QoS-Aware Sensor Allocation for Target Tracking in Sensor-Cloud

Sudip Misra, Anuj Singh, Subarna Chatterjee, Amit Kumar Mandal

School of Information Technology, Indian Institute of Technology, Kharagpur
West Bengal, India - 721302

Abstract

This work addresses the problem of Quality of Service (QoS) aware sensor allocation for target tracking in a sensor-cloud environment. In a sensor-cloud environment, whenever a target enters within a sensor deployed zone, physical sensor nodes are dynamically scheduled and allocated for the corresponding target. In this work, specifically, we address the issue of selection of an optimal set of sensors to track a particular target. However, in sensor-cloud the underlying physical sensor nodes are heterogeneous with respect to their owner, their sensing ability, transmission range, and the unit cost of usability. Considering the heterogeneity of the nodes, we propose the *QoS-aware Sensor Allocation Algorithm* (*Q-SAA*) that takes into account an assortment of parameters that determine QoS. Thereafter, using an auction-based mechanism we find the optimal solution for allocation of a subset of available sensors to achieve efficient target tracking. Experimental results on implementation of our solution show that in comparison with the considered benchmark, the proposed scheme schedules approximately 20-30 % less number of sensors for target tracking applications and still achieves the desired QoS while tracking the target.

Keywords:
Wireless Sensor Network (WSN), Cloud Computing, Target tracking, Quality of Service, Auction theory

1. Introduction

Recent research has perceived sensor-cloud infrastructure as a potential substitute of the traditional Wireless Sensor Networks (WSNs) [5, 17, 28]. Although sensor-cloud has been conceptualized and envisioned to mitigate the limitations of conventional WSNs, there is still a scarcity of research to support it from an implementation point of view. This work addresses an application specific issue within sensor-cloud.

The emergence of WSN has spawned huge enhancement in the field of research. However, such WSNs are single-user centric, and end-users who do not own sensors are unable to have access to any WSN-specific application. Also, the sensor nodes are constrained by many issues and challenges with respect to computation power, memory, and communication range. To mitigate the aforesaid issues, sensor-cloud infrastructure has been perceived as a potential replacement of the traditional WSNs [5, 17, 28]. As defined by MicroStrains, who is considered to be one of the pioneers in sensor-cloud, sensor-cloud infrastructure can be introduced as "A unique sensor data storage, visualization and remote management platform that leverages powerful cloud computing technologies to provide excellent data scalability, rapid visualization, and user programmable analysis" [5]. Sensor-cloud thrives on the principle of virtualization of physical sensor nodes and rendering them as an on-demand easily obtainable service, *Sensors-as-a-Service* (*Se-aaS*). To obtain Se-aaS, end-users are required to send their application demand to sensor-cloud, which in turn, schedules and allocates a set of physical sensor nodes to serve the application [28].

In this work, we focus on an application specific scheduling and allocation of physical sensor nodes to serve a target tracking application within sensor-cloud infrastructure [10]. The requirement of sensors in a target tracking application depends on the movement of the target. In such a scenario, there exists a cloud service provider having a number of sensors owned by different sensor owners [9]. Services of these sensors are managed by cloud controller/administrator to meet the dynamic demands of the end users. Consequently, the end-users can dynamically demand and obtain Se-aaS. Different allocated sets of sensors forming a virtual sensor group within the cloud are used by the users for their disparate application. In such a framework, the end-users are unaware of the exact physical location of the sensors.

1.1. Motivation

In a conventional target tracking application, every user-organization that wants to track a target has to deploy its own WSN. Consequently, for tracking within the same zone, multiple users need to deploy separate WSNs on behalf of one another. Also, there is no sharing of data leading to duplicity of effort and resources. Sensors of a WSN are entirely application-specific. Users of a WSN are always concerned about the issues connected with network deployment, and the actual physical location of the sensors. Moreover, as sensor nodes are highly resource-constrained, users of a WSN have to survive with different network overheads on their own. Further, WSN services are not accessible to end-users who do not own the deployed sensors.

In target tracking within sensor-cloud, the sensors are reused for the sensing ability, whereas the tracking applications are executed at the used end. Based on application demand, physical sensor nodes are allocated to serve a particular tracking application. In such a scenario, it is to be taken into consideration at the cloud end that an end-user is provisioned with an optimal set of sensor nodes that ensure the Quality of Service (QoS) at a reasonably payable cost.

1.2. Contribution

This work focuses to address the problem of QoS-aware sensor allocation for target tracking in a sensor-cloud platform. The contributions of the work are multifold and are discussed as follows.

- Initially, the work models few parameters in terms of availability of sensor nodes, accuracy of sensor nodes, dwelling time of a target within a sensor coverage, and the detection probability of a particular set of sensors, that are explicit to a target tracking application.
- The *QoS-aware Sensor Allocation Algorithm* (*Q-SAA*) is proposed, in which the "best" suited sensors are allocated to a target, based on certain parameters that quantify QoS in regards availability, accuracy, dwelling time, detection probability. The sensors are abstracted as a virtual group and data from them are delivered to the end-user through the sensor-cloud infrastructure.
- As sensor-cloud follows a pay-per-use model, in which an end-user pays only for the resources consumed by him, the cost incurred due to availing a set of physical sensor nodes is mathematically formulated. The

incurred cost is modeled by the provisioned QoS of the particular sensor set.

- The work formulates a direct revelation based auction mechanism in which the members of the maximal set of sensor nodes place a bid based on the provisioned QoS. The end-user acts as an auctioneer and chooses the subset of sensors that optimizes his/her cost and ensures a threshold QoS, simultaneously.

Apart from the the design issues of Q-SAA, the work also analyses the real-time computing ability of the algorithm, thereby inferring to implement Q-SAA in areal-world application scenario.

1.2.1. Contribution of Auction Theory

As mentioned earlier, the proposed algorithm Q-SAA is based on the direct revelation mechanism of auction theory. The basic motivation behind implementation of auction theory in this work is that an end-user of sensor-cloud may not be aware of the potential price that s/he has to pay for obtaining Se-aaS. The end-user expects to enjoy a threshold QoS at a reasonable price. In this work, the theory of auction enables the end-user to play the dominant role of the auctioneer. This allows the end-user to select an optimal subset of sensors that provision Se-aaS with QoS within the payable limits.

As the sensor nodes behave as the bidders of the system, every node tends to be within the selected subset in order to obtain an incentive (in terms of money that the end-user has to pay to the cloud service provider) on behalf of the sensor-owner. The overall scenario is conceptualized as an incomplete-information game which has a point of equilibrium, or in other words, can be stated as the revelation principle of auction theory. The control and negotiation of the pricing of the allocated physical sensors is explicitly managed through the auction mechanism.

1.3. Organization of the paper

Our work is organized as follows. In Section 2, we briefly elaborate the related work on sensor-cloud. Section 3 describes the problem statement and mathematical model of the system. In Section 4 we formulate an auction-based mechanism for the selection of an optimal set of sensors. Section 5 presents the results of simulation, and highlights the economics behind using

a sensor-cloud platform vis-a-vis a privately owned WSN for target tracking. Section 6 concludes the work with directions for future work.

2. Related Work

The ideology and dogma of sensor-cloud was proposed by Yuriyama and Kushida [28] in which the virtualization of physical sensors was proposed. Yuriyama *et al.* also propounded the model of sensor-cloud for accelerating the service innovation [29]. The work was further extended by Madria *et al.* [17] in which the different mapping configurations for virtualization was proposed. While most of these works focused on the conceptualization of sensor-cloud, very few work addressed the technical challenges from the implementation point of view. Misra *et al.* [19] theoretically characterized the aspects of virtualization and justified for a paradigm shift from conventional WSNs through their experimental results. In a very recent work, Chatterjee and Misra [10] explored the issues of target tracking within sensor-cloud and conceptualized the architecture for the virtualization of sensors serving a target tracking application.

Target tracking in WSNs are quite common and explored. A good number of research works [12, 8] have thoroughly investigated target tracking and the performance issues associated with it. Some of the works focused on target localization policies. In [24], Wang *et al.* addressed the problem of posterior target location distribution from the knowledge of the sensor network, thereby maintaining the accuracy in estimation. In [23], the authors have employed a general state evolution model to define the dynamics of the target. The work obtains a reduction in the consumption of resources as well as the precision in localization. Few works [13, 3] have focused on the issues of energy efficiency within sensor networks. A good number of research works also focuses on the aspect of sensor scheduling. Maheswararajah *et al.* [18] proposed a sensor scheduling algorithm (for tracking targets) that minimizes measurement error and sensor usage. In another work, Huber [14] propounded a pruning based sensor scheduling. However, the work schedules a single sensor node at a particular time to reduce the measurement error. The mentioned works on sensor scheduling find their applicability within traditional WSNs. However, the proposed work focuses on a sensor-cloud environment in which the underlying sensor network is subjected to dynamic allocation policies following the Service Level Agreement (SLA). In some cloud-based works [16, 11], the authors proposed a scheduling within WSNs

to optimize the tracking accuracy with the sensor usage. Few works [31, 30] have addressed the implementation of blind scheduling algorithms for multimedia cloud service providers. The works are independent of the demand of the cloud service providers. In [30], the authors focused on scheduling appropriate service providers, whereas our work focuses on the selection of physical sensor nodes. The former is based on a post packet-transmission scenario, whereas the our work concerns the relevant tracking issues (availability of sensors, detection probability of sensors, accuracy of detection, and dwelling time of a target within a sensors coverage) that arises while tracking a mobile target. However, from the perspective of physical sensor scheduling for moving targets, it is necessary to have the knowledge of the availability, accuracy, and the coordinates of the underlying sensors. In [31], the authors of this work have proposed a Blind Online Multimedia Scheduling algorithm (BOSA). The architecture proposed in this work focuses on task division and virtualization aspects within the cloud environment. However, our architecture considers the communication between physical sensor networks and sensor-cloud. In such environment, as targets enter within a sensor deployed zone, multiple sensor nodes in the vicinity of the target are allocated to serve the target. The data from the set of allocated sensors are reported to the cloud end, which in turn, transmit the data to the end-users. The proposed architecture focuses on a QoS aware sensor allocation while tracking a target in a sensor-cloud environment.

In this work, we propose a sensor scheduling and allocation algorithm to be executed within the sensor-cloud environment for serving a mobile target. The work ensures to provide a threshold QoS by scheduling an optimal number of physical sensors. However, for an optimal allocation of nodes, the proposed algorithm Q-SAA, utilizes the benefits of auction theory. There exists lot of literature on application of auction theory for the selection of required resources keeping in mind the usefulness and limitations of certain parameters and finding an optimal solution to the addressed problem [20], [27], [6]. After successful allocation of sensor nodes to targets, the work considers the execution of a standard tracking algorithm, *Probability-based Target Prediction and Sleep Scheduling Protocol* (*PPSS*) [15]. It is to be noted here that, the work explicitly focuses of sensor scheduling and allocation prior to tracking a target. The results of Q-SAA are fed to PPSS for comparison and analysis.

3. System Model

In this work, we consider a scenario where a number of sensors from different sensor service providers are available in a given area. These sensors are used to provide services to the end users through the sensor-cloud infrastructure. We consider that the service of target tracking is being provided by the sensor-cloud and a user wants to track a target using this sensor cloud infrastructure. The overall layered architecture is shown in Fig. 1(a) and for the sake of convenience the notational details are illustrated through Table 1. In this case, we consider the problem of selection of an optimal set of sensors from the available set in the sensor-cloud infrastructure for tracking a single target moving in a two-dimensional field covered by sensors deployed by different owners and form part of sensor-cloud, as shown in Fig. 1(b). When the target moves through the monitored area, it is under the coverage of multiple sensors. The cloud service provider allocates an optimal number of sensors from the set of sensors covering that target. While doing so, we aim to meet the QoS requirements of the user who wants to run his target tracking application through the sensor-cloud. For allocation of the sensors, we consider the following QoS parameters:

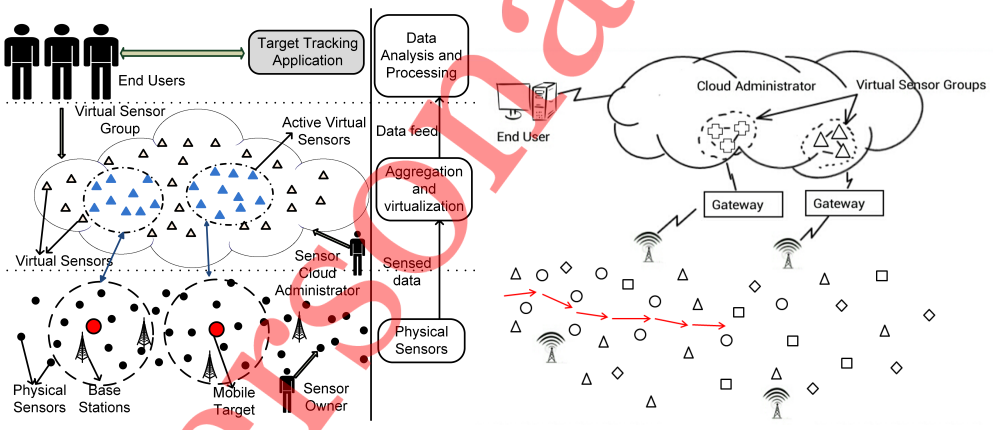
- Availability of the sensors
- Detection probability of the sensors
- Accuracy of detected location
- Dwelling time of a target in a sensor's sensing range

The QoS parameters chosen relate very closely to the target tracking application. The availability of sensors is required for any application and sensors should sustain for the time it is required to provide service. The accuracy in locating a target's position and the probability of detection form crucial factors for achieving higher efficiency in target tracking. The more time a sensor is available for tracking a target, the more beneficial it is to employ that sensor for the application. Each sensor tries to get selected by the cloud service provider to provide the service of tracking so that it can maximize its payoffs.

Initially, the target is required to be detected by n_i sensors (where $n_i \geq 3$) at time t. The position of the target can be determined by finding the point

Table 1: Table of Notation

Parameters	Values
(x, y)	Coordinates of target
(x_i, y_i)	Coordinates of sensor node s_i
r_i	Distance of target from s_i
N_t	Maximal subset of N sensor nodes of a particular target
n_t	Optimal subset of sensor nodes of a particular target
P_{sp}	Probability of detection
P_n	Cumulative probability of detection by n sensors
$P(\lambda_j\|(x, y))$	Conditional probability of noise, given (x, y)
P_{acc}	Probability of accuracy in detection
τ_k	Dwelling time of s_k
β	Available time for a sensor
b_i	Bid of s_i
w_{ij}	Weight associated with j^{th} QoS parameter of s_i
k_{ij}	Price associated with j^{th} QoS parameter of s_i
h_i	Value estimate by s_i
$p_i()$	Probability of servicing a target by s_i
$U_i()$	Utility of s_i
$U_0()$	Utility of auctioneer
$Q_{threshold}$	Measure of QoS to be provided to the end-user



(a) Layered view (b) Network view

Figure 1: Application specific architecture for target tracking in sensor cloud

of intersection of at least three circles formed by taking the distance between the sensors and target as the radius with center at the sensor location.

$$(x - x_1)^2 + (y - y_1)^2 = r_1^2 \quad (1)$$

$$(x - x_2)^2 + (y - y_2)^2 = r_2^2 \quad (2)$$

$$(x - x_3)^2 + (y - y_3)^2 = r_3^2 \tag{3}$$

In Equations (1), (2), and (3), $(x_1, y_1), (x_2, y_2)$ and (x_3, y_3) denote the coordinates of the sensors s_1, s_2, and s_3, respectively, and r_1, r_2, and r_3 are the distances of the target from the sensors s_1, s_2, and s_3, respectively. On solving these equations, we get the coordinates (x, y) of the detected target position at time t, as given below:

$$x = \frac{(y_1 - y_2)X_x - (y_1 - y_3)X_y}{2((x_1 - x_3)(y_1 - y_2) - (x_1 - x_2)(y_1 - y_3))} \tag{4}$$

$$y = \frac{(x_1 - x_2)X_x - (x_1 - x_3)X_y}{2((x_1 - x_2)(y_1 - y_3) - (x_1 - x_3)(y_1 - y_2))} \tag{5}$$

where, $X_x = (x_1^2 - x_3^2) + (y_1^2 - y_3^2) + (r_3^2 - r_1^2)$ and $X_y = (x_1^2 - x_2^2) + (y_1^2 - y_2^2) + (r_2^2 - r_1^2)$. After the target is detected, it is needed to find a set of sensors N_t, at time t, such that,

$$(x - x_j)^2 + (y - y_j)^2 < R_{max_j}^2 \tag{6}$$

where (x, y) and (x_j, y_j) are the coordinates of the detected target position and the known location of the j^{th} sensor respectively, and R_{max_i} is the maximum sensing radius of the j^{th} sensor. Thus, we have, $N_t = \{s_1, s_2.....s_N\}$.

Once the target is detected, it is required to find the next predicted location, so that N_{t+1} can be determined at time instant $t+1$. it is assumed that the present and past positions of the target are known. Let the present location of the target be denoted by (x_i, y_i) at a given time t_i, and the previous location of the target be represented by (x_{i-1}, y_{i-1}) at a given time t_{i-1}. let the next actual location of the target be represented by (x_{i+1}, y_{i+1}) at time $t + 1$. The speed v of the target is computed as:

$$v = \frac{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}}{t_i - t_{i-1}} \tag{7}$$

The direction of motion θ is computed as:

$$\theta = \cos^{-1} \frac{x_i - x_{i-1}}{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}} \tag{8}$$

Therefore, the predicted location of the target at point (x'_{i+1}, y'_{i+1}) is given as follows:

$$\begin{aligned} x'_{i+1} &= x_i + vt\cos\theta \\ y'_{i+1} &= y_i + vt\sin\theta \end{aligned} \tag{9}$$

It is assumed that the prediction of the next location (x'_{i+1}, y'_{i+1}) obeys a two-dimensional standard Gaussian distribution [7] with 0 mean and unit standard deviation. The deviation of the actual trajectory of the target from its predicted path also needs to be considered. Therefore, Equation (9) becomes:

$$\begin{aligned} x'_{i+1} &= x_i + vt\cos\theta \pm \triangle x \\ y'_{i+1} &= y_i + vt\sin\theta \pm \triangle y \end{aligned} \tag{10}$$

It can be clarified that to ensure accuracy in the process of prediction, we follow a two-dimensional standard Gaussian distribution. The authors of the work [7] have clearly discussed how a two-dimensional standard Gaussian distribution helps to preserve the accuracy in prediction. This motivated us to incorporate such a distribution while predicting the next location of the target.

At this predicted location (x'_{i+1}, y'_{i+1}), it is necessary to determine the N_t sensors, which are part of sensor-cloud, and can form virtual sensor group for target tracking. After identification of N_t, the proposed algorithm, Q-SAA, endeavors to identify an optimal set of sensors n_t, where $n_t \subseteq N_t$, which can be utilized to execute the task of target tracking efficiently. This optimal set of sensors n_t is identified on the basis of the metrics listed in the following subsections.

3.1. Probability of Detection

An important parameter for determining the QoS of a sensor in a target tracking application is the probability of detection. Probability of detection is modeled by Aitsaadi *et al.* in [4]. Once a target is in a sensor node's sensing radius, it must be detected by it for efficient tracking. We consider a probabilistic detection model, in which we assume that the detection ability of the sensor increases with the reduction in its distance from the target. There are two sensing ranges defined, R_1 is the range within which the detection probability is considered to be maximum (or equal to 1), and thereafter, it starts decreasing. Finally, it becomes zero after reaching the maximum sensing range R_{max}. The detection probability depends on the distance between the sensor location and the target.

Definition 3.1. If sp is the Euclidean distance between a sensor point s and a predicted target location p, and a and b are the constants related to sensor characteristics, then the probability of detection, P_{sp}, for a particular s and

p is defined is a function of the Euclidean distance between the point s and p [4]. Thus,

$$P_{sp} = \begin{cases} 1 & 0 \leq sp \leq R_1 \\ \frac{a}{sp^b} & R_1 < sp \leq R_{max} \\ 0 & R_{max} < sp \end{cases} \tag{11}$$

where R_1 is the range of the sensor in which the detection probability is 1. Beyond range R_{max}, the detection probability drops to zero.

When an area is sensed by a number of sensors, it is required to calculate the cumulative effect of those sensors for the detection of target. Therefore, the overall detection probability of all the sensor nodes that form part of N_t is defined as:

$$P_n = 1 - \prod_{j=0}^{N_t} (1 - P_{s_j p}) \tag{12}$$

3.2. Accuracy

Each sensor, on sensing the target, has some amount of error in observation. An error measurement model for WSN is given in [25]. Most Extended Kalman Filter (EKF) algorithms consider additive noise only, thereby leading to unstable tracking performances. Such filters are applicable generally to static targets served by a fixed set of sensor nodes. However, in our work we have considered a mobile target and a sensor-cloud environment the target is served by a set of sensors that are dynamically scheduled and allocated. In [25], the problem of non-linearity has been addressed and the work considers both additive and multiplicative noise. The actual distance between sensor j and the target is given as r_j, where:

$$r_j = \sqrt{(x - x_j)^2 + (y - y_j)^2} \tag{13}$$

In Equation (13), (x_j, y_j) are the coordinates of the location of the j^{th} sensor, and (x, y) are the coordinates of the actual position of the target. Let λ_j be the distance actually measured by the j^{th} sensor at time t. The measurement model uses additive and multiplicative noises, and is represented as given below [25].

$$\lambda_j = (1 + \kappa_j) r_j + \pi_j = r_j + u_j \tag{14}$$

where π_j and κ_j are the additive and multiplicative Gaussian noises of sensor j. The conditional probability density function for λ_j, given (x, y), is given as follows

$$p(\lambda_j|(x,y)) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(\lambda_j - r_j - \mu_j)^2}{2\sigma_j^2}} = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(r_j - (\lambda_j - \mu_j))^2}{2\sigma_j^2}} \tag{15}$$

The above equation can be utilized to find the probability of error in the process of sensing by sensor node j.

Definition 3.2. The probability of accuracy of a sensor s_j is denoted by P_{acc} and is defined as the probability that there is no sensing error in estimating the distance of the target positioned at (x, y). P_{acc} is mathematically expressed as follows:

$$P_{acc} = 1 - p(\lambda_j|(x,y)) \tag{16}$$

3.3. Dwelling Time

The parameter, dwelling time measures the time a target is likely to remain in an area covered by the sensing range of the node. This parameter enables the prediction of the time a sensor node has to serve the target after it is selected.

Definition 3.3. The dwelling time τ_k for a target at a predicted position (x'_{i+1}, y'_{i+1}) with respect to a sensor s_k at position (x_k, y_k) is defined as the time the target takes to traverse a path formed by extending a straight line joining the present location (x_i, y_i) and the next predicted position (x'_{i+1}, y'_{i+1}) to the point where it intersects the sensing circle of sensor s_k in the direction of motion.

Theorem 3.1. *The dwelling time τ_k with respect to a sensor s_k is given by:*

$$\tau_k = \frac{\sqrt{(x'_{i+1} - x)^2 + (y'_{i+1} - y)^2}}{v} \tag{17}$$

Proof. Fig. 2 shows the next predicted position (x'_{i+1}, y'_{i+1}) found with the help of the present position (x_i, y_i) and previous positions (x_{i-1}, y_{i-1}). We assume a straight line motion for the target on the line connecting the coordinates (x_i, y_i) and (x'_{i+1}, y'_{i+1}) and extended to intersect the periphery of

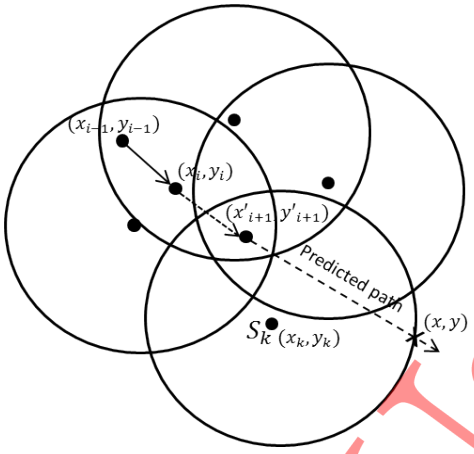


Figure 2: Calculation of dwelling time

the sensing circle at point (x, y). The line joining (x'_{i+1}, y'_{i+1}) and (x, y) gives the distance a target covers in the sensing area of sensor s_k. The equation of the sensing circle of the k^{th} sensor is given by:

$$(x - x_k)^2 + (y - y_k)^2 = r_k^2 \tag{18}$$

The equation of the line joining the present (x, y) and predicted locations (x'_{i+1}, y'_{i+1}) are given by:

$$\frac{y - y'_{i+1}}{x - x'_{i+1}} = \frac{vt\cos\theta + \triangle x}{vt\sin\theta + \triangle y} \tag{19}$$

Solving the above, we get,

$$y = (x - x'_{i+1})\frac{vt\cos\theta + \triangle x}{vt\sin\theta + \triangle y} + y_{i+1} \tag{20}$$

$$x = (y - y'_{i+1})\frac{vt\sin\theta + \triangle y}{vt\cos\theta + \triangle x} + x_{i+1} \tag{21}$$

Putting the value of y in Equation (21) to Equation (18), we get

$$\begin{aligned}(1 + \triangle p^2)x^2 - 2x(x_k + x'_{i+1}\triangle p^2 + \triangle p y_k) + \\ x_k^2 + y_k^2 + x'^2_{i+1}\triangle p^2 + 2x'_{x+1}y_k\triangle p - r^2 = 0\end{aligned} \tag{22}$$

where $\triangle p = \frac{vt\cos\theta+\triangle x}{vt\sin\theta+\triangle y} + y'_{i+1}$. The above equation is of the quadratic form and can be solved to get the value of coordinates x where the predicted path intersects with the circle on $x-$axis.Similarly we get the values of y where the predicted path intersects with the circle on $y-$axis. Therefore, the distance d the target travels on the predicted path with the sensing circle of sensor s_k is given by:

$$d = \sqrt{(x'_{i+1} - x)^2 + (y'_{i+1} - y)^2} \tag{23}$$

Hence, the dwelling time τ_k for a target in the sensing range of sensor s_k is formulated as given below

$$\tau_k = \frac{\sqrt{(x'_{i+1} - x)^2 + (y'_{i+1} - y)^2}}{v} \tag{24}$$

This concludes the proof. □

3.4. Availability of Sensor

The availability of the sensor is calculated on the basis of the residual energy in the sensor.

Definition 3.4. If ψ is the battery consumption rate for transmitting, receiving, and sensing combined together, a sensor s_k is said to be available to sense a target at the next predicted position (x'_{i+1}, y'_{i+1}), if it has residual energy E_r, which is sufficient to sense the target without interruption for the dwelling time τ_k.

$$\beta = \frac{E_r}{\psi} \tag{25}$$

where β is the time for which the sensor is available, and E_r is the residual energy of the sensor node.

Therefore, for a sensor s_k to be available for sensing the target throughout the time it is in its sensing area, the condition $\beta_k > \tau_k$ must be satisfied. The sensor nodes which do not meet this criteria are eliminated from the previously selected set of sensors N_t. It can be clarified that from the implementation perspective, before the execution of Q-SAA, we assume every node to possess 100% battery level. For each operation (sensing, communication, or computation), the node is assumed to consume variable amount of energy which are considered to be the standard values for sensing (10 nJ/event), communication (20 nJ/bit), or computation (7 nJ/bit). Based on the battery consumption rate, the availability of a sensor is calculated.

4. Auction-based selection of sensors

We formulate an auction-based mechanism for allocation of sensors, i.e., allocating n_t sensors such that $n_t \subset N_t$, using the QoS parameters discussed in Section III. The aim for this auction is to ensure a balance between achievable or desired QoS and the cost incurred by the user. An auction is based on buying and selling of products on the basis of bids proposed by potential bidders. This work is based on the *direct revelation auction mechanism* [20].

In the auction mechanism, user C is the auctioneer and there are $N_t = \{1, 2, 3, ...n\}$ sensors as the bidders in the auction. N_t sensors place their bids, b_i, on the basis of the evaluation of cost they would incur for providing the service, as follows:

$$b_i = w_{i1}k_{i1}P_{sp_i} + w_{i2}k_{i2}P_{acc_i} + w_{i3}k_{i3}\tau_i \tag{26}$$

where w_{ij} and k_{ij} are the weights and prices associated with every QoS parameter, respectively. P_{sp_i}, P_{acc_i} and τ_i are the values of probability of detection, probability of accuracy and the dwelling time of a sensor i, respectively. Let h_i be the value estimate of the bidder i, which he/she is going to reveal to all the other bidders. A continuous probability distribution over a finite interval gives the users an estimate of bidder i. Let ϕ_i represent the possible range of values which i might assign to the object. ϕ for a particular bidder can be estimated by knowing the previous value ranges in the previous auctions. Let the set of all possible combinations of bidders values estimates be denoted by H. We have

$$H = [\phi_1] \times [\phi_2] \times ... \times [\phi_n] \tag{27}$$

To find all possible assessment values by all the bidders except i, we remove the i^{th} estimate from H to get H_{-i}.

$$H_{-i} = [\phi_1] \times [\phi_2] \times ...[\phi_{i-1}] \times [\phi_{i+1}] \times ... \times [\phi_n] \tag{28}$$

The joint density function on H for the vector $h = (h_1, ..., h_n)$ of individual value estimates is:

$$f(h) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(h-\mu_j)^2}{2\sigma_j^2}} \tag{29}$$

Bidder i's his value estimate is a known quantity. Both the user and the bidder i assess the joint density function on H_i for the vector $h_{-i} = (h_1, ., h_{i-1},$

$h_{i+1}, ..., h_n)$ of values for all bidders other than i to be as follows:

$$f_{-i}(h_{-i}) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(h_{-i}-\mu_j)^2}{2\sigma_j^2}} \tag{30}$$

All the value estimates are made available to the bidders. On revelation of these estimated values for providing a service to the user, bidder i compares his/her evaluation with the others. Therefore, bidder i may reassess his/her own evaluation and change the own value of providing service by a factor $K_i(h_j - h_i)$, where K_i is an experimental constant. Thus, if bidder i has the value estimates initially held by the n bidders $h = (h_1, ..., h_n)$ available to him, then i revises his/her own evaluation of providing the service to:

$$v_i(h) = h_i + \sum_{j \in N_t} K_i(h_j - h_i) \quad where \quad j \neq i \tag{31}$$

Similarly, user may also reassess his estimated value on the basis of bidders evaluation, as follows:

$$v_0(h) = h_0 + \sum_{j \in N_t} K_i(h_j - h_0) \tag{32}$$

The probability that a user may get a chance to provide the service to a target can be derived, on the basis of the dwelling time of the target in a sensors coverage area, and can be described as:

$$p_i(h) = \frac{\epsilon\tau_i}{\sum_{i=1}^{N} \xi\tau_i} \tag{33}$$

where ϵ and ξ are constants.

In the direct revelation auction mechanism, the bidders declare their value estimates to the auctioneer secretively and concurrently. Based on these evaluations submitted by the bidders the auctioneer decides which bidder wins the auction and what he/she has to pay. Thus, the utility of a direct revelation auction mechanism is given by two outcome functions (p, b) such that, if h is the vector of value estimates declared by the bidders, $p_i(h)$ is the probability that i services the target, and $b_i(h)$ is the expected cost which bidder i incurs in providing this service to the user. Thus, the expected

utility from direct revelation auction mechanism, as given in [20], described by (p, b) for bidder i is given by:

$$U_i(p, b, h_i) = \int_{H_{-i}} (v_i(h)p_i(h) - b_i(h))f_{-i}(h_{-i})dh_{-i} \tag{34}$$

where $dh_{-i} = dh_1, ..., dh_{i-1}, dh_{i+1}, ..., dh_n$.
Similarly, the expected utility for the auctioneer from this auction mechanism is:

$$U_0(p, b) = \int_{H} (v_0(h)(1 - \sum_{j \in N_t} p_j(h)) + \sum_{j \in N_t} b_j(h))f(h)dh \tag{35}$$

where $dh = dh_1, ..., dh_n$.

Algorithm 1 QoS-Aware Sensor Allocation Algorithm (Q-SAA)

Input:

- Present location of target (x_i, y_i) at time t_i.
- Past location of the target (x_{i-1}, y_{i-1}) at time t_{i-1}.

Output: Selected subset of sensors n_t at time t.

Step 1: Compute the next predicted target position (x'_{i+1}, y'_{i+1})
Step 2: Select N_t sensors
Step 3: Compute $\beta, \tau, P_{sp}, P_{acc}$ for all $n_t \in N_t$
Step 4: Compute U_i for all $n_t \in N_t$
Step 5: Arrange N_t in the ascending order of their U_i
Step 6: Select $n_t \subset N_t | (Q_n \geq Q_{threshold}) \vee (n_t \geq \chi \cdot N_t)$
Step 7: t_i=t_{i+1}
Step 8: $x_{i-1} = x_i$, $y_{i-1} = y_i$
Step 9: $x_i = x_{i+1}$, $y_i = y_{i+1}$
Step 10: Goto Step 1

In this case, it is needed to select multiple bidders as winners in the ascending order of their utilities U_i such that it meets the desired QoS criteria of accuracy and detection of the user. With the help of this negotiation, the

user gets the desired QoS, and at the same time has to pay the least possible cost. The aim of the bidder i is to acquire the opportunity to serve the user so as to make profit from the cost of usage and also to provide the desired QoS to the user.

$Q_{threshold}$ is the measure of QoS required to be delivered to the user. Therefore, the task is to select n_t sensors (where $n_t \subset N_t$) so that we may get the desired quality of service.

Definition 4.1. The threshold QoS, $Q_{threshold}$, is defined as the weighted mean of the probability of detection, P_{sp}, and the probability of accuracy, P_{acc}, as desired by the end user.

$$Q_{threshold} = \frac{w_1' P_{sp(n)} + w_2' P_{acc(n)}}{2} \tag{36}$$

where w_1' and w_2' are the weights of the cumulative probabilities of detection and accuracy desired by the user.

Definition 4.2. An optimal set of sensors n_t at a time instant t is defined as $n_t \subset N_t$ such that the cumulative QoS, Q_n, provided by the first n sensors is arranged in their increasing order of utility U_i.

U_i is higher than the threshold QoS $Q_{threshold}$ and the number of sensors n is greater than a minimum predefined percentage of sensors.

Therefore, it can be inferred that the problem is reducible to selecting n_t sensors with lowest utilities U_i, such that:

$$Q_n \geq Q_{threshold} \quad and \; n \geq \chi \cdot N_t \tag{37}$$

where Q_n is the set of cumulative values of probability of detection and the probability of accuracy for the first n_t sensors, χ is a predefined percentage of sensors which which should be employed for tracking the target out of the total available sensors at that point. This forms the subset of n_t sensors that meets the requirements of the user. The proposed steps of execution are presented in Algorithm 1. The proposed algorithm, Q-SAA, requires an input of the present (x_i, y_i), and previous (x_{i-1}, y_{i-1}) target positions to start localizing of all sensors that are available to the sensor-cloud in that region. On the basis of the location information of the target and the sensing radius of each node, a set of N_t sensors is formed for tracking the target. All the four parameters are evaluated only for this set of N_t sensors and further

filtering is performed on the basis of availability of sensors. Availability is measured in terms of battery life that is sufficient to give lifetime for a sensor more than the dwelling time of the target in that particular sensor's coverage area. After omitting the sensors which are unavailable, from N_t, all sensors evaluate their cost incurred for providing tracking service and place their bids. Bids can be based on the assessment of each sensor and the weightage it gives to all three parameters. In case a target is likely to have a longer dwelling time in a sensors area of coverage, it will be beneficial for the user to choose such a sensor, as it may not have to disengage the sensor for long time, thereby, reducing the overheads for forming a virtual sensor, which can generate revenue by providing service for a longer time. Therefore, it may be inferred that the weightage of dwelling time can be higher as compared to the other two parameters for better results from the algorithm. On the basis of bids, the utility for each sensor is computed for the first n_t sensors with highest utility, which satisfies the QoS requirement for the user. This optimizes resource allocation in the sensor-cloud. This process is repeated at the next step if the chosen set of sensors in the last position cannot meet the QoS requirements of the user. Thus, a new group of sensors is formed, otherwise, we continue with the same set of sensors.

Theorem 4.1. *There exists a Nash Equilibrium (NE) for the bid of every sensor of the maximal subset N_t.*

Proof. Wang *et al.* [26] proved the existence of NE in an auctioned system. Wang *et al.* and Rosen [21] characterized the existence of NE for a negative second order derivative of the utility function. From Equation (35), for every $s_i \in N_t$, we obtain,

$$\frac{\delta^2 U_i(h)}{\delta h^2} = \Bigg(v_i'(h) - v_i'(h)\sum_{j\in N_t} p_j(h) - v_i(h)\sum_{j\in N_t} p_j'(h) + \sum_{j\in N_t} b_j'(h)\Bigg) f(h) + f'(h)\Bigg(v_i(h)(1-\sum_{j\in N_t} p_j(h)) + \sum_{j\in N_t} b_j(h)\Bigg) \tag{38}$$

Now, we see that,

$$f'(h) = c_2 h_i e^{\frac{-(h_i-\mu_j)^2}{2\sigma^2}}, v_i'(h) = 1 - K_i|N_t| + K_i \tag{39}$$

c_2 being a negative constant. From Equation (39) and the values of K_i, we observe that $f'(h)$ and $v_i'(h)$ are negative quantities. From this , we can directly infer that $\frac{\delta^2 U_i(h)}{\delta h^2} < 0$, as $f(h) > 0$. This concludes the proof. □

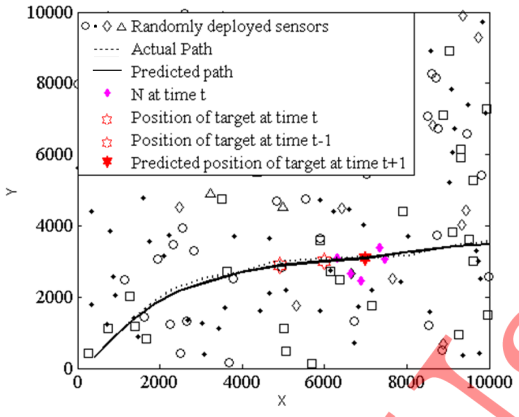
5. Performance Evaluation

In this Section, we discuss and analyze the performance of the proposed system and the algorithm under several categories as follows:

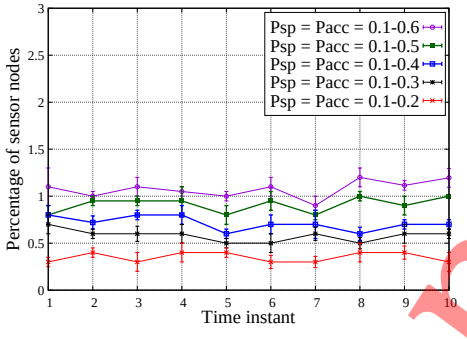
5.1. Scheduling of Sensor Nodes

The simulation setup considering a 10,000 X 10,000 units 2-D terrain with 1000 sensors randomly deployed, is depicted in Fig. 3(a). The dotted line depicts the trajectory of the target over the actual path with correction and the solid line the trajectory of the target on the predicted path. Rhombus markers represent the set of N_t sensors that satisfy the condition $(x - x_j)^2 + (y - y_j)^2 < R^2_{max_j}$. These sensors have the target within their sensing range at a given time instant for a target position. The depicted rhombuses correspond to the third position of the target in the figure. Simulation experiments were executed by changing different parameters over a period of time and also varying the weights associated with them to see their effect on the selection of sensors.

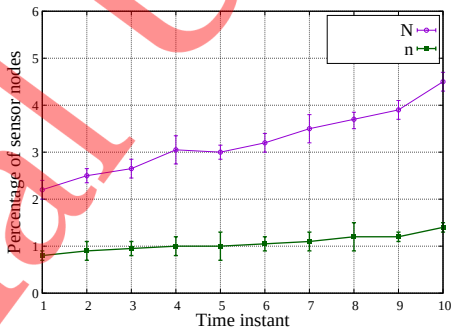
Fig. 3(b) shows the variation in the percentage of sensors in subset n_t, which is undertaken by taking the range of values for $P_{sp} = P_{acc} = (0.1 - 0.6)$, $P_{sp} = P_{acc} = (0.1 - 0.5)$, $P_{sp} = P_{acc} = (0.1 - 0.4)$, $P_{sp} = P_{acc} = (0.1 - 0.3)$ and $P_{sp} = P_{acc} = (0.1 - 0.2)$ for allocating a subset of sensors. R_{max_j} si randomly assigned within the interval $[150m, 200m]$. $P_{sp} = P_{acc} = (0.1 - 0.6)$ implies that the value of P_{sp} and P_{acc} for all sensors which are part of N_t is randomly selected within the range of 0.1 to 0.6 for the first set of experiments, i.e., the maximum value of P_{sp} and P_{acc} for a sensor is 0.6, and similarly, for the other set of experiments, the maximum value of the two parameters is $0.5, 0.4, 0.3$, and 0.2. Let us consider the percentage of sensors in the subset n_t at the 3^{rd} time instant. The percentage of allocated sensors is 0.2, 0.6, 0.8, 0.9, or 1.1 for different decreasing values in the range of P_{sp} and P_{acc}. It can be seen that on decreasing the range of parameters P_{sp} and P_{acc}, the required percentage of sensors increases. The subset with square markers which has a range of values for $P_{sp} = P_{acc} = (0.1 - 0.6)$ requires 4-5 sensors to fulfill the requirement of the user. However, the subset with star markers ($P_{sp} = P_{acc} = (0.1 - 0.2)$) requires 11-12 sensors to meet the



(a) Simulation of 2D area with predicted path.



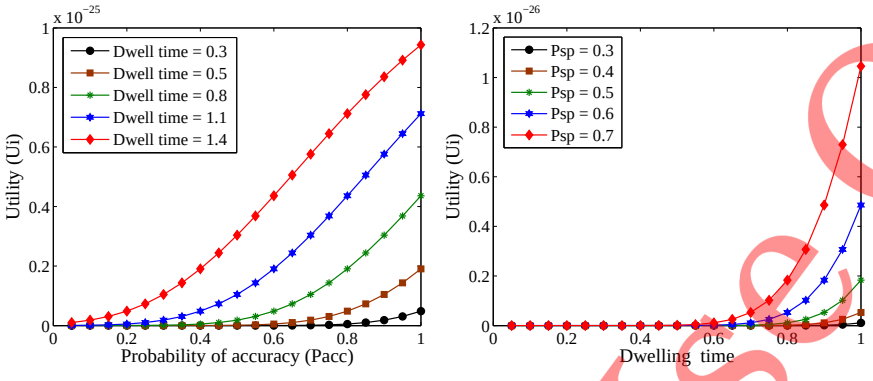
(b) Comparison on different ranges of P_{sp} and P_{ac}.



(c) Comparison between total and allocated sensors

Figure 3: Comparative study by changing system parameters

QoS parameters of the user. This is because the cumulative probability of accuracy and detection requires more number of sensors to reach $Q_{threshold}$ due to the smaller individual values of P_{sp} and P_{acc} for each sensor. Fig. 3(c) shows a comparison between the percentage of sensors allocated (n_t) to the percentage of sensors available (N_t) in that area that can engage the target at the given time instant. In this experiment we allocate $\chi\%$ of the available sensors as per their utility and meet the $Q_{threshold}$ of the user. If the selected $\chi\%$ of sensors does not satisfy the $Q_{threshold}$, then in such a case more than $\chi\%$ of sensors are allocated till the $Q_{threshold}$ is satisfied. In this experiment, we take χ as 30%. It can be seen from the graph that with the use of this algorithm, the requirement of sensors varies with the available set of sensors



(a) Ui vs Probability of Accuracy

(b) Ui vs Dwelling time

(c) Ui vs Detection Probability

Figure 4: Variation of U_i with change in different parameters

at a given point. Here, we can also infer that by using this approach, we can find an optimal number of sensors that can be allocated to utilize the resources in the sensor-cloud, thereby saving lot of sensor resources.

5.2. Utility Behavior

We also analyzed the behavior of the value of utility for a sensor by varying different parameters. Some of the results are explained as follows. The weights for these experiments are kept the same for each of the parameters. Fig. 4(a) shows the graph plotted between the probability of accuracy and the utility for different values of dwelling time, while keeping the detection probability as constant. As we see, the graph shows an upward trend with the increase in the values of dwelling time.

In Fig. 4(b) a plot between dwelling time and utility is plotted. The graph is plotted for different values of detection probability, while keeping the probability of accurate detection constant. The behavior shows a lower utility value for most of the graph, followed by which there is a steep rise in utility specially on higher values of detection probability. In Fig. 4(c) a plot between the probability of detection and utility is plotted. The weightage of all the three parameters is kept the same, the graph is plotted for different values of probability of accuracy, while keeping the dwelling time as constant. The curve shows that initially there is very little effect on the value of utility, but after a certain point, the value of utility takes a linear rise. The experimental values for the above mentioned plot are given out in Table 2 for better understanding. The results show how the three parameters affect the utility U_i associated with different sensors and a user can get an idea how to set his/her preferences for obtaining the desired quality of service. For example in the third case, where the dwelling time is kept constant, a user for lower values of P_{acc} and P_{sp} will get lower utility sensors, i.e., these sensors are more beneficial to the user in monetary terms but more number of sensors will be required for achieving higher QoS. Also, the weights associated with the parameters will have a bearing on the behavior of utility.

Table 2: Simulation Parameters

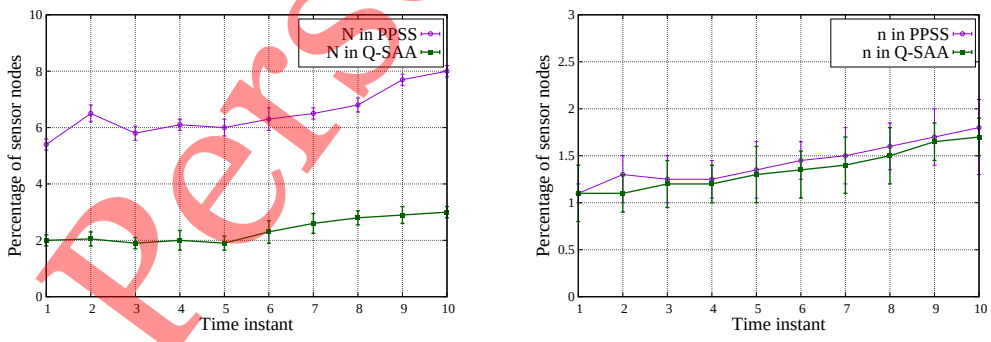
S.No.	P_{sp}	P_{acc}	τ_i	$U_i(\times 10^{-27})$
1	0.6	0.5	1	4.858
2	0.6	0.6	1	10.46
3	0.65	0.5	1	7.295
4	0.65	0.6	1	14.376
5	0.7	0.5	1	10.45
6	0.7	0.6	1	19.046

5.3. Comparison with PPSS

To verify the efficiency of Q-SAA, we compared it with an existing sensor management algorithm for target tracking application. The algorithm we chose to compare is the *Probability-based Target Prediction and Sleep Scheduling Protocol (PPSS)* [15]. The methodology of PPSS concerns the duty cycling and activation of a set of physical sensor nodes, followed by the

prediction of the target trajectory. PPSS focuses on duty cycling of sensor nodes within the vicinity of the target, thereby reducing the consumption of resources. In our work, we propose Q-SAA for a QoS aware sensor scheduling and activation. Q-SAA not only activates sensor nodes within the vicinity of the predicted target location, but also focuses on the QoS of the provisioned service, unlike PPSS. To investigate the difference in the objectives of PPSS and Q-SAA, and the consequent effects in their performance, we choose PPSS as the benchmark. Also, as the authors have also executed PPSS in TelosB motes [1] and TinyOS [2], PPSS finds its credibility even from an implementation point of view. This is also one of the primary reasons because of which we follow the tracking algorithm of PPSS (during performance evaluation), after executing Q-SAA for sensor scheduling and allocation. For simulation of PPSS, we have used the experimental setup as indicated by the authors of the work. For the simulation of utility of Q-SAA, the simulation settings are illustrated in subsections 5.1 and 5.2.

Fig. 5(a) shows the comparison of selection of a superset of sensors (N_t) by both the algorithms. We see that in Q-SAA, the percentage of sensors in N_t is approximately one-third the percentage of sensors in N_t in case of PPSS, thereby saving lot of computational overhead in selecting n_t from N_t, because all sensors in N_t need to be analyzed for selecting the best sensors. In Fig. 5(b), we show the comparison of n_t, i.e., the allocated or active sensors from both the algorithms. Results show that the percentage of sensors selected to be part of n_t in Q-SAA is lesser than that in PPSS. Therefore, Q-SAA gives an optimal set of sensors and saves on the sensor resource.



(a) Comparison of N for PPSS and Q-SAA (b) Comparison of n for PPSS and Q-SAA

Figure 5: Comparison of PPSS and Q-SAA

5.4. Economics of the Model

The very basic idea that motivates the use of sensor-cloud is the economy of scale. There are common resources that are utilized by different users through online connectivity thereby reducing the location dependence. The users get on-demand scalable and elastic resources and they are priced as usage-sensitive or pay-per-use model. The economics of using a sensor-cloud for target tracking is justified as follows. Firstly, by using Q-SAA, we reduce the the number of sensors actually required by a traditional algorithm. Secondly, the cost of ownership of a WSN vis-a-vis the pay-as-per-usage for a set of sensors from the sensor-cloud works out to be cheaper, specially on a longer run. The cost of ownership of a sensor network accounts for the cost of investment of setting up a privately owned WSN. On the other hand, in a sensor-cloud, the cost of investment is zero. However, the user has to pay as per his/her usage. The experimental setup is indicated in Table 3.

Table 3: Experimental setup

Parameters	**Values**
Time period (T)	3×10^4 units
Number of sensor nodes ($n_s = n_{sc}$)	1000
Unit cost price of a node (C_r)	Rs. 20/unit
Unit cost due to maintenance in WSN (C_{m1})	Rs. 20/unit
Unit cost due to maintenance in WSN (C_{m2})	Rs. 10/unit
η_1	0.85
η_2	0.85

A comparison between cost of ownership and cloud usage cost [22] shows that the cost of ownership evaluated for 3 years is approximately 1.58 times the cost of usage of cloud infrastructure. If the cost of buying a sensor is C_s the cost of buying n sensors to form a WSN is $n_s \times C_s$. However, in case of a sensor cloud cost, the of ownership is negligible. But, we need to pay for the time we have used a sensor, i.e., the cost is incurred on the basis of rate based on per unit time per sensor, given by C_r, such that $C_s >> C_r$. The cost of ownership further includes additional costs such as maintenance cost and cost of supporting infrastructure required for setting up a WSN. We term all these costs as C_m. On the contrary, in case of a user of sensor-cloud, these costs do not exist or they are negligible. Another factor that comes into play is the efficiency η, which is defined as the amount of resources utilized at a given time. It can be understood that a lot of resources of a privately-owned WSN

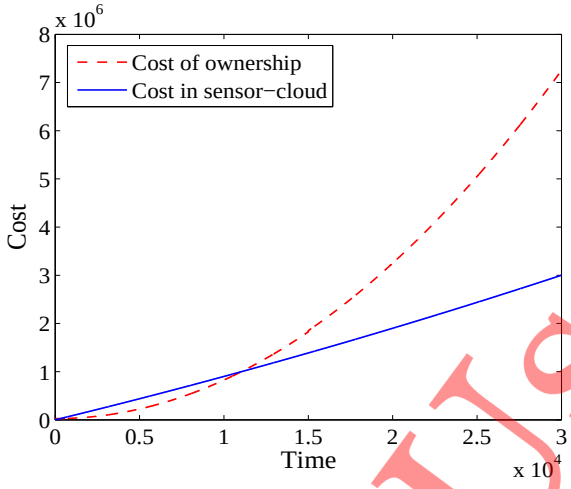


Figure 6: Cost comparison of WSN vs Sensor-Cloud.

are under-utilized because of the fact that once a target is being tracked, all the sensors are not used at any time instant and there may not be target to be tracked throughout the life-cycle of a the WSN. On the contrary, in sensor-cloud, the resources have higher utilization due to elastic and scalable nature of resources. Therefore, the efficiency of a WSN is low, as compared to a sensor-cloud. Also, in a sensor-cloud, there is more flexibility, scalability, and reduced chances of failure. Therefore, the total cost of ownership C of a WSN is calculated as follows:

$$C = \frac{n_s \times C_s}{\eta_1} + \sum_{t=0}^{T} C_{m1} \tag{40}$$

where T is the time of usage, n_s is the number of sensors deployed, η_1 is the efficiency and C_{m1} is the cost of maintenance per unit time for privately-owned WSN. The cost of using the sensor resource as a part of sensor-cloud C_{sc} is given by the number of sensors used at a given rate for a given time duration. Hence, C_{sc} is represented as;

$$C_{sc} = \sum_{t=0}^{T} (\frac{n_{sc} \times C_r}{\eta_2} + C_{m2}) \tag{41}$$

where T is the time of usage, n_{sc} is the number of sensors allocated in the sensor-cloud, η_2 is the efficiency, and C_{m2} is the cost of maintenance per unit

time for a sensor-cloud. Fig. 6 shows a graph of comparison between cost of ownership and the cost of using the sensor-cloud. The graph compares the cost of usage for using sensor-cloud vis-a-vis a privately owned WSN for approximately 30,000 hours, which corresponds to three years and four months of continuous usage. It is worth noting that the graph is plotted keeping the number of sensors the same for both WSN and sensor-cloud. Using Q-SAA we further reduce the number of sensors required and widen this gap between the two lines of sensor-cloud and private WSN. We can be intuited that this gap will increase further if the usage is not continuous. This is because the user pays as per usage in the sensor-cloud. However, the maintenance in private WSN will be undertaken irrespective of the level of usage. Also, it can be predicted that after every 5-6 years, there will arise a requirement of major upgradation in the infrastructure of the privately owned WSN, whose cost may be assumed to be similar to the initial setup of the infrastructure. The above mentioned reasons further justify the rationale for using a sensor-cloud. Hence, we observe that by using sensor-cloud we reduce the cost of usage, firstly, by cutting down on initial purchase of infrastructure, secondly, by getting over the maintenance and upgradation costs and effort required for the same and thirdly, just paying for what the user uses.

5.5. Complexity Analysis of Q-SAA

In this subsection, we discuss and analyze the runtime complexity analysis of Q-SAA as presented below.

Lemma 5.1. *The worst case asymptotic computational complexity for evaluation of the cumulative detection probability is* $O(\mid N_t \mid^2))$, N_t *being the maximal subset of physical sensor nodes for tracking a target.*

Proof. Let us assume that $T'(k)$ is the computational complexity for obtaining the cumulative detection probability involving k sensor nodes, such that, $\mid N_t \mid = k$. From the cumulative probability of detection, as shown in Equation (12), we obtain,

$$T'(k) = T'(k-1) + c', \quad T'(1) = O(1) \tag{42}$$

c being a constant. Therefore, $T'(k) = O(k^2)$ which implies that $T'(\mid N_t \mid) = O(\mid N_t \mid^2)$. This completes the proof. □

Theorem 5.1. *The worst case asymptotic computational complexity of Q-SAA involving* $\mid N_t \mid$ *number of sensors in the maximal subset is* $T(\mid N_t \mid) \simeq O(\mid N_t \mid^2)$.

Proof. We assume $T(k)$ as the computational complexity of Q-SAA in which $\mid N_t \mid = k$. From a step by step analysis of Q-SAA, as illustrated in Algorithm 1, and using the results of Lemma 5.1, the recursive equation for analysis of computational complexity can be derived as,

$$T(k) = c_1 O(k) + c_2 O(k^2) + c_3(T'(k-1) + c') + c_4, \quad T(1) = c \tag{43}$$

Therefore, we infer, $T(k) = O(k^2)$ which implies $T(\mid N_t \mid) \simeq O(\mid N_t \mid^2)$. □

6. Conclusion

In this paper, an auction-based scheme for autonomous allocation of sensors to a particular target through sensor-cloud service provider was formalized. The sensor-cloud architecture is able to retrieve and process sensor data in a cost-effective, timely, and easily accessible manner. In other words, due to visualization in sensor-cloud, a particular sensor becomes usable to multiple end-users and its employability becomes application independent. We specifically addressed the problem of resource allocation in a target tracking scenario and utilized the resources of multiple sensor network providers for achieving the aim while being agnostic about the physical locations of the nodes. It can be seen from the results that this algorithm enables the sensor-cloud service provider to autonomously allocate the optimal number of sensors based on QoS parameters to achieve the desired efficiency. The selection is based on direct revelation auction mechanism, in which all the bidders reassess their evaluation of the object based on the evaluation of other bidders before placing a bid. This auction mechanism helps the user to get a better value of the service being offered to him. We evaluated the results to find the effect of quality of service parameters on the utility in auction process and effect on of selection of optimal number of sensors.

In the future, we plan to consider scenarios involving multiple targets in a sensor-cloud environment. This improvement will further enhance the usability of sensor-cloud for more applications and their concurrent use by a number of users. Also, the sensor selection procedure may be made more efficient by incorporating application dependent QoS.

Acknowledgments

This work was partially supported by a fellowship sponsored by the Tata Consultancy Services (TCS), India.

References

[1] Telosb data sheet, http://www.willow.co.uk, 2012.

[2] Tinyos, http://www.tinyos.net, 2012.

[3] S. Aeron, V. Saligrama, and D. A. Castanon. Efficient sensor management policies for distributed target tracking in multihop sensor networks. *IEEE Transactions on Signal Processing*, 56(6):2562–2574, June 2008.

[4] N. Aitsaadi, N. Achir, K. Boussetta, and G. Pujolle. Heuristic deployment to achieve both differentiated detection and connectivity in WSN. In *IEEE Vehicular Technology Conference (VTC)*, pages 123 – 127, Singapore, 11-14 May 2008.

[5] A. Alamri, W. S. Ansari, M. M. Hassan, M. S. Hossain, A. Alelaiwi, and M. A. Hossain. A survey on sensor-cloud: Architecture, applications, and approaches. *International Journal of Distributed Sensor Networks*, 2013, November 2013.

[6] M. W. Baidas and A. B. MacKenzie. Auction-based power allocation for multi-source multi-relay cooperative wireless networks. In *IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1 – 6, Houston, TX, USA, 5-9 Dec. 2011.

[7] M. Z. A. Bhuiyan, G.-J. Wang, Z. Li, and P. Yong. Prediction-based energy-efficient target tracking protocol in wireless sensor networks. *Journal of Central South University of Technology, Springer*, 17(2):340–348, Apr. 2010.

[8] W. Blair and T. Bar-Shalom. Tracking maneuvering targets with multiple sensors: does more data always mean better estimates? *IEEE Transactions on Aerospace and Electronic Systems*, 32(1):450–456, Jan 1996.

[9] S. Chatterjee and S. Misra. Dynamic and adaptive data caching mechanism for virtualization within sensor-cloud. In *IEEE ANTS 2014*, December 2014.

[10] S. Chatterjee and S. Misra. Target tracking using sensor-cloud: Sensor-target mapping in presence of overlapping coverage. *IEEE Communications Letters*, PP(99):1–1, 2014.

[11] A. Chhetri, D. Morrell, and A. Papandreou-Suppappola. On the use of binary programming for sensor scheduling. *IEEE Transactions on Signal Processing*, 55(6):2826–2839, June 2007.

[12] K. Dogancay. Bias compensation for the bearings-only pseudolinear target track estimator. *IEEE Transactions on Signal Processing*, 54(1):59–68, Jan 2006.

[13] Y. Hamouda and C. Phillips. Adaptive sampling for energy-efficient collaborative multi-target tracking in wireless sensor networks. *IET Wireless Sensor Systems*, 1(1):15–25, March 2011.

[14] M. Huber. Optimal pruning for multi-step sensor scheduling. *IEEE Transactions on Automatic Control*, 57(5):1338–1343, May 2012.

[15] B. Jiang, B. Ravindran, and H. Cho. Probability-based prediction and sleep scheduling for energy-efficient target tracking in sensor networks. *IEEE Transactions on Mobile Computing*, 12(4):735 – 747, April 2013.

[16] J. Lin, W. Xiao, F. Lewis, and L. Xie. Energy-efficient distributed adaptive multisensor scheduling for target tracking in wireless sensor networks. *IEEE Transactions on Instrumentation and Measurement*, 58(6):1886–1896, June 2009.

[17] S. Madria, V. Kumar, and R. Dalvi. Sensor cloud: A cloud of virtual sensors. *IEEE Software*, 31(2):70–77, 2014.

[18] S. Maheswararajah, S. Halgamuge, and M. Premaratne. Sensor scheduling for target tracking by suboptimal algorithms. *IEEE Transactions on Vehicular Technology*, 58(3):1467–1479, March 2009.

[19] S. Misra, S. Chatterjee, and M. Obaidat. On theoretical modeling of sensor cloud: A paradigm shift from wireless sensor network. *IEEE Systems Journal*, PP(99):1–10, 2014.

[20] R. B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73.

[21] J. Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica*, 33:520–534, 1965.

[22] G. Shroff. *Enterprise Cloud Computing*. Cambridge University Press, 2010.

[23] J. Teng, H. Snoussi, C. Richard, and R. Zhou. Distributed variational filtering for simultaneous sensor localization and target tracking in wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 61(5):2305–2318, Jun 2012.

[24] H. Wang, K. Yao, and D. Estrin. Information-theoretic approaches for sensor selection and placement in sensor networks for target localization and tracking. *Journal of Communications and Networks*, 7(4):438–449, Dec 2005.

[25] X. Wang, M. Fu, and H. Zhang. Target tracking in wireless sensor networks based on the combination of kf and mle using distance measurements. *IEEE Transactions on Mobile Computing*, 11(4):567 – 576, Apr. 2012.

[26] X. Wang, Z. Li, P. Xu, Y. Xu, X. Gao, and H.-H. Chen. Spectrum sharing in cognitive radio networks - an auction-based approach. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 40(3):587–596, June 2010.

[27] C.-H. Yu, B. Mumey, and O. Tirkkonen. Distributed multiple relay selection by an auction mechanism. In *IEEE Global Communications Conference (GLOBECOM)*, pages 4392 – 4397, Anaheim, CA, 3-7 Dec. 2012.

[28] M. Yuriyama and T. Kushida. Sensor-cloud infrastructure - physical sensor management with virtualized sensors on cloud computing. In 13^{th} *International Conference on Network-Based Information Systems*, pages 1–8, Takayama, 14-16 Sept. 2010.

[29] M. Yuriyama, T. Kushida, and M. Itakura. A new model of accelerating service innovation with sensor-cloud infrastructure. In *SRII Global Conference (SRII), 2011 Annual*, pages 308–314, March 2011.

[30] L. Zhou and H. Wang. Toward blind scheduling in mobile media cloud: Fairness, simplicity, and asymptotic optimality. *IEEE Transactions on Multimedia*, 15(4):735 – 746, 2013.

[31] L. Zhou, Z. Yang, J. Rodrigues, and M. Guizani. Exploring blind online scheduling for mobile cloud multimedia services. *IEEE Wireless Communications*, 20(3):54–61, 2013.