considered to be the major source of power consumption in the sensors. We have adopted the energy model (Lindsey et al., 2002), according to which, the energy consumed by a sensor u in receiving an M bit message is

$$R_x = \epsilon_{elec} M \tag{1}$$

and the energy consumed by sensor node u to transmit an M-bit message to a node v is

$$T_x(u,v) = R_x + \epsilon_{amp} M d(u,v)^2 \tag{2}$$

Here, ϵ_{elec} is the constant energy required per bit by the transmitter or receiver circuitry and ϵ_{amp} is that for the transmitter amplifier to transmit a single bit over a distance of 1 m. Here, $d(u,v)$ is the Euclidean distance between u and v.

4.2 Cost of a data aggregation tree

We assume that the solution of the first sub-problem gives a set of component each with at least one BS. The definitions used for deriving the cost of a data gathering tree G_B in a component with a node set V and BS B are presented in Table 3. We consider three models of data aggregation. They are namely :

- complete aggregation model
- raw data model
- partial aggregation model

Table 3 Definitions for a data gathering Tree

$G_B = (V, E)$	Data gathering tree rooted at a BS B where E represents the set of edges in the tree G_B
E_B	$\{e : e \in E$ and e is adjacent to $B\}$
$\mathcal{B}$	set of all Base Stations
E'	$E - E_B$
V_{cov}	set of coverage node in V i.e $V_{cov} \subset V$
u_D	Number of coverage node among the descendants of a node u in G_B
$d(u,v)$	Euclidean distance between nodes u and v
$W_e = d(u,v)^2$	Weight of an edge $e = (u,v)$
$P(u)$	set of edges in the path from u to BS
$P'(u)$	$\{e : e \in P(u)$ and $e \notin E_B\}$
$h_u = \|P(u)\|$	Hop count of node u to BS
$W_{P(u)}$	$\sum_{e \in P(u)} W_e$
$W'_{P(u)}$	$\sum_{e \in P'(u)} W_e$
$C(B)$	Cost of data gathering tree G_B

4.2.1 Complete aggregation model

In this model, we assume that a node aggregates its own data with the data received from children in such a way that the packet it generates to be sent towards BS is always of size k bits. Thus, the size of the packet does not depend on how many sensor's data are aggregated in it.

Lemma 1: *The total energy cost T_{CA}^{B} for transmission and total energy cost R_{CA}^{B} for receiving in a tree G_B under complete aggregation model is given by:*

$$T_{CA}^{B} = \epsilon_{elec}k|E| + \epsilon_{amp}k \sum_{e \in E} W_e \tag{3}$$

$$R_{CA}^{B} = \epsilon_{elec}k|E'| \tag{4}$$

Proof: This follows from equations (1) and (2) because of the facts

i all packets are of size k bits

ii receiving energy is not to be considered when the recipient is BS B.

□

4.2.2 Raw data model

In a data gathering tree, there are two types of nodes. A node which covers one or more targets is termed as coverage node. A coverage node which is also a leaf node just sends a packet of size b bits to its parent. A coverage node which is not a leaf node, receives packets from all of its children and then combines its own data with it and sends to its parent. Thus, for such a node u the size of its data packet is $size_u = (u_D + 1)b$. A connecting node u just pass over the data received from its children and its packet size is $u_D b$. We summarise this mode of data aggregation by the following formula:

$$size_u = \begin{cases} (u_D + 1)b, & \text{if } u \text{ is a covereage node} \\ u_D b, & \text{otherwise} \end{cases} \tag{5}$$

Lemma 2: *The total energy cost T_{RD}^{B} for transmission and total energy cost R_{RD}^{B} for receiving in Raw Data model is given by the following equation.*

$$T_{RD}^{B} = \epsilon_{elec}b \sum_{u \in V_{COV}} h_u + \epsilon_{amp}b \sum_{u \in V_{COV}} W_{P(u)} \tag{6}$$

$$R_{RD}^{B} = \epsilon_{elec}b \sum_{u \in V_{COV}} (h_u - 1) \tag{7}$$

Proof: It is clear that any leaf node which is not a coverage node in the tree is superfluous. So, we can assume that all leaf nodes must be coverage nodes. An intermediate node can either be a coverage or a connecting node. The energy spent by a coverage node u to transmit its own data over an edge e is equal to $\epsilon_{elec}b + \epsilon_{amp}bW_e$. Every edges e_j in the path $P(u)$ also has to spent energy $\epsilon_{elec}b + \epsilon_{amp}bW_{e_j}$ to transmit data generated from u. Hence, total energy spent for transmitting u's data is given by $\epsilon_{elec}bh_u + \epsilon_{amp}b\sum_{e \in P(u)} W_e$. This justifies the expression for T_{RD}^{B}

In the expression for R_{RD}^{B} the term h_u is replaced by $h_u - 1$ as receiving energy for BS is ignored. □

4.2.3 Partial aggregation model

In the partial aggregation model, a packet has two parts: one is of fixed size k consisting of header, a trailer and a digest of all sensed data within the packet; the other is of variable length nb where n is the number of sensor's data aggregated and b is the number of bits to represent a sensor's sensed data and target id. To recover a sensor's data these b bits along with the k bits are needed. Thus, the size of a packet sent by a node u is given by

$$size_u = \begin{cases} k + (u_D + 1)b, & \text{if } u \text{ is a covereage node} \\ k + u_D b, & \text{otherwise} \end{cases} \tag{8}$$

It may appear that the data size for partial aggregation is more than that for the raw data model. But it is not so, as the b bits corresponding to raw data model includes the header, trailer, and the sensor's data and is thus more than the corresponding value for partial aggregation model.

Lemma 3: *The total energy cost T_{PA}^{B} for transmission and total energy cost R_{PA}^{B} for receiving in complete aggregation model is given by the following equation.*

$$\begin{aligned} T_{PA}^{B} = \epsilon_{elec}k(|E| + b \sum_{u \in V_{cov}} h_u) \\ + \epsilon_{amp}(k \sum_{e \in E} W_e + b \sum_{u \in V_{COV}} W_{P(u)}) \end{aligned} \tag{9}$$

$$R_{PA}^{B} = \epsilon_{elec}k|E'| + \epsilon_{elec}b \sum_{u \in V_{COV}} (h_u - 1) \tag{10}$$

Proof: As the packet size is given by equation (8), the expressions for total energy can be derived by combining results of Lemmas 1 and 2. □

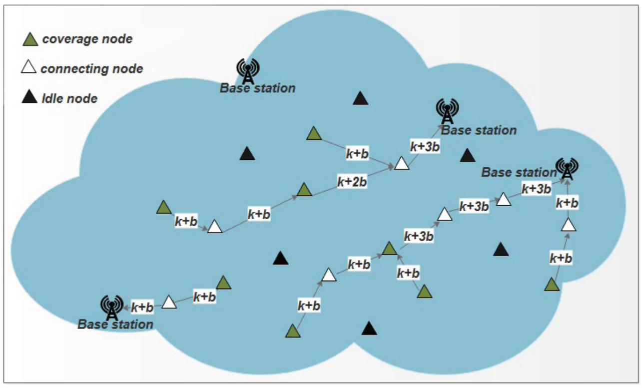
It is clear that $\sum_{e \in E} W_e$ would be minimum if G_B is a minimum spanning tree (MST) with the weight of each edge $e = W_e$. Also, $\sum_{u \in V_{COV}} W_{P(u)}$ would be minimum for a SPT. So, for the raw data model, SPT will give the tree with the minimum total Energy. But, this model unnecessarily replicates the header leading to wastage of bandwidth. In contrast, partial aggregation model seems to be more economical as it avoids replicating header and keeps the data generated by each coverage node. In subsequent discussion throughout this paper, partial aggregation model is followed. Figure 2 illustrates the partial aggregation model where triangles (green, white, black) indicates the deployed sensors and green, white and black among them are coverage, connecting and idle nodes respectively.

4.3 Data gathering forest for RaaS

Algorithm 2 is used to construct a data gathering forest $F = \{G_B : B \in \mathcal{B}\}$ and the total energy cost of F is $C(F) =$

$\sum_{B \in \mathcal{B}}(T^B_{PA} + R^B_{PA})$. The algorithm takes as input the solution of the first subproblem ie., the set of components each with at least one BS. A fictitious node r is added to the set V of all nodes. Then, an edge (B, r) with zero weight is added for each $B \in \mathcal{B}$. The weight of all other edges (u, v) are set as $d(u, v)^2$. The resulting graph has a single connected component comprising all V_i, $1 \le i \le k$. Next, the node r is used as the root to find Minimum Spanning Tree T_{MST} and SPT T_{SPT}. The children of r in each tree would be the BSs as the corresponding edges have minimum weight zero. Now, the root r and its adjacent edges are removed from both trees resulting in forests F_{MST} and F_{SPT}. It is clear that each tree of the forests will be rooted at one of the BSs. The one among the two forests which results in a minimum value of $C(F)$ is taken as the final output of the algorithm.

Figure 2 Packet size for partial aggregation (see online version for colours)



4.4 Finer alterations to reduce cost

After obtaining the data gathering forest F, we check if the cost could be further reduced by changing the parent of a node. We consider a node x whose parent is u, and there is another node v which is not a descendant of x such that $d(x, v) \le r_c$. We can make v the parent instead of u without violating the tree property since v is not a descendant of x (refer to Figure 3). We can find an expression for the change in cost if v is made the parent of x. Let h_x and h'_x be the current and changed hop counts of x. Also, let $P(x)$ and $P'(x)$ be the current and changed path from x to BS. Let $e_u = (x, u)$ and $e_v = (x, v)$, we define the terms $\Delta^T_{u,x}(v)$, $\Delta^R_{u,x}(v)$, and $\Delta^C_{x,u}(v)$ as follows:

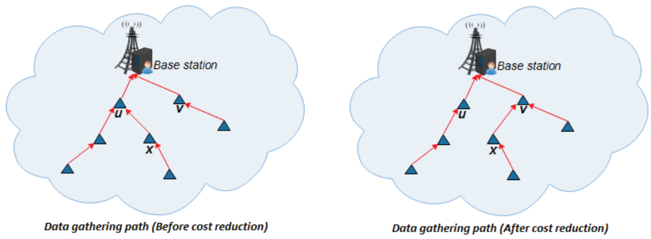
$$\Delta^T_{u,x}(v) = \epsilon_{amp}[k(W_{e_u} - W_{e_v}) + size_x(W_{P(x)} - W_{P'(x)})] + \epsilon_{elec} size_x(h_x - h'_x) \tag{11}$$

$$\Delta^R_{u,x}(v) = \epsilon_{elec}[k\delta_{u,v} + size_x(h_x - h'_x)] \tag{12}$$

$$\text{where, } \delta_{u,v} = \begin{cases} -1 & \text{if } u = BS \text{ and } v \ne BS \\ 1 & \text{if } u \ne BS \text{ and } v = BS \\ 0 & \text{otherwise} \end{cases}$$

$$\Delta^C_{u,x}(v) = \Delta^T_{u,x}(v) + \Delta^R_{u,x}(v) \tag{13}$$

Figure 3 Reducing the cost by changing parent (see online version for colours)



It is to be noted that v may belong to a tree different from the tree which has u. The next lemma shows that decrease in cost of transmission and reception are at least $\Delta^T_{u,x}(v)$ and $\Delta^R_{u,x}(v)$ respectively, if v is made the parent of x instead of u.

Lemma 4: *If $\Delta^C_{u,x}(v) > 0$, then the cost of Data gathering forest decreases if v is made the parent of x.*

Proof: Let T_{PA} and T'_{PA} be the energy cost of transmission when u and v are parents of x respectively. Then according to equation (9), $T_{PA} - T'_{PA} \ge \Delta^T_{u,x}(v)$. Here, we have considered all terms of equation (9) except the first. If the total number of edges remains the same the first term $k\epsilon_{elec}|E|$ has no effect and $T_{PA} - T'_{PA} = \Delta^T_{u,x}(v)$. But, if after the change u is found to be a connecting node with $u_D = 0$, the node u and adjacent edges would also be removed and in that case, $T_{PA} - T'_{PA} > \Delta^T_{u,x}(v)$.

Let R_{PA} and R'_{PA} be the energy cost for receiving before and after the change. We have to take into account that one of u and v may be a BS and we need not consider the receiving energy for BS. This explains the term $\delta_{u,v}$. If the number of edges remains the same, then $R_{PA} - R'_{PA} = \Delta^R_{u,x}(v)$. If number of edges decreases, by similar logic $R_{PA} - R'_{PA} > \Delta^R_{u,x}(v)$. Combining the above two inequalities

$(T_{PA} + R_{PA}) - (T'_{PA} + R'_{PA}) \ge \Delta^T_{u,x}(v) + \Delta^R_{u,x}(v) = \Delta^C_{u,x}(v)$.

Hence, if $\Delta^C_{u,x}(v) > 0$, the cost of tree decreases as v is made the parent of u. □

Algorithm cost reduction: We find for a node x whose parent is u, a set of candidate nodes A such that for any $c \in A$, $\Delta^C_{u,x}(c) > 0$. We find a node $v \in A$ for which $\Delta^C_{u,x}(v)$ is maximum and make v the parent of x. We update $size_u$ as it has lost x as a child. If $size_u$ becomes zero, we can remove u as it serves no purpose. The update is performed on each node w in the path from u to BS. If at a particular iteration no reduction is possible ($NoChange = true$), we stop.

5 Results and performance analysis

In this section we present the result of our experiment for the two subproblems

i sensor node selection

ii cost of data gathering forest.

The performance of the first subproblem is measured by the number of sensors selected and that of the second subproblem by the total energy cost per round. For our simulation, we have placed sensors and targets randomly over a surveillance region of 200m ×200m. The other parameters of the simulation are mentioned in Table 4. Algorithms 1–3 are implemented in Java and the ILPs are solved using GLPK (Available at: http://www.gnu.org/software/glpk/). The number of sensors is varied from 400 to 600 and the number of targets are varied from 100 to 200 (Table 4). For a given number of sensors and targets, we have considered 50 random placements of sensors and targets and tabulated the average values obtained from our experiments. The values of $\epsilon_{elec}$and ϵ_{amp} taken from Lindsey et al. (2002) are as follows:

ϵ_{elec} = 50 nJ/bit, ϵ_{amp} = 100 pJ/bit/m^2

Table 4 Simulation parameters

Parameter	*Value*
Surveillance region	200 m × 200 m
Sensing range	20 m
Communication range	40 m
No. of WSN provider	5
No. of sensors by each provider	80/100/120
No. of targets	100/150/200
No. of applications	5
k	10 byte
b	8 byte

Algorithm 1 Component Connectivity

Require: Adjacency Matrix Adj, Set of Base Stations $\mathcal{B}$, Set of Coverage Nodes cv
Ensure: Set of Minimum Connecting Nodes M, Set of Disconnected Nodes D

$M \leftarrow \emptyset$
$(C_{BS}, C_{WBS}) \leftarrow$ findComponents $(adj, \mathcal{B}, cv)$
repeat
  Find closest pair of components (c_1, c_2) s.t.
   $c_1 \in C_{WBS}, c_2 \in C_{BS} \cup C_{WBS}, \quad c_1 \neq c_2$
  if no such pair found **then** goto **disconnected**
  else
    $minPath \leftarrow$ set of nodes in the shortest path between c_1 and c_2
    $m_c \leftarrow c_1 \cup c_2 \cup minpath$
    replace c_2 by m_c
    remove c_1 from C_{WBS}
    $M \leftarrow M \cup minPath$
  end if
until $(C_{WBS} = \emptyset)$
disconnected: $D \leftarrow \cup_{c \in C_{WBS}}(c)$

5.1 Node selection

We look at the performance of our node selection from two perspectives. Firstly, We see how does our algorithm fare in comparison with an optimal one. Secondly, how much advantage collaboration of WSN gives compared to the case where each application is served by a separate WSN.

Algorithm 2 Data Gathering Forest

Require: Set of components with node set $V_1, V_2, \ldots, V_k$; Set of Base Stations $\mathcal{B}$
Ensure: Data-gathering forest F

$V \leftarrow \cup V_i$
$E \leftarrow \{(u,v) : u, v \in V, d(u,v) \leq r_c\}$
weight of edge $e \leftarrow W_e$ for all $e \in E$.
Add a fictitious node r to V
$E_x \leftarrow \{(r, B) : B \in \mathcal{B}\}$
$W_e \leftarrow 0$ for all $e \in E_x$
$E \leftarrow E \cup E_x$
Create a Tree T_{MST} by applying Prim's Algorithm (Cormen et al., 2009) on (V, E) starting from r.
Create a Tree T_{SPT} by applying Dijkstra's Algorithm (Cormen et al., 2009) on (V, E) using r as root.
Get F_{MST} from T_{MST} by removing r and E_x
Get F_{SPT} from T_{SPT} by removing r and E_x
$F \leftarrow$ one of F_{MST} or F_{SPT} whichever is of lower cost according to equations 9 and 10.

Algorithm 3 Cost Reduction

Require: Data gathering forest obtained by Algorithm 2; Set of all sensors V in the forest
Ensure: Data-gathering forest F of lower cost

repeat
  $NoChange \leftarrow true$
  for each $x \in V$ **do**
    $u \leftarrow$ parent of x;
    $A \leftarrow \{c : c$ is a neighbour of x, $c \neq u$, c is not a descendant of x, $\Delta^C_{u,x}(c) > 0\}$
    if $A \neq \phi$ **then**
      $v \leftarrow argmax(\Delta^C_{u,x}(c) : c \in A)$
      make v parent of x ; $NoChange \leftarrow false$;
      update w_D and $size_w$ for each node w in the path from v to BS
      for each node w in the path from u to BS **do**
        update w_D and $size_w$
        if $size_w = 0$ **then** remove w from its Tree and V
        end if
      end for
    end if
  end for
until $NoChange = true$
Find the new cost of F based on equations 9 and 10

5.1.1 Comparison with the optimal

Our approach for sensor selection involves using an ILP to select a minimum set of coverage nodes followed by Algorithm 1 to find an additional set of connecting nodes. We see that this ILP can be executed reasonably quickly even with a large number of sensors and targets. There is another alternative to select nodes, as presented in Section 3.3, where a single ILP is used to find coverage and connecting nodes simultaneously. This method even though optimal has the drawback that it has more variables and takes a very long time to execute and so can be attempted only for small sized problems.

We compare the performance of our preferred approach (ILP with Algorithm 1) with the optimal for a relatively small sized problem with 200 sensors and 40 targets. The results are shown in Table 5 with 5 random placements of sensors and targets. It is observed that our approach needs slightly more sensors than those needed by the optimal in most of the cases.

Table 5 Optimal vs. ILP followed by Algorithm 1

200 sensors 40 targets	*Optimal*	*ILP(coverage) + Algorithm 1*
seed value(1)	20	8+14 = 22
seed value(2)	18	9+12 = 21
seed value(3)	17	8+15 = 23
seed value(4)	18	8+14 = 22
seed value(5)	31	8+26 = 34

5.1.2 Standalone vs. collaboration

Here, we present a comparative study of sensor node requirement in stand-alone WSN with respect to that in a collaborative platform as shown in Table 6. The node requirement of the stand-alone case is computed as follows. We assume that there are 5 different applications each running on a separate WSN. The coverage and connecting nodes for each WSN are computed and their total value is put in columns 3 and 4 respectively. When we consider the collaboration we take the union of the set of targets for all 5 applications and then determine the coverage nodes by applying ILP and connecting nodes by the Algorithm 1. These values are put in columns 4 and 5. The figures clearly demonstrate the advantage of collaboration over stand-alone WSNs.

Table 6 Standalone vs. collaboration

		Standalone		*Collaboration*	
No. of targets	*No. of nodes*	*COV*	*CON*	*COV*	*CON*
100	400	159	152	60	25
	500	157	147	60	25
	600	154	138	57	24
150	400	219	162	72	22
	500	213	148	70	23
	600	202	143	68	24
200	400	268	142	80	19
	500	256	142	78	19
	600	247	142	74	13

5.2 Energy cost of data gathering forest

After the selection of nodes, the next phase is to create a data gathering forest using Algorithm 2 and further reduction of cost using Algorithm 3. We investigate which one of SPT or MST gives lower energy cost and what is the effect of applying cost reduction in either case. We also investigate a special case ($k = 0$ i.e., raw data model) where energy costs are given by equations (6) and (7). Finally, we give a comparison between the energy cost for standalone WSN and collaboration.

5.2.1 MST, SPT and cost reduction

Even though Algorithm 2 selects one of MST or SPT whichever gives lower cost, we have computed the energy cost of both the cases and applied cost reduction to them. Average results over 50 random cases are shown in Table 7 where 3rd and 5th columns give the energy cost of the forest employing SPT and MST respectively. The final costs after Algorithm 3 are put in 4th and 6th columns respectively. It is observed that in all cases except one (600 nodes, 100 targets) SPT gives a considerably lower cost than MST. Also, the effect of cost reduction is negligible on SPT but quite significant on MST.

Table 7 Energy cost in Joule /100 rounds)

Nodes	*Targets*	*SPT*	*SPT(F)*	*MST*	*MST(F)*
400	100	0.3764	0.3763	0.4520	0.3782
	150	0.3684	0.3683	0.4864	0.3691
	200	0.4053	0.4046	0.4830	0.4055
500	100	0.3558	0.3558	0.4131	0.3599
	150	0.4036	0.4036	0.4707	0.4057
	200	0.3902	0.3900	0.4531	0.3906
600	100	**0.4487**	**0.4487**	**0.3976**	**0.3619**
	150	0.3924	0.3924	0.0460	0.3985
	200	0.4514	0.4511	0.5118	0.4512

5.2.2 Raw data and partial aggregation

If we look at the equations (6) and (7) for the raw data model it is clear that SPT will give the minimum cost and then we expect that no change in cost would result after Algorithm 3. So, we ran some experiment to show that is indeed the case. But the argument that SPT is optimal for raw data model holds only for a single tree. If we apply Algorithm 3 for cost reduction, no change in tree structure and cost will result only if we impose a restriction that nodes are not allowed to move from one tree to another. In order to substantiate our claim, we ran Algorithm 3 with two conditions

i where the restriction applies

ii there is no such restriction.

The results are presented in Table 8 where SPTF(R) and SPTF(UR) denote the cost of the final tree in restricted and unrestricted respectively. It is observed that for raw data model $(k = 0)$ SPT and SPTF(R) are identical but SPT and SPTF(UR) are not. For partial aggregation model $(k = 40, b = 64)$, there is a considerable difference between SPT and SPTF(R) for the case where the number of targets is 200. We can make the following conclusions from these results.

- For the raw data model, SPT will produce the optimal tree and no further cost reduction is possible if nodes are not moved from one tree to another.
- Algorithm 3 has a better capacity of cost reduction by allowing nodes to change parent across different trees.

- Partial aggregation model behave closer to raw data model than complete aggregation model, and hence we are more likely to get lower energy cost by employing SPT in Algorithm 2.

Table 8 Energy cost for raw data and partial aggregation in Joules /100 rounds)

No. of sensors	*No. of target*	*k, b*	*SPT*	*SPTF(R)*	*SPTF(UR)*
400	100	0, 64	0.5809	0.5809	0.5809
		40, 64	0.6128	0.6128	0.6128
	150	0, 64	0.3216	0.3216	**0.3210**
		40, 64	0.3581	0.3581	0.3578
	200	0, 64	0.3769	0.3769	0.3769
		40, 64	0.4171	**0.4165**	0.4159

5.2.3 *Energy cost (standalone vs. collaboration)*

We have already shown in Table 5, how the collaboration helps in drastically reducing the sensor requirement compared to standalone WSNs. Naturally, we expect lower energy cost of data gathering forest for collaboration. The results are put in Table 9, where we have only considered SPT as SPTs are giving lower energy cost in most cases. The costs of the final tree after cost reduction in standalone and collaboration are shown in 3rd and 4th columns respectively. The results are also shown graphically in Figure 4 where the difference between these two cases are quite apparent.

Figure 4 Improved energy efficiency for collaboration

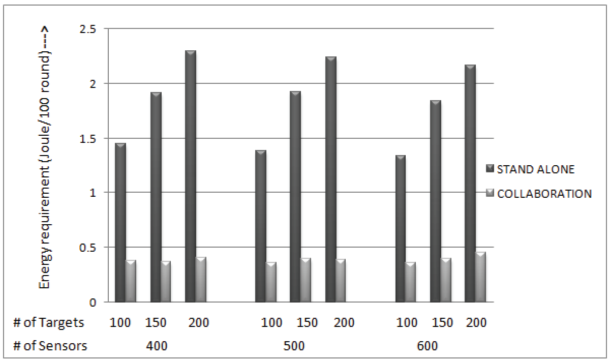


Table 9 Standalone vs. collaboration energy requirement in Joules/100 round

		Standalone	*Collaboration*
No. of nodes	*No. of targets*	*SPT*	*SPT*
400	100	1.45	0.38
	150	1.91	0.37
	200	2.29	0.41
500	100	1.38	0.36
	150	1.92	0.40
	200	2.24	0.39
600	100	1.34	0.36
	150	1.84	0.40
	200	2.16	0.45

6 Related works

Recently, networking issues in sensor-cloud like collaborative WSN platform have been conceived as a major research interest. Despite the upsurge in research on sensor-cloud, there is still a lack of research works that deal with the fundamental operations like data gathering. The underlying procedure of data gathering for multiple applications using the collaborated physical networks is one of the major challenges in sensor-cloud. In this section, we first analyse and discuss some of the established data gathering tree construction approaches in conventional WSN. Next, we review some of the major research works in the context of the sensor-cloud platform.

A series of works in the literature have adopted tree-based topology for constructing energy efficient data gathering tree (Goel and Estrin, 2005; Wu et al., 2010; Khan and Pandurangan, 2006; Dutta et al., 2013; Baksi et al., 2014) because of its simplicity. We classify these works into two categories. The first category of works is focused to solve the problem of tree construction for maximising the network lifetime, where the network lifetime is defined as the time until the first node completely depletes its energy. The second category of works is motivated to solve (Crowcroft et al., 2014; Kuo et al., 2016) the problem of tree construction for minimising the total energy spent for communication.

Goel and Estrin (2005) studied the problem of constructing efficient trees in stand-alone WSN to send aggregated information to a sink. They proposed a randomised algorithm that approximates the optimal tree. Khan and Pandurangan (2006), proposed a distributed algorithm that constructs an approximate minimum spanning tree (MST) in arbitrary networks. Wu et al. (2010) focused on the energy-efficient wake-up schedule in the data aggregation process. They also proposed algorithms to construct data aggregation trees such that both the energy consumption and the network throughput are within a constant factor of the optimal. The total energy consumption of all sensor nodes is also an important issue in data aggregation networks. Dutta et al. (2013) proposed a two-phase algorithm to compute a power-efficient data gathering tree in a distributed fashion where the energy lost in creating the tree is also taken into account. The lifetime of a node is computed as its residual energy divided by its cost. Baksi et al. (2014) adopted a technique to generate all possible spanning trees and modified it to generate only the optimal spanning tree using a branch and bound technique by iteratively discarding that partial tree with a lifetime less than or equal to the lifetime of a tree generated earlier. The lifetime obtained by this method is also shown to be better than that obtained by another heuristic algorithm like PEDAP.

Crowcroft et al. (2014) considered the power assignment problem for data aggregation. The authors measured the efficiency of a solution by total energy consumption, total transport capacity, latency and quality of the transmissions. Kuo et al. (2016) proved that the problem of constructing a data gathering tree that minimises the total energy cost is NP-complete and provided a 2-approximation algorithm. The modified version of the problem where relay nodes exist in the network is also proved to be NP-complete and a 7-approximation algorithm is also given.

Regarding Sensor-cloud, earlier research exploration was primarily concerned with the ideology, challenges, and architecture related aspects (Hassan et al., 2009; Alamri et al., 2013). A thorough survey on Sensor-cloud, its definition, concepts, benefits, and technical challenges was presented in Alamri et al. (2013). Some of the works addressed the problem of dynamic gateway allocation while transmitting the sensed data from the networks to the cloud (Misra et al., 2013). The virtualisation of sensor nodes was initially presented by Yuriyama and Kushida (2010). Subsequent research explored (Madria et al., 2014; Abdelwahab et al., 2016) in detail some aspects of virtualisation and sensor data management in the cloud. Abdelwahab et al. (2016) further explored in this direction by proposing a virtualisation algorithm to deploy VSNs on top of a set of physical devices. Nguyen and Huh (2011), presented some of the security aspects of sensor-cloud. Leontiadis et al. (2012) introduced the SenShare platform, which supports virtualisation in the sensor-node and network-level. In SenShare, a runtime layer on top of each sensor node supports multiple applications and a subset of WSN nodes form a VSN to support a specific application. Each independent subset executes an application, supporting network-level virtualisation. Chatterjee and Misra (2014) gave a theoretical model for Sensor-cloud infrastructure and then focused on a multi-organisation application scenario for tracking multiple targets. Zhu et al. (2017) proposed a multiple data delivery scheme for sensor-cloud users and compared their proposed scheme with the typical exclusive data delivery from the cloud to SC users.

The work in Sen et al. (2018), dealt with an optimal mapping of Sensor-cloud applications to host which achieves minimum latency while keeping loads on the hosts of a data centre suitably balanced. Madria (2018) discussed sensor cloud architectures with the aim to enable IoT related services using virtual sensors for the multiuser environment on top of physical wireless sensors to support multiple applications on-demand. Work in Dinh and Kim (2019), proposed an information-centric prediction-based integration model for the sensor cloud to reduce workloads and energy consumptions for resource-constrained sensors.

7 Conclusion

In this work, we have presented an approach to provide routing as a service (RaaS) to multiple applications on a Sensor-cloud platform with efficient use of resources (sensors and battery energy). We have considered different aggregation model for data gathering and derived the expressions for total energy consumption under these models. We show that under certain conditions SPT gives the optimal tree for the raw data model. We consider partial aggregation model to be most suitable in terms of packet size and retaining of all information and focus on this model for our experiments. Simulation results showed that the data gathering forest employing SPT results in minimum energy cost in most cases. We have also developed an iterative process which makes slight modifications in the structure of the forest and reduces the cost further. Finally, the advantage of Sensor-cloud over the stand-alone WSNs in terms of resources used have been demonstrated by the experimental results.

Future work in this area could be how to schedule convergecast in the Sensor-cloud taking into account the interference among simultaneous communications. The scheduling can use single frequency channels or multiple-frequency channels.

Acknowledgement

This work is an outcome of the R&D work supported by the Visvesvaraya PhD Scheme for Electronics & Information Technology, Government of India (implemented by the Digital India Corporation) and TEQIP Phase - III, University of Calcutta.

References

Abdelwahab, S., Hamdaoui, B., Guizani, M. and Znati, T. (2016) 'Cloud of things for sensing-as-a-service: Architecture, algorithms, and use case', *IEEE Internet of Things Journal*, Vol. 3, No. 6, pp.1099–1112.

Alamri, A., Ansari, W.S., Hassan, M.M., Hossain, M.S., Alelaiwi, A. and Hossain, M.A. (2013) 'A survey on sensor-cloud: architecture, applications, and approaches', *International Journal of Distributed Sensor Networks*, Vol. 9, No. 2, pp.917923.

Baksi, S., Sinha, A., Khatua, S. and Das, R.K. (2014) 'Finding the maximum lifetime data-gathering tree in sensor networks', *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 21–24 April, Singapore, pp.1–6.

Chang, J-H. and Tassiulas, L. (2004) 'Maximum lifetime routing in wireless sensor networks', *IEEE/ACM Transactions on networking*, Vol. 12, No. 4, pp.609–619.

Chatterjee, S. and Misra, S. (2014) 'Target tracking using sensor-cloud: sensor-target mapping in presence of overlapping coverage', *IEEE Communications Letters*, Vol. 18, No. 8, pp.1435–1438.

Chen, C-P., Mukhopadhyay, S.C., Chuang, C-L., Liu, M-Y. and Jiang, J-A. (2015) 'Efficient coverage and connectivity preservation with load balance for wireless sensor networks', *IEEE Sensors Journal*, Vol. 15, No. 1, pp.48–62.

Crowcroft, J., Segal, M. and Levin, L. (2014) 'Improved structures for data collection in wireless sensor networks', *INFOCOM, 2014 Proceedings IEEE*, pp.1375–1383.

Dinh, N-T. and Kim, Y. (2018) 'An efficient on-demand latency guaranteed interactive model for sensor-cloud', *IEEE Access*, Vol. 6, pp.68596–68611.

Dinh, N-T. and Kim, Y. (2019) 'An energy efficient integration model for sensor cloud systems', *IEEE Access*, Vol. 7, pp.3018–3030.

Dutta, A., Thakkar, K., Khatua, S. and Das, R.K. (2013) 'Power efficient data gathering for sensor network', *International Conference on Distributed Computing and Internet Technology*, pp.232–243.

Fazio, M. and Puliafito, A. (2015) 'Cloud4sens: a cloud-based architecture for sensor controlling and monitoring', *IEEE Communications Magazine*, Vol. 53, No. 3, pp.41–47.

Goel, A. and Estrin, D. (2005) 'Simultaneous optimization for concave costs: single sink aggregation or single source buy-at-bulk', *Algorithmica*, Vol. 43, No. 1-2, pp.5–15.

Haenggi, M. (2004) 'Opportunities and challenges in wireless sensor networks', *Handbook of Sensor Networks*, pp.21–38.

Hassan, M.M., Song, B. and Huh, E-N. (2009) 'A framework of sensor-cloud integration opportunities and challenges', *Proceedings of the 3rd International Conference on Ubiquitous Information Management and communication*, pp.618–626.

Khan, M. and Pandurangan, G. (2006) 'A fast distributed approximation algorithm for minimum spanning trees', *International Symposium on Distributed Computing*, pp.355–369.

Kuo, T-W., Lin, K. C-J. and Tsai, M-J. (2016) 'On the construction of data aggregation tree with minimum energy cost in wireless sensor networks: Np-completeness and approximation algorithms', *IEEE Transactions on Computers*, Vol. 65, No. 10, pp.3109–3121.

Leontiadis, I., Efstratiou, C., Mascolo, C. and Crowcroft, J. (2012) 'Senshare: transforming sensor networks into multi-application sensing infrastructures', *European Conference on Wireless Sensor Networks*, pp.65–81.

Lindsey, S., Raghavendra, C. and Sivalingam, K.M. (2002) 'Data gathering algorithms in sensor networks using energy metrics', *IEEE Transactions on parallel and distributed systems*, Vol. 13, No. 9, pp.924–935.

Madria, S. (2018) 'Sensor cloud: sensing-as-a-service paradigm', *2018 19th IEEE International Conference on Mobile Data Management (MDM)*, pp.3–6.

Madria, S., Kumar, V. and Dalvi, R. (2014) 'Sensor cloud: A cloud of virtual sensors', *IEEE software*, Vol. 31, No. 2, pp.70–77.

Misra, S., Bera, S., Mondal, A., Tirkey, R., Chao, H-C. and Chattopadhyay, S. (2013) 'Optimal gateway selection in sensor–cloud framework for health monitoring', *IET Wireless Sensor Systems*, Vol. 4, No. 2, pp.61–68.

Misra, S., Chatterjee, S. and Obaidat, M.S. (2017) 'On theoretical modeling of sensor cloud: A paradigm shift from wireless sensor network', *IEEE Systems journal*, Vol. 11, No. 2, pp.1084–1093.

Nguyen, T-D. and Huh, E-N. (2011) 'An efficient key management for secure multicast in sensor-cloud', *2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering (CNSI)*, pp.3–9.

Sen, B.K., Khatua, S. and Das, R.K. (2015) 'Target coverage using a collaborative platform for sensor cloud', *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS 2015)*, pp.1–6.

Sen, B.K., Khatua, S. and Das, R.K. (2018) 'Optimal mapping of applications on data centers in sensor-cloud environment', *Advanced Computing and Systems for Security*, pp.1–12.

Wu, Y., Li, X-Y., Li, Y. and Lou, W. (2010) 'Energy-efficient wake-up scheduling for data collection and aggregation', *IEEE Transactions on parallel and distributed systems*, Vol. 21, No. 2, pp.275–287.

Yuriyama, M. and Kushida, T. (2010) 'Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing', *2010 13th International Conference on Network-Based Information Systems*, pp.1–8.

Zhu, C., Leung, V.C., Wang, K., Yang, L.T. and Zhang, Y. (2017) 'Multi-method data delivery for green sensor-cloud', *IEEE Communications Magazine*, Vol. 55, No. 5, pp.176–182.