Value of information based sensor ranking for efficient sensor service allocation in service oriented wireless sensor networks

Value of Information based Sensor Ranking for Efficient Sensor Service Allocation in Service Oriented Wireless Sensor Networks

Sourabh Bharti, *Student Member, IEEE,* K. K. Pattanaik, *Senior Member, IEEE,* and Paolo Bellavista, *Senior Member, IEEE*

Abstract—In service oriented Wireless Sensor Networks (WSNs), a sensor service allocation mechanism should consider the usage context of the required sensor services in the application. Recent sensor service allocation mechanisms rank a sensor service uniformly for all incoming applications without considering its usage context in the respective application. This paper proposes a Value of Information based Sensor service Ranking Mechanism (VoISRAM) that models the rank of a sensor service as a Value of Information (VoI) attribute while taking into account its usage context in the corresponding application. Unlike existing research, VoISRAM is evaluated for its ability to complement existing gateway services for VoI based sensor service selection. Performance analysis of VoISRAM suggests its efficiency in maintaining a trade-off between application specific QoS requirements and overall energy consumption in the network. Further comparative evaluation with existing sensor service ranking mechanisms manifests the incremental addition made by VoISRAM in this area. Simulation results show that VoISRAM outperforms existing sensor service ranking mechanisms in terms of meeting application QoS requirements. In addition to this, VoISRAM shows an impressive 13% improvement in network lifetime as well.

Index Terms—Wireless Sensor Network, Value of Information, Sensor ranking, Sensing as a service

1 Introduction

While the data communication networks are assessed by measuring the QoS provided by them in terms of latency, delay, bandwidth, etc., the WSNs are evaluated in terms of the Quality of Information (QoI) provided by them to the applications [1]. However, the burgeoning involvement of sensor networks in Internet of Things (IoT) necessitates the information products' assessment in a more diverse and dynamic environment.

In service oriented WSN [30], an information product is the sensed data (constituting the sensor data points from one or more sensors) consumed by an application query. For instance, data points from temperature and smoke sensors constitute the information product deliverable to a fire detection application. Traditionally, information products are assessed uniformly for every application query in terms of their associated data quality metrices such as information accuracy [24]. However, WSN based applications have different QoS requirements and specifications. Thus, a requirement specific design i.e. no one-size-fits-all solution [2] is required.

A sensor node can be enabled with various service types such as temperature and humidity service. In service oriented WSNs, a sensor service denotes one of the services enabled on the sensor nodes. Sensor services are matched against the application requirement so that the best available sensor service(s) can serve the application. Traditional matchmaking mechanisms [11][12] rank a sensor service based on the associated QoI without considering its usage context in the respective application. Usage context is the requirements of the application query against which the information product is going to be evaluated. Evaluation of an information product in different usage contexts is termed as Value of Information (VoI), formally defined as: An assessment of the utility of an information product when used in a specific usage context [1]. *Value* of a sensor service or an information product associated with a sensor service is defined as its importance in the particular usage context. A sensor service can be more valuable in one usage context while its importance can vary in another. Taking cue from this, we argue that a sensor service in WSNs can not be ranked uniformly without considering its usage context in the respective application. We defend this hypothesis considering spatial accuracy and staleness of data as the key parameters in an application of environmental monitoring a farm [4].

In this application, sensor services measuring various environmental parameters such as temperature, humidity, etc., are deployed in the farm as to take automated decisions by the sensor-actuator control system. For an instance, the temperature readings sensed by a sensor service can actuate a water sprinkler based on the settings. We argue that a sensor service s_i can be *valued* differently in two usage contexts in terms of its spatial accuracy and staleness of the information products associated with s_i.

Requirements associated with the area/location are termed as spatial requirements. In one usage context, a task of measuring the humidity level in a farm segment (a circular region shown in Fig. 1) is injected into the WSN.

Sourabh Bharti and K. K. Pattanaik are with Wireless Sensor Networks Laboratory, ABV-Indian Institute of Information Technology and Management, Gwalior, India.

Paolo Bellavista is with University of Bologna, Bologna, Italy.

Any sensor service (s_i for that matter) of the required service type (humidity) installed in that segment can meet the spatial requirement of the task. Thus, every humidity sensor service is *valued* the same in this usage context. In another usage context, a task to examine the working status of a water sprinkler installed in the same segment at location *p(x,y)* (specified by a triangle in Fig. 1) is injected into WSN. In this usage context, the sensor service deployed nearest to that water sprinkler (let it happen to be s_i) is *valued* more than the other sensor services deployed inside the same segment. Hence, s_i is *valued* differently in the above usage contexts.

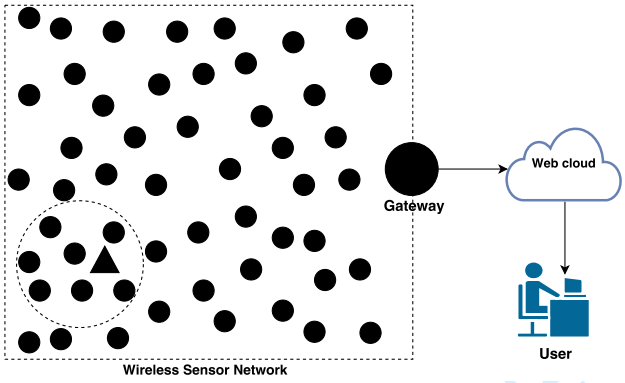


Fig. 1: VoI aspect of spatial accuracy

Staleness or freshness of sensed data is a time-related quality dimension that measures to the level of synchronization between the data originator (sensor node) and the information system processing the data (gateway)[5]. In a usage context, a task to investigate the current status of the water sprinkler is injected into WSN. This requires immediate querying the sensor service deployed nearest to the water sprinkler (s_i for that matter) and the fresh information products from s_i are *valued* more as compared to the stale ones. In another usage context, a task to investigate for how long the water sprinkler has been down due to technical failure is injected into WSN. For this usage context, the fresh information products from s_i are equally *valued* as stale ones. Hence, the stale information products from s_i are *valued* differently in the above usage contexts. Thus, it can be inferred that a sensor service can not be ranked uniformly without considering its usage context in the respective application. Our contributions in this paper can be summarized as follows.

1) After extensive need analysis, we associate the concept of VoI with sensor service ranking and propose VoISRAM that ranks the sensor services by considering their usage context in the respective application.
2) We model the sensor service rank as a VoI attribute and propose a cost function for rank assignment (Section 3.2). The cost function is designed around both application specific and network specific QoS requirements to maintain the trade-off among them. Maintaining this trade-off is crucial for resource constrained WSN.
3) VoISRAM is executed at the gateway and requires average *1-hop* delay estimation and residual energy information of sensor nodes. To make these parameters available beforehand at the gateway, we develop average *1-hop* delay estimation (Section 3.4) and residual energy prediction (Section 3.5) models.
4) VoISRAM is integrated with existing gateway service (TRAPS) [3] to evaluate its overall performance in terms of effects on application specific and network specific QoS parameters (Section 4). Later we prove our rationale behind integrating VoISRAM with TRAPS (Section 5). We compare VoISRAM with state-of-the-art sensor service ranking mechanisms in terms of network lifetime and ability to fulfill application specific QoS requirements (Section 6). Simulation results suggest an impressive 13% improvement in network lifetime.

2 Related Work

Sensor search and ranking mechanisms for WSN range from geographical indexing [6], clustering [7] to content based searching [8] which analyzes the relevancy of available sensor services regarding the incoming queries. Web based sensor discovery frameworks [26-28] do not take scarce network resources into consideration. These discovery frameworks produce a limited number of semantically equivalent services satisfying the search criteria. The discovery frameworks are not able to differentiate between the retrieved services and to rank them in an order that is beneficial for the scarce network resources as well [11]. The geospatial indexing helps the discovery engine locate the gateway(s) that are likely to contain the services with respect to the queries [6]. On the other hand, the ranking based resource allocation is the next step after sensor search and discovery. VoISRAM ranks the discovered sensor services to select the best sensor service(s) among them such that the trade-off between application specific and network specific QoS requirements is maintained.

Various QoS based ranking mechanisms [9][10] are proposed that are based on user feedback and ratings. Recent energy aware sensor ranking mechanisms [11][12] estimate the access cost of a sensor service. The cost function proposed in [11] constitutes the residual energy level of the respective sensor service and its importance in the network. Another sensor search mechanism [12] considers data accuracy, reliability, availability, etc. for sensor service ranking.

The usage context of a sensor service in a particular application plays a significant role in meeting application specific QoS requirements. Recent user feedback based mechanism [13] is not applicable in many remote scenarios. All of the existing energy aware sensor ranking mechanisms focus on network aspects but do not consider the key application requirements (spatial accuracy and staleness of data) discussed in the previous section. Moreover, a sensor service is ranked uniformly without considering its usage context in the corresponding application.

Existing methods for web services ranking [14][15] consider the application specific parameters. The extensive resources at hand help the ranking mechanisms designed for web services to be more application oriented. Moreover, the

network parameters such as residual energy are not considered while ranking a web service. In resource constrained scenarios such as WSN, the network specific issues can not be ignored while designing a ranking mechanism. Hence, the ranking mechanisms proposed for web services are not applicable in WSN.

To the best of our knowledge, none of the sensor service ranking mechanisms [11-13] is evaluated in terms of their effects on application QoS requirements. On the other hand, in WSNs, the design of a gateway service for an efficient network and system management [2] should maintain a trade-off between application specific QoS requirements and network resource consumption. Thus, a sensor service ranking mechanism is required that takes into consideration the usage contexts of sensor services and helps the gateway service meeting applications' QoS requirements while minimizing the energy consumption.

3 VoISRAM

VoISRAM uses network specific (residual energy) and application specific (spatio-temporal accuracy) QoS parameters to model the sensor service as a VoI attribute. In this section, a sensor service access cost function is proposed that serves as a metric for the ranking. The development of proposed cost function requires average *1-hop* delay and residual energy information beforehand at the gateway (place of VoISRAM execution). The average *1-hop* delay estimation and residual energy prediction models are also discussed in the following.

3.1 Preliminaries

This section discusses the application model, task model and network model used throughout this paper. An important attribute *sensor-to-task relevancy* in the context of sensing as a service is also defined.

3.1.1 Application model

This work considers both *push-based* and *pull-based* applications. *Push-based* applications involve continuous or periodic sensing and reporting of the sensed data at the gateway. On the other hand, *pull-based* applications involve on-demand querying for the fresh or stale sensor data. A *pull-based* application query is decomposed into various network level sensing tasks [29] to be scheduled inside the WSN.

3.1.2 Task model

A simple WSN query arriving at the gateway has the following semantics.

SELECT sensor data **FROM** *req_service_types* $\{s\}$ **WHERE** locations $\{L\}$ **DURATION** D **EVERY** frequency e.

The **SELECT** clause specifies the data values to be sensed, **FROM** clause specifies the set of required service types such as temperature and humidity services and **WHERE** clause may specify a set of locations. A location L_i either represents region of interest (x_i, y_i, r) or a certain location (x_i, y_i) (see section 3.2.1 for details). **DURATION** clause specifies the sensing time duration D $[t_b, t_e, d_t]$ where, t_b, t_e are beginning and ending time instants respectively. Real time queries requires immediate sensing and reporting

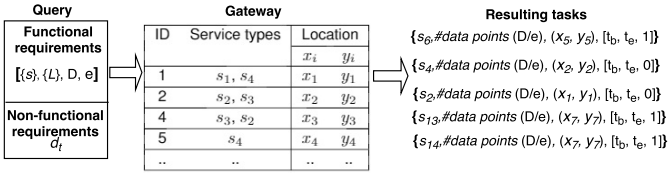


Fig. 2: Query to task derivation

of all the sensed data within a specified delivery deadline d_t. **EVERY** clause specifies the sensing time interval (frequency (e) of the observations).

The query to task derivation is performed at the gateway that is assumed to contain the sensor service meta-data such as ID, service types and location. A query may require data from different sensor services installed at different locations for its successful execution. For example, an application query may require temperature and smoke sensor services deployed at the different locations in the monitored area. Thus we adopt a distributed query execution strategy in which the query is decomposed into multiple sensing tasks inheriting the query requirements in the following manner.

Each task (T_i) is an atomic entity constituting a unique required service type (*req_service_type*) associated with a location (L_i). Given the frequency (e) and D $[t_b, t_e, flag]$ the number of data points to be sensed ($\frac{D}{e}$) are derived for each sensing task. We characterize a task as a 4-tuple $\langle req\_service\_type, number\ of\ data\ points, R^{T_i}_{spat}, R^{T_i}_{temp} \rangle$ where $R^{T_i}_{spat}$ is the required geographical (spatial requirement) location associated with T_i and $R^{T_i}_{temp}$ is a time window with a delivery deadline (in case of real time queries).

3.1.3 Network model and topology management

The whole network is governed by multiple gateways in a distributed manner where each gateway manages a group of 40-50 sensor nodes constituting a sub-network. The sub-network is modeled as a Destination Oriented Directed Acyclic Graph (DODAG), routed at a single destination termed as root/gateway. The sensor nodes are stationary and homogeneous (equipped with same sensing, computing and storage capabilities) working on IEEE 802.15.4 MAC protocol with beacon enabled mode. DODAG follows a parent-child topology paradigm in which every sensor node transmits the sensed data packets to its parent that is selected on the basis of predefined metrics. We choose the distance from the root as the parent selection metric to minimize the hop count between the gateway and the sensor service for reduced end-to-end delay required in IoT sensory environment.

Every time a node leaves the DODAG topology (due to energy drain or other failures), it is maintained using control packets called DODAG information objects (DIO) [17]. The routing protocol in DODAG employs Trickle algorithm [25] to efficiently maintain the routing topology, enabling quick reaction to topological changes while minimizing overhead during stable conditions. According to this algorithm, if a node's parent leaves the network due to energy drain, it selects another node with lower rank as its new parent. The rank of a node is decided on the basis of the pre-defined metrics (distance from root in our case). Every gateway has

the logical view of its respective sub-network's topology that gets updated every time a sensor node leaves the sub-network. The logical view represents the connections between sensor nodes in the form of an adjacency matrix.

3.1.4 Sensor-to-task relevancy

The first step in ranking the sensor services is to select the sensor services that provide the same service type as required (*req_service_type*) by the task. Based on the service type provided by a sensor service s_i, its *sensor-to-task relevancy* ($\mathrm{r}_{ct}^{s_i}$) (represented by equation 1) is calculated.

$$\mathrm{r}_{ct}^{s_i} = \begin{cases} 1, & \text{if } \textit{req\_service\_type} == \textit{service\_type} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Based on the above equation, the suitable sensor services are selected and termed as *candidate sensor services* that participate in the ranking process.

3.1.5 Gateway storage

This work considers gateway as a powerful machine capable to store and process a fair amount of data. Since the number of sensor nodes in a sub-network is kept as 40-50, the size of the storage does not go out of the bound. The sensed data is stored at the gateway as a 4-tuple ⟨*sensor id, service type, sensed value, sensing time*⟩ in a table named as ***sensor data***. The information about sensor nodes in a sub-network is stored as 4-tuple ⟨sensor id, service type, location info, residual energy⟩ in a table named as ***metadata*** that is placed at the respective gateways at the time of deployment. ***sensor data*** table is also placed at the respective gateways, initially in the form a schema that is filled with the ⟨*sensor id, service type, sensed value, sensing time*⟩ tuples as the time progresses.

3.2 Sensor service rank as VoI attribute

Quality of an information product in WSN can be determined by various parameters such data accuracy, accessibility, etc. On the other hand, quality coupled with the usage context determines the *value* of an information product. Hence, VoI is a function of QoI assessed in a specific context (equation 2).

$$\mathrm{VoI} = f_{context}(\mathrm{QoI}) \tag{2}$$

In this section, we consider different usage contexts of the required information to model VoI based spatial and temporal accuracy parameters. The modeled parameters are used to determine the cost of a sensor service that serves as a basis for the ranking.

3.2.1 Spatial accuracy

The gateway extracts the usage context from the task and identifies the *candidate sensor services*. It is assumed that the sensor services and $\mathrm{R}_{spat}^{T_i}$ are in a two dimensional euclidean space. We consider two usage contexts: (1) $Cntxt_1^{spat}$, when $\mathrm{R}_{spat}^{T_i}$ specifies a location $p(x, y)$ and (2) $Cntxt_2^{spat}$, when $\mathrm{R}_{spat}^{T_i}$ specifies $p(x, y)$ with a radius $r > 0$ and any sensor service inside the circle with $p(x, y)$ as center and r as radius satisfies $\mathrm{R}_{spat}^{T_i}$.

If the spatial accuracy (represented as QoI_1) is considered to be a function f_{QoI_1} of given usage context, it can be represented using equation 3. For $Cntxt_1^{spat}$, f_{QoI_1} is assigned the *value* of the euclidean distance $d(p, q)$ between a *candidate sensor service* q and p. For $Cntxt_2^{spat}$, $d(p, q)$ is compared with r. In case of $d(p, q) \leq r$, f_{QoI_1} is assigned a *value* of 0 and $d(p, q)$ otherwise.

$$f_{\mathrm{QoI}_1} = \begin{cases} d(p,q), & \text{if } Cntxt_1^{spat} \\ 0, & \text{if } Cntxt_2^{spat} \mid d(p,q) \leq r \\ d(p,q), & \text{if } Cntxt_2^{spat} \mid d(p,q) > r \end{cases} \tag{3}$$

Equation 3 states two spatial usage contexts of the required sensor service. A sensor service nearest to the required location is *valued* more in the $Cntxt_1^{spat}$. On the other hand, all sensor services belonging to the specified region are *valued* the same in the $Cntxt_2^{spat}$. However, sensor services not belonging to the specified region are ranked on the basis of their distance from the p. The parameters r, $d(p, q)$ and f_{QoI_1} are measured in unit of length.

3.2.2 Temporal accuracy

This work considers two usage contexts: (1) $Cntxt_1^{temp}$ and (2) $Cntxt_2^{temp}$, for temporal requirements as well. $Cntxt_1^{temp}$ represents real time usage context specifying the sensing time window [t_b, t_e] along with a delivery deadline (d_t) imposing the condition: $\mathrm{t}_{curr} \leq \mathrm{t}_b < \mathrm{t}_e$ where t_{curr} represents the current time. This involves immediate sensing of the requested data points, bundling the data points into required number of packets (*pckt*) and finally routing the data packets to the gateway. If m represents the number of sensor services between the gateway and a sensor service, $delay^{avg}$ represents the average *1-hop* delay occurred in transmitting one packet, T_{sense} is the time taken in sensing one packet of information, the total time (T_{s_i}) taken in this process can be represented as

$$\mathrm{T}_{s_i} = (pckt)\Big(\sum_{k=1}^{2m-2} delay^{avg} + \mathrm{T}_{sense}\Big) \tag{4}$$

$Cntxt_2^{temp}$ represents non real time usage context specifying a sensing time window [t_b, t_e] with the condition: $\mathrm{t}_b < \mathrm{t}_e \leq \mathrm{t}_{curr}$. This does not require immediate sensing/querying for sensed data. These tasks can be served by the stale data stored in the ***sensed data*** table at the gateway.

This work considers a multi-application hybrid data gathering (combination of *push* and *pull* based techniques) scenario. The data points sensed by *push* based applications are stored at the gateway. Since there are requirement overlaps among different applications [3], tasks with $Cntxt_2^{temp}$ usage context are served by the stale data stored at the gateway and does not involve querying the sensor services. However, this work also takes $Cntxt_2^{temp}$ usage context into consideration that is elaborated in subsequent section.
Similar to spatial accuracy, if the temporal accuracy (represented as QoI_2) is considered to be a function f_{QoI_2} of given usage context, it can be represented using equation 5.

$$f_{\mathrm{QoI}_2} = \mathrm{T}_{s_i} \tag{5}$$

Equation 5 states that for tasks with $Cntxt_1^{temp}$ usage context, sensor services with minimum estimated delay (T_s) are ranked good. The parameters $delay^{avg}$, T_{s_i}, T_{sense} and f_{QoI_2} are measured in unit of time.

3.2.3 Handling tasks with non-real time usage context

This involves searching for the required data points at the gateway storage. First of all available sensed data points stored in the ***sensed data*** table are filtered on the basis of the *service type* and *req_service_type* followed by fetching the *sensor id, sensed value* and *sensing time* of the relevant data points by

select *sensor id, sensed value, sensing time* as **T from *sensed data*** **where** *req_service_type==service type*

All sensor ids in table **T** are matched against sensor ids in ***metadata*** table to fetch their location information that helps calculating f_{QoI_1} values of the sensor services for the tasks with $Cntxt_2^{temp}$ usage contexts using equation 3. For $Cntxt_2^{temp}$, f_{QoI_1} value of a sensor service is calculated as explained in Section 3.2.1.

On the other hand, f_{QoI_2} value of a sensor service is calculated on the basis of the staleness of its sensed data points stored in ***sensed data*** table at the gateway. Staleness (S_{curr}^D) [5] of a data point D stored at the gateway varies with time and can be defined as

$$S_{curr}^D = \text{t}_{curr} - \frac{\text{t}_{last}^D \times (\text{N}^D + 1) \; - \; \text{t}_0^D}{\text{N}^D} \tag{6}$$

where t_{curr} is the current time instant, t_{last}^D is the most recent time instant at which the D was sensed by an application, t_0^D is the time instance at which D was sensed for the first time and N^D is the total number of times D is sensed [5]. S_{curr}^D shows the staleness of D with reference to the current time instance (t_{curr}). In this case, all available data points sensed by the relevant sensor services (Table **T**) recorded between [t_b, t_e] are *valued* the same (=0). If required data points recorded between [t_b, t_e] are not available at the gateway storage, the data points recorded nearest to the [t_b, t_e] are *valued* more. The *value* ($S_{[\text{t}_b,\text{t}_e]}^{D_{s_i}}$) of a data point D_{s_i} belonging to a sensor service s_i with respect to [t_b, t_e] is calculated as follows.

$$S_{[\text{t}_b,\text{t}_e]}^{D_{s_i}} = min\Big\{\Big|S_b^{D_{s_i}}\Big|, \Big|S_e^{D_{s_i}}\Big|\Big\} \tag{7}$$

Equation 6 defines the staleness of D with reference to the current time instance. We use equation 6 to calculate the staleness of D_{s_i} (equation 7) with respect to [t_b, t_e]. Thus the f_{QoI_2} value in $Cntxt_2^{temp}$ is calculated as

$$f_{\text{QoI}_2} = \begin{cases} 0, & \text{if } Cntxt_2^{temp} \mid t_{last}^{D_{s_i}} \in [t_b, t_e] \\ S_{[\text{t}_b,\text{t}_e]}^{D_{s_i}}, & \text{if } Cntxt_2^{temp} \mid t_{last}^{D_{s_i}} \notin [t_b, t_e] \end{cases} \tag{8}$$

The above equation implies that sensor services associated with the data points nearest to the specified time window are ranked good for tasks with non-real time usage context.

3.2.4 The cost function

This section discusses the cost function for both real time and non-real time usage contexts of a task. Since tasks with $Cntxt_1^{temp}$ involve querying the sensor service by sending request inside the network, we propose an energy-aware cost function that serves as the basis for sensor service rank assignment for $Cntxt_1^{temp}$ usage context. Network specific QoS parameter should be considered in any sensor service ranking mechanism designed for resource constrained WSN. To address the aforementioned, we choose average residual energy level ($Res_{s_i}^{avg} = \frac{\sum_{k=1}^{m+1} Res_{s_k}}{m+1}$ Joules) of the path from s_i to the gateway to be included in the weighted cost function (equation 9). It is to be noted that the path includes s_i as well.

$$\text{Cost}_{s_i} = min\Big\{w_i\big(\widehat{f_{\text{QoI}_1}} + \widehat{f_{\text{QoI}_2}}\big)\Big\} \tag{9}$$

$$w_i = \frac{E_{max} - Res_{s_i}^{avg}}{E_{max} - E_{min}} \tag{10}$$

where, f_{QoI_1} and f_{QoI_2} are calculated using equations 3 and 5 respectively, E_{max} and E_{min} represent the maximum and minimum energy levels of a sensor service respectively.

Tasks with $Cntxt_2^{temp}$ do not involve sending the request inside the network. Thus, w_i is set as 1 in this case and the value of f_{QoI_2} is calculated using equation 8. A sensor service with minimum cost value is ranked highest and so on. For tasks with $Cntxt_2^{temp}$, the sensor services are ranked on the basis of their location and the staleness of their sensed data available at the gateway. This ranking does not result in querying the highest ranked sensor service at all. However, the available sensed data of the highest ranked sensor service is delivered to the task. On the other hand, the highest ranked sensor service is queried for tasks with $Cntxt_1^{temp}$.

Since f_{QoI_1} and f_{QoI_2} are in different units of measure, we normalize them as $\widehat{f_{\text{QoI}_1}}$ and $\widehat{f_{\text{QoI}_2}}$ respectively to prevent biasing in the cost function. The ranking method proposed in [12] uses similar normalization technique to re-scale the context properties (e.g. accuracy, reliability, latency) in the range of [0, 1]. For every *candidate sensor service* s_i, its corresponding parameters: f_{QoI_1} and f_{QoI_2} are calculated. Suppose x_j^i represents the calculated value of jth parameter corresponding to sensor service s_i. Min-max normalization technique is used to re-scale the data in the range of [0, 1]. Once the data corresponding to every *candidate sensor service* is available, x_j^{min} (minimum value corresponding to jth parameter) and x_j^{max} (maximum value corresponding to jth parameter) can be easily estimated. Every x_j^i value in the data is normalized as $\frac{|x_j^i - x_j^{min}|}{x_j^{max} - x_j^{min}}$.

The first term of the cost function states that sensor services closer to the requested location cost less. The second term indicates that in case of non-real time temporal requirements, data points nearest to the specified time window cost less whereas, in case of real time temporal requirements, the hop-count between the gateway and selected sensor service is directly proportional to the cost which helps in reducing the end-to-end delay. The weight w_i helps maintaining the trade-off between application specific and network specific QoS requirements. In other words, paths including sensor services with high residual energy level help reducing the cost.

Since the ranking is performed on the basis of a cost function designed around the VoI aspects of spatial and temporal accuracy it can be stated that a sensor service rank is successfully modeled as a VoI attribute.

3.3 Ranking algorithm and time complexity

Algorithm 1: VoISRAM

Input : tasks with QoS requirements
Output: task allocation

1 **begin**
2 **if** *Task T_i has real time temporal requirement* **then**
3 **for** *every sensor service s_i* **do**
4 **if** $r_{ct}^{s_i} == 1$ **then**
5 *candidate*[j] = s_i;
6 j++;
7 **end**
8 **end**
9 **for** *every candidate sensor service* **do**
10 ▷ compute f_{QoI_1}, f_{QoI_2} using equations 3, 5
11 ▷ compute Res_{s_i} and $Res_{s_i}^{sum}$
12 ▷ compute the access cost using equation 9
13 **end**
14 ▷ rank the *candidate sensor services* according to lowest access cost first
15 ▷ assign task to the highest rank *candidate sensor service*
16 **end**
17 **if** *Task T_i has non-real time temporal requirement* **then**
18 // search for available data in ***sensed data*** table
19 ▷ filter on the basis of $req\_service\_type$
20 ▷ fetch the relevant sensor ids and their location using ***metadata*** table
21 **for** *every relevant sensor service* **do**
22 ▷ compute f_{QoI_1}, f_{QoI_2} using equations 3, 8
23 ▷ compute the access cost using equation 9 with w_i==1
24 **end**
25 ▷ rank the sensor services according to lowest access cost first
26 ▷ available sensed data of the highest ranked sensor service is delivered to the task
27 **end**
28 **end**

This section explains the computational steps involved in VoISRAM. The gateway service gathers the functional ($req\_service\_type$) and QoS (spatio-temporal accuracy) requirements of the incoming tasks. If the task has real time temporal requirements, all the sensor services with same service type as $req\_service\_type$ are filtered and inserted in an array named as *candidate sensor service*. These services participate in the ranking process as explained in Step 10-15 of Algorithm 1.

Tasks with non-real time temporal requirements do not require querying the sensor services. These tasks are served by the available sensed data (sensed by a *push based* mechanism) at the gateway. For them, the sensor services whose sensed data is available at the gateway participate in the ranking process and the available sensed data of the highest ranked sensor service is provided to the task (Step 25, 26 of Algorithm 1).

The complexity of the mechanism can be estimated as *O(NCS* $*m$), where *NCS* is the number of *candidate sensor services* participating in the ranking process and m is the number of intermediate sensor nodes between the gateway and the sensor service in question. Let N be the total number of sensor services in the network, then $N \gg NCS$ and $N \gg m$. Thus, the proposed ranking mechanism is computationally efficient.

Following sections discuss the average *1-hop* delay and residual energy prediction models used in the process of developing the proposed cost function.

3.4 Average *1-hop* delay model

In wireless networks such as WSN, the $delay^{avg}$ (mentioned in equation 4) is composed of two parts : (a) the average waiting time a packet spends in the queue (W_q) and (b) transmission time (T_{trans}). To analyze the average waiting time in the queue (equation 11), a queuing model of type (M/M/1) : (GD/L/∞) [18] is considered.

$$W_q = \frac{1-(\lambda/\mu)^L-(1-\lambda/\mu)(L(\lambda/\mu)^L+1-(\lambda/\mu)^{L+1})}{\mu(1-\lambda/\mu)(1-(\lambda/\mu)^{L+1})} \tag{11}$$

According to this model, the sensor service acts as a server with the queue size of L. Packet arrival rate λ follows Poisson distribution and service time μ is exponentially distributed. T_{trans} for a packet depends on various factors such as number of re-transmissions (n_R), the awake time of the receiver (T_{aw}), sleep time of a sensor service (T_{sl}) and MAC layer back-off time (T_b). Hence, the T_{trans} can be represented as follows.

$$T_{trans} = n_R(T_{aw}+T_{sl}) + E[T_b] + R(0,T_{sl}) \tag{12}$$

where T_{aw} and T_{sl} are MAC layer parameters that depend upon 802.15.4 MAC protocol operating in beacon-enabled/non beacon-enabled mode. $R(0,T_{sl})$ is a random number distributed between 0 and T_{sl} for the case when the receiver is sleeping, and the sender has to wait until the receiver wakes up [19]. The average number of re-transmissions can be modeled using Geometric distribution [18] which represents the probability of number of transmission failures before first successful transmission (Pr) as

$$Pr = (1-p)^K p \tag{13}$$

where p is the probability of successful transmission which can be represented as

$$p = \frac{1}{\#nodes\ contending\ for\ channel\ access} \tag{14}$$

Using equations 13 and 14, the n_R can be calculated as $\frac{1-p}{p}$. According to the exponential back-off mechanism in CSMA/CA [19], the number of slots a sensor node has to wait for (n_s) after collision detection is 2^c-1, where c is the number of collisions occurred so far for that sensor node. It is to be noted that c and n_R are closely related and can be used interchangeably. The expected back-off time for a sensor node can be calculated as

$$E[T_b] = \frac{1}{n_s+1}\sum_{j=0}^{n_s} j = \frac{n_s(n_s+1)}{2(n_s+1)} = \frac{n_s}{2} = \frac{2^{\frac{1-p}{p}}-1}{2} \tag{15}$$

Hence, the equation 12 can be re-written as

$$T_{trans} = \frac{1-p}{p}(T_{aw}+T_{sl}) + \frac{2^{\frac{1-p}{p}}-1}{2} + R(0,T_{sl}) \tag{16}$$

$$delay^{avg} = W_q + \frac{1-p}{p}(T_{aw} + T_{sl}) + \frac{2^{\frac{1-p}{p}} - 1}{2} + R(0, T_{sl}) \quad (17)$$

Equation 17 represents the average *1-hop* delay in transmitting a packet. IEEE 802.15.4 standard fixes the T_{aw} as 15.36 ms (at 250 kbps in 2.4GHz band) while T_{sl} is calculated in beacon-enabled mode as follows. In IEEE 802.15.4, sleep time of a sensor node is calculated as *Beacon Interval (BI) - Super frame Duration (SD)* whereas *BI* and *SD* are calculated as

$$SD = BaseSuperframeduration \times 2^{superframe\ order} \quad (18)$$

$$BI = BaseSuperframeduration \times 2^{beacon\ order} \quad (19)$$

TABLE 1: Simulation parameters for $delay^{avg}$ evaluation

Parameter	Value
T_{aw}	$15.36ms$
T_{sl}	$600ms$
p	.25
Total number of sensor nodes	100
Arrival rate (λ) range	[10-200]
Service rate (μ) range	[5-30]
Queue size (L)	20
Number of sensor nodes	100

We tested the proposed $delay^{avg}$ model using OPNET 18.0 modeler. The experiment considers a *BaseSuperframeduration* of 5.36 ms, *superframe order* as 4 and *beacon order* as 7. Since DODAG network topology is considered throughout this paper, we limit the maximum number of children for a parent sensor node to be 3. For the experiments, λ and μ are varied to observe the effect of W_q on $delay^{avg}$ while other parameters are set as shown in Table 1. Fig. 3 depicts the observed $delay^{avg}$ with varying λ and μ. To manifest the subtle effects of W_q on $delay^{avg}$, we represent the observed delay in the normalized form. The delay in seconds can be calculated as $\frac{Normalized\ delay}{10^6} + 1.85$. $delay^{avg}$ decreases with the λ (= 200 to 50) that indicates its dependability on the traffic intensity. The results shown are the average of 500 simulation instances.

Fig. 4 shows the comparison among $delay^{avg}$ estimated using proposed model, delay model proposed in [20] and delay observed through simulations. The *1-hop* delay modeled in [20] does not take W_q into account which has a minute effect on the average *1-hop* delay (see Fig. 3). Extensive simulations with the simulation parameters as stated in Table 1 are performed with λ = 50. The proposed model has a good accuracy rate (Fig. 4) with overall Root mean square deviation (RMSD) of 0.1438 as compared to 0.1470 of the model proposed in [20].

3.5 Residual energy model

This section discusses estimating the residual energy (equation 10) of a sensor service with the minimum message exchange overhead. Piggybacking is the standard approach used for energy state information exchange. Energy consumption in transmitting l- bit data from source to the destination is represented by equation 20.

$$E_{tx}(l, src, des) = \sum_{k=1}^{m} E_{tx}(l, distance) \quad (20)$$

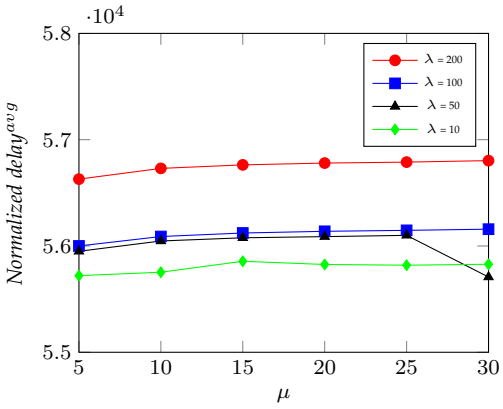


Fig. 3: Effect of W_q on $delay^{avg}$

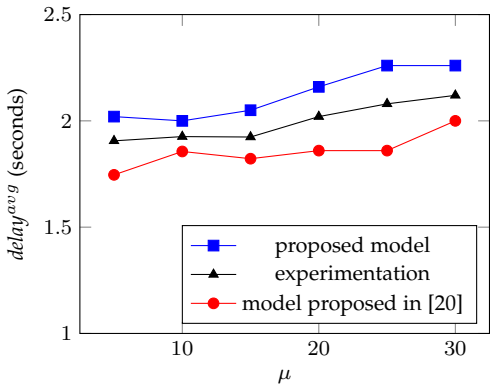


Fig. 4: Accuracy evaluation of proposed $delay^{avg}$ model

where m represents the minimum hop count between source and destination and l represents the size of the data being transmitted. Though this approach is better than the previous research on energy information exchange in which the energy information is shared using periodic control message exchanges between sensor service and gateway, there is still an energy overhead due to the large size of l. The size of l is increased due to the addition of residual energy information bits. In resource constrained WSN, energy consumption due to every bit of communication must be taken into consideration [3,16]. By using prediction based approach, we minimize the value of l such that there is no need of sharing residual energy information with the gateway. Thus mechanism incurs no energy overhead in gathering energy state information.

A sensor node can be in either of the three states - *idle*, receiving (*rcv*) and transmitting (*trans*). A sensor node's state transition is modeled as Discreet Time Markov Chain (DTMC) process. The estimation of **P** is a one-time offline process to avoid any control data transmission regarding energy information of sensor nodes. To compute **P**, we injected over 1000 tasks with different start and stop time (see Table 4) in the network of 100 sensor nodes and calculated the probability of a sensor node being in a state as follows. The probability of a sensor node being in *rcv* state is the number of times the sensor node was in *rcv* state and went to *idle* or *trans* state divided by the total number of time-steps the sensor node was in *rcv* state. The probabilities of

a sensor node being in *trans* and *idle* states are computed similarly.

The expected values for a sensor node's probability of being in *idle*, *rcv* and *trans* state are estimated as P_{NS} = (0.21, 0.45, 0.34). We use discrete time event simulation based mechanism taking input as the P_{NS} to estimate **P**. Sensor node's state distribution is shown in Table 2. To estimate **P** we generate 100 random numbers in the range of [1, 100] for 50 slots considering 20 tasks in each slot (=1000 total tasks) and assigned the state of a sensor node (Table 3).

TABLE 2: Sensor node's state distribution

State	Probability	Cumulative probability	Random digit assignment
idle	0.21	0.21	1-21
receive	0.45	0.66	22-66
transmit	0.34	1.0	67-100

TABLE 3: Sensor nodes' generated states for each slot

Sensor node	Random number (slot 1)	State (slot 1)	..	Random number (slot 50)	State (slot 50)
1	24	receive	..	19	idle
2	68	transmit	..	45	receive
..	..	..	..	..	..
100	54	receive	..	9	idle

For each sensor node, the average number of transitions from one state to another over 50 slots is estimated using Table 3. The expected values in **P** exactly represent the average number of transitions between each pair of two states. The sample **P** for a sensor node is represented by equation 21.

$$\mathbf{P} = \begin{pmatrix} 2/7 & 5/7 & 0 \\ 1/5 & 2/5 & 2/5 \\ 2/13 & 7/13 & 4/13 \end{pmatrix} \tag{21}$$

The tasks specify their attributes including *number of data points* to be sensed in a given time interval so that the sensors can adjust their sending rate accordingly. A sample task format is shown in Table 4. A sensor node's initial state probability matrix $(p_{idle} \quad p_{rcv} \quad p_{trans})$ changes according to the requirements of the tasks. Let the incoming task in the current slot requires the service of sensor node *i*. The initial state probability matrices for sensor node *i* becomes $P^i_{current\_slot} = (0.0 \quad 0.5 \quad 0.5)$. A sensor node's state probability matrix (P^i_{SPM}) gets updated after every slot according to equation 22.

$$P^i_{SPM} = P^i_{current\_slot} \times \mathbf{P} \tag{22}$$

Let total energy consumption in *trans*, *rcv* and *idle* state is E_{trans}, E_{rcv} and E_{idle} respectively. The energy model is adopted from [21] that includes B as the bit rate. In our case B represents the $info\_bits$ required by the task. The energy consumption information along with sensor node's state probability matrix enables the computation of the total energy consumed by a sensor node i (E^i_{Total}) in a slot according to equation 23.

$$E^i_{Total} = P^i_{SPM} \times \begin{pmatrix} E_{idle} \\ E_{rcv} \\ E_{trans} \end{pmatrix} \text{ Joules} \tag{23}$$

TABLE 4: A sample task format to obtain **P**

Attribute	Value
Start time	2 seconds
Stop time	20 seconds
Required information	100 bits
Target location	[45.4, 10.7]
Service_type	temperature data

TABLE 5: Simulation parameters for energy prediction

Parameter	Value
e_t	50×10^{-9} Joules/bit
e_r	50×10^{-9} Joules/bit
Power index	2
e_{id}	40×10^{-9} Joules/bit
Sensing range	10 m
Transmission power	30 dBm

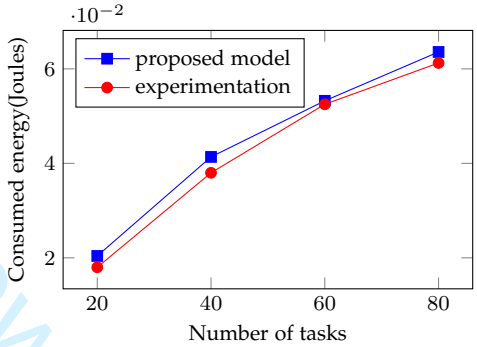


Fig. 5: Energy consumption prediction accuracy

To test the correctness of the proposed energy prediction mechanism, we tested our model for a different number of tasks in the network. Simulation parameters used for this module are shown in Table 5. The proposed energy prediction mechanism is the first of its kind for IoT scenario. Thus we could not find any other mechanism to compare our results. We first injected our generated task graphs in the network where sensor nodes are distributed uniformly. Each sensor node's residual energy levels are recorded according to each task. To record the actual energy consumption of a sensor node required by the incoming task, we divided the energy consumption into three main blocks [22]: E_{NET} (for data communication/networking), E_{ACQ} (for data acquisition) and E_{PRC} (for data processing). Total energy consumption is the sum of E_{NET}, E_{ACQ} and E_{PRC}. Once the actual energy consumption is recorded, our prediction mechanism is applied on the same network with same task graphs. Fig. 5 shows that the proposed model has a good accuracy rate with overall RMSD value of 0.00198.

Since the gateway has a virtual view of the network topology, the intermediate sensor nodes between the gateway and a sensor service are identified, and their energy

consumption in the current slot is calculated similarly. Moreover, the residual energy prediction module keeps a check on the residual energy level of sensor services so that the energy drained sensor services can be prohibited from taking part in the ranking process.

4 Integration with a Gateway Service

To test the utility of VoISRAM, we integrate it with a recent gateway service design named as TRAPS (Task Requirement Aware Pre-processing and Scheduling [3]). It is a middleware layer gateway service designed around the information sharing among multiple sensing tasks. It groups the similar tasks at the gateway and selects representative tasks from each group to be executed inside WSN. The representative tasks fetch the sensed data that is shared by other members of the group the representative tasks belong to. This reduces the down-link traffic in WSN and increases the network lifetime [3]. Selection of a sensor service is solely based on the mapping between spatial requirement of the task and the associated geographical location of the sensor service.

Rather than distributing the task among selected sensor services in proportion to their residual energy levels (as in TRAPS), VoISRAM sorts the *candidate sensor services* in ascending order of their ranks. The sensor service with the highest rank is chosen to serve the task. However, if the highest ranked sensor service is not available (busy serving other tasks or failed due to technical reasons), the task is assigned to the second highest ranked sensor service and so on. QoS requirements in IoT sensory environment can be characterized into application specific QoS and network specific QoS [3]. Application specific QoS requirements include spatio-temporal accuracy of the sensed data (see section 3.2). Network specific QoS requirements include network lifetime that is critical for resource constrained WSN. VoISRAM is assessed for its utility to complement TRAPS in maintaining a trade-off between application specific and network specific QoS requirements. We simulated a WSN environment using OPNET modeler with simulation parameters listed in Table 6.

The experiments are performed over a network with 4 sub-networks, each having one gateway. The proposed VoISRAM is executed on each gateway. For simplicity, the experiments described in this section are shown for a sub-network with different node densities (10, 20, 30 and 40 sensor nodes) and a single gateway. A multi-application scenario is considered in which different applications using *pull* and *push* based mechanisms are executed to fetch the sensed data. The number of *pull based* applications are varied from 5-20 where each application consists of at-least 4 sensing tasks. Maximum tasks that can be injected into the network are limited to 80. Tasks with real time temporal requirements ($Cntxt_1^{temp}$) are considered to be critical for many IoT based applications such as smart health-care. Similarly $Cntxt_1^{spat}$ tasks pose a stricter location based spatial requirements as compared to their counterparts ($Cntxt_2^{spat}$). Thus, we study the effect of VoISRAM on the temporal (section 4.1) and spatial (section 4.2) requirements of the $Cntxt_1^{temp}$ and $Cntxt_1^{spat}$ tasks respectively.

TABLE 6: Simulation parameters

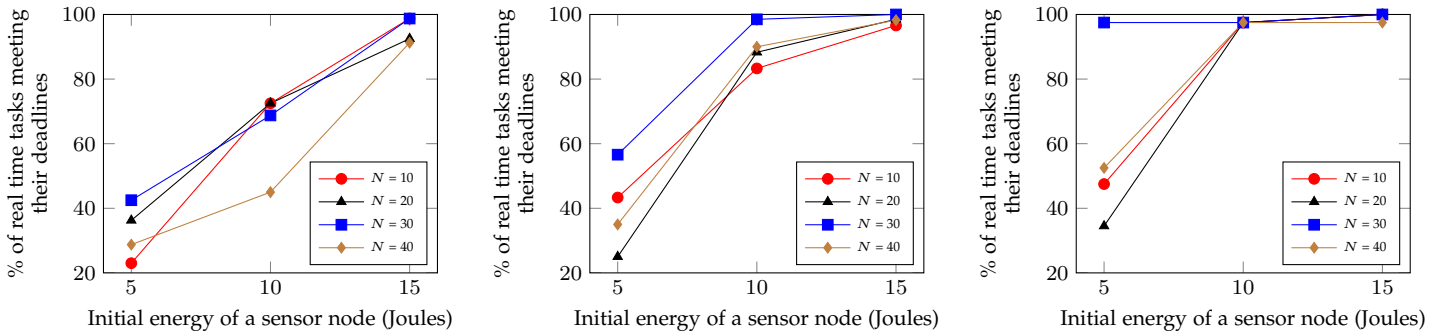
Parameter	Value
Number of tasks	20-80
Simulation time	60 minutes
Number of sensor nodes (N)	10-40
Deployment area	100×100 m^2
Sensor node's initial energy	5-15 Joules
Bit rate (B)	$.25Mbit/s$
Sensing range	10 m
Data packet size	80 bits

4.1 Effects on $Cntxt_1^{temp}$ tasks

In this experiment, we study the effect of VoISRAM on temporal requirements of the real time tasks. A $Cntxt_1^{temp}$ task (see section 3.2.2) specifies a deadline (d_t) before which all the required data points must be available at the gateway. We generated equal number of $Cntxt_1^{temp}$ and $Cntxt_2^{temp}$ tasks and measured the percentage of $Cntxt_1^{temp}$ tasks able to meet their deadlines. Experiments shown in this section are performed while varying the following network parameters: node density, initial energy level of a sensor node and number of tasks in the network (see Table 6). Fig. 6(a) shows the percentage of $Cntxt_1^{temp}$ tasks meeting their temporal requirements out of 80 tasks injected into the network with varying node density and initial energy level of a sensor node. For initial energy level of 5 Joules, the percentage of $Cntxt_1^{temp}$ tasks meeting their temporal requirements is less. This is because the low initial energy level of sensor nodes causes early node failure due to energy drain. This leads to increased end-to-end delay causing $Cntxt_1^{temp}$ tasks miss their respective deadlines. As the initial energy level of nodes increases, the percentage of $Cntxt_1^{temp}$ tasks meeting their temporal requirements increases sharply for node density of 10, 20 and 30. Interestingly this increment is rather gradual for node density of 40 due to the increased number of hops contributing to the end-to-end delay. As the initial energy level of nodes reaches 15 Joules, node density does not affect much the performance of the mechanism. However, with less number of tasks (=40) in the network, the scenario with node density of 30 results in high percentage (Fig. 6(c)) of $Cntxt_1^{temp}$ tasks meeting their temporal requirements when initial energy level of sensor nodes is 15 Joules. This indicates that a dense network (node density = 40) with the high initial energy level of sensor nodes (=15 Joules) results in increased percentage of $Cntxt_1^{temp}$ tasks meeting their temporal deadlines. On the contrary, high node density (=40 nodes) increases the end-to-end delay causing $Cntxt_1^{temp}$ tasks miss their respective temporal deadlines. Similarly, scenarios with low node densities (=10 and 20) perform poorly for low initial energy level (=5 Joules) of sensor nodes. This is due to the network partition caused by low initial energy level of sensor nodes. On the other hand, the scenario with node density of 30 succeeds in maintaining the trade-off between network wide energy consumption and end-to-end delay that results in the high percentage of $Cntxt_1^{temp}$ tasks meeting their respective temporal deadlines.

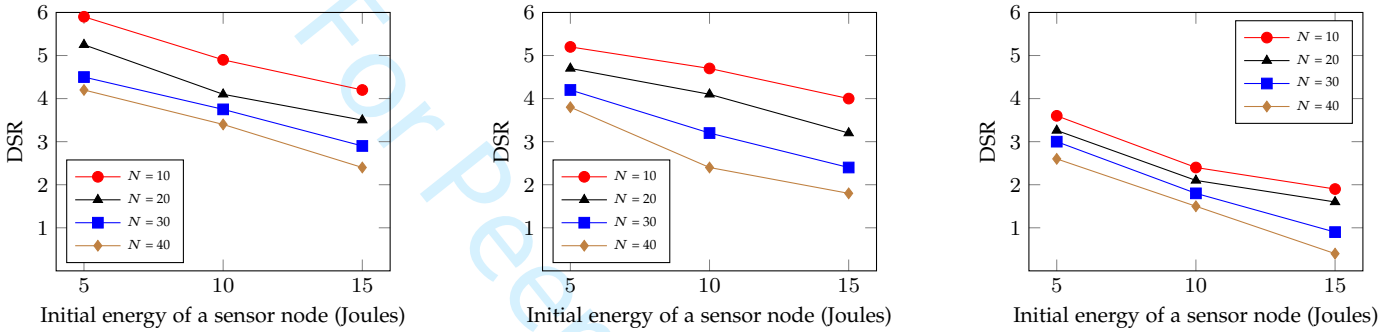
4.2 Effects on $Cntxt_1^{spat}$ tasks

In this experiment, we study the effect of VoISRAM on the spatial requirements of the location specific tasks. Equal



(a) Number of tasks = 80 (b) Number of tasks = 60 (c) Number of tasks = 40

Fig. 6: Percentage of real time tasks meeting their temporal requirements with different network conditions.



(a) Number of tasks = 80 (b) Number of tasks = 60 (c) Number of tasks = 40

Fig. 7: Effect on location specific tasks with different network conditions

number of $Cntxt_1^{spat}$ and $Cntxt_2^{spat}$ (see Section 3.2.1) tasks are injected into WSN. Since the $Cntxt_1^{spat}$ specifies a location in the form of coordinates, there may be a possibility that no sensor node is deployed at that location. In this case, sensor nodes nearest to the required location serve the task (see equation 3). VoISRAM's ability to meet the spatial requirement of $Cntxt_1^{spat}$ tasks is measured using Deviation from Spatial Requirement (DSR) metric defined as follows. Suppose there are n $Cntxt_1^{spat}$ tasks injected into the WSN. If a task T_i specifies the required location such as (x_i, y_i) whereas it is assigned to a sensor node located at (p_i, q_i), the DSR value for n such tasks is calculated using equation 24.

$$\text{DSR} = \sqrt{(|x_1 - p_1| + |y_1 - q_1|)^2 + ... + (|x_n - p_n| + |y_n - q_n|)^2} \quad (24)$$

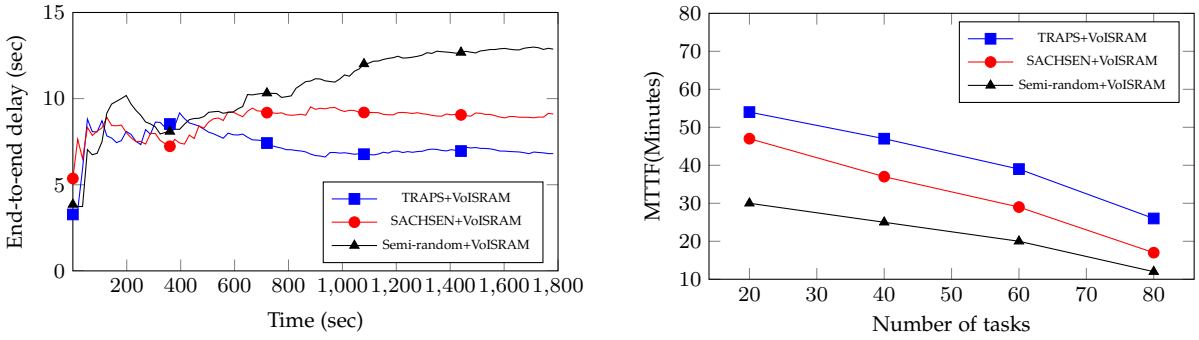
DSR is the collective measure for n tasks and represents the sum of distances that indicates the deviation in $Cntxt_1^{spat}$ tasks' spatial requirements from the locations of sensor nodes such tasks are assigned to. Fig. 7 shows the obtained DSR values for various network conditions. The scenario with node density of 40 observes low DSR throughout the experiments (Fig. 7(a)-(c)). This indicates that a dense network helps $Cntxt_1^{spat}$ tasks meeting their spatial requirements. The effect of different initial energy levels shows the availability of a sensor node at the required location since a sensor node with low initial energy level (=5 Joules) can drain out quickly as compared to the one with high initial energy level (=10 and 15 Joules).

It can be inferred from the above experiments (Fig. 6 and 7) that scenario with network density of 30 outperforms others in terms of meeting $Cntxt_1^{temp}$ tasks' temporal requirements. Moreover, it performs reasonably good in terms of meeting $Cntxt_1^{spat}$ tasks' spatial requirements as well. Scenario with node density of 40 outperforms others in terms of meeting the spatial requirements of $Cntxt_1^{spat}$ tasks. However, it performs poorly in terms of meeting the temporal requirements of $Cntxt_1^{temp}$ tasks when the initial energy of sensor nodes is low (=5, 10 Joules). Experiments in the following sections are performed on the scenario with node density of 30 while keeping the initial energy level of nodes as 10 Joules.

5 Justification for Integration with TRAPS

In this section, we compare the performance of TRAPS with other gateway services such as SACHSEN [16] and Semi-random [16] when integrated with VoISRAM. We integrate VoISRAM with mentioned gateway services and compare the results in terms of application end-to-end delay and MTTF. The end-to-end delay is measured using M/G/1 queuing model that includes service time, network transmission time and waiting time in the queue. The simulation is performed for 30 minutes with node density of 30, and initial energy of sensor nodes as 10 Joules. Other simulation parameters are same as mentioned in Table 6.

SACHSEN is a QoS aware gateway service similar to TRAPS with the objective of minimizing the number of tasks entering the network. Unlike TRAPS, SACHSEN does not consider the temporal requirements of real time tasks before assigning them to the appropriate sensor services.



(a) Number of tasks = 80 (b)

Fig. 8: End-to-end delay and MTTF comparison when VoISRAM is integrated with Semi-random, SACHSEN and TRAPS.

Moreover, SACHSEN uses piggybacking to get the residual energy information from the sensor nodes that results in increased energy consumption (see Section 3.5). On the other hand, Semi-random approach randomly selects a sensor service from the list of *candidate sensor services*. Fig. 8(a) shows the comparison of mentioned approaches in terms of application end-to-end delay when integrated with VoISRAM. For this experiment, we injected 80 tasks in the network with an equal number of real time and non-real time tasks. The recorded end-to-end delay is the average of over 100 simulations. Initially all three gateway services perform equally well (Fig. 8(a)) due to control message exchanges for network stabilization. However, as the network stabilizes TRAPS outperforms SACHSEN and Semi-random. This is because VoISRAM enables TRAPS to further reduce the end-to-end delay by taking the hop count between gateway and sensor service into consideration. This reduces overall end-to-end delay experienced by the application.

Fig. 8(b) shows the comparison among TRAPS, SACHSEN and Semi-random in terms of network lifetime. Semi-random gateway service does not prevent redundant tasks from entering into the network, thus results in low network lifetime. TRAPS and SACHSEN experience the similar amount of energy consumption while working in standalone manner [3]. Network lifetime is a function of overall energy consumption and the number of messages exchanges. However, the amount of energy consumption and network lifetime are two different parameters. The amount of energy consumption depends upon the number of message exchanges in the network. On the other hand, network lifetime depends upon the fair and balanced allocation of tasks to the sensor services. Unlike SACHSEN, TRAPS does not use piggybacking to gather the residual energy information from sensor services. This results in reduced energy consumption and leads to high network lifetime. Moreover, VoISRAM further boosts the network lifetime by considering the residual energy of intermediate sensor nodes between the gateway and a sensor service before scheduling a task. Since end-to-end delay and network lifetime are the crucial evaluation parameters for IoT based applications, the experimental results shown this section justify the integration of VoISRAM with TRAPS.

6 Comparative Evaluation

In an attempt to manifest the incremental addition towards the state-of-the-art solutions in the area of sensor service ranking, this section compares the VoISRAM with other energy aware sensor service ranking mechanisms such as CASSARAM [12] and the mechanism proposed in [11]. For notation purpose, we represent mechanism proposed in [11] as Energy Aware Ranking Mechanism (EARM). Similar to the earlier experiments, spatio-temporal accuracy (application specific QoS parameters) and network lifetime (network specific QoS parameter) are considered as evaluation parameters to compare VoISRAM, CASSARAM, and EARM. Neither EARM nor CASSARAM provides any mechanism to serve the non real time tasks requiring stale data ($Cntxt_2^{temp}$). Thus, we choose only real time tasks for the comparative evaluation. CASSARAM addresses location specific ($Cntxt_1^{spat}$) tasks and do not take $Cntxt_2^{spat}$ tasks into consideration. On the other hand, EARM does not take spatial accuracy into consideration and assigns the tasks to the gateway that contains the required sensor service. The gateway assigns the task to the sensor service on the basis of the path length between the gateway and the sensor node. We choose DSR value as the spatial accuracy measurement metric for VoISRAM, EARM and CASSARAM.

EARM models a sensor service access cost as a function of its residual energy level. CASSARAM calculates Comparative Priority-based Weighted Index (CPWI) to rank the sensor services. Though CASSARAM has no limitation on number of parameters, we consider residual energy (network specific), response time (temporal requirement) and precision (spatial requirement) as the specified parameters with weights α_1, α_2 and α_3, respectively to calculate the CPWI so that it can be reasonably comparable with VoISRAM and EARM. Response time in CASSARAM is defined as the time taken by a task to fetch the required sensed data and is used as the measured parameter to evaluate mechanism's temporal accuracy. Thus, only real time tasks with delivery deadlines are considered for the temporal accuracy assessment.

The experiments are performed on a network size of 40 sensor nodes with initial energy level of 15 Joules. Other simulation parameters are same as mentioned in Table 6. Since EARM does not use a weighted cost function and mostly focuses on energy efficiency, α_1 in CASSARAM is

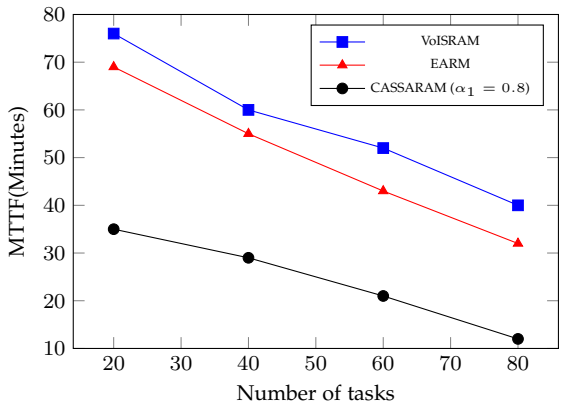


Fig. 9: Network lifetime comparison among VoISRAM, EARM and CASSARAM

set as 0.8 ($\alpha_2 = 0.1, \alpha_3 = 0.1$) (assigning highest priority to residual energy) for a fair comparison in terms of MTTF. To assess the application specific QoS parameters α_2 in CASSARAM is set as 0.8 ($\alpha_1 = 0.1, \alpha_3 = 0.1$) for temporal requirements assessment and α_3 is set as 0.8 ($\alpha_1 = 0.1, \alpha_2 = 0.1$) for spatial requirements assessment. Fig. 9 shows the comparison among VoISRAM, EARM and CASSARAM in terms of MTTF of a sensor node while varying the number of tasks in the network. Unlike CASSARAM, VoIRSAM and EARM include residual energy levels of sensor nodes along the best probable path in their cost functions. It is observed that VoISRAM outperforms CASSARAM and shows upto 13% improvement in the MTTF when compared with EARM. This improvement in the MTTF is caused due to following reasons. EARM involves the exchange of periodical updates about the sensor service residual energy information that contributes to the overall energy consumption. On the other hand, VoISRAM uses prediction based residual energy estimation mechanism that involves zero message exchange once the network is initialized. Moreover, f_{QoI_2} helps selecting paths with minimum hop counts that results in reduced overall energy consumption and leads to increased network lifetime. CASSARAM considers the residual energy level of the sensor service but does not take into account the residual energy levels of the intermediate sensor nodes between the gateway and the sensor service in question that results in low MTTF.

Due to the topological management overhead, MTTF dips further as the number of tasks increases (Fig. 9) in the network. A large number of tasks causes more energy consumption that may possibly result in topological changes. The sensor services may leave at any point due to low residual energy levels. On the other hand, new sensor services may discover their parents and join the network in between. In both the cases, the VoISRAM follows DODAG topology management protocol [17] according to which, when a sensor service discovers a new child/parent it communicates this information to the gateway so that the logical view of the topology can be updated. However, these topological updations will be more frequent in mobile scenarios that have the following implications on the VoISRAM. In mobile scenario, suppose a sensor node leaves the network owned by the gateway it belongs to and moves to an area owned by another gateway. This causes the change in sensor service location that has to be updated at the gateway (assumed to contain the sensor service location information). Moreover, due to the joining/leaving of the sensor service(s), various procedures such as route discovery will be invoked to maintain the logical view of the network topology at the gateway. These updations contribute to the overall network traffic that leads to the increased energy consumption in the network. Since residual energy is the key attribute in the VoISRAM, the tasks are assigned to the sensor services with low residual energy levels compared to that of in static scenario. This can affect the overall QoS experience of the application.

Since the sensor services considered in this paper are static, there are sporadic topological changes observed. Moreover, residual energy prediction module is linked with the metadata information and timely informs it about the status of a sensor service so that the energy drained sensor services can be prohibited from participating in the ranking process. Energy consumption in other cases such as retransmission and re-issuing route discovery is beyond the scope of this paper.

Fig. 10 and 11 show the comparison among VoISRAM, EARM and CASSARAM in terms of the temporal and spatial accuracy respectively. EARM and CASSARAM treat every task uniformly, and the VoI aspect is ignored. Though EARM outperforms VoISRAM initially, its performance degrades as the number of tasks increases in the network (see Fig. 10). EARM completely ignores the temporal requirement part at the time of ranking. However, its energy aware mechanism contributes in reducing the number of intermediate sensor nodes that results in reduced end-to-end delay and more number of tasks meeting their temporal requirements. CASSARAM ranks the sensor service on the basis of response time that is proportional to the amount of traffic in the network. VoISRAM considers response time along with the usage context in which the received information is going to be used. The degradation in CASSARAM's performance as the number of tasks increases in the network is the direct outcome of increased network traffic.

Since CASSARAM addresses the location based spatial requirements of the task, we consider both $Cntxt_1^{spat}$ and $Cntxt_2^{spat}$ tasks for spatial accuracy assessment. CASSARAM performs equally well as VoISRAM in terms of meeting spatial requirements (see Fig. 11). This is because CASSARAM considers precise location in the cost function. Similar to the temporal requirement, the VoI aspect of the spatial requirement is completely ignored in EARM and CASSARAM. Usually $Cntxt_1^{spat}$ tasks are considered to be stricter than their $Cntxt_2^{spat}$ counterparts. CASSARAM considers the $Cntxt_1^{spat}$ tasks that help $Cntxt_2^{spat}$ tasks meeting their spatial requirement as well (low DSR value). On the other hand, EARM simply assigns the task to the gateway that contains the required sensor service. Ignoring the spatial usage context, EARM assigns the task to the sensor service near to the gateway. This results in high DSR value in case of EARM.

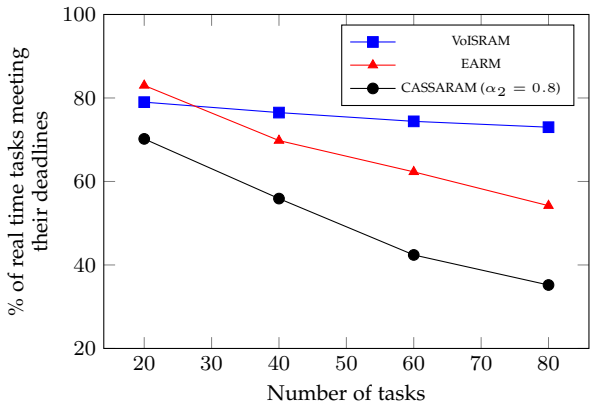


Fig. 10: Temporal accuracy comparison among VoISRAM, EARM and CASSARAM

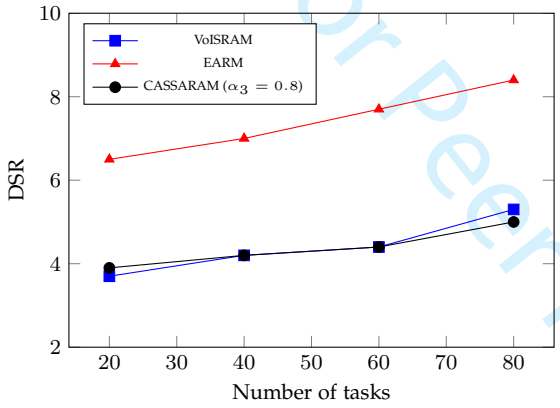


Fig. 11: Spatial accuracy comparison among VoISRAM, EARM and CASSARAM

7 Conclusion

This paper addressed the issue of VoI based sensor service ranking for the first time and proposed a novel energy aware sensor service ranking mechanism named as VoISRAM. Mathematical model of VoISRAM states that a sensor service ranking can be modeled as VoI attribute. The proposed VoISRAM is evaluated for its ability in meeting application specific and network specific QoS requirements when integrated with existing state-of-the-art gateway services. The comparison with recent state-of-the-art service ranking mechanisms manifests the incremental addition made by VoISRAM in the area of sensor service ranking. Moreover, the low time complexity of VoISRAM indicates its suitability for resource constrained WSNs.

The future work includes integrating VoISRAM with a mobility-aware sensor indexing mechanism to test the implications of mobility on its performance. The cost function can also be improved by developing model to measure the workload of a sensor service as to incorporate this in the cost function. This will reduce the incurred communication cost in determining if the highest ranked sensor service is busy.

References

[1] C. Bisdikian, L. M. Kaplan and M. B. Srivastava, "On the quality and value of information in sensor networks," *ACM Transactions on Sensor Networks*, vol. 9, no. 4, pp. 48.1-48.26, 2013.

[2] V. C. Gungor and G. P. Hancke, "Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 4258-4265, 2009.

[3] S. Bharti, K.K.Pattanaik, "Task requirement aware pre-processing and Scheduling for IoT sensory environments," *Ad Hoc Networks, Elsevier*, vol. 50, pp. 3816-3825, November 2016.

[4] J. Hwang, C. Shin and H. Yoe "Study on an Agricultural Environment Monitoring Server System using Wireless Sensor Networks," *Sensors*, vol.10, no. 12, pp. 11189-11211, 2010.

[5] O. Chayka, T. Palpanas, P. Bouquet "Defining and measuring data-driven quality dimension of staleness," University of Trento, Italy, Technical Report # DISI-12-016, 2012.

[6] W. Wang, S. De, G. Cassar, K. Moessner, "An experimental study on geospatial indexing for sensor service discovery,", *Expert Systems and Applications, Elsevier*, vol. 42, no. 7, pp. 3528-3538, 2015.

[7] J. Timpner, D. Schrmann and L. Wolf, "Trustworthy Parking Communities: Helping Your Neighbor to Find a Space," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 1, pp. 120-132, 2016.

[8] R. Mietz and K. Rmer, "Exploiting correlations for efficient content-based sensor search," *2011 IEEE SENSORS Proceedings*, Limerick, 2011, pp. 187-190.

[9] V. Deora, J. Shao, W. A. Gray and N. J. Fiddian, "Modeling Quality of Service in Service Oriented Computing," *2006 Second IEEE International Symposium on Service-Oriented System Engineering (SOSE'06)*, Shanghai, 2006, pp. 95-101.

[10] Z. Xu, P. Martin, W. Powley, F. Zulkernine "Reputation-Enhanced QoS-based Web Services Discovery," *In Proceedings of IEEE conference on web services*, , 2007, pp. 249256.

[11] W. Wang, F. Yao, S. De, K. Moessner and Z. Sun "A ranking method for sensor services based on estimation of service access cost," *Information Sciences, Elsevier*, vol. 319, pp. 1-17, 2015.

[12] C. Perera, A. Zaslavsky, C. H. Liu, M. Compton, P. Christen and D. Georgakopoulos, "Sensor Search Techniques for Sensing as a Service Architecture for the Internet of Things," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 406-420, 2014.

[13] W. Niu, J. Lei, E. Tong, G. Li, L. Chang, Z. Shi and S. Ci, "Context-Aware Service Ranking in Wireless Sensor Networks," *Journal of Network and System Management, Springer*, vol. 22, no. 1, pp. 50-74, 2014.

[14] G. Cassar, P. Barnaghi, K. Moessner, "Probabilistic matchmaking methods for automated service discovery," *IEEE Transactions on Services Computing*, vol. 7, pp. 654-666, 2013.

[15] G. Cassar, P. Barnaghi, W. Wang and K. Moessner, "A hybrid semantic matchmaker for IoT services," *2012 IEEE International Conference on Green Computing and Communications*, IEEE Computer Society, Los Alamitos, CA, USA, 2012, pp. 210-216

[16] W. Li, F. C. Delicato, P. F. Pires, Y. Ch. Lee, A. Y. Zomaya, C. Meceli and L. Pirmez, "Efficient allocation of resources in multiple heterogeneous Wireless Sensor Networks," *Journal of Parallel and Distributed Computing, Elsevier*, vol. 74, pp. 1775-1788, 2014.

[17] O. Iova, P. Picco, T. Istomin and C. Kiraly, "RPL: The Routing Standard for the Internet of Things... Or Is It?," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 16-22, 2016.

[18] Sheldon M. Ross, *Introduction to Probability Models*, Sixth edition, Academic Press, U.S.A., 2006

[19] B. Bellalta, A. Faridi, D. Staehele, J. Barcelo, A. Vinel and M. Oliver "Performance analysis of CSMA/CA protocols with multi-packet transmission," *Computer Networks, Elsevier*, vol. 57, no. 14, pp. 26752688, 2013.

[20] J. Wang, W. Dong, Z. Cao and Y. Liu, "On the Delay Performance in a Large-Scale Wireless Sensor Network: Measurement, Analysis, and Implications," *IEEE/ACM Transactions on Networking*, vol. 23, no. 1, pp. 186-197, 2015.

[21] Q. Gao, K.J.Blow, D.J.Holding, I.W.Marshal, and X.H.Peng, "Radio Range Adjustment for Energy Efficient Wireless Sensor Networks," *Ad Hoc Networks, Elseveir* vol. 4, no. 1, pp. 75-82, 2006.

[22] B. Martinez, M. Montn, I. Vilajosana and J. D. Prades, "The Power of Models: Modeling Power Consumption for IoT Devices," *IEEE Sensors Journal*, vol. 15, no. 10, pp. 5777-5789, 2015.

[23] H. AbdelSalam, S. Olariu, "Toward Efficient Task Management in Wireless Sensor Networks," *IEEE Transactions on Computers*, vol. 60, no. 11, pp. 1638-1651, 2011.
[24] L. Li, S. Li, S. Zhao, "QoS-Aware Scheduling of Services-Oriented Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1497-1505, 2014.
[25] P. Levis et al., "The Trickle Algorithm," IETF RFC 6206, Mar. 2011.
[26] S. Jirka, A. Broring, C. Stasch, "Discovery Mechanisms for the Sensor Web," *Sensors*, vol. 9, pp. 2661-2681, 2009.
[27] Z. Ding, Z. Chen, Q. Yang, "IoT-SVKSearch: a real-time multi-modal search engine mechanism for the internet of things," *International Journal Of Communication Systems*, vol. 27, no. 6, pp. 871-897, 2014.
[28] Y. Zhou, S. De, W. Wang, K. Moessner and M. S. Palaniswami, "Spatial Indexing for Data Searching in Mobile Sensing Environments," *Sensors*, vol. 17, no. 6, 1427, 2017.
[29] C. H. Liu, J. Fan, J. W. Branch and K. K. Leung, "Toward QoI and Energy-Efficiency in Internet-of-Things Sensory Environments," *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 4, pp. 473-487, 2014.
[30] A. E. Al-Fagih, F. M. Al-Turjman, W. M. Alsalih and H. S. Hassanein, "A priced public sensing framework for heterogeneous IoT architectures," *IEEE Transactions on Emerging Topics in Computing*, vol. 1, no. 1, pp. 133-147, 2013.

Sourabh Bharti is currently working towards his Ph.D. from ABV-Indian Institute of Information Technology and Management, Gwalior, India. He also worked as a visiting researcher from 2015-16 with Anglia Ruskin University, U.K. under UKIERI project. His current research interests include network load balancing, in-network processing and QoS provisioning in IoT sensory environments.

K. K. Pattanaik received Ph.D. in Engineering with Computer Science as major from Birla Institute of Technology Mesra, Ranchi, India in the year 2010. Currently, he is associated with wireless sensor networks laboratory at ABV-Indian Institute of Information Technology and Management Gwalior, MP, India. His research interests are Distributed Systems, Grid Computing, Mobile Computing, Multi Agent Systems and Wireless Sensor Networks.

Paolo Bellavista is Associate Professor of Distributed and Mobile Systems at the University of Bologna. His research activities span from mobile computing to middleware, from network/systems management to adaptive location/context-aware services, from federated cloud computing to big data online stream processing. He is senior member of IEEE and ACM. He serves in the Editorial Boards of IEEE T. Computers, IEEE T. Services Computing, IEEE T. Network and Service Management, Springer J. Network and Systems Management, and Elsevier Pervasive and Mobile Computing J. He serves in the TPC of many international conferences about software support for advanced distributed systems and has chaired several ones. He authored more than 150 papers in most prestigious journals/magazines, books, and conferences.