Ad Hoc Networks 000 (2016) 1–13

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Task requirement aware pre-processing and Scheduling for IoT sensory environments

Sourabh Bharti*, K.K. Pattanaik

Wireless Sensor Networks Laboratory, ABV-Indian Institute of Information Technology and Management, Gwalior, India

ARTICLE INFO

Article history:
Received 21 January 2016
Revised 14 June 2016
Accepted 8 July 2016
Available online xxx

Keywords:
Internet-of-Things
Wireless sensor networks
Task allocation
Quality of service

ABSTRACT

In Internet of Things (IoT) sensory environment Wireless Sensor Networks (WSNs) are connected to the Internet through gateways and this gives birth to many real time sensor based applications. Applications from Internet querying for Spatio-temporal information within WSN may require query decomposition. Decomposition of queries may result in tasks having similar functional and Quality of Service (QoS) requirements. Thus pre-processing the tasks results in decreased number of task executions within WSN as compared to executing tasks individually. Moreover prior knowledge of sensor nodes' residual energy can help in deciding if a task can be scheduled for its successful completion. Existing energy monitoring protocols in WSNs incur considerable amount of message traffic and energy. Thus an Energy Monitoring System (EMS) that works on reduced message traffic and yet provide the correct energy information will be of great value. This paper proposes Task Requirements Aware Pre-processing and Scheduling (TRAPS) mechanism comprising a task pre-processor, EMS and scheduler within the gateway. The scheduler allocates the best available sensor nodes to the incoming tasks while meeting their QoS requirement. Task pre-processor and EMS are tested individually to measure the key performance metrics that include number of tasks entering WSN and energy information prediction accuracy respectively. Further the performance evaluation of TRAPS is extended to study its effects on various network parameters. It is found that TRAPS outperforms the findings of recent research on task scheduling in IoT sensory environments in terms of the identified network performance metrics.

1. Introduction

Research in the WSNs has mainly focused on MAC protocol, routing protocol and location management of the sensor nodes. Internetworking between WSN and Internet; and enabling applications to handle the sensed information is going to be the future of ubiquitous computing. Research related to this is divided into two fields. One is the gateway-based which is performed by gateway located between the WSN and the Internet; the other is overlaying-based integration which is performed by overlay networks constructed on either the WSNs or the Internet [19].

In IoT sensory environment, applications on Internet query [1] for a variety of physical quantities in Spatio-temporal domain. Extensive interaction between applications from the Internet and sensors is the next big thing in the era of Internet where smart devices (sensors and RFIDs) will get connected to the Internet [1–4]. Being an advancement in the technology, IoT offers many challenges for researchers in both academia and industry. One of those challenges is to manage WSNs to process queries by Internet applications. For instance, in a simplistic scenario of intense aquaculture, the application on Internet may query for the information about the dissolved oxygen concentration in different parts of the pond. Such a query can be characterized as Spatio-temporal since the oxygen concentration can vary from one part of the pond to the other and may change over time. However, a complex query can have multiple tasks to be executed in WSN. These tasks may have specific functional and QoS requirements. In the above scenario, the required service (oxygen concentration) from the target location and the information bits required are the functional requirements whereas maximum tolerable delay is the QoS requirement associated with the task. Since the given network technology has its own resource constraints, meeting application requirements while carefully managing the constrained resources is the foremost challenge. Grouping of tasks having similar requirements and injecting the selected ones into WSN conserves network resources. Further, scheduling them on the most appropriate sensor nodes leads to their successful completion. Moreover, in IoT

* Corresponding author.
E-mail address: bharti.sourabh90@gmail.com (S. Bharti).

[1] The term "query" is an abstract used at application level which further decomposed into network level tasks.

http://dx.doi.org/10.1016/j.adhoc.2016.07.005
1570-8705/

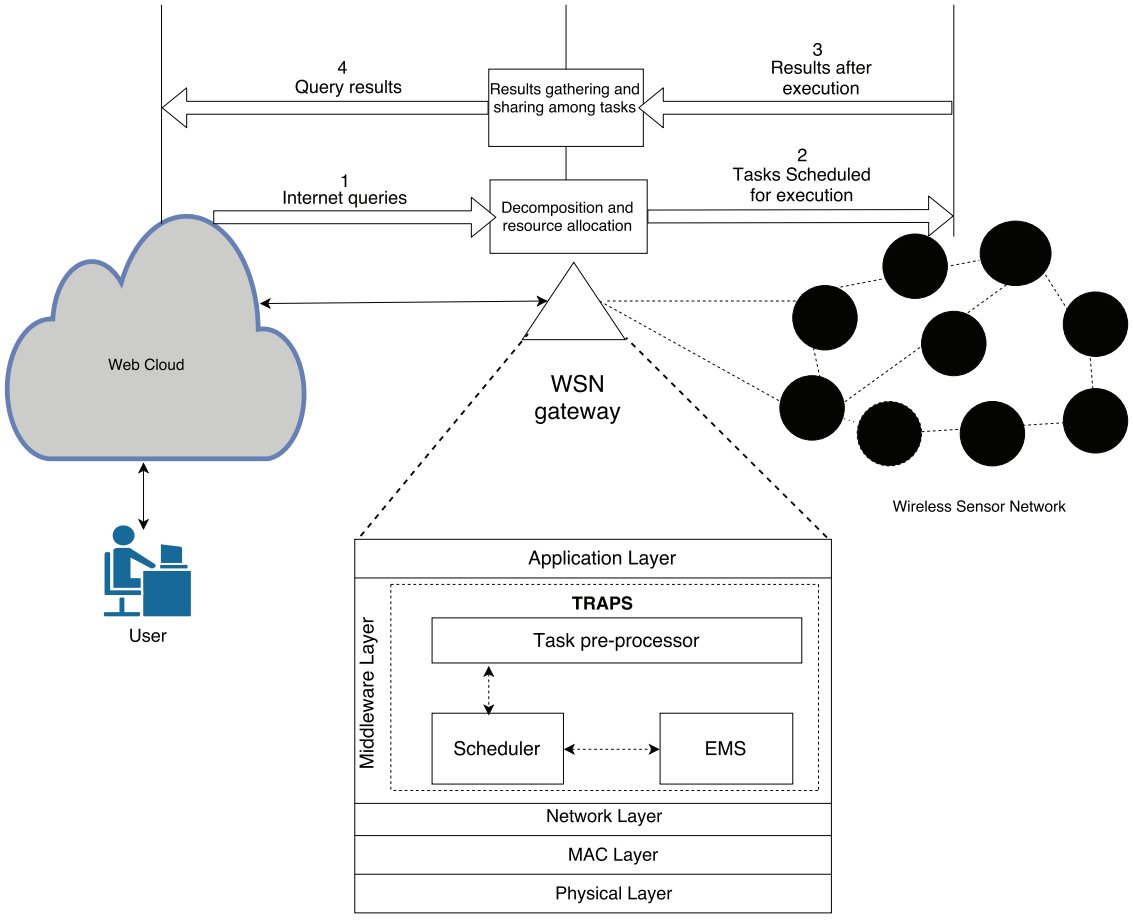

Fig. 1. Proposed WSN gateway design.

scenario, downstream traffic dominates the regular upstream traffic which is not the case in standalone WSNs. Effective resource management to increase network lifetime, fulfilling QoS requirements of incoming tasks and most importantly efficient scheduling of tasks to avoid execution redundancy and reduce the resulting inter-node communication are the key issues in such scenario. To meet the above requirements, this paper proposes an enhancement in the existing gateway architecture by introducing a task pre-processor and energy monitoring system. Fig 1. depicts the functional architecture of such a gateway. Efficient gateway design is an established research problem in distributed computing where various resource management techniques have been proposed. The research related to scheduling of tasks in WSNs [5–8] primarily focus on selecting suitable nodes to satisfy the QoS requirements.

TRAPS in the gateway (Fig. 1) consists of a task pre-processor module (responsible for gathering the requirements of incoming tasks and further classifying the tasks based on their requirements), an EMS module (for monitoring the residual energy of sensor nodes) and a scheduler module (to schedule the tasks). We propose a concept of Critical Covering Tasks (CCT) set that represents the subset of a group of tasks sharing similar spatial requirements. Subsequently, leader(s) are selected from each CCT set and scheduled to be serviced by sensor nodes. This reduction in the number of tasks has a direct consequence on network parameters. The decision for selecting potential sensor node from a set of nodes is based on its sensory capability and residual energy. The scheduler maintains the location and capability of sensor nodes in the network while EMS provides residual energy information. The EMS is designed around providing energy information of nodes with no message exchange in the network.

Rest of the paper is structured as follows. Section 2 discusses the related work. Section 3 describes the problem formulation. The mechanism design and correctness of its modules is discussed in Sections 4 and 5 respectively. Experiment design and evaluation of the overall system is presented in Section 6 followed by the conclusion and future scope in Section 7.

2. Related work

Task scheduling is an established research area in parallel and distributed computing paradigm. IoT offers different set of challenges in this context. Task scheduling in IoT sensory environment requires minimizing the resource consumption since it consists of constrained devices such as sensors and RFIDs. Thus any mechanism developed for such environments should complement to the existing mechanisms for increasing network lifetime and reducing inter-node communication. Moreover such mechanisms should also be light weight. The energy management system proposed in [5] for efficient duty cycling of sensor nodes focuses on selecting critical sensors for a given task on the basis of their residual energy levels. In their work, a new concept of Quality of Information (QoI) aware sensor to task relevancy is proposed to measure the sensory capabilities of sensor nodes against tasks' QoS requirements. Moreover energy management decision is made at run time in order to ensure the fairness in task distribution. Task transition is modeled using Discrete Time Markov Chain (DTMC) which models the changes in the incoming tasks in order to wake up corresponding sensor nodes.

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

Table 1
Key issues addressed in recent research.

Mechanism	Information sharing among applications	Sensor energy state information gathering	Fair task allocation
QoI [5]	No	No	Yes
UMADE [15]	No	No	No
MARAEE [20]	Yes	Yes (incurs message/energy overhead)	No
SACHSEN [14]	Yes	Yes (incurs message/energy overhead)	No
TRAPS	Yes	Yes (no message/energy overhead)	Yes

In [6] a modular approach for scheduling the tasks is followed to satisfy the energy, delay and reliability requirements using different priority level queues.

A cross layer solution in [7] for QoS aware scheduling argues that QoS management in IoT is still poorly studied [8] and new QoS attributes are proposed such as information accuracy, required energy consumption and coverage for IoT sensory environments. In this work, decision making processes for QoS are proposed at each layer. A new layer: sensing layer and its QoS attributes are proposed whereas QoS attributes at application and network layer are measured with the traditional QoS attributes. Sensing layer does the task of selecting appropriate resources for the incoming tasks.

[9] discusses the task mapping problem that addresses finding the best way to allocate the tasks to sensor nodes. In [14,20], an efficient resource allocation mechanism is proposed in which the applications' requirements are classified on the basis of data accuracy and network lifetime. The scheduling decision is composed of selecting candidate nodes followed by best candidate selection. Network lifetime based applications are scheduled on sensors providing minimum energy consumption while precision based applications are scheduled on sensors with high sensing accuracy. Moreover candidate sensors are selected by considering their locations, sensing ranges and the spatial requirements of tasks. The residual energy levels of candidate nodes are not taken into consideration which is an important decision parameter. As a result, the QoS requirements of an ongoing task is affected in the event of insufficient residual energy [18].

IoT sensory environments may be required to host tasks from Internet with soft and strict temporal requirements. Tasks can involve internode communication among resource constrained sensor nodes affecting the overall service cost. In most of the research on task scheduling, each task is allocated to a node on the basis of its capability and enough residual energy to run the task to its completion. This may cause a sensor node getting selected repeatedly by scheduler, eventually leading to node's death. Allowing such failures can cause energy holes in the network and can lead to network disconnection. We define capability of a sensor node as its ability to sense a given physical quantity and its ability in terms of enough residual energy to run a task to its successful completion. TRAPS mechanism distributes the tasks among multiple sensor nodes in proportion to their residual energy levels. Table 1 shows the key issues addressed in recent research on task allocation.

Most of the related research on IoT sensory environments focus on minimizing overall energy consumption and scheduler level mechanisms. Such scheduling mechanisms treat each incoming task independently causing redundant computations and internode communication. Sharing of sensed information among tasks with similar QoS requirements is mentioned in [14] but authors provide no description about its functioning. In this paper, we propose a task clustering mechanism that identifies and groups the tasks based on similar functional requirements (mainly spatial, *service_type* and temporal requirements). Leader(s) from each group is selected and scheduled for execution. We name this process as task pre-processing. This process reduces the redundant tasks entering into the network and as a result, overall network traffic is reduced. The proposed task clustering mechanism is the first of its kind in the literature to the best of our knowledge.

Another important problem the paper tackles is the increased network traffic due to energy information shared between the sensors and gateway. Most of the recent research in this field assumes that gateway has the residual energy information of all sensor nodes which is possible if sensor nodes send their residual energy information to the gateway either periodically or on demand. This information exchange introduces additional message traffic resulting in energy depletion of sensor nodes. In order to minimize the energy information message exchange overhead, we propose an energy state prediction mechanism based on DTMC that results in efficient estimation of sensor nodes energy levels without any message exchange. The proposed mechanism is built into EMS and supplies nodes' energy information when needed by the scheduler.

3. Problem formulation

This paper considers sensing tasks which are injected into the network to monitor the environment. Such tasks trigger the sensors to sense and route the required environmental parameter without involving any computation on data. In context of this work, tasks are independent execution entities which may have similar functional requirements [20]. Two different applications interacting with WSN have the following requirements, for example: The query from fire detection application is composed of three tasks to gather the temperature, smoke and pressure data respectively from a given location. Another application queries for the visibility parameter consists of a task to measure the visibility of the same given location and the required *service_type* of this task is to gather the smoke data. In this case, both the applications have two tasks (smoke data gathering) with similar *service_type* and location.

We characterize a task as a 4-tuple $\langle service\_type, info\_bits, R_{spat}^{T_i}, R_{temp}^{T_i} \rangle$ where *service_type* denotes the augmented physical quantities temperature, humidity etc, $info\_bits$ is the number of bits information required, $R_{spat}^{T_i}$ is the required geographical (spatial requirement) location associated with the task, $R_{temp}^{T_i}$ is the time period in which the task should be completed (temporal requirement).

In this section, we formulate the problem of task scheduling with the objective of minimizing the network traffic and energy consumption in WSN.

Summary of important symbols

T_i	a sample i from task set
$R_{temp}^{T_i}$, $R_{spat}^{T_i}$	temporal and spatial requirements of T_i
r^{T_i}	composite requirements of T_i
S_i	set of incoming tasks
S_{CCTs}	critical covering tasks set
L_{CCTs}	leader(s) of a critical covering tasks set
T_{CCTs}^i	task i of a critical covering tasks set
$T_{S_i}^i$	task i of incoming tasks set S_i
d_i	delay bound of task i

(*continued on next page*)

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

Summary of important symbols

$dist_{12}$	spatial distance between two task groups
$d(i, j)$	spatial distance between task i and task j
x_k^i	requested spatial location by task i in k^{th} dimension
P	transition probability matrix
N_i	sensor node i
$p_{i,j}$	sensor node state transition probability
P_{slot}^i	initial state probability matrix of node i in a slot
P_0	initial state probability matrix
p_{idle}, p_{rcv}, p_{trans}	probability of a node being in state idle, receiving and transmitting respectively
E_{Total}^i	total energy consumed by node i
E(t)	total energy consumed by a node in a slot
E	initial energy of a sensor node
N	set of sensor nodes

To achieve the set objectives a mapping function Γ: $T_i \rightarrow N_i$ where $T_i \in S_i$ and $N_i \in N$ is formulated. After gathering the requested *service_type* and QoS requirements of incoming tasks, scheduler communicates with EMS (see Fig. 1) to get energy state information about the candidate nodes.

In this task scheduling problem, the relevancy of a sensor to a task [5] (r_{st}) is measured as a function of sensor capabilities (c_s) and task's requirements (q_t) (Eq. 1).

$$r_{st} = f(c_s, q_t) \quad \forall s \in N, \forall t \in S_i \tag{1}$$

The relevant sensor nodes are termed as potential nodes. To prolong the network lifetime, the task should be distributed among the potential nodes in proportion to their remaining energy levels. To ensure fair task distribution the scheduler assigns weights ψ_i to potential nodes. ψ_i is estimated for each potential node dynamically (Eq. 18) based on its residual energy Res_i. Finally tasks are distributed among potential nodes in proportion to their Res_i.

If the total information to be sensed for incoming tasks is s bits and information to be sensed by a node is s_i bits then the objective function is modeled as:

$$maximize : s = \sum_{i \epsilon N_i} \psi_i s_i,$$

subject to:

$$Res_i \geq \xi, i \epsilon N_i \tag{2}$$

where $\psi_i s_i$ signifies the amount of information a node i has to sense. ξ denotes the threshold remaining energy after which the node will no longer be able to service any task. The system must establish a trade-off between the QoS requirements of the incoming tasks and the above mentioned optimized energy consumption.

4. Proposed mechanism design

A modular level discussion on the gateway design is presented in the following.

4.1. Task pre-processor

The pre-processor is responsible for tasks' requirement gathering, classification and leader(s) selection. Algorithm 1 describes the process of task pre-processing.

4.1.1. Task requirement gathering

Task pre-processor extracts required *service_type*, *info_bits*, spatial and temporal requirements from tasks arriving at the gateway node (see Fig. 2). Tasks can have requirements involving information from specific geographic locations. Such requirements are termed as spatial requirements. Similarly delay constraints or information at a particular time are termed as temporal requirements. The extracted information are then used for task classification.

Algorithm 1: Task pre-processor.

```
begin
    1. Requirements = RequirementGathering(Tasks)
    2. if Requirements == Negotiable then
        if Requirements == Time critical then
            send to scheduler
        end
        else
            construct CCTs
            select leaders among CCTs
            send to scheduler
        end
    end
    3. else
        send to scheduler
    end
end
```

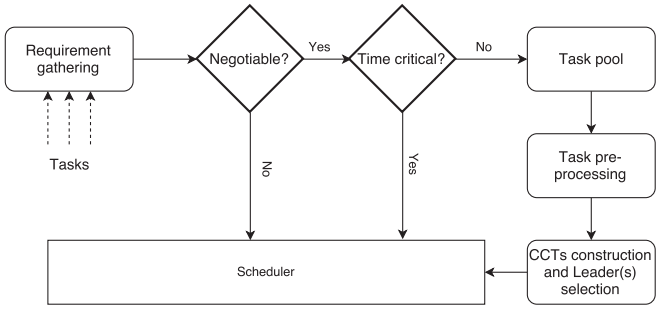

Fig. 2. Task pre-processor.

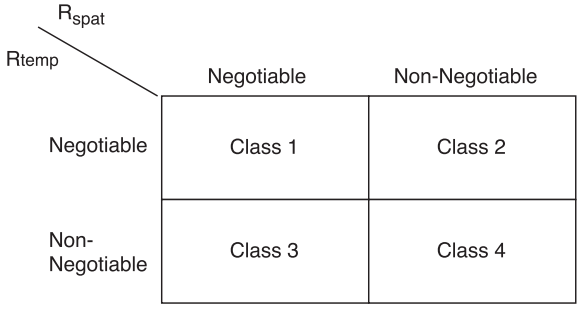

Fig. 3. Task classification.

4.1.2. Task classification based on requirements

Tasks in IoT sensory environments can be Negotiable or Non-negotiable (see Fig. 3). For example one temperature sensing application may tolerate a slight deviation in location specific temperature data and delay (termed as negotiable) whereas another temperature sensing application may not tolerate even a slightest deviation (termed as non-negotiable). In other words, negotiable and non-negotiable tasks have soft real time and hard real time requirements, respectively. Thus tasks can be broadly classified into four types based on their temporal and spatial requirements.

Class 1. Both temporal and spatial requirements are negotiable

Class 2. Temporal requirement is negotiable but spatial requirement is non-negotiable

Class 3. Temporal requirement is non-negotiable but spatial requirement is negotiable

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

Class 4. Both temporal and spatial requirements are non-negotiable

Tasks corresponding to Class 3 and 4 are exempted from pre-processing and scheduled directly (see Fig. 2) since their temporal requirements are non-negotiable. Thus the complexity in fulfilling the requirements increases as we move from Class 1 through 4.

4.1.3. CCTs and leader(s) selection

In the task pool (see Fig. 2), negotiable tasks with same *service_type* are put together in a group and tested for spatial similarity and further grouped where each group consists of tasks with similar spatial requirements. These groups of tasks are called as CCT sets. We introduce the concept of CCT sets whose formation is solely based on the spatial requirements of the tasks. In order to include the temporal requirement of tasks, we select leader(s) from each CCT set. Leader(s) are the tasks from CCT set having strictest temporal requirements. For example, a CCT set having four tasks $\{T_1, T_2, T_3, T_4\}$ with delay requirements in time units as $\{R^1_{temp} < 3, R^2_{temp} < 4, R^3_{temp} < 5, R^4_{temp} < 2\}$ respectively. Here, task T_4 is the leader in the CCT set since it has the strictest delay requirement. CCT sets and leader(s) are formally defined as follows.

Definition 1. CCTs contain tasks which cover nearly all spatial requirements but not necessarily the temporal requirements of the tasks belonging to that group.

If S_i represents a set consists of n tasks $T_1T_2..T_n$; requirements of a task T_i can be represented by Eq. 3 as a function of its temporal and spatial requirements

$$r^{T_i} = f(R^{T_i}_{spat}, R^{T_i}_{temp}) \quad \forall T_i \in S_i \tag{3}$$

CCTs are represented by Eqs. 4 and 5 as a set $S_{CCTs} = T_1, T_2, .., T_m \in S_i$ where $1 \leq m \leq n$.

$$R^{S_{CCTs}}_{spat} \simeq R^{S_i}_{spat} \tag{4}$$

$$R^{S_{CCTs}}_{temp} \neq R^{S_i}_{temp} \tag{5}$$

It is assumed that more than one task may require the information about the same location(spatial requirements) within a delay bound(temporal requirements).

Definition 2. Leaders are those tasks which impose the strictest temporal requirements among all tasks belonging to CCTs.

Leader(s) of a S_{CCT} set having m tasks with delay bounds $R^1_{temp} \leqslant d_1, R^2_{temp} \leqslant d_2, .., R^m_{temp} \leqslant d_m$ is represented by Eq. 6.

$$L_{CCTs} = T^i_{CCTs} \; such \; that \; d_i = \min\{d_1, d_2, \ldots, d_m\} \tag{6}$$

Since leaders(s) impose strictest delay bound, they assure to get the required information within specified delay bound defined by the tasks belonging to corresponding CCT set.

To construct CCT sets initially the incoming n tasks are divided into n groups where each group contains a single task. Since CCT set construction is solely based on the spatial requirements of the tasks, we consider a two dimension decision space where (x_i, y_i) represents the coordinates of the required location by T_i.

Eq. 7 estimates the spatial distance between two tasks to check the closeness of their spatial requirements.

$$d(i,j) = \sqrt{\sum_{k=1}^{2} \left(\frac{x^i_k - x^j_k}{x^{max}_k - x^{min}_k} \right)^2} \tag{7}$$

where x^i_k and x^j_k are the required locations by T_i and T_j respectively in k^{th} dimension and the denominator term $x^{max}_k - x^{min}_k$ is used for normalization. A bottom up hierarchical clustering [11,12,17] approach is used to cluster the similar tasks. We used method proposed in [16] to determine the number of clusters (K) for a given data set. In our case $1 \leq K \leq n$ where n is the total number of tasks. Small value of K represents all the tasks have the same spatial requirement. Equation 9 (see Algorithm 2) is used to estimate the spatial distance between every pair of groups. Two groups with minimum spatial distance are merged together. This process continues until the number of groups reaches a predefined value K which signifies the desired number of CCT sets.

A group of tasks with similar spatial requirements are represented by CCT set given by Eq. 8.

$$S_{CCTs} = \{(T_1, T_3, .., T_l), (T_2, T_6, .., T_n), (T_4, T_8, .., T_p), ..\} \tag{8}$$

The process of leader(s) selection (see Definition 2) is carried at Step 7 in Algorithm 2 . Once the leader(s) are selected, the next step is to assign leader(s) to appropriate sensor nodes. The potential sensor nodes are selected on the basis of their relevancy to the task (see Section 3). Since leader(s) are the tasks executed in the network on behalf of other members of the CCT set, their fitness is equally important as the sensor to task relevancy (Eq. 1). The fitness of selected leader(s) is tested with the help of a fitness function (see Eq. 10). The idea of fitness function is based on to test the leader(s) about their capability to satisfy the requirements of the tasks belong to same CCT set. In real time scenario, it may not always be the case where a sensor node is deployed exactly at the required location by the tasks. So the fitness of leader(s) depend upon two factors: (1) Its average distance from the potential sensor nodes (d_a) and (2) Its minimum tolerable delay. Based on these two factors, the fitness function is defined as

$$\text{fitness} = \frac{1}{\text{tolerable delay} \times d_a} \tag{10}$$

The above fitness function includes spatial and temporal requirements as well.

Algorithm 2: CCTs and leader(s) selection

Data: tasks with QoS requirements
Result: selected leader(s)
begin
1. Choose maximum number of desired groups(K)
2. *CurrentGroups* = 0
3. Divide N tasks into N groups
4. **if** $N \leqslant K$ **then**
   go to Step 7
end
5. **else**
   go to Step 6
end
6. **while** *CurrentGroups != K* **do**
   calculate spatial distance between two groups

$$dist_{12} = \frac{1}{|G_1||G_2|} \sum_{i \in G_1, j \in G_2} d(i,j) \tag{9}$$

   combine two groups with minimum group distance
   CurrentGroups = CurrentGroups+1
end
7. Choose leader(s) from every group having strictest delay bound
end

4.2. Node state modeling and EMS

The logic for node state modeling is deployed in EMS where it shares nodes' energy state information with scheduler. A sensor

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

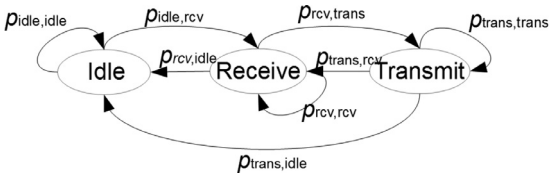

Fig. 4. Sensor node state transition.

Table 2
Node energy state table.

Node	$E(T_1)$	$E(T_2)$	$E(T_3)$	$E(T_4)$	.	.	$E(T_k)$
N_1	6	4	0	2	.	.	12
N_2	6	3	0	3	.	.	9
N_3	6	2	1	3	.	.	7
N_4	6	2	1	3	.	.	7
.	6	2	1	3	.	.	7
N_n	6	2	1	3	.	.	7

node can be in either of the three states – idle, receiving and transmitting and node's state is modeled as DTMC process (see Fig. 4). Eq. 11 represents a node's transition probability matrix **P**.

$p_{i,j}$ denotes the transition probability of a node from state i to state j. **P** can be deduced on the basis of a sensor node's state transition history or discreet time event simulation.

$$\mathbf{P} = \begin{pmatrix} p_{idle,idle} & p_{idle,rcv} & 0 \\ p_{rcv,idle} & p_{rcv,rcv} & p_{rcv,trans} \\ p_{trans,idle} & p_{trans,rcv} & p_{trans,trans} \end{pmatrix} \tag{11}$$

Thus initial state probability matrix $P_0 = \begin{pmatrix} p_{idle} & p_{rcv} & p_{trans} \end{pmatrix}$. A sensor node's initial state probability matrix changes according to its requirements by the tasks. Let the incoming tasks in the current slot requires the service of node a. The initial state probability matrices for nodes $i(=a)$ and $j(=b)$ becomes $P^a_{slot} = \begin{pmatrix} 0 & 0.5 & 0.5 \end{pmatrix}$ *and* $P^b_{slot} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$ respectively. Sensor nodes' state probability matrix gets updated after every slot according to Eq. 12.

$$P^a_{nextslot} = P^a_{currentslot} \times \mathbf{P} \tag{12}$$

Total energy consumption in transmitting, receiving and idle state can be represented by Eqs. 13, 14 and 15 respectively;adopted from [13].

$$E_{trans} = (e_t + e_r r^n) B \; Joules \tag{13}$$

$$E_{rcv} = e_r B \; Joules \tag{14}$$

$$E_{idle} = e_{id} \; Joules \tag{15}$$

where B is the information bits transmitted/received by a sensor node. In our case B represents the $info\_bits$ required by the task (see Section 3). The energy consumption information along with node's state probability matrix enables the EMS compute the total energy consumed by a node in a slot according to Eq. 16.

$$E^i_{Total} = \begin{pmatrix} p_{idle} & p_{rcv} & p_{trans} \end{pmatrix} \times \begin{pmatrix} E_{idle} \\ E_{rcv} \\ E_{trans} \end{pmatrix} Joules \tag{16}$$

Where E^i_{Total} represents the total energy consumed by node i in a slot time. EMS contains a node energy state table (Table 2) where each row represents respective node's energy consumption in successive slots.

Generally, piggyback approach is used for energy state information exchange. Whereas the EMS mechanism does not require any message exchange for the nodes' energy state information gathering. As depicted in [14] the energy consumption of transmitting l-bit data from source to the destination is:

$$E_{tx}(l, src, des) = \sum_{i=1}^{k} E_{tx}(l, d) \tag{17}$$

where k represents the minimum hop count between source and destination and l represents the size of the data being transmitted. Though this approach is far better than the previous research on energy information exchange but there is still an energy overhead due to the increased l. The energy consumption is a function of k and l. In resource constrained environment such as WSN, energy consumption due to every bit of communication must be considered [14,20]. By using prediction based approach, we are minimizing the value of l since only sensed data is being transmitted. This mechanism incurs no energy overhead in gathering energy state information and saves significant energy thus improving network lifetime.

4.3. Scheduler

We assume that scheduler has the information about the sensor nodes' locations and their capabilities. Tasks coming out of the pre-processing module are sent to the scheduler (see Fig. 2). On the basis of tasks' spatial requirements, scheduler selects the potential nodes which can serve them. Selection of nodes is solely based on the mapping between tasks' spatial requirement, nodes geographical location and their capability. Once the potential nodes are selected, scheduler consults with EMS to get the residual energy information about the potential nodes. Residual energy information plays the key role in task scheduling since if the selected sensor nodes' residual energy is less compared to the energy required for completing the task, the sensor node dies and the QoS is affected. To maintain uniformity in residual energy among the nodes, scheduler assigns a weight ψ to each potential node (Eq. 18).

$$\psi_i = \frac{\phi_i}{\sum_{i=1}^{N_c} \phi_i} \tag{18}$$

where $\phi_i = \exp(\frac{-Res_i}{E})$ and $\Sigma_{i \epsilon N} \psi_i = 1$.

E is the initial energy of a node, Res_i is the residual energy of node s_i, N_c represents the total number of potential nodes and ψ_i represents the bits of information to be sensed by sensor s_i. Scheduler assigns the task of sensing bits of information to the potential nodes in proportion to their estimated ψ_i values. In other words, potential nodes having low ψ_i values receive tasks that require less bits of information. This approach of task assignment ensures fair distribution of workload among sensor nodes that avoids the possibility of node failure due to energy drain and allows successful completion of tasks assigned to them. Algorithm 3 depicts the steps involved in the scheduling process. After the completion of the task, all such nodes send the sensed information to the gateway where the received information is consolidated and sent back to the application according to the requirement. The distribution of the data required by different applications is handled by gateway.

4.4. Sensed data distribution

Distribution of information fetched by leader(s) among the tasks within the corresponding CCT set(s) is performed at the gateway and the process is exemplified with the help of an example. Let us consider fire detection and visibility measurement as the two applications (A and B respectively) consisting of different tasks to be performed at the same location (spatial aspect) with in the sensor network. Fire detection application has three tasks (A_1, A_2, A_3) of sensing temperature, smoke and pressure data respectively.

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

Algorithm 3: Scheduling algorithm.

Data: tasks with QoS requirements, location of sensor nodes
Result: task allocation
begin
1. When there are m tasks $T = \{T_1, T_2, .., T_m\}$ in the system at time t
// Task pre-processing
2. **for** *each task* T_i *in* T **do**
  if T_i==*time critical* **then**
    insert into scheduler queue Q
  end
  else
    perform task grouping using Algorithm 2
    construct CCT sets and identify the leader(s)
    insert leader(s) into Q
  end
end
// Potential nodes selection and task allocation
3. **while** *size(Q)* > 0 **do**
  select potential sensor nodes $(c_1, c_2, .., c_k)$ on the basis of tasks' spatial requirements
  assign weights to each potential node c_i
  divide the sensing work among potential sensors
end
end

The visibility application on the other hand has two tasks (B_1, B_2) of sensing smoke and luminosity data respectively. According to Algorithm 2, after considering the *service_type* and spatial requirements of the tasks, there will be four CCT sets $\{A_1\}$, $\{A_2, B_1\}$, $\{A_3\}$ and $\{B_2\}$. Since A_2 poses the strictest delay requirement among its belonging CCT set members (see Table 3), it is selected as a leader. While other CCT sets have only a single member. Once A_2, A_1, A_3 and B_2 are selected as leaders, scheduler searches for the potential sensors based on the spatial requirement and required *service_type* of the leaders. This process is followed by the assignment of leaders to the potential sensor nodes.

Tasks A_2 and B_1 require data for a duration of t_1 (10 time units) and t_2 (5 time units) respectively given that the sensing rate of the potential sensors is s_r bits/unit time where $t_2 < t_1$, then task B_1 is said to be less demanding. Thus, the fetched bits are distributed among A_2 and B_1 as $t_1 \times s_r$ bits and $t_2 \times s_r$ bits respectively. The fetched bits of information by tasks A_2, A_1, A_3 are provided to the fire detection application (A). Similarly information bits fetched by A_2 is shared with task B_1 of visibility measurement application (B) along with data fetched by B_2.

5. System analysis

The performance analysis of task pre-processor and EMS is presented in this section. Further the proposed mechanism is validated by theoretical analysis.

5.1. Performance analysis of task pre-processor

Task pre-processor's efficiency is measured in terms of minimizing the number of Class 1 and Class 2 type tasks entering into the WSN. Further the credibility of selected leader(s) are evaluated using a fitness function (Eq. 10). We considered the same simulation parameters (task arrival follows the Poisson distribution with a mean of 20 time units and the maximum number of tasks is 18) as considered in [14] to test the efficiency of our model. We conducted a number of experiments for varying amount of incoming tasks in each time unit. The task sharing concept used in [14] does not provide a clear mechanism to reduce the number of tasks in the network. Thus we do not choose to compare our results with SACHSEN. We choose UMADE [15] and Semi-random approaches to compare the results in terms of number of tasks entering in WSN to be scheduled. UMADE is a QoS aware approach with the objective of quality of monitoring while Semi-random is a non QoS aware approach which does not consider the QoS requirements of the tasks and assigns a sensor node randomly from a list of candidate sensor nodes. Fig. 5 shows a comparison between UMADE, Semi-random schemes and TRAPS in terms of number of tasks scheduled in WSN. Since UMADE and Semi-random schemes do not pre-process the tasks the number of tasks scheduled is equal to the number of tasks arrivals at gateway. Due to task pre-processing the number of tasks scheduled in TRAPS is less than the number of task arrivals at gateway. Since number of tasks in the network affects network parameters including traffic, application end-to-end delay and energy consumption, reducing number of tasks entering into WSN helps in reduced redundant network traffic increased network lifetime.

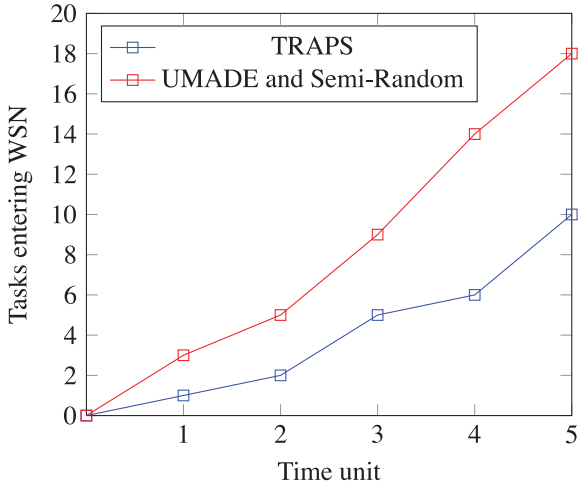

Fig. 5. Redundant task removal.

To evaluate the efficiency of the leader(s) selection process, it is necessary to evaluate the fitness of the selected leader(s). Leader(s) selection is modeled as a two-objective problem where one is to minimize the difference between the spatial requirements and available sensor node location; and the second is to minimize the delay in order to meet the temporal requirements of the tasks belonging to CCT sets. One can assume CCT set members as the solutions of the aforementioned problem of leader(s) selection. This section evaluates the fitness of CCT set members and selects the fittest member as leader(s).

Table 4 shows six sample tasks $\{T_1, T_2, T_3, T_4, T_5, T_6\}$ in a time slot with their spatial and temporal requirements, respectively. Algorithm 2 computes the CCT sets as $\{T_1, T_5\}$, $\{T_2, T_6\}$, $\{T_4\}$, $\{T_6\}$ and the selected leader(s) are $\{T_5, T_6, T_3, T_4\}$. We used five random locations where the sensor nodes are available to serve them and conducted experiments to test the fitness of T_1, T_5 and T_2, T_6.

Figs. 6 and 7 show the fitness comparison of Tasks. The fitness values are obtained using Eq. 10 where both variables are normalized. Tasks with higher fitness value are selected as leader(s). Task's higher fitness value indicates low d_a and *tolerable delay* which implies the selected leader(s) can meet the requirements of itself and other tasks in its CCT set. This experiment states that when a sample of incoming tasks is taken at a time unit, the number of tasks to be scheduled in the WSN is reduced from six to four.

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

Table 3
Example of two applications with their corresponding tasks.

Task	*x* coordinate	*y* coordinate	*service_type*	Tolerable delay (Time units)	Execution duration (Time units)
A_1	4	5	Temperature	2	7
A_2	4	5	Smoke	3	10
A_3	4	5	Pressure	2	8
B_1	4	5	Smoke	4	5
B_2	4	5	Luminosity	3	9

Table 4
Tasks' spatial and temporal requirements.

Task	*x* coordinate	*y* coordinate	Tolerable delay (seconds)
T_1	2	3	8
T_2	3	4	7
T_3	1	2	9
T_4	1	5	4
T_5	2	4	5
T_6	3	5	4

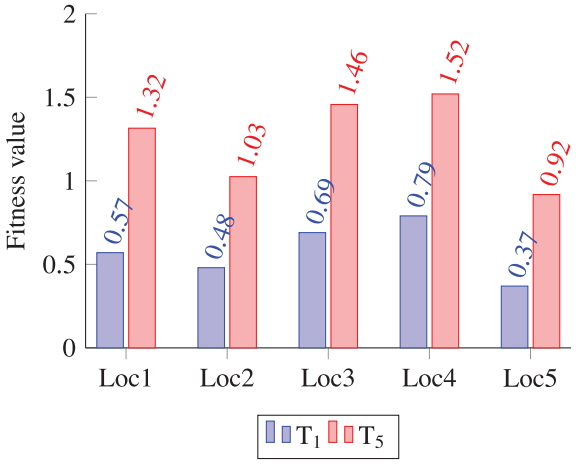

Fig. 6. Fitness comparison of Task 1 and Task 5.

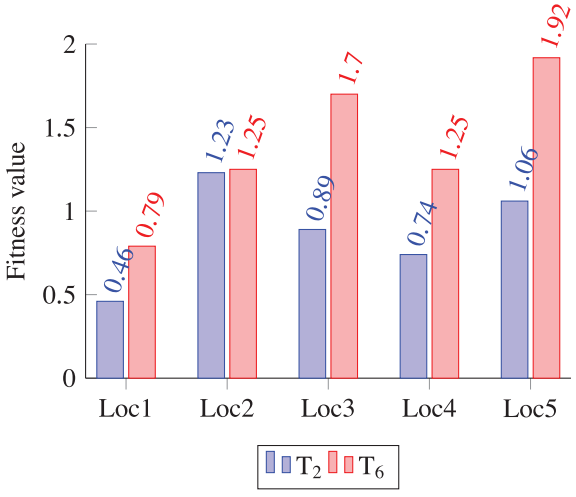

Fig. 7. Fitness comparison of Task 2 and Task 6.

In order to observe the reaction of task pre-processor when tasks with different temporal requirements are supplied and the effect of number of CCT sets; value of K (see Algorithm 2), experiments were performed with different values of K with increasing variance in the temporal requirements of the tasks. The variance in the temporal requirements of the tasks represents the difference in tasks' time requirements. For example (see Fig. 8), variance value of 2 represents that supplied tasks' temporal requirements are close to each other. Similarly, a higher variance represents the temporal requirements of tasks are wide apart. The following experiment was performed with 50 tasks in each task set. Five samples of task sets with different variance in temporal requirements (2, 4, 6, 8, 10) are supplied and the reaction of task pre-processor is recorded in terms of percentage of tasks meeting deadline from each set.

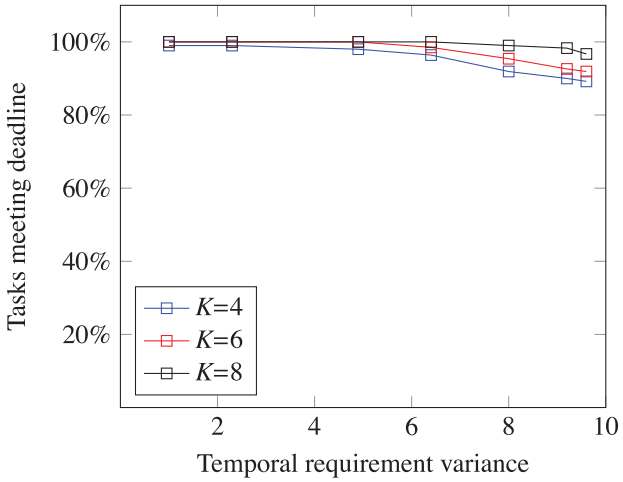

Fig. 8. Impact of different temporal requirements and value of K on tasks' temporal deadline.

Fig. 8 depicts that when $K = 8$, there are more number of clusters implying more number of smaller size CCT sets that helps in satisfying temporal constraints of the tasks. Thus the temporal requirement variance does not affect much in this case but as the cluster size increases ($K = 6$ and 4) the percentage of tasks meeting their temporal requirements decreases (more number of tasks in each CCT set). On the other hand, large value of K does not help much at task reduction stage whereas small value of K put tasks temporal requirements at risk. So, the optimum value of K should be taken in order to avoid such issues.

5.2. Performance analysis of EMS

As discussed in Section 4.2, we use DTMC in order to predict the energy consumption of sensor nodes. We assume that the transition probability matrix **P** is known *a* priori to EMS. **P** is obtained by using discreet time event simulation and realized during the deployment stage. In order to calculate **P**, more than 50 (70% negotiable and 30% non-negotiable) sensing tasks (no data processing requirement) with varying attribute values are supplied. The tasks specify their attributes including number of bits 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 5.

The expected values in **P** exactly represents the average number of transitions between each pair of two states. After the simulation of over 1000 sensor nodes transitions the transition

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

Table 5
A sample task format to obtain **P**.

Attribute	Value
Start time	2 s
Stop time	20 s
Required information	100 bits
Target location	[45.4, 10.7]
Service type	Temperature data

Table 6
Simulation parameters for EMS.

Parameter	Value
e_t	$50 \times 10^{-9} J/bit$
e_r	$50 \times 10^{-9} J/bit$
Power index (n)	2
e_{id}	$40 \times 10^{-9} J/bit$
Bit rate (B)	$.25 Mbit/s$
Range (r)	1342m
Transmission power (T)	30dBm

probability matrix obtained is

$$\mathbf{P} = \begin{pmatrix} 2/7 & 5/7 & 0 \\ 1/5 & 2/5 & 2/5 \\ 2/13 & 7/13 & 4/13 \end{pmatrix} \tag{19}$$

Simulation parameters used for this module(mentioned in Section 4.2) are shown in Table 6.

In order to check the correctness of the proposed energy prediction model, we tested our mechanism for different number of tasks in the network. This 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 nodes' residual energy levels are recorded according to each task. In order to record the actual energy consumption of a sensor node required by incoming task, we divided the energy consumption into three main blocks [21] : E_{NET} (for data communication/networking), E_{ACQ} (for data acquisition) and E_{PRC} (for data processing). The energy consumed by communication block can be expressed (by Eq. 20) in terms of the energy required to send a bit E_{trans}, and the time between the consecutive bits T^i_{bit}.

$$E_{NET} = \frac{\sum_{i=1}^{N_{bits}} E_{trans}}{T^i_{bit}} \tag{20}$$

E_{ACQ} is expressed by Eq. 21 for the regular monitoring of the required parameter.

$$E_{ACQ} = E_{SMP}.N_S \tag{21}$$

where E_{SMP} is the energy needed to acquire on sample and N_S is the total number of samples acquired. Since there is no processing of data, the E_{PRC} has been ignored in this paper. Then, our prediction mechanism is applied on the same network with same task graphs. Fig. 10 illustrates a sample task graph where T_1 is the leader and T_2, T_3 and T_4 are the other members of CCT set which depend on T_1. Fig. 9 shows the energy consumed by a node when number of tasks targeted to that node are varying. The proposed mechanism has a good accuracy rate with overall Root mean square deviation [10] value of 0.028.

The objective of EMS is mainly to reduce the traffic generated due to energy information exchange between sensor nodes and sink. Proposed mechanism is non-message exchange based and thus reduces the network traffic which results in energy saving of sensor nodes.

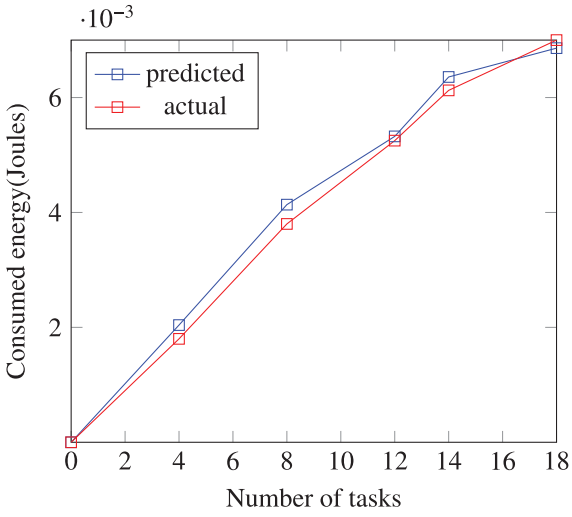

Fig. 9. Energy consumption prediction.

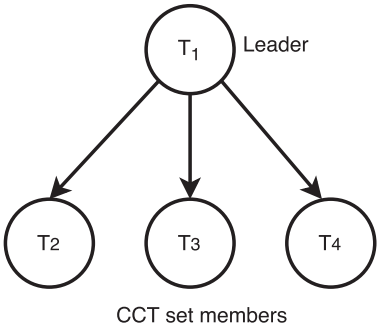

Fig. 10. Sample task graph.

5.3. Performance analysis of TRAPS

In order to analyze the proposed mechanism, a queuing model of type $M/G/1$ [9] is considered. According to this model, the entire sensor network acts as a server where task arrival rate λ follows Poisson distribution, tasks service time μ is unknown;follows a general distribution and of no specified queue size.

The average number of tasks in the network according to $M/G/1$ model can be represented as

$$L = \frac{\lambda^2 E[\mu^2]}{2(1-\lambda E[\mu])} + \lambda E[\mu] \tag{22}$$

Average amount of time a task spends in the network is represented as

$$W = \frac{\lambda^2 E[\mu^2]}{2(1-\lambda E[\mu])} + E[\mu] \tag{23}$$

Eqs. 22 and 23 depicts both $E[\mu]$ and λ are tunable parameters that affect L and W. In the context of our work if the average number of tasks entering (L) and the average amount of time a task spends in the network (W)can be controlled while meeting the desired QoS aspects, the overall service cost of servicing a task is reduced. As a result, a fair amount of network resources is saved which is one of the objectives of our work. Since all time critical tasks are scheduled directly, we consider only the tasks coming from task pre-processing module for analysis.

Considering four tasks belonging to a CCT set with temporal requirements $R^1_{temp} \leqslant d_1, R^2_{temp} \leqslant d_2, R^3_{temp} \leqslant d_3, R^4_{temp} \leqslant d_4$ and task 4 having strictest delay requirement among all then task 4 is selected as leader and will be the one serviced by the network. Task

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

4 will complete its required services within a delay constraint of $d_4 \ll d_1, d_2, d_3$ which reduces the overall service time of entire CCT set.

This argument confirms that proposed mechanism helps in reducing the average time spent by a task in the network. It can also be inferred that small values of W results in enhanced network lifetime. Another objective of the proposed mechanism is to reduce the number of tasks in a slot τ. Leader(s) being a subset of CCT set it results in reduced λ, leading to small values of L.

Considering Eqs. 22 and 23, the total service cost C_s for servicing the tasks is:

$$C_s = L \times W \tag{24}$$

It is evident that the proposed mechanism minimizes both L and W and so the total service cost.

Proposition. Assuming that selected leader(s)' temporal requirements are satisfied by WSN, every task's temporal requirements belonging to corresponding CCT set will be satisfied.

Proof. Assume, for the purpose of contradiction, in spite of ignoring network delay and congestion, leader(s) are not able to satisfy the temporal requirements of the tasks belonging to corresponding CCT set. In this scenario, leader(s) do not have strictest delay bounds among the belonging CCTs. In that case, there is some other task g belonging to the CCT set having delay bound $d_g < d_{leader(s)}$. According to Definition 2, leader(s) are selected on the basis of strictest delay bounds. So both task g and selected leader(s) can not be the leaders until and unless $d_g = d_{leader(s)}$. So $d_g \nless d_{leader(s)}$. This contradicts the assumption and ensures the proposition. □

6. Performance evaluation

Performance of the proposed gateway design is evaluated by simulating the scenario in OPNET 18.0 modeler. Since the proposed gateway design is modular, we test the performance of task preprocessor and EMS individually. As the scheduler uses these two modules to schedule the incoming tasks from Internet in energy efficient manner, the performance of the complete gateway design when all modules put together is measured in terms of its effects on sensor network lifetime, application end-to-end delay and sensor network load.

6.1. Parameters setup

The simulation parameters used in the experiments includes task, system and energy parameters. The task arrival follows Poisson distribution with the mean time of 100 s. The number of applications (= 3) and number of tasks in each application (= 2 to 6) are considered same as in SACHSEN for the fair comparative analysis. The information bits required by a task are generated from a uniform distribution ranging from 1 to 20. And the time spent in scheduling process is considered to be far less than the minimum temporal requirement of the task belonging to CCT set.

System consists of a heterogeneous WSN with 1 pan coordinator and sensor nodes ranging from 1 to 100. All sensor nodes are considered to have the same sensing ranges. A cluster based multihop WSN topology is considered for the simulation. The energy parameters settings are given in Table 4. As shown in Fig. 11, as the number of non-negotiable tasks increases in the network, the energy consumption also increases. But in many real time scenarios, the number of non-negotiable tasks in the network are much less than the negotiable ones. The experiments stated in the next section are performed with the 70% of negotiable and 30% of non-negotiable tasks in the network. The value of K for all the experiments is set to be 6. The comparison with SACHSEN and MARAEE is done for the β value of 1 throughout the experiments.

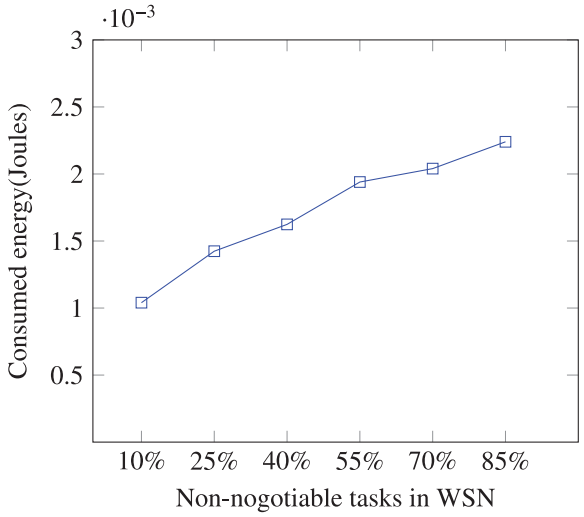

Fig. 11. Impact of increased non-negotiable tasks on energy consumption.

6.2. Effects on network parameters

TRAPS is evaluated by studying its effects on Network output load and Energy consumption. SACHSEN and Semi-random task allocation schemes are considered as references for comparative analysis.

SACHSEN is a QoS aware task allocation scheme with the objective of coming up with a trade-off between available network resources and QoS requirements. SACHSEN selects the candidate nodes for a task based on their location and task's spatial requirements which is similar to our proposed scheme. Instead of selecting one sensor node for a task (as in SACHSEN), TRAPS selects more than one sensor nodes on the basis of their proximity to the task's spatial requirement and divides the task among candidate sensor nodes based on their residual energy values. SACHSEN uses application data sharing approach to reduce the traffic in the network. Semi-random approach is a non QoS aware approach which selects the candidate sensor nodes based on the spatial requirements of task but selects one sensor node randomly among the candidate sensor nodes. Fig. 12 shows the effects of all three schemes on network load which is the function of the total traffic in the network. As seen from the figure, the network load in SACHSEN and TRAPS is less when compared with semi-random approach. Network load is measured in terms of number of bits transmitted per second. Due to the task sharing mechanism used by SACHSEN and TRAPS, they are able to reduce the network traffic which results in less network load. On the other hand Semi-random does not use such approach to reduce the network traffic which results in increased network load. As the number of tasks increases in the network, the network load measured in TRAPS is less as compared to SACHSEN. SACHSEN uses piggybacking for energy state information exchange which increases the packet size and number of bits transmitted, while maintaining the number of packets same as in TRAPS. On the contrary, in TRAPS, EMS uses prediction based energy state information monitoring and incur no extra overhead which results in low network load.

Energy is the key parameter in resource constrained environments such as WSNs. Fig. 13 shows the average energy consumption of a sensor node in the deployed network. When all three

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

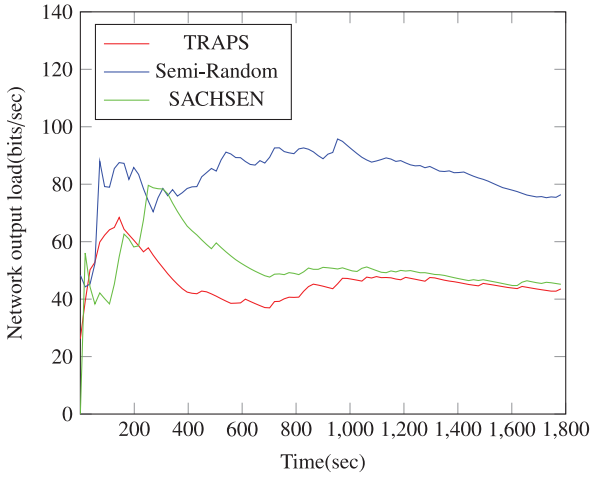

Fig. 12. Network output load for all three allocation schemes.

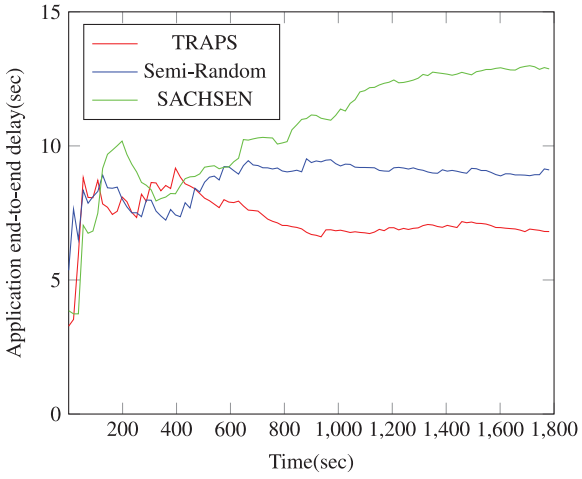

Fig. 14. Application end-to-end delay for all three schemes.

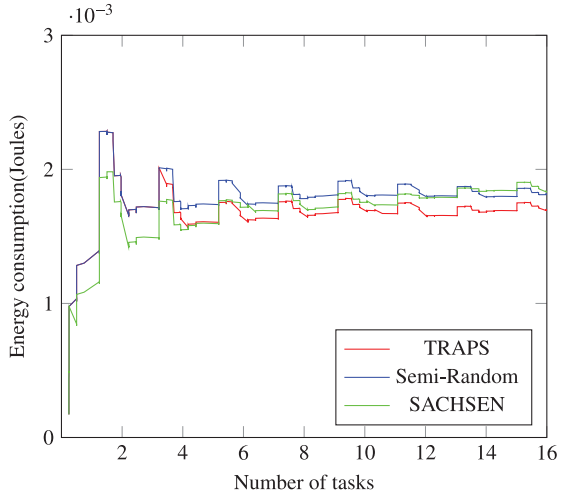

Fig. 13. Average energy consumption of a node for all three allocation schemes.

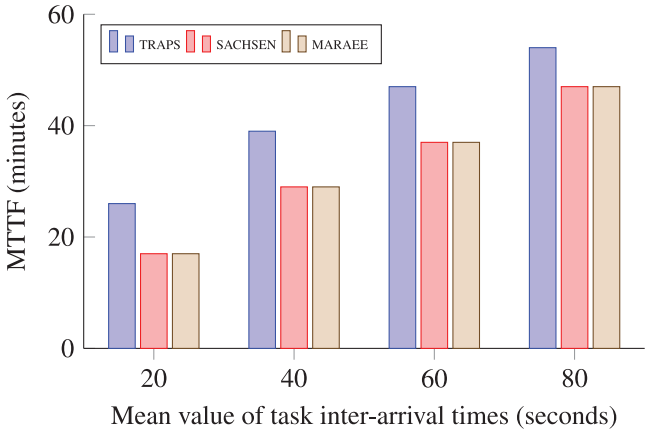

Fig. 15. System lifetime comparison for different values of task inter-arrival times.

schemes were tested for their energy efficiency, it was found that energy consumption increases as the number of tasks increases in the network (as shown in Fig. 13). In this experiment SACHSEN performs better over semi-random since only lifetime based application are considered [14]. The reason behind SACHSEN outperforming semi-random is due to its task sharing mechanism. On the other hand semi-random is a non QoS aware scheme which schedules every task. This increases the network traffic which results in high energy consumption. It is shown in Fig. 13 that, initially, SACHSEN and TRAPS perform equally well but as the number of tasks increases in the network, TRAPS is found to be more energy efficient. As can be inferred from Eq. 17, energy consumption increases with number of bits transmitted. The piggybacking approach used in SACHSEN results in extra bits overhead in the network which leads to more energy consumption as compared to TRAPS.

6.3. QoS assessment

The WSN QoS requirements can be classified into two categories: application specific QoS and network specific QoS [20]. For the experiments, we consider end-to-end delay and network lifetime as application specific QoS and network specific QoS respectively. The end-to-end delay is measured using M/G/1 queuing model which includes service time, network transmission time and waiting time in the queue. SACHSEN does not study the effects of the task allocation mechanism on application end-to-end delay. Since we are considering negotiable and non-negotiable applications in this paper, the application end-to-end delay becomes the crucial parameter of study. As shown in Fig. 14, as the number of tasks increases, end-to-end delay of SACHSEN and Semi-random approaches increases as well. On the other hand, TRAPS experiences less end-to-end delay as compared to the other schemes. This is because the task pre-processor module in TRAPS considers both spatial and temporal requirements before scheduling the leader(s) in the network. Tasks with strictest delay requirements are selected as leader(s) so that they can satisfy the temporal requirements of all the tasks belonging to the CCT set. SACHSEN does not consider such temporal requirements of the tasks which results in poor performance when comes to end-to-end delay. Semi-random experiences less end-to-end delay than SACHSEN because it allocates the sensor nodes randomly to the incoming tasks without considering any QoS requirement which reduces the best sensor node selection time.

Network lifetime is the most important QoS parameter in context of resource constrained environments such as WSN. Since the failure of a single node in the network can leave the network disconnected. In order to assess the network lifetime, we consider Mean Time to Failure (MTTF) as the measured parameter. In this experiment, MTTF is the time a sensor node is expected to last in operation. Network lifetime is directly proportional to the sensor node's MTTF as a single node failure may lead to network disconnection and halt the ongoing task executions. Fig. 15 shows the

comparison between TRAPS, SACHSEN and MARAEE in terms of MTTF when tasks with different inter arrival times are injected in the network. MARAEE is another QoS aware task scheduling algorithm with an objective of increasing network lifetime while maintaining the applications requirements. As shown in Fig. 15, initially when the number of tasks are more in the network, the MTTF for a node is less as compared to when the task arrival rate decreases.

TRAPS outperforms SACHSEN and MARAEE in terms of MTTF of sensor nodes in the network. It is to be noted that average energy consumption in the network and network lifetime are two different parameters considered in this paper. SACHSEN performs nearly as good as TRAPS in terms of energy consumption but lacks when comes to network lifetime. Network lifetime depends upon the fair task distribution among the sensor nodes. In SACHSEN and MARAEE, a single node serves the task whereas in TRAPS, task is distributed among the available sensor nodes in proportion to their residual energy levels. Fair task distribution increases MTTF of sensor nodes which results in increased network lifetime.

7. Conclusion and future work

This paper presents TRAPS mechanism at the gateway. An analytical model for the problem scenario is developed and evaluated with the help of simulation. Obtained results justify the efficacy of the proposed concept in terms of reduction in the amount of task executions and overall service cost. Further simulations reveals that proposed scheme outperforms existing task allocation schemes in terms of application end-to-end delay, network load and energy consumption. Scheduling on the basis of nodes' remaining energy resulted in fair distribution of tasks which implies improved network lifetime.

Future work includes partially executed tasks and other application specific QoS parameters.

Acknowledgments

The authors would like to thank the reviewers for their valuable suggestions.

References

[1] L. Atzori, A. Iera, G. Morabito, The internet of things: A survey, Comput. Netw. 54 (15) (2010) 2787–2805.
[2] Y. Zhang, R. Yu, M. Nekovee, Y. Liu, S. Xie, S. Gjessing, Cognitive machine–to-machine communications: Visions and potentials for the smart grid, IEEE Netw. 36 (3) (2012) 6–13.
[3] G. Wu, S. Talwar, K. Johnson, N. Himayat, K. Johnson, M2M: From mobile to embedded internet, IEEE Commun. Mag. 49 (4) (2011) 36–53.
[4] J. Zheng, D. Simplot-Ryl, C. Bisdikian, H.T. Mouftah, The internet of things, IEEE Commun. Mag. 49 (11) (2011) 30–31.
[5] C.H. Liu, J. Fan, J.W. Branch, K.K. Leung, Toward qoi and energy-efficiency in internet-of-things sensory environments, IEEE Trans. Emerg. Top. Comput. 2 (4) (2014) 473–487.
[6] D. Djenouri, I. Balasignham, Traffic-differen-tiation-based modular qos localization routing for wireless sensor networks, IEEE Trans. Mob. Comput. 10 (6) (2011) 797–809.
[7] L. Li, S. Li, S. Zhao, Qos-aware scheduling of services-oriented internet of things, IEEE Trans. Ind. Inf. 10 (2) (2014) 1497–1505.
[8] B. Billet, V. Issarny, From task graphs to concrete actions: A new task mapping algorithm for the future internet of things, in: Proceedings of IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems, 2014, pp. 470–478.
[9] S.M. Ross, Introduction to Probability Models, Sixth edition, Academic Press, U.S.A., 2006.
[10] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, second edition, MIT Press, Cambridge, U.S.A., 2001.
[11] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley and Sons, 2005.
[12] D.E. Goldberg, Genetic Algorithms for Search, Optimization, and Machine Learning, Reading, MA: Addison-Wesley, 1989.
[13] Q. Gao, K.J. Blow, D.J. Holding, I.W. Marshal, X.H. Peng, Radio range adjustment for energy efficient wireless sensor networks, Ad-hoc Netw. J. Elseveir Sci. 4 (1) (2006) 75–82.
[14] W. Li, F.C. Delicato, P.F. Pires, Y.C. Lee, A.Y. Zomaya, C. Meceli, L. Pirmez, Efficient allocation of resources in multiple heterogeneous wireless sensor networks, J. Parallel Distrib. Comput. Elsevier Science 74 (2014) 1775–1788.
[15] S. Bhaattacharya, A. Saifullah, C. Lu, G.-C. Roman, Multi-application deployment in shared sensor networks based on quality of monitoring, 16th IEEE Real-Time and Embedded Technology and Applications Symposium(RTAS), 2010.
[16] R. Tibshinani, G. Walther, T. Hastie, Estimating the number of clusters in a data set via the gap statistic, J. R. Stat. Soc. B 63 (2) (2001) 411–423.
[17] L. Rokach, O. Maimon, in: Clustering Methods. Data Mining and Knowledge Discovery Handbook, Springer US, 2005, pp. 321–352.
[18] P. Gil, A. Santos, Dealing with outliers in wireless sensor networks: An oil refinery application, IEEE Trans. Control Syst. Technol. 22 (4) (2014) 1589–1596.
[19] J.-H. Kim, D.-H. Kim, H.-Y. Kwak, Y.-C. Byun, Integration between WSNs and internet based on address internetworking for WEB service, Comput. Inf. 27 (2008) 707–718.
[20] W. Li, F.C. Delicato, P.F. Pires, A.Y. Zomaya, Energy-efficient task allocation with quality of service provisioning for concurrent applications in multi-functional wireless sensor network systems, Concurr. Comput. 26 (2013) 1869–1888.
[21] M. Borja, M. Monton, I. Vilajosana, D.J. Prades, The power of models: Modeling power consumption for iot devices, IEEE Sensors J. 15 (10) (2015) 5777–5789.

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005

Sourabh Bharti received his Bachelor degree in Computer Science and Engineering from Guru Gobind Singh Indraprastha University, New Delhi, India in 2010 and received Master's degree in Computer Science and Engineering from ABV-Indian Institute of Information Technology and Management, Gwalior, India in 2013. He worked as a Software Engineer in Tata Consultancy Services from 2013 to 2014. Currently he is pursuing Ph.D. from ABV-Indian Institute of Information Technology and Management, Gwalior, India and an Academic Visitor at the Anglia Ruskin University, Chelmsford, U.K. His research interests include network engineering, Data Center Networks and QoS provisioning in IoT and WSNs.

K.K. Pattanaik received Bachelor of Engineering in Electrical and Electronics Engineering from Kuvempu University, Shimoga, in 1997, Master of Technology in Computer Science and Engineering from Motilal Nehru National Institute of Technology Allahabad and subsequently, PhD in Engineering with Computer Science as major from Birla Institute of Technology-Mesra, Ranchi in the year 2010. Currently, a full time faculty at ABV-Indian Institute of Information Technology and Management Gwalior, India. His research interests are Distributed Systems, Grid Computing, Mobile Computing, Multi Agent Systems and Wireless Sensor Networks.

Please cite this article as: S. Bharti, K.K. Pattanaik, Task requirement aware pre-processing and Scheduling for IoT sensory environments, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.07.005