SENSORS-AS-A-SERVICE: TOWARDS THE CONCEPTUALIZATION OF SENSOR-CLOUD

Subarna Chatterjee

SENSORS-AS-A-SERVICE: TOWARDS THE CONCEPTUALIZATION OF SENSOR-CLOUD

Thesis submitted to the
Indian Institute of Technology Kharagpur
for award of the degree

of

Doctor of Philosophy

by

Subarna Chatterjee

Under the guidance of

Dr. Sudip Misra

Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721 302, India
July 2017

©2017 Subarna Chatterjee. All rights reserved.

CERTIFICATE OF APPROVAL

20/07/2017

Certified that the thesis entitled **Sensors-as-a-Service: Towards the Conceptualization of Sensor-Cloud** submitted by **Subarna Chatterjee** to the Indian Institute of Technology, Kharagpur, for the award of the degree Doctor of Philosophy has been accepted by the external examiners and that the student has successfully defended the thesis in the viva-voce examination held today.

(Member of DSC) (Member of DSC) (Member of DSC)

(Supervisor)

(Internal Examiner) (Chairman)

CERTIFICATE

This is to certify that the thesis entitled **Sensors-as-a-Service: Towards the Conceptualization of Sensor-Cloud**, submitted by **Subarna Chatterjee** to Indian Institute of Technology Kharagpur, is a record of bonafide research work under my supervision and I consider it worthy of consideration for the award of the degree of Doctor of Philosophy of the Institute.

Date: 20/07/2017

Dr. Sudip Misra
Associate Professor
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721 302, India

DECLARATION

I certify that

a. The work contained in the thesis is original and has been done by myself under the general supervision of my supervisor.

b. The work has not been submitted to any other Institute for any degree or diploma.

c. I have followed the guidelines provided by the Institute in writing the thesis.

d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.

f. Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Subarna Chatterjee

Dedicated to
My parents, my sister, and Subhadeep

ACKNOWLEDGMENT

Writing this part of the dissertation is probably the toughest. Though the list of people to acknowledge is long, making this list is not difficult. The difficult part is to find the words that convey the sincerity and magnitude of my gratitude. People unknown to me till a few years before, have become indispensable in my life, while the people already known to me, remain pillars of support and encouragement all these days. It is because of their presence that I am here now.

First and foremost I would like to express my deepest gratitude to my supervisor Dr. Sudip Misra for his invaluable guidance and encouragement throughout my work. His constant motivation, support and infectious enthusiasm have guided me towards the successful completion of my doctoral study. My interactions with him have been of immense help in defining my research goals and in identifying ways to achieve them. His encouraging words have often pushed me to put in my best possible efforts. Above all, the complete belief that he has entrusted upon me and has instilled a great sense of confidence and purpose in my mind, which I am sure, will stand me in good stead throughout my career.

It gives me immense pleasure to thank my doctoral scrutiny committee members Dr. P. Mitra, Dr. K. S. Rao, and Dr. G. Das for their valuable suggestions during my research tenure. My sincere thanks to the ex-Head of the Department of Computer Science & Engineering, Dr. Rajib Mall, and the current Head of the Department of Computer Science & Engineering, Dr. Sudeshna Sarkar, for the world class infrastructure provided in the department to the research students. In this regard, it is also worthy to thank Tata Consultancy Services (TCS), Google, and Facebook for supporting my Ph.D. in terms of monthly fellowships, flexible contingency grants, and travel grants. I also thank all faculty members of Department of Computer Science & Engineering for their helpful comments and constant encouragement. I sincerely remember the support of office staffs Mithunda, Somadi, Malayda, Vinodda, Pratap da, Utpal Da, and others.

I wish to convey my special thanks to my friends Subhadeep, Amit, Manasa, Anirban, and Procheta Di for their constant support and help during the various stages of

my work. I would really like to thank Mr. Samir Banerjee for his constant support that has helped me sort out several additional things. I am greatly indebted to many of my friends for their constant inspiration. The support of my group mates namely Samaresh Da, Ayan, Arijit Da, Sanku, Barun Da, Sankar Da, Prosenjit Da, Anandarup Da, Tamonghna Da, Goutam da, Nabiul Da, Judhistir Da, Soumen Da, Sumit Da and many more. It is a great fun and source of ideas and energy to have friends like Pratik Da, Snighdha Di, Debasmita, Achutananda, Parakrant, Tuhin Da, and many more during my stay at IIT Kharagpur.

Acknowledgment certainly remains incomplete if I do not write anything regarding my parents and my sister, whom I love the most. Due to my studies, they have sacrificed a lot since my childhood days. Their priceless affection, spontaneous encouragement, dedication to build myself a good human being are the pillars of my strength to achieve goals which are almost next to impossible. No word is enough to express their contributions to my life. Finally, I am grateful to my school teachers, well wishers, elders in our native place for their blessings.

Subarna Chatterjee

Abstract

Wireless Sensor Networks (WSNs) have enhanced the standard of living of mankind with the touch of advanced technology and the manifestations of this fact are found in numerous real-life applications. However, all of these WSN-based applications are *single-user centric*, in which a user-organization owns and deploys its personalized sensor network and typically does not share the accessed data to another party (user/organization). Also, the data sharing policies vary across organizations and an external user-organization is able to retrieve sensor information that is specific only to the region that is administered by the network administrator. Thus, generally, only user-organizations that own a sensor network have satisfactory access to sensor data. To address these issues, recently, *sensor-cloud* infrastructure has been conceived as a potential solution for multi-organization WSN deployment and data access.

Sensor-cloud infrastructure acts as the interface that connects the physical world (attributes of which can be measured using sensor based devices) and the cyber world of inter-computer communication through excellent data scalability, on-demand service provisioning, remote data visualization, and user programmable analysis. The idea of sensor-cloud thrives on the principle of virtualization of physical sensor nodes. The user-organizations possess their own applications, and request the sensor-cloud for retrieval of sensed data. These requests are interpreted within the sensor-cloud environment, and the physical sensor nodes are dynamically consorted to form Virtual Sensors (VSs), as per requirements. Aggregated data from the VSs are transmitted to the end-users in the form of a simple obtainable service (just as electricity, or water), named *Sensors-as-a-Service* (*Se-aaS*). Sensor nodes are hereby transformed from a typical 'hardware' to a simple 'service', which is cheap, convenient, user-friendly, and scalable.

Currently, the principles, ideology, and challenges involved in this paradigm shift from traditional WSNs to sensor-cloud platforms are being explored. However, the technicalities that are required from an implementation perspective, inclusive of the theoretical modeling, experimental analysis, architectural designs, and development of this platform, are unexplored till date. This dissertation focuses to resolve the principal technical challenges associated with the complete conceptualization of sensor-cloud and eventually aims to build a fully-functional prototype of sensor-cloud infrastructure.

A summary of the major works reported in this dissertation is described as follows.

From an implementation point of view, the scope of technical and theoretical research have been identified in this domain. Initially, the focus is on the theoretical modeling of virtualization of physical sensor nodes within sensor-cloud. The necessity for this paradigm shift to a sensor-cloud platform is mathematically justified for all WSN-based applications. Eventually, the work endeavors to establish the idea of Se-aaS. Next, the design issues of sensor-cloud platforms are addressed. Conventional data transmission techniques involve periodic packet transmissions from sensor nodes to the cloud-servers for computation, storage, and processing. However, the rate of change of the physical environment may not be reasonably significant, thereby, leading to redundant packet transmissions and inefficient utilization of network resources. In this regard, a dynamic, adaptive, and optimal caching has been designed that preserves the accuracy of information, and conserves the network resources, simultaneously. Followed by this, the economics of the infrastructure is investigated. Within sensor-cloud infrastructure, the end-users utilize the physical sensors and the cloud infrastructure as per their demand and pay as per their usage, to the cloud service provider (CSP). Thus, to quantify the usage of the end-users and charge them accordingly, a dynamic and optimal pricing scheme is designed, specifically for Se-aaS. The networking dimensions of sensor-cloud have also been considered – the problem of routing and channelization of the data of the VSs, originating from multiple regions, to geographically distributed sensor-cloud data centers (DCs) is investigated. To resolve this issue, an algorithm is proposed for the dynamic scheduling of a cloud DC that would serve a particular user application with data from the respective VSs.

Combining the solutions of the afore-mentioned research challenges, a holistic prototype of sensor-cloud infrastructure is developed using real sensor hardware and a real cloud platform. To validate the correctness of the infrastructure, an application-specific scenario of multiple target tracking is investigated and experimented within it. An experiment is also performed using the real (non-simulated) setup to examine the performance of the prototyped sensor-cloud infrastructure in big-data environments. It is observed that the infrastructure performs significantly well in practical scenarios involving real sensor-hardware, huge and voluminous data requests, and large number of end-users.

Keywords: Cloud computing, wireless sensor networks, big data, data centers, energy constraints, pricing, virtualization

Contents

List of Figures

List of Tables

List of Abbreviations

BS	Base Station
BSCI	Big-Sensor-Cloud Infrastructure
BSCSP	Big-Sensor-Cloud Service Provider
CCCI	Client-side Client-Cloud Interface
CCF	Coverage Contraction Factor
CCS	Credit Clearance Service
CSP	Cloud Service Provider
DC	Data Center
DCL	Data Control Language
DDL	Data Definition Language
DML	Data Manipulation Language
DQS-Cloud	Data Quality-Aware Sensor Cloud
DSR	Dynamic Source Routing
DWDM	Dense Wavelength Division Multiplexing
EC	External Cache
HDFS	Hadoop Distributed File System
HMDP	Hierarchical Markov Decision Process
HMTT	Hierarchical Markov Decision Process for Target Tracking
IaaS	Infrastructure-as-a-Service
IC	Internal Cache

IIA	Independence of Irrelevant Alternatives
RMSE	Root Mean Square Error
NSNR	Nodal Signal to Noise Ratio
OpenFS	Open Field Server
OLS	Open Line System
OSGi	Open Service Gateway Initiative
pH	pricing attribute due to Hardware
pI	pricing attribute due to Infrastructure
P	Pareto Axiom
PaaS	Platform-as-a-Service
PPC	Potential Pareto Criterion
PPM	Packet Purse Model
PPSS	Probability-based Prediction and Sleep Scheduling
QoI	Quality of Information
QoS	Quality of Service
RSS	Received Signal Strength
SaaS	Software-as-a-Service
Se-aaS	Sensors-as-a-Service
S-DMA	Social-choice based Dynamic Mapping Algorithm
SAF	Social Aggregation Function
SCCI	Server-side Client-Cloud Interface
SCF	Social Choice Function
SOA	Service Oriented Architecture
VM	Virtual Machine
VS	Virtual Sensor
VSG	Virtual Sensor Group
UPnP	Universal Plug and Play

WSN	Wireless Sensor Network

Chapter 1

Introduction

In the recent past, Wireless Sensor Networks (WSNs) have emerged at a very fast pace and have been an integral part of enumerable real-life applications such as target-tracking [1], battlefield monitoring [2, 3], telemonitoring [4], ubiquitous health monitoring [5], and several other applications [6, 7]. WSNs are built of several individual sensor units coupled with multiple communicator units, together functioning as sensor nodes. The individual sensor nodes are equipped with sensing hardware, but are essentially resource-constrained devices with low power and limited computational abilities. The end-user organizations of the WSNs are required to own and purchase the individual sensor nodes, and deploy the same over a terrain of their interest. Subsequently, these nodes communicate among one another to form the sensor network. The sensor nodes of a particular WSN comprise of operating systems with monolithic kernel in which the application remains compiled, and the nodes are specific to that particular application only. Existing WSN-based applications are generally owned and used by commercial organizations and these organizations are also responsible for the maintenance of the sensor network over time. The WSN for a particular end-user organization is designed to serve the interests of that organization only, and hence, the applications running within the sensor nodes are programmed accordingly to meet the requirements of the

organization.

Recent research has discovered multifold limitations [8, 9] of conventional WSNs, which are summarized as follows:

- Existing WSNs are generally single-user centric and sensors of a particular WSN are configured for a single application. Therefore, the renderability of customized WSN applications is infeasible in traditional WSNs, unless the WSN is reconfigured afresh, and the customized application is reloaded within the sensor-nodes.

- Existing WSN-based applications require the end user-organizations to own the sensor nodes, deploy those over a region, and also be responsible for the maintenance of the same. The purchase, deployment, maintenance, and management of WSNs are extremely expensive [10] and require the supervision of technical personnel.

- Organizations which do not own sensor nodes are deprived of access to applications of the WSNs that are deployed in the field. Further, as mentioned above, it is extremely expensive for the common mass of people and organizations to purchase and deploy sensor networks on their own. This applies to any WSN specific application, in general, and results in redundancy and unoptimized utilization of sensor network resources [11].

- Because of monolithic kernels, it is infeasible to schedule and load multiple applications (of the same type) within a single sensor node. However, in some cases, a two-stage bootloader is used to support switching between multiple applications. But such operations involve manual intervention, and significantly high overhead cost due to memory management, operating system independence, and complicated process management [12, 13].

Evidently, with the existing state-of-the-art (i.e., the traditional WSNs), the common mass of people cannot enjoy the very emerging sensor technology without being directly

involved with the purchase, deployment, maintenance, and management of the sensor nodes. The problem is intense and nontrivial, as most of the end-users are naive and the overheads associated with a WSN are expensive.

Recent research [14,15] envisions sensor-cloud infrastructure as a potential substitute for traditional WSNs. Sensor-cloud infrastructure is essentially an offshoot of conventional cloud computing [16–18] that thrives on the principle of virtualization of physical sensor nodes, thereby rendering a powerful infrastructure that interfaces between the physical and cyber worlds. According to MicroStrain[1], who stands among one of the pioneers in inventing this technology, sensor-cloud infrastructure is defined as [14]:

> *A unique sensor data storage, visualization and remote management platform that leverage powerful cloud computing technologies to provide excellent data scalability, rapid visualization, and user programmable analysis.*

Unlike the usual WSNs, sensor-cloud disseminates the usability of the physical sensors to the common mass of end-users who do not have to own, maintain, or manage the physical sensor nodes. In order to avail of the sensor-cloud infrastructure, the end-users are required to possess their own WSN-based applications. These applications are fed by the required sensed information, directly from the sensor-cloud service provider, on-demand from the end-users. The underlying procedure of obtaining the raw sensed data from the physical networks and the complex processing of those data are completely abstracted from the end-users. The physical sensors are virtualized to form Virtual Sensors (VSs) and data from the VSs are sent to the end-users. From the end-user organization perspective, it appears that the organization is continuously being served by a set of dedicated sensor nodes. However, in reality, a particular physical sensor node serves multiple end-users and is dynamically allocated or deallocated to serve different end-users. Thus, the virtualization of the physical sensor nodes enables the end-users to

[1] http://www.sensorcloud.com/system-overview

envision the *Sensors-as-a-Service*, commonly known as *Se-aaS*. Se-aaS breaks the conventional perception of the sensor nodes as typical hardwares and enables the end-users to envision them simply as a service, just like water or electricity.

1.1 Background

In this Section, the background concepts and the architectural details of sensor-cloud infrastructure are illustrated. The Section includes the various actors and their interrelationships, the different functional components of the infrastructure, and the work flow of the platform.

1.1.1 Actors of Sensor-cloud

Sensor-cloud infrastructure comprises of three different actors [19]:

1. **End-user**: End-user is a person (or an organization) who (that) possesses his/her (its) own applications, which are to be fed with sensor-data from the physical sensor networks. As the type and amount of the demand changes with time, the end-users enjoy scalability of Se-aaS, provided by the Cloud Service Provider (CSP), i.e., the end-users are privileged to demand for different sensor services at different time instants from heterogeneous sensor devices, and the services are offered instantaneously by the CSP. In return, the end-users are liable to pay as per their usage of Se-aaS to the CSP.

2. **Sensor-owner**: The sensor-owners are the business-actors in sensor-cloud infrastructure as they primarily impact the economics of the infrastructure. They purchase physical sensor devices and lend these devices to the CSP. The sensor-owners earn a monthly monitory profit as per the usage of their respective sensor devices.

3. **Sensor-cloud administrator**: The sensor-cloud administrator primarily manages and controls the entire cloud processing activities involving virtualization of

the physical sensor devices into distinct VSs, maintenance and monitoring of the physical sensor devices, organization of the unstructured data, executing computationally intensive queries over the big data sets, and real-time service provisioning of Se-aaS.

1.1.2 Architecture of Sensor-cloud

In this subsection, the internal architecture of sensor-cloud is presented. The fundamental difference between conventional WSNs and sensor-cloud platforms is that, unlike WSNs, in sensor-cloud platforms the applications are decoupled from the physical sensor nodes. The sensor-nodes are used only for sensing and performing local data aggregation while the entire application-processing logic is handled within the cloud computing unit.

Sensor-cloud infrastructure essentially follows a three-tier architecture [19], as shown in Figure 1.1. Tier 1 is the bottom-most layer of the infrastructure, which comprises of physical sensor nodes that are heterogeneous in terms of their vendor, sensing hardware and sensing capabilities, communication module and mechanism, and applications. The data from Tier 1 are pulled by the cloud computing unit that comprise Tier 2. In Tier 2, the entire sensor-cloud computation is performed, i.e., Tier 2 is the center for execution of sensor allocation and deallocation, sensor grouping and virtualization, pulling sensor data from Tier 1, aggregation of raw sensor data, and transmission of aggregated sensor data to Tier 3. In Tier 3, lies multiple end-user organizations with their own applications. Data from Tier 2 are fed to these applications running at Tier 3. Any modification in the structure and design of the applications in Tier 3 does not call for any change in the sensors of Tier 1, i.e., if an application in Tier 3 is re-designed or re-structured, it is not necessary to reload or reconfigure the sensor nodes to serve the modified application.

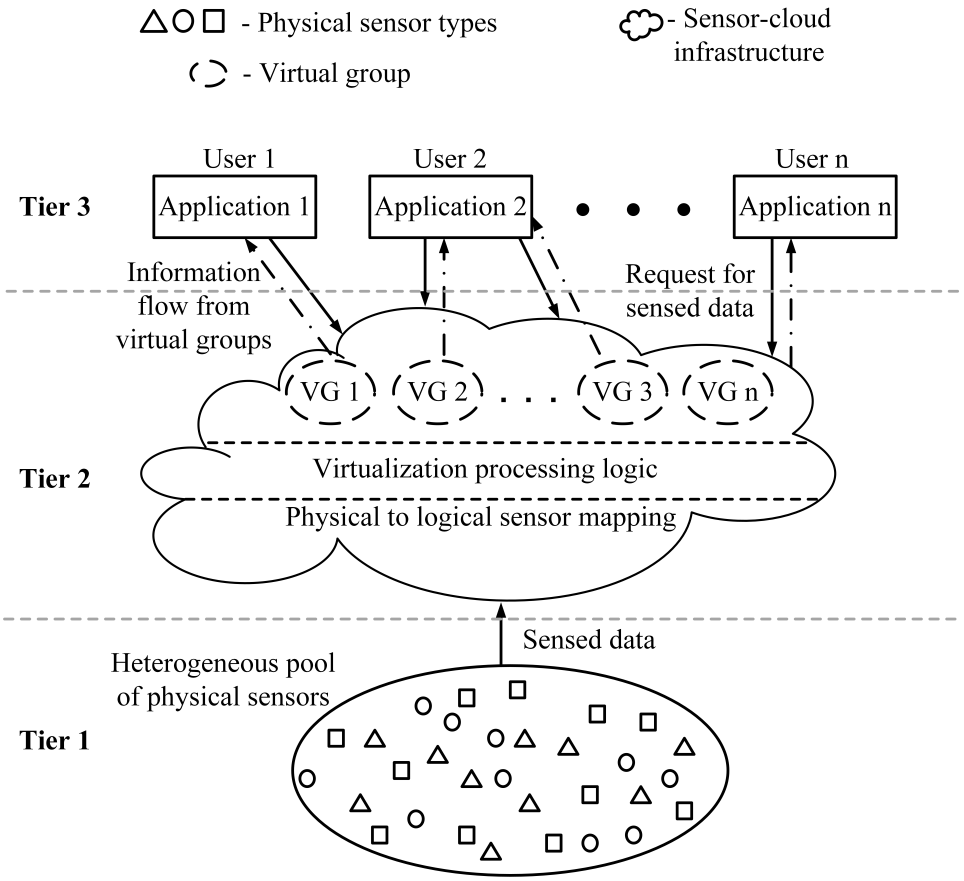


Figure 1.1: Architecture of sensor-cloud

1.1.3 Views of Sensor-cloud

This subsection presents the details of the architectural aspects of sensor-cloud from two different points of view: (a) *user-organization's view* or the *logical view*, and (b) *algorithmic view* or the *real view*.

1.1.3.1 User-organization's View (Logical View)

The logical view or the user-organization's view of obtaining Se-aaS is the perspective of how a sensor-cloud end-user envisions the infrastructure. Figure 1.2 depicts the logical view of the architecture from the viewpoint of the end-user-organization. The communication interface of a user-organization is primarily a Web interface running at the site

of the CSP. It is a Web portal through which the user-organization requests Se-aaS [19]. After the user-organization logs into the portal, the end-user is presented with some specific templates that collect information relevant to the type of application such as the type of sensor nodes that the user is expecting, and the region that the user is interested in. The templates enable the end-user to specify his requirements of sensor data at a very high level. The templates are then interpreted within the cloud to transform the high level requirements in terms of allocation physical sensor nodes. The subsequent details of this process are discussed in Subsection 1.1.3.2.

Having specified the relevant details, the user-organization is kept abstracted from the underlying complex processing logic involved with the physical sensor node allocation, application-specific aggregation, and virtualization. Following the consolidated data processing within cloud, the user-organization receives the sensed information from the cloud through the portal, which, in turn, is fed into the intended application(s).

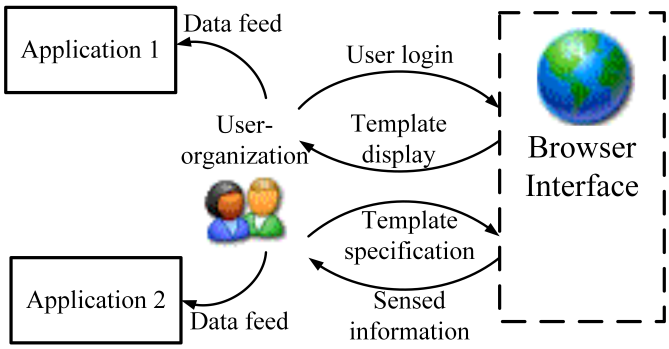


Figure 1.2: User-organization's view of sensor-cloud

1.1.3.2 Algorithmic View (Real View)

The real view of the architecture for the actual processing, which is required within the sensor-cloud infrastructure, is discussed here. Figure 1.3 shows the diagrammatic representation of the philosophy behind sensor node virtualization, which is the principal component of the real view. Instructions obtained from the end-user organization are

extracted from the templates. As sensor-cloud architecture deals with sensor nodes with heterogeneous specifications, the sensor nodes are standardized using Sensor Modeling Language (SensorML), defined by the Open Geospatial Consortium [19, 20]. To make the processing flexible, manageable, and platform-independent, SensorML uses XML encoding while maintaining the sensor metadata [21]. The process of translating the high level user-requirements, in terms of physical sensor allocation, is one of the major contributions of this dissertation, and his discussed in detail in Chapter 3. Once the physical sensors are allocated to serve a particular application, the data from all of these sensors are virtualized to form the respective VSs. Different VSs serving a particular application are further grouped to form Virtual Sensor Groups (VSGs). After the formation of the VSs and VSGs, the data are pulled from the underlying physical sensor nodes in an on-demand and application-specific manner. The data from the sensors comprising a VS are meaningfully aggregated and dispatched to the respective end-user applications.

Every physical sensor node reports its sensed data to the sensor-cloud storage. Within the cloud environment, the sensed data are efficiently aggregated in real-time. These data from a consolidated group of sensor nodes are transmitted to the end-user-organization. Thus, an end-user anticipates the source of the data to be dedicated for his/her application, whereas, practically the set of source sensor nodes is chosen from an infinite pool of resources and changes dynamically. Further, the single sensor node may also serve multiple applications at a particular time instant and the composition of VSs may change based on sensor allocation policies.

From Figures 1.1 and 1.2, the usefulness of sensor-cloud architecture is well perceived. The end-users of sensor-cloud can be any naive person/organization possessing its own WSN application(s). Unlike conventional WSNs, the end-users can obtain *Se-aaS*, just as water or electricity, which can be obtained on-demand, in no time. Thus, sensor-cloud brings in a revolutionary change by enabling the dissemination of WSN technology to the common mass of people/organizations who do not really own WSNs.

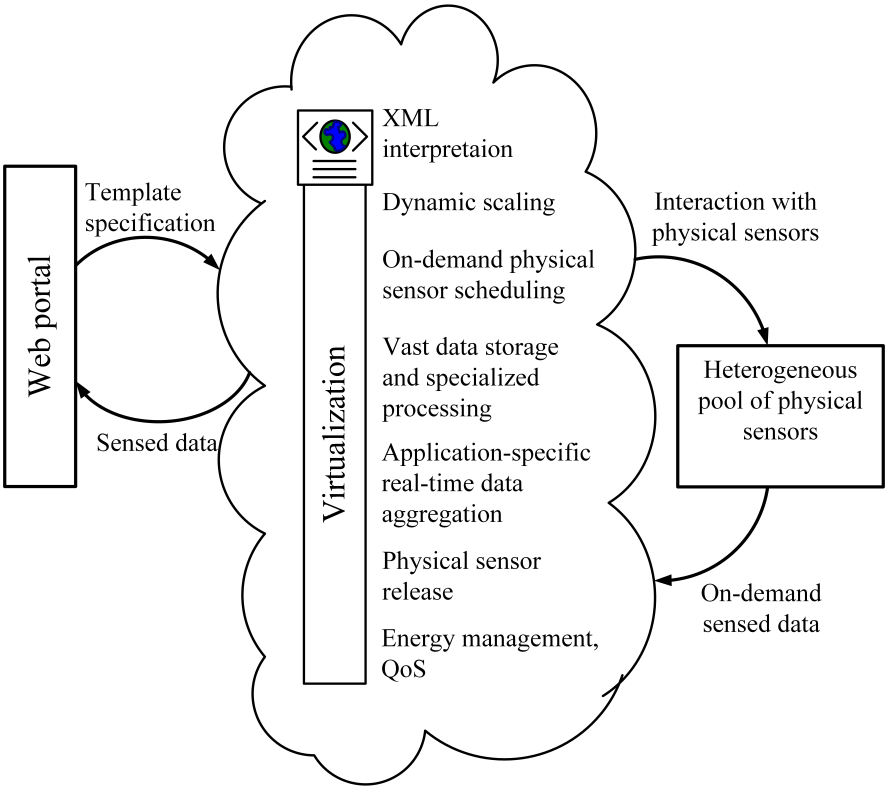


Figure 1.3: Real view of complex processing within sensor-cloud

As mentioned previously, by virtualization, sensor-cloud enables run time switching of applications, and real-time data and resource provisioning, without the user being aware of the complex processing logic. On the contrary, in a WSN, the nodes are statically configured for a fixed set of applications. From the architectural aspects of sensor-cloud infrastructure, it can be inferred that only the virtualization aspect of it makes it so convenient, accessible, beneficial, and adaptable for public interests. The pay-per-use policy within sensor-cloud also adds on to the benefits of the end-users by diminishing the huge expenditure incurred for setup, maintenance, and management of WSNs.

1.2 Scope and Objectives

Sensor-cloud platforms are relatively recent, and are recognized as potential substitutes of the traditional WSNs. Presently, very few research works exist on this technology, and to the best of our knowledge, all of the existing research works focus primarily on the dogma, the principles, and the conceptualization of sensor-cloud. However, research work providing scientific and technological concreteness to the concept of sensor-cloud infrastructure are scarce. From an implementation point of view, this dissertation identifies the scope of technical and theoretical research. Considering the above-mentioned scopes of research in this domain, the primary *scope* and *objectives* of the dissertation are outlined as follows:

(a) **Theoretical characterization of virtualization and justification for a paradigm shift from conventional WSNs**: Initially, this dissertation focuses on the theoretical characterization of virtualization of physical sensor nodes – one of the first attempts in this direction. Existing related research works on sensor-cloud have primarily focused on the ideology and the challenges that WSN-based applications typically encounter. However, none of the works has addressed theoretical characterization and analysis, which can be used for building models for solving different problems to be encountered in using sensor-cloud. This objective also experimentally justifies the necessity for the shift of paradigm from traditional WSNs to sensor-cloud platforms.

(b) **Dynamic and adaptive data caching for virtualization within sensor-cloud**: Conventional data transmission techniques involve periodic packet transmissions to the cloud-end. However, the rate of change of the physical environment may not be reasonably significant, thereby, leading to redundant packet transmissions and inefficient utilization of network resources. Hence, an optimal caching mechanism is designed to be executed within sensor-cloud to obtain resource efficiency in terms

of energy and network lifetime. The proposed data caching mechanism is dynamic, and is adaptive to the change of the physical environment, and thereby, preserves the accuracy of information, and conserving the network resources, simultaneously.

(c) **Dynamic and optimal pricing scheme, specifically for Se-aaS**: As sensor-cloud infrastructure conceptually extends cloud computing, it complies with the features that are intrinsic to the latter. A cloud computing platform generally conforms with a pay-per-use model, in which the end-users pay only for those units of resources that are utilized. Within sensor-cloud technology, the end-users utilize the physical sensors and the cloud infrastructure as per their demand and pay as per their usage, to the CSP. Thus, it is necessary to develop a pricing scheme for Se-aaS to quantify the usage of the end-users and charge them accordingly. This objective focuses to design a dynamic and optimal pricing scheme, specifically for Se-aaS. The proposed scheme considers issues that are inherent to the heterogeneity of services of sensor-cloud infrastructure.

(d) **Optimal data center scheduling for Quality of Service management in sensor-cloud**: The networking dimensions of sensor-cloud is also totally unexplored to this date. Research scopes on sensor-cloud networking has also been identified due to routing and channelization of the data of multiple VSs, originating from multiple regions, to geographically distributed sensor-cloud data centers (DCs). Existing research works have not considered these challenges, and uncertainties that might come up while executing the traditional sensor-based applications using sensor-cloud. Therefore, the objective is to design and propose an algorithm for the dynamic scheduling of a cloud DC that would serve a particular user application with data from the multiple VSs.

(e) **Development of a prototype of sensor-cloud infrastructure using real sen-**

sor hardware and cloud platform: For the sake of establishing sensor-cloud platforms for practical purposes, this dissertation also presents a real prototype implementation of sensor-cloud infrastructure using real hardware. In the endeavor of prototyping sensor-cloud, there were certain limitations in the conceptualization of sensor-cloud that were investigated and became evident. With the vastness of contemporary data, especially when multiple organizations tend to access the infrastructure simultaneously with heterogeneous demand for Se-aaS, sensor-cloud encounters serious bottleneck as its processing mechanisms were traditional [19]. Further sensor-cloud infrastructure provisions sensor data for application-feed only, i.e., it cannot execute customized analytics on huge data sets to obtain meaningful, intelligent information. This objective of the dissertation addresses the aforesaid limitations of sensor-cloud and proposes a new infrastructure – *Big-Sensor-Cloud Infrastructure* (*BSCI*) – by including an innovative component within the infrastructure. A prototype of BSCI is constructed and comparatively tested with the sensor-cloud platforms to investigate the difference in performance. It was observed that BSCI is an enhanced and modified version of sensor-cloud platforms in terms of performance of data (or query) processing, storage, and management.

(f) **Application specific analysis of sensor-cloud infrastructure: Target tracking**: After developing an enhanced version of sensor-cloud infrastructure, referred to as BSCI, it is important to look into challenges, and uncertainties that might come up while executing the traditional sensor-based applications within sensor-cloud platforms. Such a scenario is investigated and simulated for executing a multiple target tracking application within a sensor-cloud platform.

1.3 Contribution of the Dissertation

The proposed research focuses on building a holistic prototype of sensor-cloud infrastructure, rendering Se-aaS. The primary contributions of the dissertation are as follows:

- *Theoretical characterization of sensor-cloud and justification for a paradigm shift from conventional WSNs*: Initially, the work focuses on the theoretical modeling of virtualization of physical sensor nodes. The necessity for a paradigm shift for all WSN-based applications to a sensor-cloud platform is experimentally justified. The proposed work has suggested a framework for performance analysis of sensor-cloud based on few chosen metrics such as fault-tolerance, lifetime of a sensor node, and energy consumption, in contrast to that of a WSN. Finally, this work endeavors to conceive the idea of using physical Se-aaS.

- *Data caching policies with the infrastructure*: An optimal caching mechanism within sensor-cloud to obtain resource efficiency in terms of energy and network lifetime. The proposed data caching mechanism is dynamic, and is adaptive to the change of the physical environment. Thereby, it preserves the accuracy of information and conserves the network resources, simultaneously.

- *Designing of a dynamic and optimal pricing scheme, specifically for Se-aaS*: As a cloud computing platform generally conforms with a pay-per-use model, within sensor-cloud platforms, the end-users utilize the physical sensors and the cloud infrastructure as per their demand and pay as per their use, to the CSP. Thus, a pricing scheme is developed for Se-aaS to quantify the use by the end-users and charge them accordingly.

- *Optimal DC scheduling and networking withing DCs for QoS management*: The work proposes a DC scheduling algorithm for routing and channelization of the data of the VSs originating from multiple regions, to geographically distributed

sensor-cloud data centers (DCs). The work also focuses to choose a single DC serving a particular application by reducing the notwork overhead simultaneously.

- *Functional prototype development for Se-aaS*: This work deals with the development of a prototype of sensor-cloud infrastructure using real sensor hardware and cloud platform. The work identifies the limitations of the basic sensor-cloud infrastructure and proposes a modified infrastructure.

- *Application specific analysis of sensor-cloud infrastructure*: This work considers the challenges and uncertainties that might arise while executing the traditional sensor-based applications using sensor-cloud. The problem is investigated in a multiple target tracking application scenario, using the sensor-cloud platform.

1.4 Organization of the Dissertation

The rest of the Chapters in this dissertation are organized as follows:

Chapter 2: Literature Survey

In this Chapter, a thorough survey of the literature on sensor-cloud is presented. The Chapter is divided into four distinct subsections. Initially, the evolution of sensor-cloud is elaborated and studied. Followed by this, the literature survey on pricing aspects of sensor-cloud is presented. Subsequently, the prior work done on the networking issues of sensor-cloud are discussed. Lastly, the different implementation models of sensor-cloud are presented and discussed.

Chapter 3: Theoretical Characterization of Virtualization and Experimental Justification for a Paradigm Shift

This Chapter of the dissertation focuses on the theoretical characterization of virtualization of physical sensor nodes is performed. The necessity for a paradigm shift for

all WSN-based applications to a sensor-cloud platform is experimentally justified. The proposed work suggests a framework for performance analysis of sensor-cloud, based on few chosen metrics such as fault-tolerance, lifetime of a sensor node, and energy consumption, in contrast to that of a WSN. Finally, it endeavors to establish the idea of using physical Se-aaS.

Chapter 4: Dynamic and Adaptive Data Caching Mechanism Within Sensor-cloud

This Chapter introduces both internal and external caching techniques to ensure efficiency in resource utilization of the underlying physical network. The proposed data caching mechanism is dynamic, optimal, and is adaptive to the change of the physical environment, and thereby, preserves the accuracy of information, and conserving the network resources, simultaneously.

Chapter 5: Dynamic and Optimal Pricing Scheme for Se-aaS

In this Chapter, a dynamic and optimal pricing scheme is proposed for provisioning Se-aaS comprising of two components – *pricing attributed to Hardware* (pH) and *pricing attributed to Infrastructure* (pI). pH addresses the problem of pricing the physical sensor nodes subject to variable demand and utility of the end-users. It maximizes the profit incurred by every sensor owner, while keeping in mind the end-users' utility. pI mainly focuses on the pricing incurred due to the virtualization of resources. It takes into account the cost for the usage of the infrastructural resources, inclusive of the cost for maintaining virtualization within sensor-cloud. pI maximizes the profit of the sensor-cloud service provider by considering the user satisfaction.

Chapter 6: Optimal Data Center Scheduling for QoS Management in Sensor-cloud

This Chapter focuses on the problem of scheduling a particular DC that congregates data from various VSs, and transmits the same to the end-user application. The work follows the general pairwise choice framework of the *Optimal Decision Rule*. The scheduling of the DC is performed under several network constraints, such as data migration cost, data delivery cost, and service delay of an application that ensures the preservation of the QoS and maintenance of the user satisfaction. The work quantifies the effective QoS of Se-aaS and determines an optimal decision rule for electing a particular DC. While arriving at a collective decision, the work incorporates the fallible decision making ability of a DC, thereby, excluding the loss of generality.

Chapter 7: Development of a Working Prototype of Sensor-cloud Infrastructure

This Chapter addresses the aforesaid limitations of sensor-cloud and proposes a new infrastructure – *Big-Sensor-Cloud Infrastructure* (*BSCI*) by including an innovative component within the infrastructure. As BSCI is cloud-based, it ensures the features of scalability, pay-per usage, and implementation of user programmable logic. The architecture allows the common people to envision Se-aaS. This Chapter presents the details of the limitation of sensor-cloud and illustrates the modified architecture for BSCI. Every functional module of BSCI is described and the development details of each of these modules are thoroughly presented. A comparative study is also performed to examine the improvement of BSCI over sensor-cloud in terms of performance in querying and data processing.

Chapter 8: Application Specific Analysis of Sensor-cloud Infrastructure: Target Tracking

In this Chapter, an application specific analysis is performed within sensor-cloud envi-

ronment. As sensor-cloud end-users may be multiple organizations, the traditional target tracking application encounters typical difficulty. As targets enter the coverage zone of multiple sensors, it becomes crucial to schedule sensors and generate distinct clusters of sensors for each target as different targets might be of interest to different organizations. It becomes challenging to correctly map sensors to targets, in the presence of overlapping coverage, to maintain their privacy and correctness of sensed information about the targets. To resolve this issue, the Social-choice based Dynamic Mapping Algorithm (S-DMA) is proposed, based on the *Theory of Social Choice*, for ensuring a 'fair' and unbiased mapping of sensors to targets. The details of this work are discussed in the corresponding chapter.

Chapter 9: Summary and Conclusion

This Chapter concludes the dissertation and highlights the key findings from the entire dissertation. The Chapter also discusses the future scopes of work and illustrates the future directions that may induce research interest.

Chapter 2

Literature Survey

In this Chapter, a survey of literature, related to the contributions made in this dissertation, is reported. Initially, the prior work on sensor-cloud are thoroughly discussed and analyzed. Followed by this, the literature survey is divided into four distinct Sections which are the major focused areas of the dissertation. Thus, in Section 2.1, the details on the evolution of sensor-cloud is presented followed by a thorough discussion on the current state-of-the art research towards realization of sensor-cloud. Section 2.2 presents the survey of work in network and cloud pricing models and their implications on sensor-cloud infrastructure. In Section 2.3, the work done on the networking aspects of sensor-cloud are thoroughly investigated. Section 2.4 discusses thoroughly all the real-life implementation models of sensor-cloud. Finally, Section 2.5 summarizes and concludes the Chapter.

2.1 Evolution of Sensor-cloud

Before the concept of sensor-cloud was actually proposed, many works explored the real-time communication aspects of cloud computing [22, 23]. Some works [24, 25] focused on the integration of sensors to a cloud framework. Misra *et al.* [26] studied the problem of integration of sensors with cloud from a perspective of health monitoring. The work

focuses on an optimal selection of gateway in order to obtain the maximum bandwidth required for health data transmission. The proposed idea was to transmit health data to a cloud platform in an atomic manner. In another work, Zhu *et al.* [27] focused on the reliability issues while integrating sensor networks to mobile cloud platforms. The work addresses two problems – time and priority-based data transmission and priority-based sleep scheduling for energy efficiency of sensor networks. Both the problems focus to improve the issues for integration WSNs to cloud platform. However, the dissertation not only focuses on the integration of sensor networks to cloud computing platforms, but also contributes by formally modeling the virtualization of sensors within the sensor-cloud environment.

Some of the fundamental issues were also addressed by Eugster *et al.* [28]. They proposed a publish/subscribe model that demonstrates the interaction between a publisher and subscriber based on notification of an event. This work is considered to form the basis of integrating sensor nodes in a cloud environment, as it focuses on data transfer between dissimilar entities of a system. Hassan *et al.* [29] discussed the challenges normally encountered while integrating WSN with cloud. The work proposed a sensor-cloud framework focusing mainly on Software-as-a-Service (SaaS) applications. The work also proposed Statistical Group Index Matching (SGIM) scheme, which can be used to transfer data to cloud applications, and the authors evaluate the work to exemplify its remarkable performance, compared to the existing algorithms. A similar effort has been put by Eggert *et al.* in [30]. The work highlighted the challenges faced due to the difficulty in understanding the diverse nature, implementation of the varied and scalable functionalities, and ensuring privacy in sensor-cloud. Additionally, the work drew a baseline for addressing the aforesaid issues. In another work, Kumar *et al.* [31] devised a mechanism for transferring large volume of sensed data from the local memory of sensor nodes to cloud storage. The authors proposed the idea of transferring the responsibility of data processing to the cloud gateways, thereby achieving high energy efficiency. The

authors exercised the algorithm for back propagation networks within the cloud-gateways to execute data filtration. Using neural networks, the authors showed that they were able to achieve efficiency in bandwidth consumption as well. It is observed that most of the existing works primarily enlisted and discussed the possibilities of sensor-cloud and the challenges involved with the same. However, this dissertation focuses on complete characterization of sensor-cloud form an implementation perspective.

Alamri *et al.* [32] presented a thorough survey on sensor-cloud, its definition, the intrinsic concepts, and the benefits of using it. The paper also presented a comparison of the type of message flows for different algorithmic approaches. Eventually, the authors also briefed the possible technical challenges in this aspect. Another work, which proved to be highly advantageous and constructive towards sensor-cloud research, was by Yuriyama and Kushida in [19]. This work clearly carved out the constructive and opportunistic aspects of sensor-cloud architecture to a great extent. Few works also focused on virtualization in sensor networks. Olariu *et al.* [33] contributed in this domain by proposing a very simple and general-purpose virtual infrastructure for WSNs. It is a protocol independent work that can be used by the existing routing or data aggregation protocols. Ojha *et al.* [34] dealt with topology virtualization by self-organization of nodes in underwater sensor networks. Few works focused on designing an application-specific framework and the data transmission methodologies [35–37]. Thus, the above works focused more on the designing aspects, whereas the dissertation concentrates on construction a fully-functional prototype of sensor-cloud infrastructure. For this purpose, all the technical research issues are delved into and studied.

Evidently, despite the upsurge in research on sensor-cloud, there lacks mathematically-based theoretical works that can help in supporting performance evaluation and analysis of sensor-cloud based systems. Thus, Chapter 3 of the dissertation provides a detailed formalization of the mathematical model behind *virtualization* – a key enabler of the sensor-cloud technology. Chapter 3 also justifies the necessity for a shift of technology,

from the conventional WSNs to sensor-cloud platforms, in the near future.

Out of the very few works that addressed the technical aspects of sensor-cloud, Nguyen, and Huh [38] focused on the security of data transmission in a sensor-cloud environment. Chandra *et al.* [39] addressed the benefits of using Ethernet enabled Arduino micro-controller for data communication at the sensor-cloud end. Bhunia *et al.* [40] addressed the problem of data acquisition from the underlying physical sensor nodes by fuzzification of the data. In all of the above works, the process of data transmission from the underlying physical networks to the cloud was considered to be periodic, and continuous. However, as mentioned in Section 1.2, the rate of change of environment is random, and inconsistent. Therefore, the goal of Chapter 4 is to achieve a double-caching mechanism, which is dynamic and adaptive to the change in the physical environment, thereby providing an efficient utilization of network resources, and simultaneously maintaining the accuracy of the provisioned data.

2.2 Pricing in Sensor-cloud

Some prior works exist on network pricing [41–44]. Ng and Seah [45] applied game theory for analyzing the truthful cooperation of physical nodes in a sensor network. The work considered the behavior of colluding nodes involved in data delivery and message acknowledgment in a lossy, multihop wireless network. Buttyan and Hubaux [46] proposed a secured pricing technique, which encourages the physical node to cooperate in message delivery and prevents from network overloading. In fact, some of the works [41, 42, 44] also focused on the energy-efficiency aspects in which the authors envisioned the problem of maintaining resource efficiency as a functional objective of pricing. However, such pricing schemes considered only the network attributes to be shared among the sensor nodes. In Chapter 5 of the dissertation, the goal is not to distribute the network parameters, but to provision Se-aaS through routing and forwarding of data packets. In the process of involving intermediate sensor nodes, the aim is to optimize the energy

efficiency and maintain the user-satisfaction, simultaneously.

On cloud pricing, specifically, several schemes were proposed for utilizing various cloud computing resources [47]. In a recent article, Kash and Key [48] discussed the open issues in cloud pricing and the futuristic evolution of pricing algorithms. Arevalos *et al.* [49] studied the famous the Spot Price Prediction (SPP) and presented a comparative analysis with other predictive algorithms in terms of three different metrics – mean-squared error, maximum positive error, and mean positive error. Li and Li [50] proposed a hierarchical pricing model, which considers the issues related to Quality of Service (QoS) and the utility of both users and service providers, thereby, enforcing a fair approach for both the parties. The authors of [51–53] proposed dynamic pricing schemes by adopting a revenue management framework from economics. The works suggest a pricing model in which the provider makes a profitable margin without affecting the customers' demands in the near future. The major challenge of this work is to predict how the demands of the users change with the change in the price, based on which the dynamic pricing model is suggested. Sharma *et al.* [54] proposed a pricing model that mainly focuses on two constraints: (a) the QoS to provide greater service satisfaction from the user perspective and (b) profitability aspects from the cloud service provide perspective. Son and Sim [55] studied both the pricing and time-slot negotiation for the various usages of cloud services. In another work, Nasiriani *et al.* [56] proposed simple ways of distribution of cloud's costs to its different tenants. The performance of the system is evaluated using commercial DCs of IBM and by using non-cooperative game theory. Dabbagh *et al.* [57] addressed the problem of maximizing cloud profits while having elasticity of resources. The work was tested on Google data traces. Similar to this work, Chi *et al.* [58] also focused on price calculation for resource allocation algorithms based on revenue redistribution. However, both of the works find their applicability on in Paas or Iaas environments.

Few works focused on dynamic resource pricing within a pre-specified time-limit and

fixed resource budget ensuring QoS [59–62]. Qin *et al.* [63] proposed a dynamic pricing model, which is flexible to the change in the demand of the end-users and accordingly, it adjusts the pricing of the cloud resources. Jangjaimon and Tzeng [64] implemented an 'enhanced adaptive incremental checkpointing' (EAIC) meant to significantly reduce the effective monetary cost involved in the expected turnaround time of an end-user application. Kantere *et al.* [65] designed a pricing scheme for the data cache of the cloud infrastructure. Few works [66, 67] have focused on price-based load balancing or resource sharing. Greenwell *et al.* [68] worked on Pricing Intelligence as a service (PINaaS) and carried out a case study of Amazon EC2 pricing to examine the dynamism and intelligence of pricing policies. Another interesting work presented by Gohad *et al.* [69] discussed how the pricing model can be designed for small or medium sized CSPs. However, for all of the afore-mentioned works, the pricing does not involve charging separate hardwares, such as physical sensors, and hence does not find their applicability for Se-aaS. All of the above are designed only for a specific service. As Se-aaS is built on a heterogeneous SOA, serving both infrastructure and hardware, Chapter 5 discusses the design and implementation of a dynamic and optimal pricing scheme for Se-aaS. The proposed scheme considers issues that are inherent to the heterogeneity of services of sensor-cloud infrastructure.

2.3 Networking in Sensor-cloud

Number of works explored, and addressed the technical issues of cloud networking [70–73]. Yuchi and Shetty [74] focused on teh diversity of cloud networks and introduced the hierarchical network resource graph for modeling of cloud networks. A tabu search algorithm is proposed for optimal positioning of the cloud DCs [75]. The work also focuses on efficient routing of information while taking the link capacities into account. The credibility of the work is also strengthened by the case study of a Web search engine. Mastroianni *et al.* [76] addressed the problem of managing power consumption of

DCs. The work propounds *ecoCloud* for the consolidation of VMs within DCs. Another work [77] also addressed the problem of consolidations of VMs under QoS constraints. In [78], Bruneo examined the performance of the cloud DCs for Infrastructure-as-a-Service platforms of cloud computing. The work proposes an analytical model for evaluating several performance metrics such as utilization, availability, waiting time, and responsiveness. Various other works explored the issues of traffic management in cloud systems [79, 80]. For mobile cloud systems, research has found the directions in cooperative resource management [47], energy-efficient offloading policies [81], and inter domain resource allocation [82]. However, for all of the above works, the data are transmitted from the DCs directly to the client. None of the work concerns intra-DC scheduling that is essential primarily for sensor-cloud infrastructure as it involves collection of data from various geographical locations to a single DC. Additionally, the flow of request in the aforementioned cases is downward, from the client to the cloud [83–85], whereas, in our case, the flow of data is bottom-up, from the physical networks to cloud.

Some works have differently worked on cloud networking by exploring some interesting issues of cloud networking. Massonet *et al.* [86] discussed the issue related to the security in federated cloud networks. The work proposed local and global security policies on different units of federation and the authors evaluated the work by inspection of a deep packet. Filer *et al.* [87] thoroughly examined the network infrastructure of Microsoft's cloud platform to analyze the effects of elasticity and flexibility of cloud networks. The authors assembled an Open Line System (OLS) using Dense Wavelength Division Multiplexing (DWDM) signals, Ethernet switches and a controller of the software defined networks. Jiao *et al.* [88] addressed the problem of dynamic allocation of cloud network resources that is highly difficult due to the highly varying demands, the reconfiguration cost, and the variety of resources. The authors address these challenges and evaluate the performance of the proposed approach on real-world cloud environments. Demydov *et al.* [89] proposed an intriguing issue of firewall migration within

cloud networks. The authors utilized it as an intermediate cloud platform that have a high-defense mechanism to stand heavy threats and security attacks. Murugesan and Bojanova [90] authored a book on cloud networks where the authors discussed about the various network and I/O virtualization techniques and the different possibilities and results. However, it is to be noted that none of the works focused on the network issues involving cloud DCs. The dissertation addresses the problem of migration of geographically distributed data from VSs to different DCs of sensor-cloud.

Chapter 6 of the dissertation propounds scheduling of a single DC for serving a particular application. The scheduled DC collects information from several temporary DCs, which are used in intermediate storage and logging of data from the physical sensors directly to the cloud. The scheduler process is designed under constraints to ensure user satisfaction and maintenance of QoS, simultaneously.

2.4 Implementation Models of Sensor-cloud

Several sensor-cloud frameworks and models have been implemented till date [91]. For example, in the work of Chandra *et al.* [92], the authors addressed the problem of establishing a connection with a cloud server and Ethernet enabled Arduino microcontroller based sensors. In this work, three different cloud services – ThingSpeak, Nimbits, and Open.Sen.se are explored for integration of sensors. Sen *et al.* [93] proposed a real-life implementation sensor-cloud architecture that encourages end-users to use their own sensor-based applications. The architecture is associated with the two different phases – "coverage" in which the optimal set of sensors is determined to locate a target and the second phase is "connectivity" to establish communication connectivity among all the selected sensors. However, this work just considers the collection and dumping of sensor data for real-life applications. Sen and Madria [94], in another work, focused on the risk-assessment aspects of sensor-cloud platforms. The authors proposed a risk-assessment framework to assess the impacts and implications of the potential risks of

such platforms and analyzed the likelihood of such risks. The work aims to aid the security administrator to accordingly design and characterize security parameters for sensor-cloud platforms. From a security perspective, Saha [95] focused on the intrinsics of secured data processing within a WSN. Although the author targets sensor-cloud integration, the work is primarily based on in-network management and processing of sensor data from various applications. All of the above works focus on proposing new concepts and architecture for real-life sensor-cloud applications. However, eventually, none of the works constructed a functional model or prototype of a sensor-cloud system.

From a development perspective, Kedia [96] developed a project on water-quality monitoring with the help of a sensor-actuator system. The project comprises of deployment of HydroQual, HydroQual-A, and Hydro-Depth sensors for monitoring basic water level, pressure of water, and depth of eater, respectively. Similar to the previous work, Neto *et al.* [97] proposed the SmartComponent Framework using technologies viz. Open Service Gateway Initiative (OSGi) Apache Felix Framework, Universal Plug and Play (UPnP) Basedriver, and iPOJO and the framework collects and processes sensor data. However, both the works assume that the subsequent integration of these sensors to a cloud set-up is trivial and do not mention much on the integration issues of the sensors to a cloud server. Hirafuji *et al.* [98] built an OpenFS (Open Field Server) hardware. The developed open-source hardware is was designed for high-performance sensor-networks and the authors employed Twitter-cloud within the hardware to share the obtained data. In another work [99], Srimathi *et al.* proposed an underwater sensor cloud model using Hadoop framework as a middleware for the purpose of data aggregation and management. The model provides real-time underwater status but does not utilize the big data management capabilities of Hadoop. Kothari *et al.* [100] focused on an Internet of Things (IoT) perspective. The authors proposed a Data Quality-Aware Sensor Cloud (DQS-Cloud) for pervasive data management, intelligent computing, and sensor stream management. However, none of the afore-mentioned cloud-based works

consider sensor virtualization aspects. Further, although Kothari *et al.* considered the IoT aspect, the work precisely does not consider the high volume, velocity, and variety of contemporary data. Therefore, Chapter 7 of the dissertation focuses to build a hoilstic prototype of sensor-cloud considering the above aspects.

2.5 Summary

Combining the limitations of the existing state-of-the-art for sensor-cloud infrastructure as discussed above, this dissertation focuses to address the afore-mentioned limitations in the subsequent Chapters. Eventually, the dissertation focuses to build a holistic prototype of a modified form of sensor-cloud infrastructure – the Big-Sensor-Cloud Infrastructure (BSCI). Followed by this, the prototype is validated by mounting a WSN-based multi-target tracking application. The proper execution of the application in the new paradigm is also examined.

The subsequent Chapter presents the theoretical characterization of virtualization within sensor-cloud and further presents and experimental evaluation to justify the paradigm shift from traditional WSNs.

Chapter 3

Theoretical Characterization of Virtualization and Experimental Justification for a Paradigm Shift

The previous Chapter presents a thorough review of the prior literature on sensor-cloud. Evidently, despite the upsurge in research on sensor-cloud, there lacks mathematically-based theoretical works that can help in supporting performance evaluation and analysis of sensor-cloud based systems. This Chapter proposes a detailed formalization of the mathematical model behind *virtualization*, a key enabler of the sensor-cloud technology. Motivated by the work of Dong *et al.* [101] that proposes an idea for a high-level model for virtualization, the main focus, in this specific work, is to characterize virtualization of sensor nodes and experimentally justify the necessity for a shift of technology from the conventional WSNs to a sensor-cloud platform in the near future.

In this Chapter, a detailed formalization of the mathematical model behind virtualization, a key enabler of the sensor-cloud technology, is presented, followed by an experimental justification for the paradigm shift from conventional sensor networks to sensor-cloud. The two afore-mentioned topics are covered in separate Sections.

3.1 Characterization of Virtualization

Prior to mathematically formulating the virtualization model, this work defines the entities and the sub-entities, which play active roles in the process of virtualization.

Definition 1. *The type of a physical sensor node, along with its specification, T_i, is interpreted to be an element from the set $T = \{T_1, T_2, ..., T_\alpha\}$, where α is the number of distinctly registered sensor types.*

For example, T_1 may represent a ADXL345 3-Axis 3g accelerometer, whereas T_2 may be the type indicator of a Laser Doppler Vibrometer. It is to be noted here that, multiple types of a particular sensor type can be allocated to an end-user. For example, an end-user may be allocated 10 rainfall measuring sensors across different geographical locations of a country and the values obtained can be averaged to get the overall rainfall metric for the country.

Definition 2. *Every sensor owner is denoted by O_i, such that, $O_i \in O = \{O_1, O_2, ..., O_\beta\}$, where β is the total number of sensor owners who contribute towards the sensor-cloud architecture.*

A sensor owner can voluntarily register into or deregister from the sensor-cloud.

Definition 3. *The location of a physical sensor node is denoted by a 2-tuple $Loc =< l_1, l_2 >$, where l_1 and l_2 represent the latitude and longitude of the position of the sensor node, correctly, upto a negotiated precision value.*

The location of a physical sensor node is stored within the cloud storage at the time of its registration, following its deployment. It is to be mentioned here that, the sensors can be deployed indoor or outdoor.

Definition 4. *The state of a sensor is denoted by a Boolean variable $st = \{1, 0\}$, to indicate whether the sensor is active (serving any user-organization), or inactive, respectively.*

A sensor will be allocated to serve an end-user either if it active (in that case, we can reuse the sensed data for multiple organizations) or, if it can be made active by external signalling [102].

Although the CSP is generally visualized as a centralized authority for provisioning cloud services, the realistic scenario involves a role-specific or region-specific distribution of service providers under a common roof. Thus, distributed cloud service providers are expressed as, $CSP = \{CSP_1, CSP_2, ..., CSP_\gamma\}$, where a total of γ number of cloud service providers are authorized. The QoS of a physical sensor node is also a significant component to identify it. It is a composite tuple that is computed including several sensor node parameters such as sensing range, transmission range, energy status, and sensing accuracy [103]. The set of currently running applications, the set of physical sensor nodes, and the set of virtual sensor nodes available within the sensor-cloud are denoted as A, S, and V, respectively.

Definition 5. *A physical sensor node is represented as a 7-tuple:*

$$s =< id, t, o, Loc, st, csp, QoS >,\ t \in T, o \in O, csp \in CSP$$

where, s.id is a sensor identification number, locally unique under s.csp and the other parameters are mentioned above in Definitions 1 *through* 4 *and in the previous paragraphs as well.*

Definition 6. *An application App running at the end of a user-organization is a 3-tuple notion expressed as,*

$$App =< A_{id}, A_{type}, A_{span} >$$

where A_{id} is a system generated unique identification for the application, A_{type} is the type of the application, and A_{span} is the span of the application, as defined in Definition 7.

Definition 7. *The span of an application, A_{span}, is a 4-tuple expressed as,*

$$A_{span} =< Loc_1, Loc_2, Loc_3, Loc_4 >$$

where Loc_1, Loc_2, Loc_3, and Loc_4, respectively, indicate the location attributes of the four vertices (in sequence) of a rectangular region that is of interest to the application.

In a practical scenario, span of an application would essentially refer to a rectangular sensor network comprising of sensor nodes that are of interest to the application. For a non-rectangular network, the nearest rectangular approximation is to be considered.

Based on the A_{type} and A_{sec}, a compatibility function f_1 is introduced to select a subset of sensor types ($T' \subset T$) and expressed as,

$$f_1(App.A_{type}, App.A_{sec}) = \{T_i : T_i \in T\} = T' \tag{3.1}$$

After the types of sensor nodes are decided for an application, the selection of sensor nodes is done using a simple allocation function, $f_{alloc}()$.

The allocation function, defined as $f_{alloc} : A \rightarrow S_1$, maps the set of applications to a subset of physical sensor nodes S_1, such that, $S_1 \in 2^S$, i.e., the set S_1 belongs to the set of subsets of S, as mentioned earlier. The principle of $f_{alloc}()$ involves a sequence of other intermediate functions $f_1()$, $g_1()$, and $g_2()$. the functionality of g_1 is to select a subset of sensor nodes of one or more given types. Thus, $g_1 : T \rightarrow 2^S$. $g_1()$ is defined as,

$$g_1(T_j) = \{s_i | s_i \in S, s_i.t = T_j\} \tag{3.2}$$

The principle of g_2 is to choose the physical sensors based on their physical locations. The chosen sensor nodes comply with the span of an intended running application. It is expressed as, $g_2 : S_1 \rightarrow S_2, S_1, S_2 \in 2^S$.

Combining the definitions of $g_1()$, and $g_2()$, the working model of $f_{alloc}()$ is obtained, and is shown below:

$$
\begin{aligned}
f_{alloc}(App) &= g_2(g_1(f_1(App.A_{type}, App.A_{sec}))) \\
&= g_2(g_1(T')) \\
&= g_2(\hat{s}, |\hat{s} \in S', S' \subset S, \hat{s}.t \in T') \\
&= \{s \in S_1, S_1 \subseteq S', circ(s.Loc, R_s) \subset App.A_{span}, \\
&\quad s.st = 0, s.QoS \geq \delta\}
\end{aligned}
$$

where R_s is the sensing radius of the sensor node, and δ is a pre-negotiated QoS threshold value with the CSP and a user-organization. After defining a physical sensor node resource and an application, mathematically, now a mapping $f_{vir} : S \rightarrow V$ is introduced and is expressed as,

$$f_{vir}(f_{alloc}(App_i)) = v_{App_i} \tag{3.3}$$

Sensor virtualization is essentially a complex logical partitioning and mapping of physical sensors into logical groups. Herein lies the significance of mapping functions that we have been used in this context.

A user-organization visualizes that each of its applications running through sensor-cloud, is mapped to a virtual sensor. Thus, $f(App) = v_{App}$. Our model considers an application App as input. After computing $f_{alloc}(App) = S_1$, f_{vir} takes S_1 as input. Therefore,

$$f_{vir}(S_1) = v_{App} | x \in S_1 \wedge x.st = 1 \tag{3.4}$$

Also, $f(App)$ is defined mathematically as,

$$f(App) = y | y \in G, f_{vir}(f_{alloc}(App)) = G = v_{App} \tag{3.5}$$

Now some interesting characteristics of the functions of the virtualization model are presented in Propositions 1 and 2.

Proposition 3.1.1. *The mapping $f(\cdot)$ from an application App_i to a virtual sensor v is injective.*

Proof. Let us assume that the co-domain of f is V. In a sensor-cloud, the virtual sensors are created on a demand-based manner. Thus, the range V' of f is never a proper subset of co-domain, i.e., $V' \not\subset V$. The CSP cannot have a virtual sensor v that is created, but not assigned to any user-organization. Thus, $\forall v \in V, \exists App_i \in A | f^{-1}(v) = App_i$. From this, it is inferred that $V' = V$.

Let us assume, $f(App_i) = v_{App_i}$. We try to allocate v_{App_i} to another application App_j. The physical sensor nodes within v_{App_i} is $S_1 = f_{alloc}(App_i)$. So, we have to allocate S_1 to App_j. But, $\forall s \in S_1, s.st = 1$. We have $f_{alloc}(App_j) \neq S_1$. Thus, the following inequalities hold.

$$f_{alloc}(App_i) \neq f_{alloc}(App_j)$$

$$\text{or, } f_{vir}(f_{alloc}(App_i)) \neq f_{vir}(f_{alloc}(App_j))$$

$$\text{or, } v_{App_i} \neq v_{App_j}$$

Thus, it can be inferred that, $v_{App_i} = v_{App_j} \Rightarrow App_i = App_j$. This completes the proof.

□

Proposition 3.1.2. *The mapping $f_{vir}(\cdot)$ of physical to virtual sensor for an application App_i is surjective (onto).*

Proof. It can be proved by the method of contradiction. Let us assume that a particular running application, App_i, requires a single physical sensor node, and f_{vir} does not have

a pre-image, i.e., $f_{vir}^{-1}(\cdot) = \emptyset$. As mentioned in Equation (3.3), $f_{vir}(f_{alloc}(App_i)) = v_{App_i}$. Thus,

$$f_{vir}^{-1}(v_{App_i}) = f_{alloc}(App_i) \Rightarrow f_{alloc}(App_i) = \emptyset \Rightarrow S_1 = \emptyset \tag{3.6}$$

This means that no physical sensor node serves application App_i. Thus, App_i is not currently served by the sensor-cloud. This completes the proof.

□

Proposition 3.1.3. *The worst case asymptotic computational complexity of $f_{alloc}(\cdot)$ for an application App_i, involving t type of sensors, $t \in T$, is O(n(t)), where n(t) is the total number of physical sensors of type t.*

Proof. From Equation 1, t of App_i, $t \in T$ can be obtained. After that $f_{alloc}()$ computes and selects sensor nodes s, such that, $s.t = t, s \in \hat{S}, |\hat{S}| = n(T)$. Thus, all sensor nodes of type t are picked up. Followed by this, functions $g_1()$ and $g_2()$ are executed. Hence the worst case asymptotic computational complexity of $f_{alloc}(\cdot)$ is $O(n(t))$. This completes the proof. □

From Proposition 1, it directly follows that every virtual sensor uniquely maps to a single application in progress and viceversa. However, a physical sensor can serve multiple applications at a given time. Using Propositions 1 and 2, an example runtime scenario is analyzed as shown in Table 3.1, consisting of 100 sensor nodes and 3 running applications. The services of the physical sensor nodes for an application App_i, at a particular time instant t, constitute a virtual sensor $v_{i,t}$. It is observed that, $v_{1,t_0} = \{s_1, s_3, s_7\}$. Thus, $f_{vir}(s_1) = v_1$. Due to the surjective property of f_{vir}, $\nexists v_i | \exists s_j \in S f_{vir}(s_j) = v_i$. Also, it is evident that, at a particular time instant t, $\forall v_i, v_j \in V$, v_i and v_j are disjoint. Thus, $\nexists s_k \in S : (s_k \in v_i) \wedge (s_k \in v_j)$.

Table 3.1: Illustration of a runtime scenario within sensor-cloud

Applications	Serving time	Resources	Virtual sensor
App_1	t_0	s_1, s_3, s_7	v_1
App_2	t_0	s_8, s_4, s_2, s_5	v_2
App_3	t_1	s_1, s_9, s_{10}	v_3
App_2	t_2	s_1, s_7, s_6	v_2
App_1	t_2	s_2, s_3, s_8	v_1
App_3	t_2	s_4, s_9, s_{10}	v_3
App_1	t_3	s_3, s_9, s_{10}	v_1

3.2 Experimental Justification for Paradigm Shift

In this subsection, a detailed analysis is made based on the performance metrics of sensor-cloud and a comparative study is performed with conventional WSNs. From a network point of view, the energy consumption of the nodes and the fault tolerance of the network are studied and analyzed. From a business point of view, a thorough evaluation of the cost-effectiveness of sensor-cloud is also done by examining the cash inflow and outflow characteristics of every actor of sensor-cloud. The experimental setup of this work is illustrated in Table 3.2.

3.2.1 Performance Metrics

As mentioned above, the following are the metrics to evaluate the performance of sensor-cloud systems compared to traditional WSNs:

(i) *Energy Consumption*: The consumption of energy E is analyzed as

$$E = E_{tr} + E_r + E_s + E_{proc} \tag{3.7}$$

where, E_{tr}, E_r, E_s, and E_{proc} are the energy expenses due to transmission, reception, sensing, and computation, respectively. The unit of energy consumption for each of these components are assumed to be same for both WSN and sensor-cloud.

Table 3.2: Experimental setup

Parameters	**Values**
Time period	5 simulation years (60 simulation months)
Deployment area	500 m × 500 m
Deployment	Uniform, random
Number of sensor nodes (N)	1000
Number of sensor owners (η_1)	5
Number of end-users (η_2)	10
Unit cost price of a node (C_s)	20 currency unit
Unit cost due to deployment (C_{deploy})	3 currency unit/sensor
Unit cost due to maintenance ($C_{maintain}$)	10 currency unit/month
Unit cost due to rent (C_{rent})	10 currency unit/month
Cost per unit usage of Se-aaS (C_{Se-aaS})	10 currency unit/month
Communication range	[50, 100] m
Transmission energy (E_{tr})	7 nJ/bit
Computation energy (E_{proc})	5 nJ/sec
Sensing energy (E_s)	6 nJ/event
Time interval for nodes being faulty (Ω)	5

(ii) *Fault Tolerance*: Fault-tolerance, $\mathcal{F}$ of a network is defined as the total number of non-faulty nodes present in the network at a particular time. Mathematically,

$$F_t = F_{t-1} - P_f \times F_{t-1}, \quad \mathcal{F}_0 = N \tag{3.8}$$

where N and P_f are the total number of operative nodes initially present in the network and the percentage of faulty nodes, respectively. Here, we assume that, the number of faulty nodes change with time, However, the percentage of faulty nodes at a particular time is constant. For example, if there are 100 nodes initially, after unit time 5 nodes will fail and 95 would remain. Then, at the next time instant, 4.75 would fail and 90.25 would remain approximately, and so on.

(iii) *Cost-effectiveness*: For evaluating cost-effectiveness, an analysis of flow of cash for every actor and a WSN user is studied. Lines of cumulative cost along the

negative ordinate represents a cash outflow CO from the actor, whereas the one along the positive ordinate represents cash inflow CI to the actor. The costs due to deployment, maintenance, and rent are denoted by C_{deploy}, $C_{maintain}$, and C_{rent} respectively.

(a) Sensor-owner: For a sensor-owner, the flow of cash is governed by the following Equations:

$$CO_{sensor-owner} = n_1 \times (C_s + C_{deploy}) \tag{3.9}$$

$$CI_{sensor-owner} = n_1 \times C_{rent} \tag{3.10}$$

where, n_1 is the number of sensors registered by the sensor-owner. C_s is the unit cost price of a sensor node.

(b) End-user:

- WSN user: For a WSN user,

$$CO_{wsn} = n_2 \times (C_s + C_{deploy} + C_{maintain}) + n_3 \times C_{deploy} \tag{3.11}$$

where n_2 and n_3 are the total number of sensor nodes in the WSN and the number of faulty nodes, respectively. A WSN user is basically served in terms of the sensed data, and there is no cash inflow for such user.

- Sensor-cloud end-user: From a sensor-cloud end-user point of view, the cash outflow is expressed as follows:

$$CO_{end-user} = n_4 \times C_{Se-aaS} \tag{3.12}$$

where n_4 is the total number of sensors nodes of which the user has obtained service in a particular month. C_{Se-aaS} is the cost incurred per unit

usage of Se-aaS per month.

(c) CSP[1]: For a CSP, the monthly inflow and outflow of cash are also analyzed with the help of the following equations.

$$CO_{csp} = \eta_1 \times CI_{sensor-owner} + \Omega \times n_5(C_{deploy} + C_{maintain}) \tag{3.13}$$

$$CI_{csp} = \eta_2 \times CO_{end-user} \tag{3.14}$$

where η_1, η_2, and Ω are the total number of registered sensor-owners, total number of end-users, and the monthly time interval after which maintenance and deployment activities are performed by the CSP, respectively. The number of faulty sensor nodes after Ω interval of time is denoted by n_5.

3.2.2 Explanation of Parameters

The parameters used for this work have been selected on the basis of prior related works [104–106]. The values of these parameters are set as per the related works in this domain that are most cited [107–111].

3.2.3 Approach Taken

For the performance evaluation and analysis of the network parameters, i.e., the energy consumption and fault tolerance, a simulation is conducted over 5 simulation years (60 simulation months). A uniform random deployment of 1000 sensor nodes is conducted across a deployment area of about 500m × 500m. For simulation of the functionality of sensors, every sensor is assigned a communication range, and rates of energy consumption due to transmission, computation, and sensing. Under this setup, the entire environment

[1]It is to be noted that the third actor in a sensor-cloud infrastructure is the sensor-cloud administrator. But, the administrator is essentially a non-human actor and hence, does not participate in the economics of the model. However, although the CSP is not a potential actor of the infrastructure, it is the most significant business entity of sensor-cloud, and hence the cash flow analysis of the CSP is of high interest.

is simulated for the afore-mentioned time and is repeated for 50 iterations. The mean of the data obtained for the different metrics, as discussed in subsection , is plotted and presented.

For the analysis of the cost-effectiveness of the two paradigms, η_1 number of sensors-owners and η_2 number of end-users are considered as per Table 3.2. For the purpose of evaluation, unit costs for purchase, deployment, maintenance, rent, and for consumption of Se-aaS are considered as per Table 3.2 and the sensor virtualization, allocation, and de-allocation are simulated for the simulation period. The demand of end-users are varied randomly following a Poison distribution with mean sensor requirement of 5 and for a period of 1 hour.

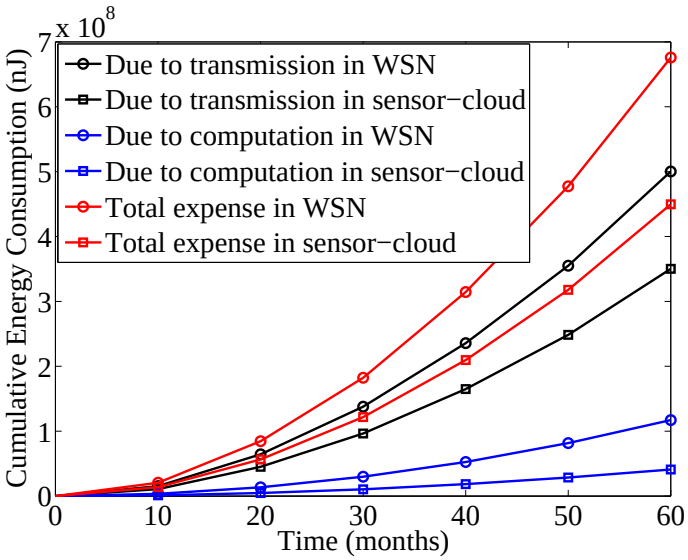
For the profit analysis of the CSP, 6 different experiments were conducted by varying the number of end-users from 1 to 20 for a theoretical case study. Each experiment is repeated over 50 iterations to determine the mean value that is eventually plotted.

3.2.4 Performance Analysis

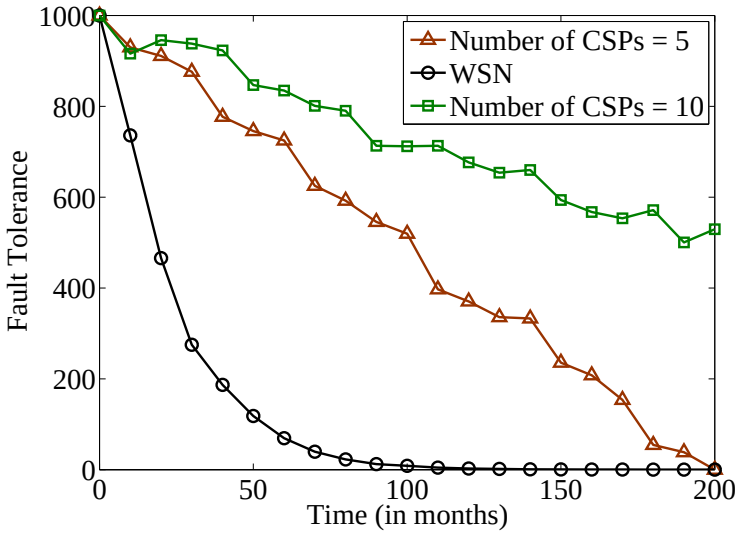
In this subsection, the performance of sensor-cloud platforms are analyzed thoroughly. The analysis is performed separately for each of the metrics indicated in subsection 3.2.3.

Firstly, the performance of a single sensor node is investigated in terms of its energy consumption. Figure 3.1(a) shows the cumulative energy expenses of a sensor node in terms of sensing, computation, and transmission of packets. In a WSN, intra-network communication occurs by repetitive multi-hop communication followed by transmission of packets to a data center. However, in a sensor-cloud environment, energy expenses due to transmission are mainly attributed to reach the cloud platform via multi-hop communication. Communication among sensor nodes is very rare (or does not occur), and, hence, large amount of energy is conserved. Moreover, unlike WSN, a particular sensor node does not necessarily serve a user-organization, even if it is application-compatible. Periodic scheduling is followed by the CSPs among multiple application-

compatible sensor nodes with a view to distribute load and conserve resources. The figure presents that sensor-cloud achieves 36.68% decrease in energy consumption, compared to that of a WSN.



(a) Comparative analysis for cumulative energy consumption

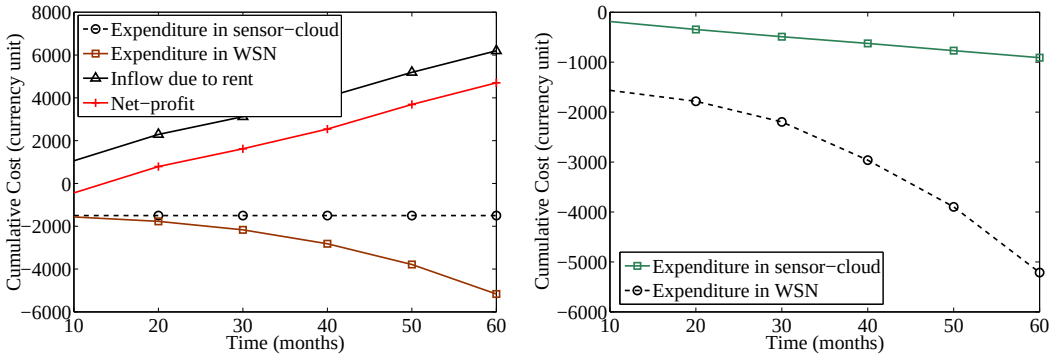


(b) Comparative analysis of cumulative fault tolerance

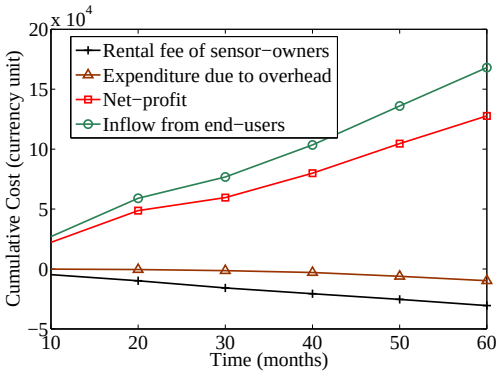
Figure 3.1: Comparative performance analysis of sensor-cloud and WSN

Next, the performance is analyzed from a network point of view in terms of fault

tolerance of a network. Figure 3.1(b) illustrates a comparative study of fault-tolerance in WSNs and sensor-cloud. Fault-tolerance is a major cause of concern in WSN. Assuming a specific fault-tolerance rate, a WSN reaches a dead state unless a redeployment scheme is considered atleast once during its lifetime. On the other hand, sensor-cloud involves multiple service providers who can render the best possible sensor nodes at any point of time to address fault-tolerance of resources. Once a user-organization's application demand is recognized, the cloud infrastructure allocates a CSP, which can best serve the user-organization in terms of energy level, accuracy, QoS, compatibility of sensor node specification, and location specific feasibility. Figure 3.1(b) indicates the increase in network performance with the increase in the number of CSPs.



(a) Analysis of cost-effectiveness for sensor-owner (b) Analysis of cost-effectiveness for end-user



(c) Analysis of cost-effectiveness for CSP

Figure 3.2: Comparative cost-effectiveness analysis of sensor-cloud and WSN

Now, a comparative study of various sensor-cloud actors and a WSN-user is investigated from a profit perspective. Figure 3.2(a) illustrates the perspective of a sensor-owner, who simply owns and deploys his/her sensor nodes within the sensor-cloud environment. In a WSN, the sensor-owner is eventually the WSN user. It is the responsibility of a WSN user to buy, deploy, maintain and redeploy sensor nodes, as and when needed. The cumulative cash outflow of a WSN user and a sensor-owner are indicated over time. The cash outflow of the sensor-owner occurs only once during the network lifetime, due to ownership and deployment of sensor nodes. The inflow of the sensor-owner is measured by the monthly rental fee that it obtains from the CSP. Finally, the overall profit of the sensor-owner is also denoted in the figure. Figure 3.2(a) depicts that a single sensor-owner can reduce 33.83% of cash-outflow in sensor-cloud environment, compared to a WSN.

The perspective and profit analysis for an end-user is different. Figure 3.2(b) illustrates a comparison with respect to the cost incurred by an end-user. End-user of a WSN is responsible for several jobs involving maintenance and overhead. However, in sensor-cloud, an end-user perceives a sensor as an instantaneous service (just like electricity, water), rather than as a hardware. Thus, s/he is liable to pay for only those units of Se-aaS that s/he has actually consumed. The profit of an end-user cannot be measured in terms of monetary units as it is relevant in terms of countable units of Se-aaS. The figure shows an average of 14.72% decrease in the expenditure of an end-user-organization.

In Figure 3.2(c), the profit perspective of a CSP within sensor-cloud is depicted. As mentioned earlier, the CSP has to pay a monthly rental-fee to each sensor owner, from whose resources s/he renders services to the end-users. Figure 3.2(c) illustrates the cumulative cash outflow for multiple sensor-owners. Some amount of cash outflow occurs due to the periodic maintenance and redeployment of the physical sensor nodes. The principal source of cash inflow is the end-users, who use the on-demand service and pay to the CSP accordingly. The net profit of the CSP is also indicated over time.

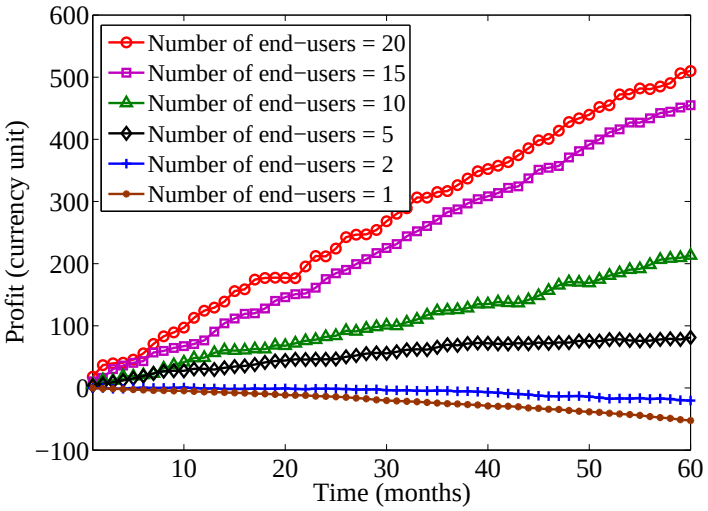


Figure 3.3: Profit analysis of CSP in a sensor-cloud

It is worthy to mention that a sensor-cloud can perform, only when the required resource type is actually available. Therefore, some sensor nodes have to be deployed by some sensor-owner. If a sensor-type is quite uncommon or less-demanded, it involves high overhead and maintenance cost compared to that of usage. Thus, if the number of end-user-organizations demanding for a particular resource type T_i is typically low, the performance of sensor-cloud reduces almost similar to that of WSN. Figure 3.3 reflects a scenario where end-user organizations demand a specific resource type. As the number of such users reduces, the profit of CSP reduces, eventually turns into loss. In such cases it is better to deploy a customized sensor network on behalf of the end-user-organizations.

3.3 Summary

The proposed Chapter presents a theoretical modeling of virtualization for sensor-cloud environment. A detailed description of the perspectives of end-user organizations and the CSP are illustrated and analyzed. The process of mapping an application to its physical resources and the procedure for virtualization of the resources are also discussed. Finally, a comparative evaluation of performance between sensor-cloud and WSN is presented.

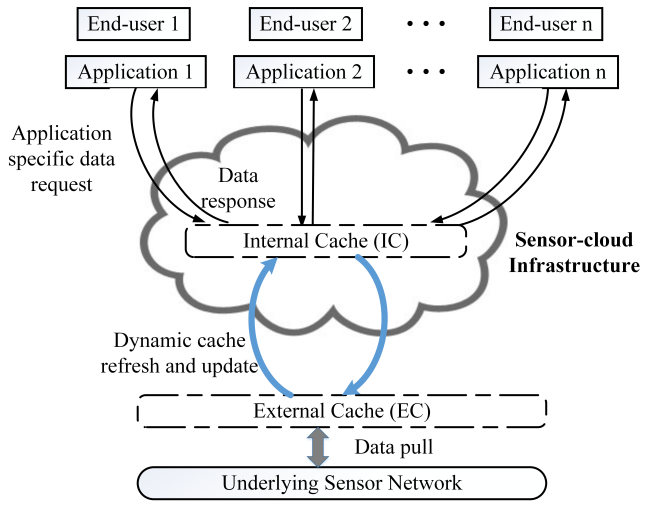
Results show that sensor-cloud accomplishes better performance compared to WSN in most of the cases. However, in some exceptional situations, sensor-cloud was found not to perform reasonably better than traditional WSNs. Thus, a paradigm shift is suggested from the existing WSN-based technology to a sensor-cloud platform as it would be beneficial in terms of performance, usability, and profit.

Chapter 4

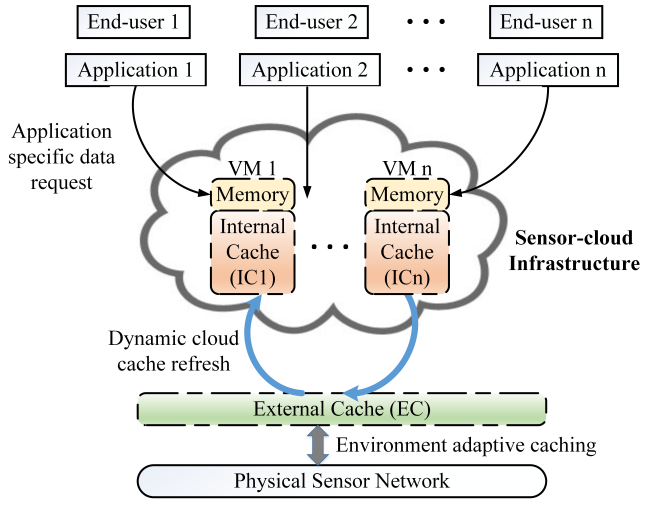
Dynamic and Adaptive Data Caching Within Sensor-cloud

In the architecture of sensor-cloud, as mentioned in Chapter 1, it is observed that sensor-cloud directly accesses the underlying physical nodes, as per the application demand. Therefore, as illustrated in Figure 4.1(a), whenever a physical node becomes an active component of a VS, it is selected, and data is fetched from it. Thus, if a running application has a high rate of demand from its VSs, the component nodes of the VSs engage themselves in continuous data communications with the sensor-cloud. Thus, the continuous process of data retrieval and data logging occurs with time. However, in a real-life scenario involving a practical sensor-based application, the change in the environment may be the following – (a) sufficiently slow, thereby resulting in a negligible rate of change of data of the sensed attributes (e.g. applications monitoring air temperature/pressure) or (b) very fast, thereby resulting is a faster rate of change of environment (e.g. applications in vehicular networks). For a small or moderate change in the environment, the sensed data remains almost unaltered. Thus, continuous transmission of the unaltered sensed data leads to unnecessary energy loss due to redundant transmissions. Moreover, it takes a toll on the lifetime of the individual sensor nodes, as well as the lifetime of

the network. The significance of data caching in sensor-cloud becomes important in this context. Data caching should be adaptive to the rate of change of the environment.



(a) Existing architecture



(b) Cache-enabled architecture

Figure 4.1: Existing and proposed architectures of caching in sensor-cloud

4.1 Contributions of the Chapter

As mentioned above, in real-life scenarios, the change of the physical environment in terms of sensor reading is imperceptible, when diagnosed very frequently, i.e., the change of the sensed data is generally negligible within a small time interval. The contribution of this Chapter is to design an optimal caching mechanism within sensor-cloud to obtain resource efficiency in terms of energy, and network lifetime. The proposed data caching mechanism is dynamic, and is adaptive to the change of the physical environment, thereby preserving the accuracy of information, and conserving the network resources, simultaneously. The user requests for the sensed data are served from the cache when the change of the physical environment is gradual. The work determines an optimal caching interval beyond which fresh data is requested from the physical sensors, and cached again. Thus, the proposed solution can significantly reduce the expenditure of transmission energy, and enhances the network lifetime.

4.2 Proposed Architecture for Caching

In this Section, the proposed cache-enabled architecture of sensor-cloud infrastructure is presented. As illustrated in Figure 4.1(b), two caches are considered – the *primary cache* or the *External Cache* (*EC*), and the *secondary cache* or the *Internal Cache* (*IC*). The EC is external to the cloud system and is an interfacing cache that resides between the cloud, and the underlying network. EC is sensitive to the change in the physical environment, thereby dynamically updating and refreshing its content to retain synchronization with the current state of the physical environment. The working principle of EC is based on the expected rate of change of the environment. The IC, on the other hand, is located within the cloud, and is updated in coherence with the dynamism of the EC. IC obtains the information from EC and synchronizes it with the variable demand rate of the end-user applications.

4.2.1 Rationale behind Two Cache Units

The position of the IC is within a Virtual Machine (VM) allocated to an end-user. When an end-user E_1 requests for Se-aaS from a particular set of sensors for the first time, the data query is redirected to the EC. If the data has been previously cached, and the cached data has not yet expired, the corresponding data is fetched from the EC and transmitted to the VM (VM_1) of E_1. The data received at VM_1 is further cached within IC_1 so that any subsequent query generated by E_1 involving data retrieval from the same set of sensors may be obtained from IC_1. Now, if another end-user E_2 wishes to access the sensor data from the same set of sensors, the corresponding data query is fed with cached data retrieved from EC subjected to the recency of the data.

4.3 Model of the External Cache

In this Section, the working principle of the EC is discussed. The steps are also indicated in algorithm format at the end of this Section. The goal of EC is to determine an optimal caching interval Δt, based on the history of the sensed data from a particular sensor s. Thus, given the last k readings $R_1 = \{r_1, r_2, ..., r_k\}$ at timestamps $T_1 = \{t_1, t_2, ..., t_k\}$, the goal is to find k', such that the caching interval $\Delta t = k' - k$ is maximum, subject to certain constraints.

Initially, some of the required metrics of the work are defined.

Definition 8. *The current memory m of EC, at time t ($m(t)$), is a k tuple, where k is a pre-negotiated system value. $m(t)$ is expressed as,*

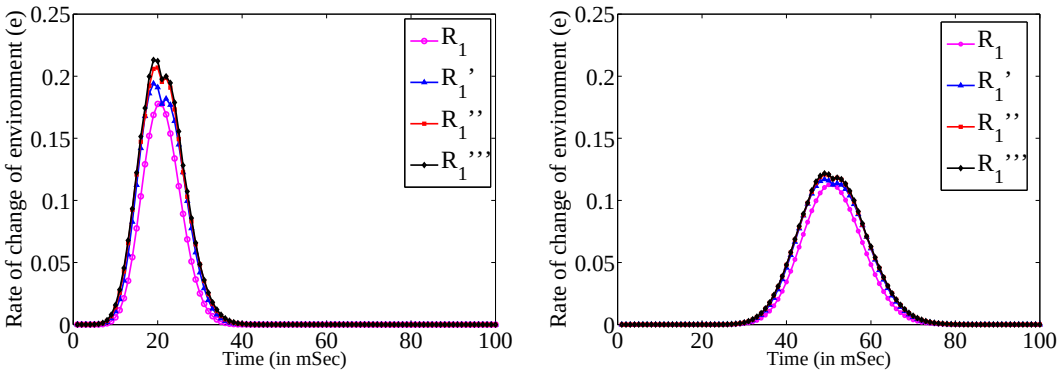
$$m(t) = \{(r_1, t_1), (r_2, t_2), ..., (r_k, t_k)\}, r_i \in R_1, t_j \in T_1 \tag{4.1}$$

Thus, at a given time, the mean rate of change of environment, e, is obtained as,

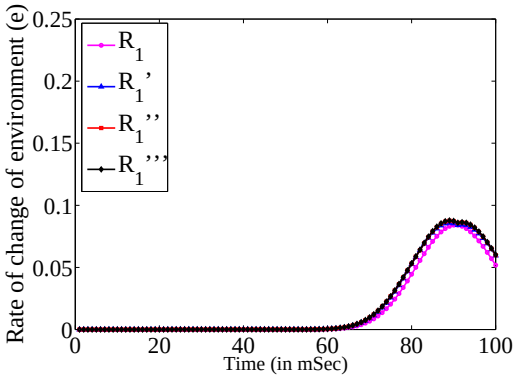
$$e = \frac{\sum_{i=2}^{k} \mid m(t).r_i - m(t).r_{i-1} \mid}{\sum_{i=2}^{k} m(t).t_i - m(t).t_{i-1}} \tag{4.2}$$

Definition 9. *The expected rate of change of environment, e, for a particular physical sensor, is based on the i^{th} degree of the rate of change of the environment, $1 \leq i \leq k$. Thus,*

$$E(e) \propto \left(\frac{dR_1}{dT_1} + \frac{d^2R_1}{dT_1^2} + ... + \frac{d^kR_1}{dT_1^k} \right) \tag{4.3}$$



(a) $R_1 \sim P(100, 20)$ (b) $R_1 \sim P(100, 50)$



(c) $R_1 \sim P(100, 90)$

Figure 4.2: Analysis of rate of change of physical environment with time

For the sake of justification of Equation (4.3), an experiment was performed for

the 100 time instants (in milli second), and the data were approximated to follow a Poisson distribution [112] with $n = 100$, and varied mean $\mu = (20, 50, 90)$, as shown in Figures 4.2(a), 4.2(b), and 4.2(c), respectively. It is observed that the rate of change of the environment is significant up to the 2^{nd} order, beyond which the rate of change is minimal. Hence, for the sake of simplicity, Equation (4.3) is revised as,

$$E(e) = c_1 \frac{dR_1}{dT_1} + c_2 \frac{d^2 R_1}{dT_1^2} \tag{4.4}$$

$E(e)$ should be less than a threshold value $e_{threshold}$, beyond which re-caching should occur to maintain accuracy of data. Having estimated the expected rate of change of environment, now the constraint is designed to satisfy energy efficiency of the underlying physical sensor network. Assuming α and β as the respective energy cost associated with per unit of communication (both transmission and reception), and per state transition, the cost $\mathcal{C}$ incurred for obtaining non-cached data directly from the physical network is inferred as,

$$\mathcal{C} = 2\alpha E_{tr} + \beta E_{st} \tag{4.5}$$

where E_{tr}, and E_{st} are the energy expended due to communication, and state transition, respectively. A factor of 2 is associated with E_{tr}, as it involves the transmission of a signal to the respective sensor and obtaining data packet from it. E_{st} is consumed mainly due to transition of the state of the node from passive (idle or not transmitting) to active (transmitting) [113]. The cumulative energy expenditure till the next time instant for caching (k') is expressed as,

$$\begin{aligned} f_2(k') &= \sum_{t_i=2}^{k'} \Big(2\alpha E_{tr} + \beta E_{st} + (t_i - t_{i-1})\gamma E_s \Big) \\ &= k'\mathcal{C} + \sum_{t=2}^{k} (t_i - t_{i-1})\gamma E_s + (k' - k)\gamma E_s \end{aligned} \tag{4.6}$$

where γ is the energy cost associated with per unit sensing activity, and E_s is the energy expended due to sensing. Now, a multi-objective optimization problem is formulated as,

$$\begin{aligned} &\textit{Maximize} \;\; \Delta k = k' - k \\ \text{i.e., } &\textit{Minimize} \;\; f_1(k') = \frac{1}{\Delta k} = \frac{1}{k' - k} \end{aligned} \tag{4.7}$$

$$\textit{Minimize } f_2(k') \tag{4.8}$$

$$\text{subject to, } f_3(T_1) = c_1 \frac{dR_1}{dT_1} + c_2 \frac{d^2 R_1}{dT_1^2} < e_{threshold} \tag{4.9}$$

Equation (4.9) accounts for the adaptability of EC. Thus, using the method of scalarization, and combining Equations (4.7) through (4.9), the resulting minimization problem is stated as,

$$\begin{aligned} &\textit{Minimize} \;\; L(k', T_1) = k'\mathcal{C} + \sum_{t=2}^{k} (t_i - t_{i-1})\gamma E_s + (k' - k)\gamma E_s \\ &-\alpha_1 \left(\frac{1}{k' - k} \right) - \alpha_2 \left(c_1 \frac{dR_1}{dT_1} + c_2 \frac{d^2 R_1}{dT_1^2} - e_{threshold} \right) \end{aligned} \tag{4.10}$$

Therefore,

$$\frac{\delta L}{\delta k'} = \mathcal{C} + \gamma E_s + \frac{\alpha_1}{(k' - k)^2} = 0, \tag{4.11}$$

$$\frac{\delta L}{\delta T_1} = \alpha_2 \left[c_1 \frac{d^2 R_1}{dT_1^2} + c_2 \frac{d^3 R_1}{dT_1^3} \right] = 0 \tag{4.12}$$

Using Equations (4.10) through (4.12), the optimal value of k'^* obtained. Thus, EC defines a optimal caching interval $\Delta k^* = k'^* - k$ that minimizes the energy consumption of the physical nodes, and is adaptive to the dynamics of the physical environment. The schedule for re-caching commences at the end of every caching interval. Data requests,

within the caching interval, are served from the EC itself.

Input:

1. Current memory m of EC, at time t: $m(t)$.

Output:

The next caching instant at EC: k'.

Compute the mean rate of change of environment e (Equation 4.2)
Compute the expected rate of change of environment $E(e)$ (Equation 4.4)
while $f_3(T_1) = c_1 \frac{dR_1}{dT_1} + c_2 \frac{d^2R_1}{dT_1^2} < e_{threshold}$ **do**
    for $j = 1$ *to* k **do**
        Compute Δk and store in a temporary variable
        Update temporary variable if Δk is bigger
    end
    Output temporary variable
end

Algorithm 2: Algorithm with EC

4.4 Model of the Internal Cache

This Section illustrates the working model of the IC. The steps are also indicated in algorithm format at the end of this Section. The IC primarily handles the data requests from the user-applications, and decides to serve the data either directly from the cache or re-caches the data from the EC and then serves it. If the sequence of the data provisioned to the end-users at $\{t_i\}$ be d, $d = \{d_i\}, 1 \leq i \leq p$. Initially, the first p readings from EC are directly fed into IC for preparing the history. Thus, the expected rate of change of EC, e', is given as,

$$E(e') = \frac{\sum_{j=2}^{k} \mid d_j - d_{j-1} \mid}{\sum_{i=2}^{k} t_i - t_{i-1}} \tag{4.13}$$

Definition 10. *The mean accuracy $\hat{\mathcal{A}}$ of data provisioning is defined as the inverse of*

the Root Mean Square Error (RMSE) of the sensor readings at EC and IC, evaluated for the previous j time instants. Thus, mean accuracy of a data at time t is expressed as,

$$\hat{\mathcal{A}} = \sqrt{\frac{j}{\sum_{i=t-j+1}^{t} (m(t).r_i - d_i)^2}} \tag{4.14}$$

For accurate servicing of data, $\sum_{i=t-j+1}^{t} (m(t).r_i - d_i)^2 \rightarrow r$, where r is an extremely small value, $r \neq 0$. Assuming that data has been cached within IC at time k, the minimization problem for IC is expressed as,

$$\textit{Maximize } (k'' - k) \text{ i.e., } \textit{Minimize } g_1(k') = \frac{1}{k'' - k} \tag{4.15}$$

where k'' is the next scheduled instant for caching within IC, subjected to the constraints,

$$\sum_{i=k''-j+1}^{k''} (m(k'').r_i - d_i)^2 - r \simeq 0 \tag{4.16}$$

Thus, using Equations (4.15) and (4.16), the solution set for k'' is expressed as,

$$k \in \min_{k<h<g} \left\{ \max \left\{ \sum_{i=h-j+1}^{h} (m(h).r_i - d_i)^2 \right\} \right\} \tag{4.17}$$

$$\sum_{i=g-j+1}^{g} (m(g).r_i - d_i)^2 > r \tag{4.18}$$

After obtaining k'', the dynamic caching of IC can be executed by maintaining data

provisioning accuracy, simultaneously.

Input:

1. Sequence of the data provisioned to the end-users at $\{t_i\}$: d,

$d = \{d_i\}, 1 \leq i \leq p$.

Output:

The next caching instant at IC: k''.

Compute the expected rate of change of environment in EC $E(e')$ (Equation 4.13)

Compute the mean accuracy of data provisioning $\mathcal{A}$ (Equation 4.14)

while $\sum_{i=k''-j+1}^{k''} (m(k'').r_i - d_i)^2 - r \simeq 0$ **do**

  for $j = 1$ *to* d **do**

    Compute $k'' - k$ and store in a temporary variable

    Update temporary variable if $k'' - k$ is bigger

  end

  Output temporary variable

end

Algorithm 4: Algorithm with IC

4.5 Theoretical Analysis

Proposition 4.5.1. *If λ and $\hat{d}$ are, respectively, the non-uniform demand rate, and the data provisioning rate for p time instants, the mean accuracy $\hat{\mathcal{A}}$ has a lower bound $\hat{\mathcal{A}}_{min}$.*

Proof. A non-uniform demand sequence is characterized by, $\lambda = \{\lambda_j\}, 1 \leq j \leq p$, where $\lambda_i - \lambda_{i-1} \neq \lambda_l - \lambda_{l-1}, i \neq l, 2 \leq i, l \leq p$. Also, $d = \{d_j\}, 1 \leq j \leq p$. Let us assume that $\hat{\mathcal{A}}$ has no lower bound. Therefore, since θ^{th} time instant caching did not occur to preserve accuracy. Thus, using Equation (4.14), it can be obtained that,

$$\sum_{i=t-j+1}^{t} (m(t).r_i - d_\theta)^2 > \sum_{i=t-j+2}^{t+1} (m(t).r_i - d_\theta)^2 \tag{4.19}$$

Thus, for a very high value γ, at a particular time t',

$$\sum_{i=t-j+1}^{t} (m(t).r_i - d_\theta)^2 \rightarrow \gamma \tag{4.20}$$

However, as $\gamma \gg r$, $\exists g$ as per Equation (4.18). Thus, caching must have occurred at least once to reflect the data of IC, as $d_{\theta+1}$. Naturally, $\{(m(t).r_i - d_\theta)^2\}$ is an increasing sequence till g, from which it can be inferred that $\hat{\mathcal{A}}_{min}$ is bounded by a lower value. This concludes the proof. □

Proposition 4.5.2. *Assuming k and k' as the previous, and the next instant of caching within EC, respectively, $\Delta k = k' - k$ always possesses a lower and an upper bound as Δk_{min}, and Δk_{max}, respectively.*

Proof. Δk is minimum, when $E(e) \gg e_{threshold}$, i.e. when the environment is highly changing . Thus,

$$\alpha_2(c_1 \frac{dR_1}{dT_1} + c_2 \frac{d^2R_1}{dT_1^2} - e_{threshold}) \rightarrow h \tag{4.21}$$

where h is a value of high magnitude, and Equation (4.21) becomes the dominant constraint. Thus, as per Equation (4.10), $L(k', T_1)$ will have its minimum value at k_{new}, $k_{new} \rightarrow k$. Thus, $\Delta k = k_{new} - k = \Delta k_{min} \simeq 0$. On the other hand, for a gradually changing environment, $c_1 \frac{dR_1}{dT_1} + c_2 \frac{d^2R_1}{dT_1^2} \ll e_{threshold}$. However, if $\{q_i\}$ be an increasing sequence, $\sum\limits_{i=j}(q_i - q_{i-1})\gamma E_s > \sum\limits_{i=j+1}(q_i - q_{i-1})\gamma E_s$. Thus, $L(k', T_1)$ obtains its maximum value at k_{new}, where $k_{new} - k \not\rightarrow 0$, and $\Delta k_{max} = k_{new} - k = b$, where b is of significant magnitude. This concludes the proof. □

4.6 Performance Evaluation

This Section presents the overall evaluation of the performance of the proposed dynamic adaptive caching mechanism. The experiments are performed in Matlab, and the experimental setup is illustrated in Table 4.1.

Table 4.1: Experimental setup

Parameters	Values
Deployment area	500 m × 500 m
Deployment	Uniform, random
Number of nodes	100
Transmission energy (E_{tr})	7 nJ/bit
Energy due to state transition (E_{st})	30 nJ
Sensing energy (E_s)	6 nJ/event
Number of time instants	100

4.6.1 Explanation of Parameters

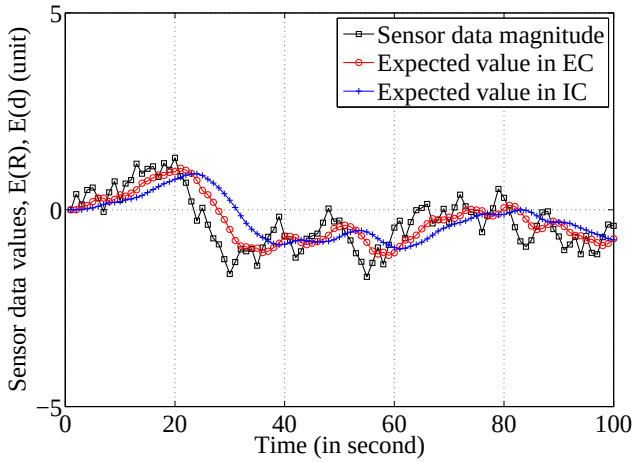
The parameters used for this work have been selected on the basis of prior related works [104–106]. The values of these parameters are set as per the related works in this domain that are most cited [107–111].

4.6.2 Approach Taken

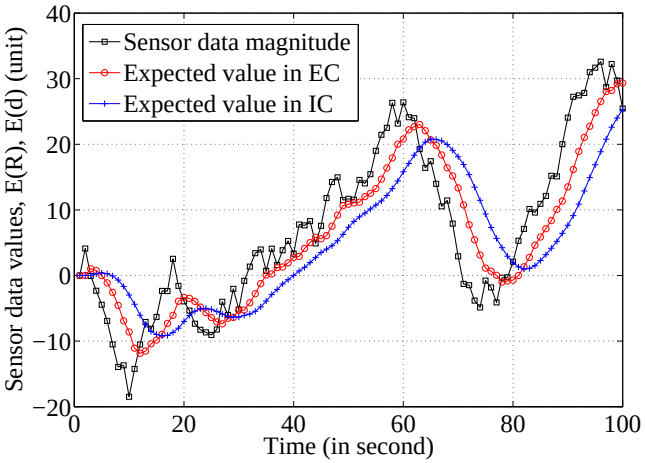
For the performance evaluation and analysis of the network parameters, i.e., the energy consumption and network lifetime, a simulation is conducted over 100 time instants. A uniform random deployment of 100 sensor nodes is conducted across a deployment area of about 500 × 500. For simulation of the functionality of sensors, every sensor is assigned a communication range, and rates of energy consumption due to transmission, computation, and sensing as per Table 3.2. Under this setup, the entire environment is simulated for the afore-mentioned time and is repeated for 50 iterations. The mean of the data obtained for the different metrics is plotted and presented.

To justify the correctness of the computation of $E(e)$ and $E(e')$, as mentioned in Equation (4.4) and (4.13), respectively, an experiment is performed on randomized sensor readings, over 100 instants of time. The experiment is repeated for stable (gradually changing), and unstable (fast and sudden changing) scenarios of the physical environment, as shown in Figures 4.3(a), and 4.3(b), respectively.

For the analysis of adaptiveness and dynamism of caching, a set of random sensor values are enforced within a logical pipeline and data is cached as per the proposed algorithms for EC and IC. This whole experiment is continued for 100 time instants and the data is eventually plotted in Figure 4.6.

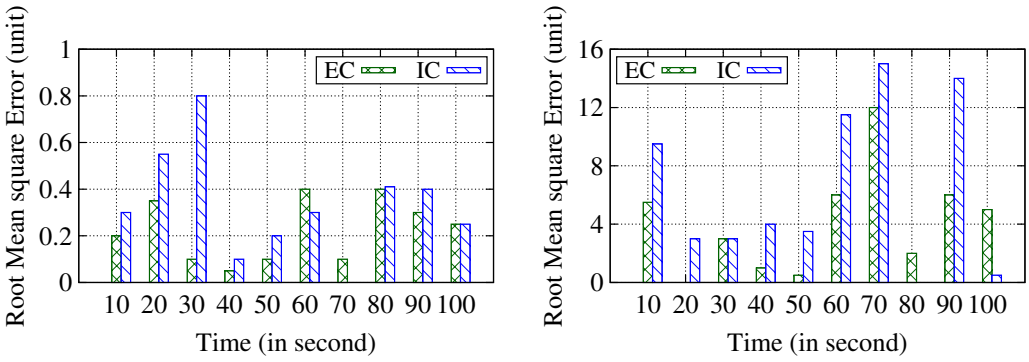


(a) Stable environment



(b) Unstable environment

Figure 4.3: Study of the expectation of the sensed data with time



(a) Stable environment

(b) Unstable environment

Figure 4.4: Analysis of the RMSE in computation of the expectation of sensed data

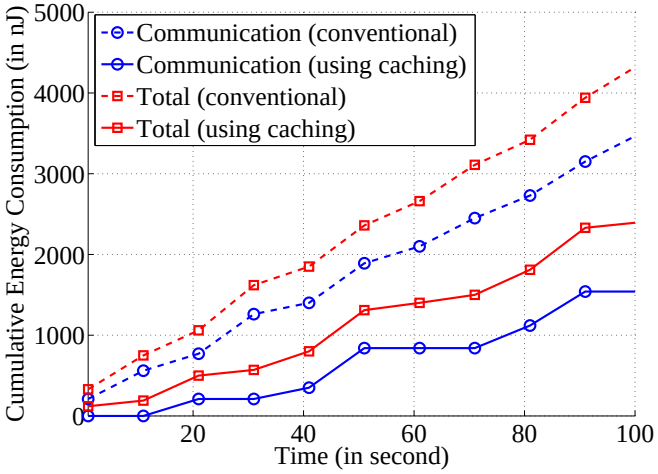
4.6.3 Performance Analysis

Figure 4.3(a) exhibits a gradual, and slow-paced change in the environment, thereby leading to close estimation of the sensed values in EC, and IC. On the other hand, Figure 4.3(b) shows the rate of change of the data for a turbulent environment leading to little deviations in the process of estimation.

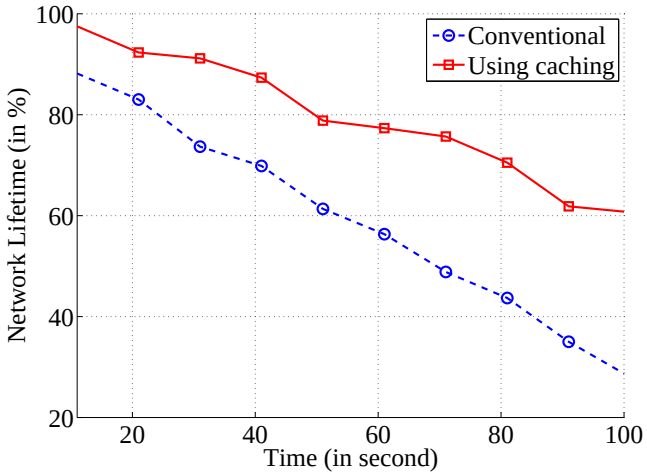
The accuracy of computation, as given in Definition 10, is evaluated in terms of the computation of the RMSE for the above two scenarios. Figure 4.4(a) clearly shows the RMSE obtained for expecting the rate of change of environment in a stable condition. The error obtained for the first few time instants are initially high, due to the gradual learning or adaptiveness of the caching process, after which the error falls to a negligible value. For an unstable environment, as depicted in Figure 4.4(b), the RMSE in expecting the change of the environment rises and falls sporadically based on the rate of change of the environment.

The energy efficiency of the proposed caching mechanism within EC is studied in Figure 4.5(a) in terms of the cumulative energy consumption, $\mathcal{E}$, expressed as,

$$\mathcal{E}(j) = \sum_{t_i=1}^{j} \Big(2\alpha E_{tr} + \beta E_{st} + (t_i - t_{i-1})\gamma E_s \Big) \tag{4.22}$$



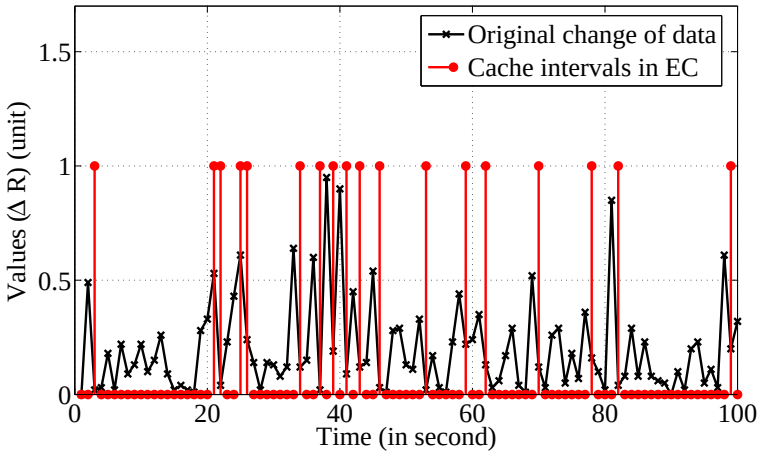
(a) Analysis of energy efficiency



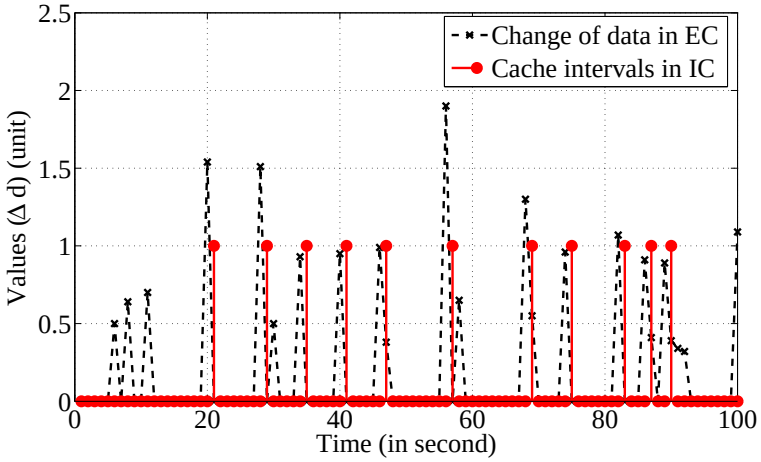
(b) Analysis of network lifetime

Figure 4.5: Overall analysis of the network resources

$\forall 1 \leq j \leq 100$. Conventional techniques generally follow a periodic data transmission scheme [114], thus, consuming more energy, compared to those that follow caching in which the transmission of the physical nodes is sensitive to significant change of the environment. Mathematically, the consumption of energy of individual sensor nodes is also reduced by 37.1%. This, in turn, enhances the network lifetime $\mathcal{N}$ of the nodes, as shown in Figure 4.5(b). $\mathcal{N}$ is computed as,



(a) Caching in EC



(b) Caching in IC

Figure 4.6: Analysis of adaptiveness and dynamism of caching

$$\mathcal{N}(t) = \frac{\mathcal{N}_{max} - \mathcal{E}(t)}{\mathcal{N}_{max}} \times 100\% \tag{4.23}$$

where $\mathcal{N}_{max}$ is assumed to be $6000nJ$. Figure 4.5(b) clearly indicates the improvement of the network lifetime by using caching mechanisms, compared to the conventional ones. The network lifetime was observed to increase by 48.43%.

Figure 4.6 examines the performance of the proposed caching techniques in EC, and

IC. The stem plots in Figures 4.6(a), and 4.6(b) indicate the scheduling of caching in both EC and IC, respectively. A 0 indicates no caching, and a 1 indicates re-caching. Based on the substantial change in the readings of the physical sensors, the caching is dynamically performed within EC, as shown in Figure 4.6(a). The caching within IC depends on the rate of change of data within EC. However, the frequency of caching is much less in IC, compared to EC. Over 100 time instants, the deviation of the cached data was observed, compared to the original data and it is found that, due to caching, data within EC, and IC are respectively maximally deviated by 7.79%, and 14.09%, only. Therefore, in the worst case, the demand at $t = 100$ is served with a data of $t = 85.91$, thereby accounting for a minimum of 85.91% recentness of the provisioned data.

4.7 Summary

This Chapter introduces an adaptive caching mechanism to prevent the unnecessary depletion of network resources. The Chapter proposes external, and internal caches, which dynamically and optimally cache the sensor data, based on the variable rate of change of the physical environment, thereby achieving reduction in redundant transmissions of data packets from the underlying sensor network to the sensor-cloud.

Chapter 5

Dynamic and Optimal Pricing Scheme for Se-aaS

As sensor-cloud infrastructure is the extension of cloud computing, it abides by the features that are intrinsic to cloud based environments. A cloud platform generally conforms with a pay-per-use model [51, 115], in which the end-users pay only for those resources that they have utilized. Within sensor-cloud infrastructure, end-users utilize the physical sensors and the cloud infrastructure as per their demand and pay as per their usage, to the CSP. Thus, it is necessary to develop a pricing scheme for Se-aaS to quantify the usage of the end-users and charge them accordingly. The profit incurred from the payment obtained from the end-users is not only enjoyed by the CSP, but is also shared among the several sensor owners whose physical sensors are registered within sensor-cloud [19].

This Chapter focuses to design a dynamic and optimal pricing scheme, specifically for Se-aaS. Currently, different pricing models are suggested for the various service oriented architectures (SOAs), namely Infrastructure-as-a-service (IaaS) [116, 117], Platform-as-a-Service (PaaS) [118], and Software-as-a-Service (SaaS) [119]. However, these pricing models have been designed for homogeneous types of services such as infrastructure,

platform and software. On the contrary, Se-aaS follows a heterogeneous SOA in which service is provisioned in the form of hardware as well as infrastructure to the end-users. Therefore, there arises a necessity to design a pricing scheme specifically for Se-aaS.

5.1 Contributions of the Chapter

The significant research contributions of this Chapter are stated below:

i) In this Chapter, a pricing model is designed for heterogeneous SOA, in Se-aaS, in which the end-users need to pay for utilizing the services of the physical sensor nodes and the sensor-cloud infrastructure, as per their application demand.

ii) The proposed algorithm for pricing of the physical nodes is context-aware, and the price charged is purely based on the Quality of Information (QoI) that the end-user obtains finally.

iii) The work takes into account the end-users' satisfaction and their net utility as one of the factors to establish the optimality in the pricing. The objective is to maximize the expected individual profit made by the several registered sensor owners along with the profit made by the CSP.

iv) The proposed pricing model is energy-efficient, as computations are primarily performed at the sensor-cloud end, rather than at the physical sensor network, thereby, reducing the complexity of pricing computation among the physical sensor nodes.

v) The work presents a comparative study of the proposed algorithms with some of the traditional hardware pricing algorithms. The former clearly outperforms the latter in terms of residual energy, proximity with Base Station (BS), Received Signal Strength (RSS), and overhead.

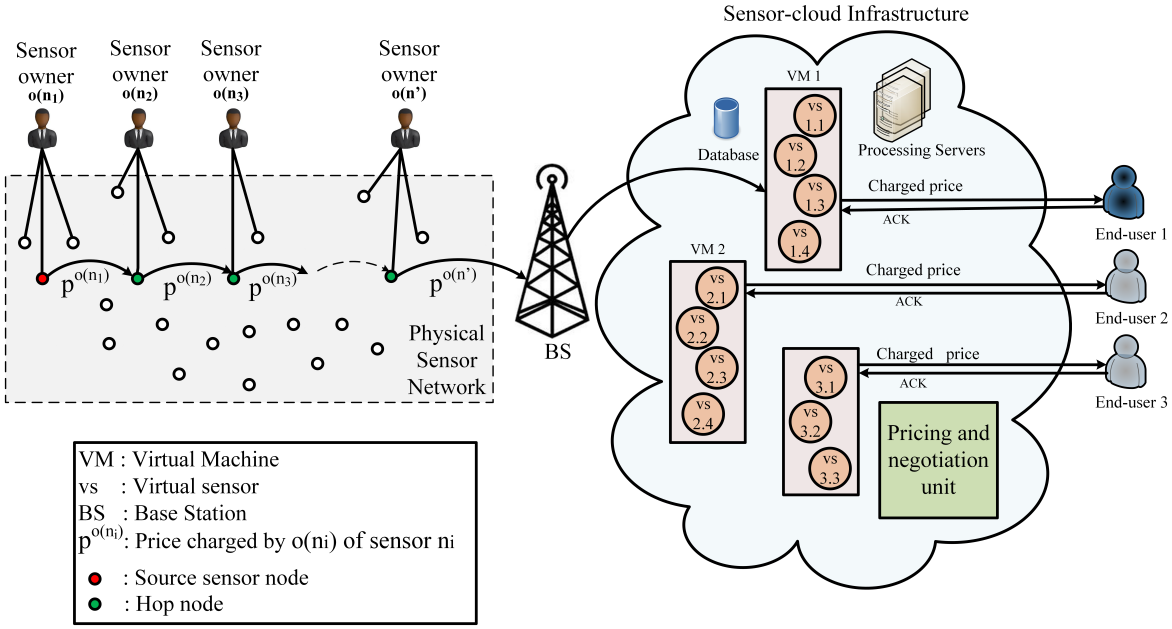


Figure 5.1: Network architecture of sensor-cloud

5.2 Problem Scenario

This Chapter focuses on determining the price to be charged by the CSP from the end-users (based on his/her usage), to achieve the following goals:

i) Optimizing the profit made by the CSP.

ii) Optimizing the profit of the sensor owners whose physical sensor nodes either participate as the source sensor nodes or as the intermediate hop nodes.

iii) Ensuring that the end-users are not overcharged, thereby achieving end-users' satisfaction

As per the requirement of the end-users, the CSP determines the source sensor node, and the other participating physical sensor nodes that are to be activated. The source sensor node may not be within direct reach of the BS, thereby leading to a multi-hop transmission. The other nodes of the network are encouraged to participate as the

intermediate hops, as they are offered incentives for their participation. The incentives are determined as per the policy to gain a net positive profit.

Some cost is also incurred for using and maintaining the sensor-cloud's infrastructural resources – the virtual machines, the virtual sensors, the IT resources, the processing ability of the cloud, and so on. Considering all these related aspects, the CSP regulates the price to be paid by the end-users. Figure 5.1 provides a pictorial illustration of the scenario.

5.3 System Model

There is a set of m physical sensor nodes, $N = \{n_1, n_2, n_3, \ldots, n_m\}$, within the physical sensor network of the sensor-cloud infrastructure, registered by their respective sensor owners. O represents the set of sensor owners. The owner of sensor node n_i is denoted by $o(n_i)$. $E = \{e_1, e_2, e_3, \ldots, e_l\}$, represents the set of end-users requesting for the data from the CSP. The components of the proposed system are formally defined as follows:

- $O' = \{o(n_1), o(n_2), o(n_3), ..., o(n')\}$, $n' < n$, where $O' \subset O$ represents the sensor owners whose physical sensor nodes are actually utilized during the data transmission for a particular end-user e.
- n_1 represents the source sensor node, n_i, $2 \leq i \leq n'$, represents the hop node.
- ${p_t}^{o(n_j)}$,$1 \leq j \leq n'$ represents the price charged by the sensor owner $o(n_j)$ for utilizing its physical sensor node at time instant t.
- VM_e represents the Virtual Machine created for the end-user e, $e \in E$.
- $VS_e = \{vs_1, vs_2, ...vs_{k(t)}\}$, where VS_e represents the set of virtual sensors created within VM_e at time instant t for e.
- $C_{VM_e}(t)$ represents the cost of VM_e at time t.

- p_{VM_e} represents the price charged by the CSP from end-user e for using VM_e.
- $p_{vs_i}(t)$ represents the price charged by the CSP to the end-user e for the virtual sensor vs_i at time instant t.
- $\lambda^e_{vs_i}(t)$ represents the demand by the end-user e for the virtual sensor vs_i at time instant t.
- c represents the criticality of the data per unit time.
- R represents the total number of requests made by all the end-users.
- w represents the service rate of the CSP.

5.3.1 Assumptions of the Model

i) A single CSP and multiple sensor owners are present in the system, i.e., the system is monopolized with respect to the CSP, and oligopolized with respect to the sensor owners.

ii) An end-user is allocated a single VM. However, allocation of multiple virtual sensor nodes within the VM is permitted.

iii) An end-user continues to accept the prices charged at time t, until s/he is dissatisfied at time $t + 1$[1].

iv) The physical sensor nodes periodically transmit control packets to the cloud end to enable the CSP to be aware of the health information of the nodes.

v) Every physical sensor node is static, and is aware of the location coordinates of itself, its neighbors and the corresponding BS.

In Figure 5.1, the sensor owner $o(n_1)$ owns the source sensor node which generates the required data. $o(n_1)$ needs the help of any immediate physical sensor node in order

[1]The upper bound of $t + 1$ is discussed under strategy profile in subsection 5.3.2.2.

to transmit the data. The CSP encourages the neighboring physical nodes of the source sensor node to participate in the data transmission. The source sensor node n_1 chooses one of its neighbors as the next hop node n_2, based on a utility value. $o(n_1)$ charges $o(n_2)$ with price ${p_t}^{o(n_1)}$ [1]. ${p_t}^{o(n_1)}$ is accepted by the hop node owned by the sensor owner $o(n_2)$. With the intention to make profit, $o(n_2)$ charges a price ${p_t}^{o(n_2)}$ greater than ${p_t}^{o(n_1)}$ to its next willing participant. This pricing scheme continues until the data finally reaches the last participating hop node. The last hop node owned by the sensor owner $o(n')$ charges a price ${p_t}^{o(n')}$ to the end-user e who actually requested the data. Furthermore, it is intuitive that ${p_t}^{o(n')} > {p_t}^{o(n'-1)} > ... > {p_t}^{o(n_2)} > {p_t}^{o(n_1)}$. In order to transfer the required data, the infrastructural resources of the CSP are utilized. Based on the end-user demand, the virtual machines and the component virtual sensors are created for which the CSP charges some amount of price. This profit is solely enjoyed by the CSP for provisioning infrastructure as a service. The pricing scheme of Se-aaS is broken up into two distinct modules and propose two different algorithms:

a) *Pricing attributed to Hardware* (pH)

b) *Pricing attributed to Infrastructure* (pI).

5.3.2 pH: Pricing attributed to Hardware

The pricing attributed to the usage of the physical sensor nodes concern the profit of the respective sensor owners. As the source sensor node n_1 generates the raw sensed data, it either directly transmits it to the BS in a single-hop, or follows a multi-hop route. Motivated by the pricing strategies mentioned in [112, 120], the proposed pricing model is designed for the hardware usage. A context aware optimal pricing scheme is propounded for the usage of the physical sensor nodes.

[1] Although it appears that the price charged by one sensor owner is paid by another, the net price is essentially paid by the end-user.

5.3.2.1 Selection of the next hop node

In order to transmit data from the source sensor node n_1 to the Base Station BS [1], n_1 selects the next hop node n_2 with the maximum utility η among all the nominated hop nodes, in set H_{n_1}. The transmission radius of n_1 at t is denoted as $r_{n_1}(t)$. The set of physical sensor nodes that are located within the transmission area $A_{n_1}(t) = \pi r_{n_1}(t)^2$ is considered to be the nominated hop nodes. Thus, $H_{n_1} = \{h_1, h_2, h_3, ..., h_b\} \mid \xi(h_j, n_1) \geq r_{n_1}(t), 1 \leq j \leq b$, where $\xi()$ computes the inverse of the Euclidean distance between two nodes. The node h_i with the maximum utility η_{max} emerges as the winner hop node among all the participating physical nodes. The utility of node h_i at time instant t is dependent on the following factors.

- *Residual Energy:* The utility $\eta_{h_i}(t)$ of h_i at t is dependent on its residual energy level, $Q_{h_i}(t)$, which expressed as,

$$Q_{h_i}(t) = \frac{E_{h_i}^{cur}}{E_{h_i}^{init}} \tag{5.1}$$

  where $E_{h_i}^{init}$ and $E_{h_i}^{cur}$ are the initial and current energy level of h_i, respectively.

- *Proximity to the BS:* $\eta_{h_i}(t)$ is dependent on the inverse of the Euclidean distance $\xi(h_i, BS)$ between node h_i and BS.

$$\xi(h_i, BS) = \left(\sqrt{(BS_{x_i} - h_{x_i})^2 + (BS_{x_i} - h_{y_i})^2}\right)^{-1} \tag{5.2}$$

  h_{x_i}, h_{y_i}, BS_{x_i}, and BS_{y_i} being the abscissa and ordinate of h_i, and the BS, respectively.

[1] It is to be noted that to ensure fault tolerance and efficiency, in practice, the system model may support multiple BSs. However, for the sake of simplicity and understandability, a single BS is considered in this work.

- *Received Signal Strength:* The Received Signal Strength of h_i, RSS_{h_i}, is also one of the factors affecting its utility at time instant t. Thus,

$$RSS_{h_i}(t) = \psi_{h_i} \frac{P^{tr}_{h_i}(t)}{\xi(h_i, n_1)^a} \tag{5.3}$$

  where $P^{tr}_{h_i}$ is the transmitted power, ψ_{h_i} is the constant that takes into account all the other factors affecting RSS such as the antenna gain and antenna height, and a denotes the propagation constant [121]. In our problem scenario, $a = 2$.

- *State Transition Overhead:* Node h_i exists in either of the three states —active ($S^0_{h_i}$), passive ($S^1_{h_i}$), and sleep ($S^2_{h_i}$). For the data transmission, h_i needs to exist in the active state $S^0_{h_i}$. The state transition overhead in terms of energy dissipation while switching from $S^1_{h_i}$ or $S^2_{h_i}$ to $S^0_{h_i}$ is denoted by, $P_{pq}, p = \{S^0_{h_i}, S^1_{h_i}, S^2_{h_i}\}, q = \{S^0_{h_i}\}$. Quite intuitively, $P_{S^1_{h_i} S^0_{h_i}} \ll P_{S^2_{h_i} S^0_{h_i}}$. However, when h_i remains in the active state, there is ideally no overhead. It is assumed that $P_{S^0_{h_i} S^0_{h_i}} \to 0$.

Definition 11. *The utility $\eta_{h_i}(t)$ of a hop node h_i, $\forall i = \{1, 2, 3, ..., b\}$, at time instant t, is a function of its residual energy $Q_{h_i}(t)$, its Received Signal Strength $RSS_{h_i}(t)$, its proximity to the BS $\xi(h_i, BS)$, and its state transition overhead P_{pq}. $\eta_{h_i}(t)$ is expressed as,*

$$\eta_{h_i}(t) = \left(Q_{h_i}(t) + g \times \frac{RSS_{h_i}(t)\xi(h_i, BS)}{P_{pq}} \right)$$

g being a normalization factor with the same unit as that of ξ.

Having computed the utility of every nominated hop node, the node (n_i) with the maximum utility emerges as the winner hop node. Thus, without the loss of generality it can be inferred,

$$n_i = \max_{\forall h_k \in H_{n_{i-1}}} \eta_{h_k}(t) \tag{5.4}$$

5.3.2.2 Context-aware pricing

Having decided the next hop node, a context-aware pricing scheme is now proposed. Initially, the expected price to be charged by the sensor-owner of the source sensor node $o(n_1)$, which is denoted by ${p_t}^{o(n_1)}$, is determined. The context of the data is examined in terms of few parameters described below.

Definition 12. *Transmission confidence of the data between a pair of nodes $< a, b >$ at time t, $f_{a,b}(t)$, is expressed in the form of profit/loss factor based on the difference of the raw sensed data between the sender and the receiver nodes [122].*

$$f_{a,b}(t) = \begin{cases} \frac{1}{N} f_{a,b}(t-1) e^{(\rho\delta)(t)}, & \rho = |D_a - D_b| < \rho_{th} \\ \frac{1}{N} f_{a,b}(t-1) e^{-(\rho\delta)(t)}, & otherwise \end{cases} \tag{5.5}$$

where N is a factor for normalization, ρ is the absolute deviation of the transmitted data D_a from the received data D_b, and δ is the profit/loss factor.

Definition 13. *The temporal relevance of the data $\mathcal{T}$ at time t is defined as the tolerable time interval, beyond which the data is assumed to be insignificant. Thus,*

$$\mathcal{T}(t) = \frac{t_d - t'}{t_r - t'}, \quad 0 \leq t_r - t_d \leq k \tag{5.6}$$

where t_d, t', and t_r are the time instant of transmitting data packet, time instant of detecting an event at n_i, and the time instant of receiving the data at n_{i+1}, respectively. If $t_r - t_d$ exceeds k, $\mathcal{T}$ is considered to be negligible, i.e., $\mathcal{T} \sim 0$.

Motivated by the general design for the metric QoI [123], the QoI of node n_i at time t is modeled as, $\mathcal{Q}_{n_i} = \omega_{n_i} \mathcal{Q}_{n_i - 1} + \eta_{n_i}, \quad \mathcal{Q}_{n_1} = 1$, where ω_{n_i} is the discounting factor at time t, which is expressed as $\omega_{n_i} = Q_{n_i} f_{n_{i-1}, n_i} \mathcal{T}/a$ where a normalizes the value within

0 to 1 and makes ω_{n_i} unit less. Thus,

$$\mathcal{Q}_{n_i} = \prod_{j=2}^{n_i} \omega_{n_j} \mathcal{Q}_{n_1} + \sum_{k=2}^{n_i-1} \prod_{l=k+1}^{n_i} \omega_{n_l} \eta_{n_k} + \eta_{n_i} \tag{5.7}$$

Definition 14. *The price ${p_t}^{o(n_1)}$ charged by sensor owner of the source node $o(n_1)$, is directly proportional to the QoI of the data of n' at time t,*

$${p_t}^{o(n_1)} \propto \mathcal{Q}_{n'}(t) \Rightarrow {p_t}^{o(n_1)} = c_1(t)\mathcal{Q}_{n'}(t) \tag{5.8}$$

where c_1 is a multiplicative factor that accounts for both the signal attenuation in terms of the Nodal Signal to Noise Ratio (NSNR) [124] and the total number of transmission attempts for the corresponding packet. Thus,

$$c_1(t) = g\frac{P_{signal}(t)}{P_{noise}(t)} \tag{5.9}$$

where P_{signal}, P_{noise}, and g are the power of signal and the background noise, and the count of the transmission attempts, respectively.

Definition 15. *The utility $\mathcal{U}$ of the end-user e is defined as the amount of data received per virtual sensor vs_i per unit time. Thus, $\mathcal{U} \sim U(\gamma_1, \gamma_2)$.*

Motivated by the works of Lam *et al.* [112], and Fudenberg and Tirole [125], the strategy profile of the proposed system is illustrated as follows.

Strategy profile:

- The end-user e obtains data from a virtual sensor vs_i for a time period, τ. The end-user follows a myopic strategy: it retains a virtual sensor vs_i at time t, if $(t \leq \tau)$ and $(\mathcal{U} \geq {p_t}^{o(n')})$ i.e., within the time period τ, the end-user accepts the service if and only if the utility $\mathcal{U}$ is higher than the price to be payed by the end-user.

- The sensor owner $o(n_i)$, $\forall i = \{2, 3, \ldots, n'\}$, of a participating hop node, charges a price $p^{*o(n_i)}(p^{o(n_{i-1})})$ which is dependent on the price charged by the previous sensor owner $o(n_{i-1})$.

$$p^{*o(n_i)}(p^{o(n_{i-1})}) \in \arg\max_{p^{o(n_i)}} \left[\left(p^{o(n_i)} - p^{o(n_{i-1})} \right) P\left(\mathcal{U} \geq m^{o(n_i)}(p^{o(n_i)}) \right) \right] \quad (5.10)$$

  where $m^{o(n_i)}(p^{o(n_i)})$ is the mark up function, as defined in Definition 16. As depicted in Equation (5.10), a sensor-owner $o(n_i)$ strategically claims his/her price by probabilistically determining the effective price payable (by the end-user) to the stream of sensor-owners $o(n_i)$ to $o(n')$. For the strategy to be effective, it also considers the probability of the end-user to be willing to pay the price, $\left(P(\mathcal{U} \geq m^{o(n_i)}(p^{o(n_i)})) \right)$.

- The owner of the last hop node $o(n')$ charges a decreasing price sequence $\{p_t^{o(n')}\}$.

Definition 16. *The mark-up function $m^{o(n_i)}(p^{o(n_i)})$ of the proposed system is defined as the price that an end-user has to pay for the stream of nodes from node n_1 to n_i after the price is fixed at n_i. Thus, $m^{o(n_i)}(p^{o(n_i)})$ is expressed as,*

$$m^{o(n_i)}(p^{o(n_i)}) = \begin{cases} p^{*o(n_i)}(p^{*o(n_i-1}(...(p^{*o(n_2)}(p^{o(n_1)}))...)), \\ \qquad\qquad i = \{2, ..., n' - 1\} \\ p^{o(n_1)}, \qquad\qquad i = 1 \end{cases} \quad (5.11)$$

With the assumption that γ_1, γ_2 are known, the optimal price $p^{*o(n')} \in [\gamma_1, \gamma_2]$ charged by $o(n')$, is determined as,

$$\left(p^{o(n')} - p^{o(n'-1)} \right) P\left(\mathcal{U} \geq m^{o(n')}(p^{o(n')}) \right) = \left(p^{o(n')} - p^{o(n'-1)} \right) \left(\frac{\gamma_2 - p^{o(n')}}{\gamma_2 - \gamma_1} \right) \quad (5.12)$$

On differentiating Equation (5.12) w.r.t $p^{o(n')}$ and equating to zero, the optimal price

$$p^{*o(n_i)} = \begin{cases} (2^{n'-i})\gamma_1 - (2^{n'-i}-1)\gamma_2, & \text{if } \frac{p^{o(n_{i-1})}+\gamma_2}{2} < (2^{n'-i})\gamma_1 - (2^{n'-i}-1)\gamma_2 \\ \frac{p^{o(n_{i-1})}+\gamma_2}{2}, & \text{if } \frac{p^{o(n_{i-1})}+\gamma_2}{2} \in [(2^{n'-i})\gamma_1 - (2^{n'-i}-1)\gamma_2, \gamma_2] \\ \gamma_2, & \text{otherwise} \end{cases} \tag{5.14}$$

$p^{*o(n')}$ is obtained as,

$$p^{*o(n')} = \begin{cases} \gamma_1, & \text{if } \frac{p^{o(n'-1)}+\gamma_2}{2} < \gamma_1 \\ \frac{p^{o(n'-1)}+\gamma_2}{2}, & \text{if } \frac{p^{o(n'-1)}+\gamma_2}{2} \in [\gamma_1, \gamma_2] \\ \gamma_2, & \text{otherwise} \end{cases} \tag{5.13}$$

On iterating the above process, the optimal price $p^{*o(n_i)}$ charged by the owner, $o(n_i)$ of any participating hop node is derived as: $\forall i = \{2, 3, ..., n'\}$, as shown in Equation (5.14). The optimal price $p^{*o(n_1)}$ charged by the owner of the source sensor node $o(n_1)$ is,

$$p^{*o(n_1)} = \begin{cases} \frac{\gamma_2}{2}, & \text{if } (2^{n'-1})\gamma_1 - (2^{n'-1}-1)\gamma_2 \leq \frac{\gamma_2}{2} \\ (2^{n'-1})\gamma_1 - (2^{n'-1}-1)\gamma_2, & \text{otherwise} \end{cases} \tag{5.15}$$

Theorem 5.3.1. *The theoretical maximum of an end-user utility, γ_2 , is dependent on the price charged by the last hop node, $p^{*o(n')}$ and the price charged by the second last hop node, $p^{o(n'-1)}$.*

Proof. The optimal price charged by $o(n')$ is obtained from Equation (5.13). Thus, to maintain the optimality in price, the utility provisioned to an end-user has an upper bound γ_{2max}. It is observed that, as $\frac{p^{o(n'-1)}+\gamma_2}{2} \in [\gamma_1, \gamma_2]$, $\gamma_2 = 2p^{*o(n')} - p^{o(n'-1)}$, and as $\frac{p^{o(n'-1)}+\gamma_2}{2} > \gamma_1$, $\gamma_2 = p^{*o(n')}$. Now,

$$\begin{aligned} \gamma_2 &= 2p^{*o(n')} - p^{o(n'-1)} \\ &= p^{*o(n')} + p^{*o(n')} - p^{o(n'-1)} \end{aligned}$$

Since $(p^{*o(n')} - p^{o(n'-1)})$ is the net profit of $o(n')$, it is expected to be a positive quantity. Thus, it is inferred that $\gamma_{2max} = 2p^{*o(n')} - p^{o(n'-1)}$. This implies,

$$\gamma_{2max} = \begin{cases} 2p^{*o(n')} - p^{o(n'-1)}, \\ \qquad \text{if } \gamma_2 \in max\Big(2\gamma_1 - p^{o(n'-1)}, p^{o(n'-1)}\Big) \\ p^{*o(n')}, \text{ otherwise} \end{cases} \tag{5.16}$$

This concludes the proof. □

Corollary 5.3.2. *The maximum utility γ_2 obtained by an end-user e, at a particular time instant, is dependent on the number of hop nodes n'.*

Justification: Ideally, every $o(n_i), \forall i \in \{1...n'\}$, makes a net positive profit. Therefore, $\{p^{*o(n_i)}\}$ is a non-decreasing sequence. Hence as n' increases, $p^{*o(n')}$ also increases. This justifies the statement.

Proposition 5.3.3. *For a single hop case, i.e., when the source node n_1 behaves as the only hop node, the maximum utility that can be provisioned is twice the price charged by $o(n_1)$.*

Proof. For a single hop case, n_1 directly connects to BS, i.e., $n' = 1$. From Equation (5.15), it is inferred that, as $\gamma_2 \geq 2\gamma_1$, $\gamma_2 = 2p^{*o(n_1)}$, and when $\gamma_2 < 2\gamma_1$, $\gamma_2 < 2p^{*o(n_1)}$. Thus, without the loss of generality it can be said that, $\gamma_{2max} \leq 2p^{*o(n_1)}$. This completes the proof. □

5.3.3 pI: Pricing attributed to Infrastructure

In terms of the usage of infrastructure within the sensor-cloud platform, whenever end-user e requests the CSP for some data to be fed into his/her application, the CSP creates a VM dedicated to e, VM_e. The number of virtual sensors within VM_e that are created and deleted depends upon the requirement of e, and, thereby, being time dependent,

and is denoted by $k(t)$. Based on the demand $\lambda^e_{vs_i}(t)$ of e for virtual sensor vs_i, the price charged by the CSP is $p_{vs_i}(t)$ at time instant t. $C_{VM_e}(t)$ is the cost of creating VM_e within the cloud platform, inclusive of the initial cost for creating the instance of VM_e, B_{VM_e}, and the cost for maintaining it over time. The maintenance cost of a VM_e is charged from the time it is built (t_{built}) till it is discarded. The maintenance cost of a VM_e per unit time, M_{VM_e}, comprises of the cost for creating its component virtual sensors $vs_i \in VS_e$, in addition to the maintenance cost per unit time, for each of them. Thus,

$$C_{VM_e}(t) = B_{VM_e}(t) + M_{VM_e}(t - t_{built}) \tag{5.17}$$

$$M_{VM_e}(t - t_{built}) = \sum_{i=1}^{k(t)} \Big(B_{vs_i} + M_{vs_i}(t - t_{0i}) \Big) \tag{5.18}$$

where t_{0i} represents the time instant at which the virtual sensor vs_i is created. The final equation of the cost incurred by the CSP for the creation and maintenance of VM_e and its corresponding virtual sensors, at time t is,

$$C_{VM_e}(t) = B_{VM_e}(t) + \sum_{i=1}^{k(t)} B_{vs_i} + \sum_{i=1}^{k(t)} M_{vs_i}(t - t_{0i}) \tag{5.19}$$

A virtual sensor comprises of a set of homogeneous (with respect to sensing hardware) physical sensors serving a particular application. The creation and deletion of the virtual sensors is completely dependent on the end-user's requirement. However, if a virtual sensor is unused for a long time duration, the maintenance cost exceeds the cost of creating the same. In such cases, it is preferred to delete a virtual sensor and create it when required.

Proposition 5.3.4. *The optimum time interval $\triangle t$ between two consecutive demands for a particular virtual sensor vs_i is $\frac{B_{vs_i}}{M_{vs_i}}$.*

Proof. We assume that the last time instant at which the maintenance cost equals the

cost of creation of vs_i is t_{max} and t represents the current time instant. Thus, $M_{vs_i}(t_{max} - t) = B_{vs_i}$. Thus, for all $t' > t_{max}$,

$$M_{vs_i}(t_{max} - t') > B_{vs_i} \Rightarrow t_{max} = \frac{B_{vs_i}}{M_{vs_i}} + t \tag{5.20}$$

$$\text{Thus, } \triangle t = t_{max} - t = \frac{B_{vs_i}}{M_{vs_i}} \tag{5.21}$$

□

Corollary 5.3.5. *The instantaneous cost incurred at the cloud end, for a virtual sensor vs_i, at time t', ($C^{inst}_{vs_i}(t')$), is dependent on the time instant when the last demand was placed.*

Proof. We assume that the last demand for vs_i was placed at t_{last}. From Proposition 5.3.4, it follows that, at current time instant t', if $t' - t_{last} < \triangle t$, then the instantaneous cost for vs_i will be only due to maintenance at t'. Otherwise, it includes both the creation and maintenance cost. Thus,

$$C^{inst}_{vs_i}(t') = \begin{cases} M_{vs_i}, & t' - t_{last} < \frac{B_{vs_i}}{M_{vs_i}} \\ B_{vs_i} + M_{vs_i}, & otherwise \end{cases} \tag{5.22}$$

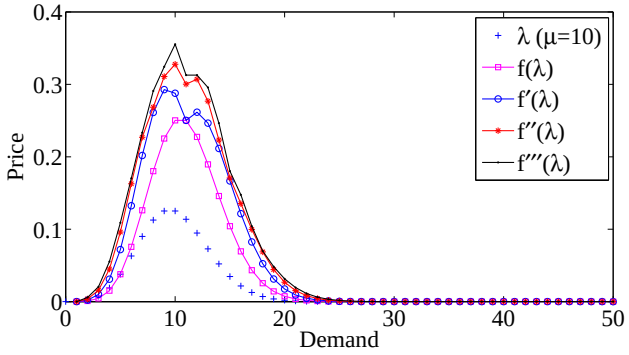
This completes the proof.

□

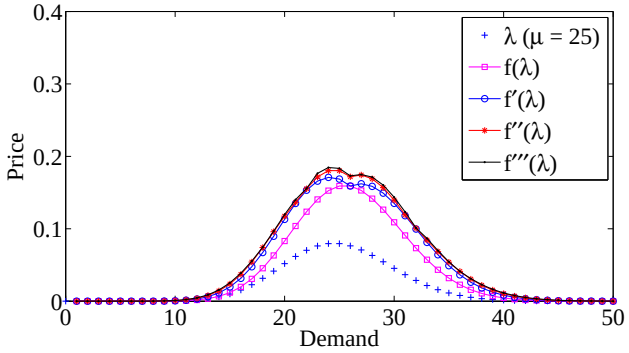
Definition 17. *The net profit of the CSP at time t, $r(t)$, is defined as the difference of the total price charged from the end-user and the sum of the cost incurred in creating and maintaining the VM for a particular end-user e and the overall price charged through pH for e ($p_e^{o(n')}(t)$). Thus, $r(t)$ is expressed as,*

$$r(t) = \left(\sum_{i=1}^{k(t)} \lambda^e_{vs_i}(t) p_{vs_i}(t) \right) + p_{VM_e} - C_{VM_e}(t) - p_e^{o(n')}(t) \tag{5.23}$$

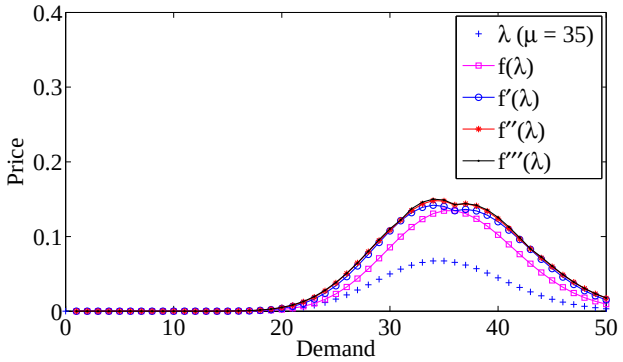
where the price charged for each virtual sensor is a function of the rate of change of demand for each $vs_i(t)$.



(a) $\lambda_{vs_i} \sim P(50, 10)$



(b) $\lambda_{vs_i} \sim P(50, 25)$



(c) $\lambda_{vs_i} \sim P(50, 35)$

Figure 5.2: Analysis of price-demand relationship

Theorem 5.3.6. *The price charged for a virtual sensor,* vs_i*, is based on the memory of*

demand: the price $p_{vs_i}(t)$ charged at a particular time instant t, is based on the previous demands $\lambda_{vs_i}(t-1)$, and the j^{th} order of rate of change of demands over time, $\frac{d^j \lambda_{vs_i}}{dt^j}$, $1 \leq j \leq n$, $n \in \mathbb{N}$.

Proof. As in Corollary 5.3.5, the instantaneous cost of the virtual sensor depends upon the time instant at which the demand was last placed. Thus, as the rate of demand increases within $\triangle t$, the cost decreases accordingly. Therefore, for a vs_i,

$$C_{vs_i}(t) = f\left(\lambda^e_{vs_i}, \frac{d\lambda^e_{vs_i}}{dt}, \ \ldots, \frac{d^{n-1}\lambda^e_{vs_i}}{dt^{n-1}}, \frac{d^n \lambda^e_{vs_i}}{dt^n}\right) \tag{5.24}$$

From Equation (5.23), it can be seen that an increase in the cost, increases the price of vs_i, for the CSP to make positive net profit, i.e., price and cost are linearly connected. Thus,

$$p_{vs_i}(t) = f\left(\lambda^e_{vs_i}, \frac{d\lambda^e_{vs_i}}{dt}, \ \ldots, \frac{d^{n-1}\lambda^e_{vs_i}}{dt^{n-1}}, \frac{d^n \lambda^e_{vs_i}}{dt^n}\right) \tag{5.25}$$

Figure 5.2 shows the relationship between demand and price. It has been assumed that in Figures 5.2(a), 5.2(b), and 5.2(c), demand follows a Poisson distribution ($n = 50$) with varying mean ($\mu = 10$, $\mu = 25$, $\mu = 35$), respectively. It can be observed that the change in price is significant with the first order derivative of the demand. However, there is not much effective change reflected from the higher order derivatives of the demand. Therefore, for the sake of simplicity, in this work, the following Equation holds true:

$$p_{vs_i}(t) = \alpha \frac{d\lambda^e_{vs_i}(t)}{dt} + \beta \lambda^e_{vs_i}(t) \tag{5.26}$$

where the parameters α, β are assumed to be system-modeled coefficients. This completes the proof. □

At a particular time t', R represents the total number of requests made by all the end-users $\in E$.

$$R = \sum_{j=1}^{l} \sum_{i=1}^{k(t')} \lambda_{vs_i}^{e_j} \quad , \forall e_j \in E, \forall vs_i \in VS_{e_j} \tag{5.27}$$

Since the service rate of CSP is w, the expected time to finish serving a request, inclusive of the waiting time and the time being served is, $\frac{1}{w-R}$ [117, 126].

Therefore, the time spent for waiting is $\frac{1}{w-R} - \frac{1}{w}$ [117]. Thus, the expected finishing time for e is,

$$\frac{1}{w-R} - \frac{1}{w} + \frac{\sum_{i=1}^{k(t')} \lambda_{vs_i}^{e}}{w} = \frac{R}{w(w-R)} + \frac{\sum_{i=1}^{k(t')} \lambda_{vs_i}^{e}}{w} \tag{5.28}$$

Definition 18. *The user satisfaction $u_e(t)$ for a particular end-user e at any time instant t, is a function of the total demand made by e for all the virtual sensors within VM_e, the total cost incurred at the sensor-cloud end for serving the demand, and the total price charged by the CSP.*

$$u_e(t) = \sum_{i=1}^{k(t)} \lambda_{vs_i}^{e} - c\Big[\frac{R}{w(w-R)} + \frac{\sum_{i=1}^{k(t)} \lambda_{vs_i}^{e}}{w}\Big] - c'\Big[\sum_{i=1}^{k(t)} p_{vs_i}(t) + p_{VM_e}\Big] \tag{5.29}$$

where c' is the constant incorporated to make the term $\sum_{i=1}^{k(t)} p_{vs_i}(t) + p_{VM_e}$ unitless.

The main objective of our work is to maximize the total profit of the CSP over time T, while considering the user satisfaction, i.e.,

$$\mathcal{F}(T) = \sum_{t=0}^{T} r(t) \tag{5.30}$$

$$\text{subjected to,} \quad \lambda_{vs_i}^{e} \geq 0, \forall i = 1, 2, 3, \ldots, k(t) \tag{5.31}$$

$$\sum_{i=1}^{k(t)} \lambda_{vs_i}^{e} - c\Big[\frac{R}{w(w-R)} + \frac{\sum_{i=1}^{k(t)} \lambda_{vs_i}^{e}}{w}\Big] - c'\Big[\sum_{i=1}^{k(t)} p_{vs_i}(t) + p_{VM_e}\Big] > v_{opt}^{e} \tag{5.32}$$

where v_{opt} is the threshold value, below which the values of $u_e(t)$ are not allowed. From the Equation (5.30), it is observed that $r(t)$ can be maximized for every time instant t. Accordingly, $\mathcal{F}(T)$ can be maximized. $r(t)$ is simplified as,

$$\begin{aligned} r(t) &= \sum_{i=1}^{k(t)} \Big(\lambda^e_{vs_i}(t) p_{vs_i}(t) \Big) + p_{VM_e} - C_{VM_e}(t) - p_e^{o(n')}(t) \\ &= F\Big(\lambda_{vs_1}, \lambda_{vs_2}, \ldots, \lambda_{vs_{k(t)}}, t_{01}, t_{02}, \ldots, t_{0k(t)} \Big) \end{aligned}$$

The aim is to maximize F using the approach of Lagrange Multiplier. Thus,

$$\nabla F\Big(\lambda_{vs_1}, \lambda_{vs_2}, \ldots, \lambda_{vs_{k(t)}}, t_{01}, t_{02}, \ldots, t_{0k(t)} \Big) = \theta \nabla u\Big(\lambda_{vs_1}, \lambda_{vs_2}, \ldots, \lambda_{vs_{k(t)}} \Big) \quad (5.33)$$

where θ is the Lagrangian multiplier. Therefore,

$$\frac{\partial F}{\partial \lambda^e_{vs_i}} = \alpha \frac{d\lambda^e_{vs_i}}{dt} + \beta \lambda^e_{vs_i} + \lambda^e_{vs_i} \Big(\alpha \frac{d}{d\lambda^e_{vs_i}} \frac{d\lambda^e_{vs_i}}{dt} + \beta \Big) \quad (5.34)$$

$$\frac{\partial u_e}{\partial \lambda^e_{vs_i}} = 1 - c\Big(\frac{w^2 - 2wR}{w^2(w-R)^2} + \frac{1}{w} \Big) - \Big[\alpha \frac{d\lambda^e_{vs_i}}{dt} + \beta \lambda^e_{vs_i} + \lambda^e_{vs_i} \Big(\alpha \frac{d}{d\lambda^e_{vs_i}} \frac{d\lambda^e_{vs_i}}{dt} + \beta \Big) \Big] \quad (5.35)$$

Using Equations (5.33) through (5.35), the following is obtained:

$$\begin{aligned} \alpha \frac{d\lambda^e_{vs_i}}{dt} + \beta \lambda^e_{vs_i} + \lambda^e_{vs_i} \Big(\alpha \frac{d}{d\lambda^e_{vs_i}} \frac{d\lambda^e_{vs_i}}{dt} + \beta \Big) = \theta \Big[1 - c\Big(\frac{w^2 - 2wR}{w^2(w-R)^2} + \frac{1}{w} \Big) \\ - \Big(\alpha \frac{d\lambda^e_{vs_i}}{dt} + \beta \lambda^e_{vs_i} + \lambda^e_{vs_i} \Big(\alpha \frac{d}{d\lambda^e_{vs_i}} \frac{d\lambda^e_{vs_i}}{dt} + \beta \Big) \Big) \Big] \end{aligned} \quad (5.36)$$

At a particular time instant t', $\frac{d\lambda^e_{vs_i}}{dt'} = 0$. Assuming

$$\alpha \frac{d\lambda^e_{vs_i}}{dt} + \beta \lambda^e_{vs_i} + \lambda^e_{vs_i} \Big(\alpha \frac{d}{d\lambda^e_{vs_i}} \frac{d\lambda^e_{vs_i}}{dt} + \beta \Big) = K \quad (5.37)$$

Therefore, $K = 2\beta\lambda^e_{vs_i}$. Using K and Equation (5.36), it is obtained:

$$\theta = \frac{2\beta\lambda^e_{vs_i}}{1 - c\left(\frac{w^2-2wR}{w^2(w-R)^2} + \frac{1}{w}\right) - 2\beta\lambda^e_{vs_i}} \tag{5.38}$$

Therefore, after the value of θ is known, the values of $\mathcal{F}(t)$ and $r(t)$ can be successfully evaluated.

5.4 Experimental Results

In this Section, the results of experimentation are presented and analyzed. Followed by this, the complexity analysis of both pH and pI are provided. The generic test-bed information for pH and pI is provided in Table 6.2. Although this work is one of the first attempts to design a pricing scheme for sensor-cloud, some of the hardware pricing solutions that are found similar (but not exact), are discussed in Subsection A. The experimentation setup for pH is shown in Table 5.2. Some comparative analysis of the proposed solutions with the benchmark approaches are also performed.

Table 5.1: Testbed information for pH and pI

Parameters	**Values**
Processor	Intel(R) Core(TM) i5-2400 CPU @ 3.10 GHz
RAM	4GB, DDR3
Disk space	320 GB
Operating system	Ubuntu 14.04 LTS
Application software	MATLAB R2013a

5.4.1 Explanation of Parameters

The parameters used for this work have been selected on the basis of prior related works [104–106]. The values of these parameters are set as per the related works in this domain that are most cited [107–111].

Table 5.2: Experimentation setup for pH

Parameters	**Values**
Deployment Area	500 m × 500 m
Deployment	Uniform, random
Number of nodes	500
Communication range	[100, 200] m
Transmission energy	7 nJ/bit
Computation energy	5 nJ/sec
Sensing energy	6 nJ/sec
Number of end-users	10
Average user utility	10000
Percentage of confidence	95 %

5.4.2 Analysis of pH

Initially, the proposed *pH* algorithm is compared with few identified benchmark solution approaches. Followed by this, the performance of *pH* is also evaluated separately. The performance metrics that are considered for comparison are:

- Mean residual energy
- Mean proximity with BS
- Mean RSS
- Mean state transition overhead
- Cumulative energy consumption
- Packet delivery rate

The first four are already defined in Section 5.3. The simulation metric for cumulative energy consumption is discussed later in this subsection, with the corresponding results. The packet delivery rate is defined below.

Definition 19. *Packet delivery rate is defined as the percentage of the total packets successfully delivered from any source sensor node to the BS.*

5.4.2.1 Benchmark Solutions

In order to find the solution for the proposed model, the following existing benchmark solutions are used as the basis for comparison,

- The Packet Purse Model (PPM) [46]
- Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks [41]

In PPM [46], the sender bears the total cost of transmitting sensed data from the source sensor node to the BS. This cost is calculated in terms of the virtual currency called nuglets. If the amount is under estimated by the sender, then the packet is dropped mid-way, and if it is over estimated, then the sender suffers a loss of nuglets. Moreover, this model requires a tamper- proof hardware established at each node for proper deduction and addition of nuglets. Also, the size of the Packet Purse Header increases than the actual packet size resulting in slow inefficient packet transmission.

In Sprite [41], a central authority, known as Credit Clearance Service (CCS), is implemented. It evaluates the amount of nuglet to be charged or credited to each node involved in the packet transmission, based on the submitted receipts of a message. For message authentication, the sender transfers a signed message to the next immediate hop node, which accepts the message only after proper verification of the signature of the sender nodes. The digital signature and verification procedure involves a significant processing overhead. Moreover, the CPU processing time exceeds an acceptable limit if any node attempts to send huge number of messages. The storage and the bandwidth requirement increases due to the addition of the authentication header with each message packet.

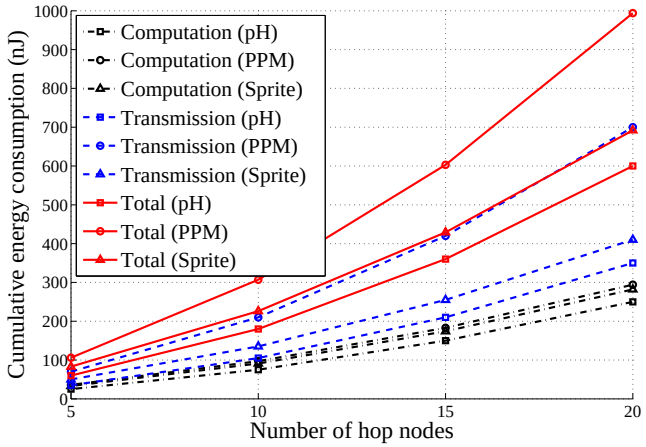
Table 5.3: Comparative study of pH with PPM and Sprite

	Mean Residual energy (in %)	Mean Proximity with BS (in metre)	Mean RSS (in units)	Mean state transition overhead (in units)
pH	72.15	203.91	36.6	1.05
PPM, Sprite	37.33	223.67	35.9	1.93

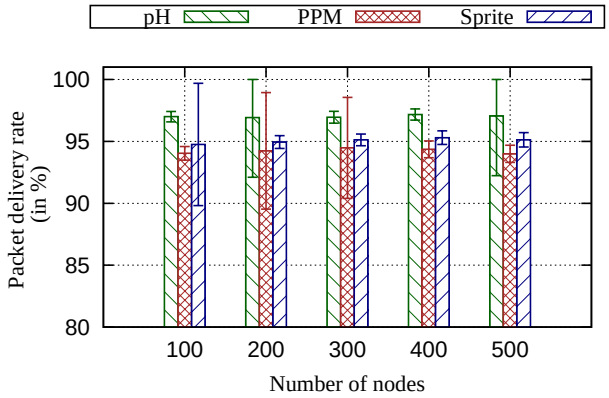
In pH, the selection of the next-hop node is evaluated using Equation (5.4), whereas in PPM [46] and Sprite [41], the standard selection of next hop node is based on simple Dynamic Source Routing (DSR) [127, 128] protocol, in which the physical sensor node closest to the source sensor node is expected to emerge as the next hop node under ideal channel conditions. The experiment is repeated 50 times and the mean of several node parameters is compared for both the approaches, and is shown in Table 5.3.

From Table 5.3, it is evident that the pH selects a better node, compared to PPM or Sprite, in terms of the mean residual energy, mean proximity with the BS, mean RSS, and mean state transition overhead. As the hop selection algorithm in DSR does not consider the other node parameters, e.g., energy level of a node, RSS intensity, and state transition overhead, the hop nodes in PPM, and Sprite are likely to have poor residual energy, or a low RSS intensity. pH outperforms the other approaches in this regard, thereby choosing the nodes with the maximum utility.

Figure 5.3(a) illustrates the cumulative energy consumption of the 10 end-users with the increase in the number of hop nodes. For every end-user, any source sensor node is subjected to identical sensing phenomenon for pH, PPM, and Sprite. Hence, the performance comparison is significant in terms of energy consumption due to transmission, and computation, only. As shown in Figure 5.3(a), PPM incurs the maximum computation due to repeated estimation of nuglets for every round of transmission. In Sprite, the node



(a) Cumulative energy consumption



(b) Packet delivery rate

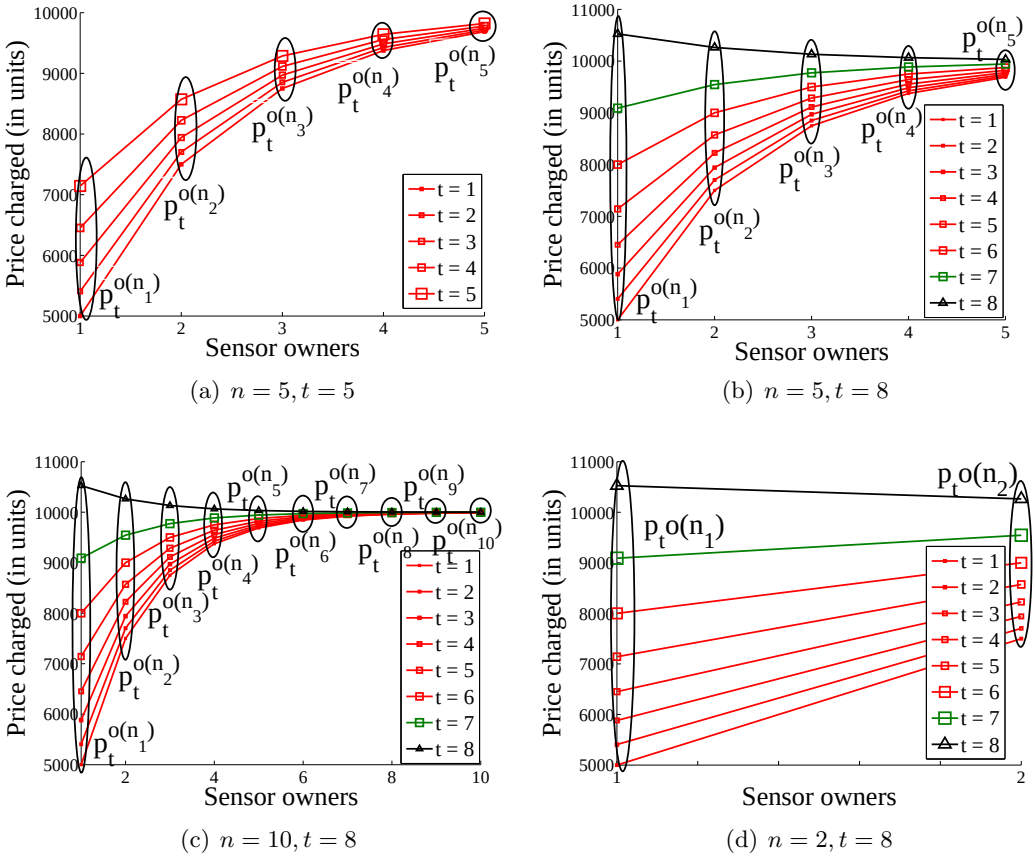
Figure 5.3: Comparative study of performance in terms of network parameters

maintains a receipt after every transmission. The computation overhead is less, and is mainly because of the processing and generation of the receipt. Unlike PPM, and Sprite, the energy consumption due to computation in pH is primarily handled at the cloud-end. The computational parameters are periodically fed to the sensor-cloud end through control packets (as per the assumptions of the model). Thus, the energy consumption due to computation within the physical sensor nodes is the least in pH. As observed in Figure 5.3(a), Sprite leads in terms of the energy expenditure due to transmission. This is because Sprite periodically communicates with the CCS, sending packets containing

the receipts of the currency to be obtained by every physical node. For both PPM, and pH, the energy expended due to transmission is significantly low. However, in PPM, retransmission of packet is required sometimes because of under-estimation of nuglets, thereby incurring an additional energy overhead. The overall effect is indicated by the line-plots for the total energy expenditure.

Figure 5.3(b) compares pH, PPM, and Sprite in terms of packet delivery rate. For 10 end-users, $n = \{100, 200, 300, 400, 500\}$ number of nodes, every node is allowed to transmit data to the BS, under identical channel conditions using pH, PPM, and Sprite. PPM estimates the nuglets before start of packet transmission. However, sometimes due to underestimation of the nuglets, the packets are dropped midway. On the other hand, Sprite, periodically transmits the receipt of the messages from each node to the CCS, thereby overloading the network, and reducing the packet delivery rate. However, for pH, pricing does not affect the network load at all. The prices charged are transmitted along with the data packets to the cloud-end. The computation, and the monetary transactions are handled outside the network, which increases the chance of the packet delivery rate. Figure 5.3(b) depicts the variation of the packet delivery rate with the increase in the number of nodes. For every iteration, the experiment is repeated for 50 times and the data plot is shown within a 95% Confidence Interval (CI).

The price charged at various time instants by different sensor owners for a single end-user is also shown. Figure 5.4 highlights the sequence of the price charged and the point at which the optimality is reached. Figure 5.4(a) demonstrates a 5-hop scenario ($n = 5$) involving 5 different sensor owners, where $o(n_1)$ is the owner of the source sensor node. As indicated in the figure, $o(n_i)$ initially charges a price, based on which the price charged by $o(n_{i+1})$ depends. The price charged at $t = 1$ increases with time. However, it does not exceed the equivalent user utility, that has been assumed to be 10000. Thus, the tendency is to reach the user-utility as close as possible, but not exceed it. For the sake of simulation, a new metric is defined below.



(a) $n = 5, t = 5$ (b) $n = 5, t = 8$

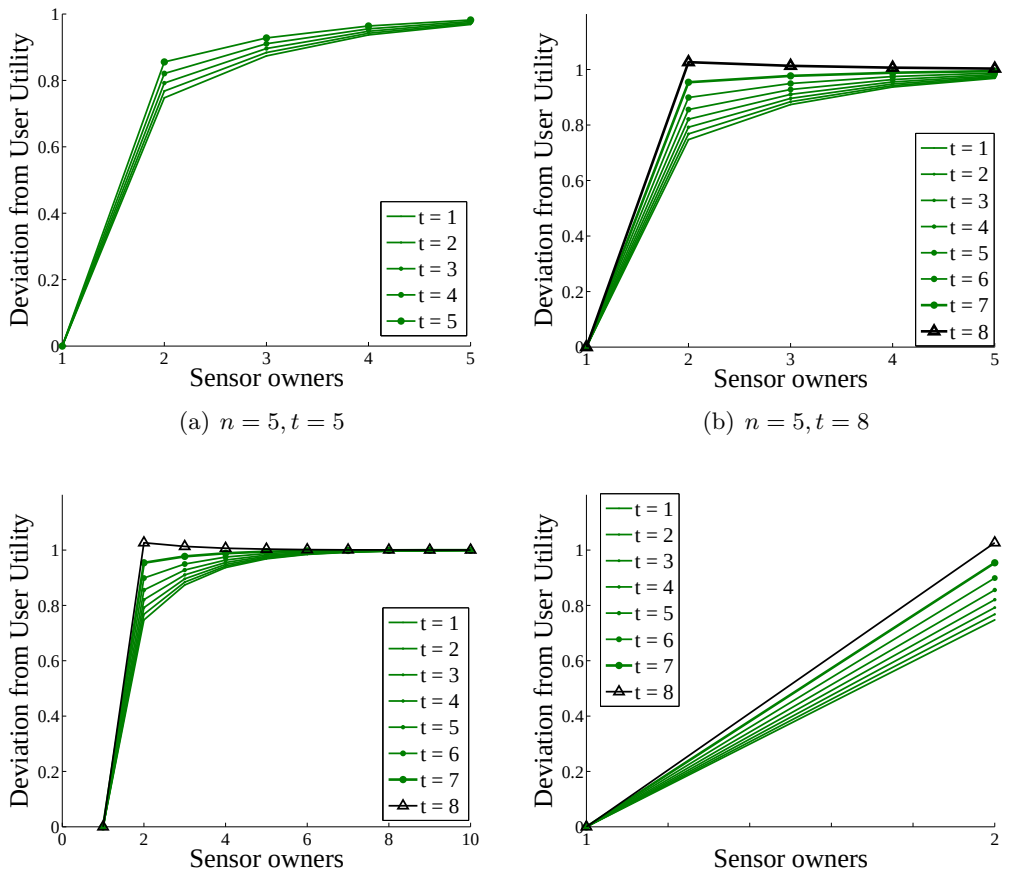
(c) $n = 10, t = 8$ (d) $n = 2, t = 8$

Figure 5.4: Analysis of price charged (due to hardware) with time

Definition 20. *Deviation from the user utility (d) is a metric in the scale of 0 to 1 that indicates the degree of convergence of the price charged by the sensor owners to the utility. It is computed as,*

$$d = 1 - \frac{\mathcal{U} - p^{o(i)}}{\gamma_2 - \gamma_1} \tag{5.39}$$

Practically, $d \to 1$, but $d \neq 1$. Corresponding to Figure 5.4(a), Figure 5.5(a) shows the tendency of convergence of the price charged with the user utility. In Figure 5.4(b), the experiment was done for $n = 5$, $t = 8$. At $t = 8$, it is found that $o(n_1)$ exceeds the user-utility. From this, it is concluded that the price charged by the sensor owners attains optimality at $t = 7$, for this simulation setup. Figure 5.5(b) indicates the asymmetry



(a) $n = 5, t = 5$ (b) $n = 5, t = 8$

(c) $n = 10, t = 8$ (d) $n = 2, t = 8$

Figure 5.5: Analysis of the tendency of the charged price to converge with the user utility

of the pattern at $t = 8$, as $d > 1$. To infer with generality, the same experiment is performed over a different setup, where $n = 10$, $t = 8$, as shown in Fig 5.4(c). Even with the increase in the number of hops, it is found that that equilibrium is reached at $t = 7$. Figure 5.5(c) supports the equilibrium at $t = 7$. Figs. 5.4(d), and Figure 5.5(d) demonstrate the same effect with a setup of $n = 2$, $t = 8$. Thus, the system attains its equilibrium at $t = 7$. Hence, without the loss of generality, it can be inferred that for a particular network setup, the system attains equilibrium after a finite period of time t_f, after which the sequence $\{p_t^{o(n_i)}\}$ stabilizes, i.e., $\forall t \geq t_f, p_{t+1}^{o(n_i)} = p_t^{o(n_i)}$, and $p_t^{o(n_i)} = p_{t_f}^{o(n_i)}$.

5.4.3 Analysis of pI

This subsection puts forth the performance analysis of pI. pI primarily provides the pricing scheme for the infrastructure of virtualization. The experimentation setup for pI is illustrated in Table 5.4.

Table 5.4: Experimentation setup for pI

Parameters	**Values**
Building cost of VM	4 unit
Building cost of vs	3 unit
Price of VM per unit	5 unit
Price of vs per unit	4 unit
Maintenance Cost of vs per time slot	2 unit
Number of end-users	[1, 10]
Number of VMs per user	1
Service rate of CSP	15 demand/sec
β	0.5

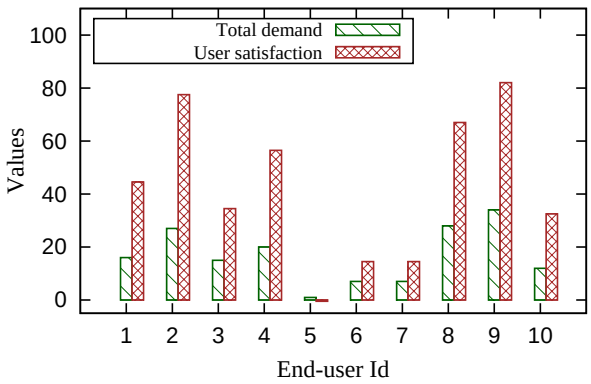
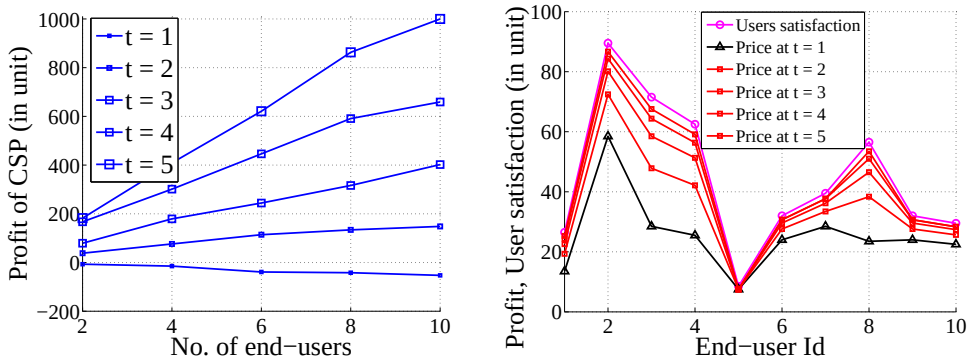


Figure 5.6: Analysis of demand and user satisfaction

Figure 5.6 shows the demand and the user satisfaction $u_e(t)$ provided by the cloud for 10 end-users. As per Definition 18, and also following Figure 5.6, it is evident that the $u_e(t)$ varies with the demand λ_{vs_i}. The increase in $u_e(t)$ is clearly reflected by the increase in demand for the virtual sensors. However, if the demand is too small, as in the case of end-user 5, the processing overhead at the sensor-cloud end increases,



(a) Number of end-users vs profit of CSP (b) Increment in price towards user satisfaction

Figure 5.7: Overall analysis of the profit made by the CSP

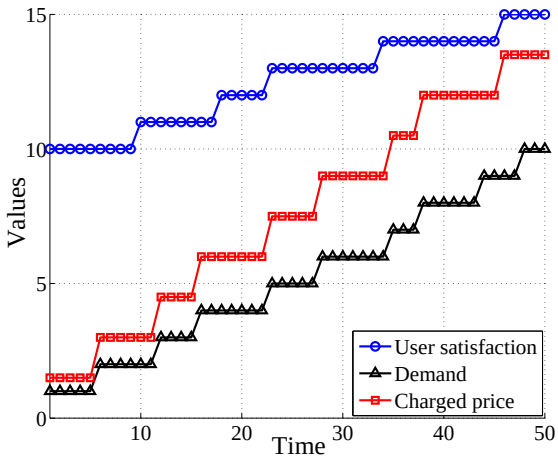
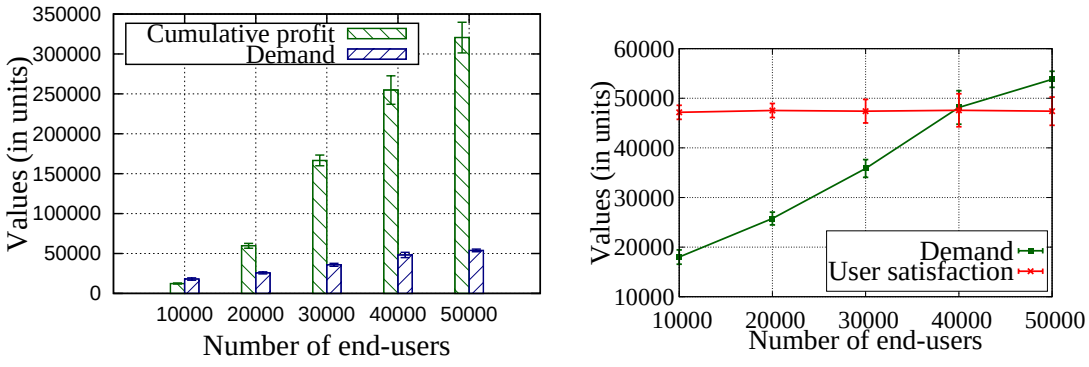


Figure 5.8: Analysis of the correlation of price, demand, and user satisfaction

thereby reducing the user satisfaction. Figure 5.7(a) illustrates the variation of the profit acquired by the CSP, with time. Initially, at $t = 1$, the CSP runs at a loss for serving the requests of 10 end-users, indicated by the negative y axis. Only after a period of time, i.e., $t = 2$ onwards, significant profit is incurred with the increase in the number of end-users. Figure 5.7(b) illustrates the timely increment of the price charged by the CSP, for a fixed user satisfaction, and a fixed demand for 5 time instants. Clearly, end-users 2 and 3 face 4 increments in the charged price, whereas the price charged from end-users 9, and 10 are incremented only thrice. This is because, the CSP has a

tendency to charge a price close to $u_e(t)$, but not exceed it. This strategy ensures that the end-users are not over-charged with time. The user satisfaction value of end-user 5 is significantly low (because of low demand and low data urgency), and hence, the CSP does not get the opportunity to increase the charged price with time. To examine the stability of the proposed system, the system was simulated for a longer period of time, i.e., for 50 time units, for a single end-user, as shown in Figure 5.8. The increasing demand λ_{vs_i} of the end-user, and the corresponding satisfaction $u(t)$ are shown. The price charged by the CSP increases with the increase in λ_{vs_i}. However, at $t = 44$, it can be seen that the price remains constant, i.e., $p_{vs_i}(45) = p_{vs_i}(44) = p_{vs_i}(43)$, although the demand increases ($\lambda(44), \lambda(45) > \lambda(43)$, and $u(43) = u(44) = u(45)$), mainly to prevent the price from exceeding the user satisfaction.

5.4.3.1 Scalability Analysis



(a) Study of demand and cumulativeprofit

(b) Study of demand and user satisfaction

Figure 5.9: Analysis of scalability of the system

Motivated by the works of [16,17], for analysis of the system scalability, an experiment was performed on an increased set of end-users, as shown in Figure 5.9. The experiment involves 10,000 to 50,000 end-users, denoted by e_{tot}. The total demand (λ_{tot}) for the end-users are also varied in terms of the number of virtual sensors allocated and is computed as, $\lambda_{tot} = \sum_{j=1}^{e_{tot}} \sum_{i=1}^{n_{e_j}} \lambda_{vs_i}^{e_j}$ where, n_{e_j} is the total number of component physical

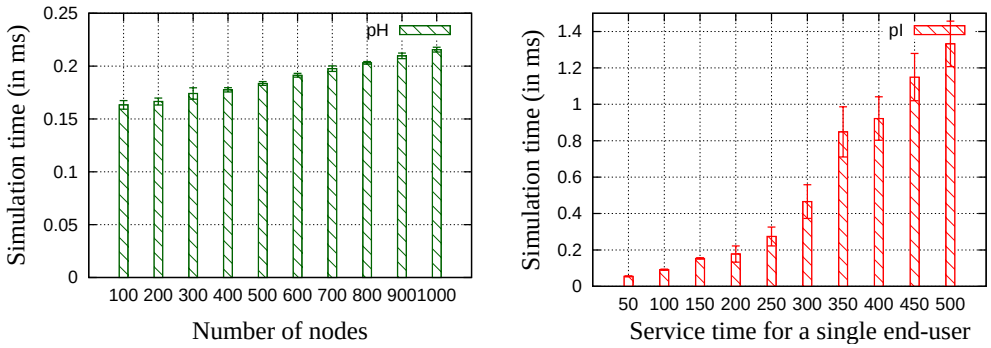
sensor nodes of vs_i for e_j. With the change in the request for the vs_i, the number of the allocated physical sensors is altered and by changing the number of allocated vs_i, $\lambda_{vs_i}^{e_j}$ is altered for multiple end-users. As depicted in Figure 5.9(a), with the increase in λ_{tot}, the profit of the CSP $r(t)$ increases as per Equation (5.23). Therefore, the cumulative profit over all the end-users also increases and is evaluated as $\sum_{j=1}^{e_{tot}} r(t)_{e_j}$. However, as illustrated through Figure 5.9(b), it is observed that the average user satisfaction $u_e(t)$ is above the threshold v_{opt} and remains almost unchanged with the increase in demand. It is also observed that with $10,000$ and $20,000$ end-users, the $u_e(t)$ has a mean of approximately $47,500$ and lies within the interval of $[48,900, 45,600]$ with 95% confidence. However, at larger demands, the mean user satisfaction tends to lie at a slightly wider interval of $[44,000, 51,000]$ with 95% confidence, but, the mean satisfaction stands at $47,400$. From this it is inferred that even with the increase in larger demands for a greater and varying number of end-users, the CSP incurs an increasing positive profit and the user satisfaction is simultaneously maintained. This justifies the scalability of the system.

5.4.3.2 Complexity Analysis

In this subsubsection, the asymptotic computational complexity of pH, and pI are analyzed to examine its real-time processing ability. The complexity of computation is measured in terms of the simulation time required for the execution of the algorithms. Figure 5.10(a) demonstrates the variation in the computational time with the increase in the number of the underlying physical sensor nodes. The mean simulation time is observed to be within the interval $[0.27, 0.82]$ with 95% confidence. Thus, it is found that the increase in the number of the physical sensor nodes has significantly low impact of the computational complexity of pH.

Figure 5.10(b) depicts the computational complexity of pI. The experiment is executed for serving the requests of a single end-user with varying demands for a varied period of time, from 50 to 500 time instants, and the corresponding execution time is

calculated and analyzed to examine the computational complexity of pH. The mean simulation time was found to lie between $[0.21, 0.81]$ with 95% confidence. Therefore, both pH and pI are suitable for real-time implementation.



(a) Number of physical nodes vs simulation time (b) Number of physical nodes vs simulation time

Figure 5.10: Overall analysis of the profit made by the CSP

5.5 Summary

In this Chapter, a dynamic pricing model was proposed for rendering Se-aaS. The proposed pricing model was divided into two different sections – pH and pI. pH deals with the pricing scheme for hardware with the aim to maximize the profit of several sensor owners involved in the data transmission. It presents the pricing scheme for maximizing the profit of the CSP, by considering the user satisfaction at different time instants. A comparative study of the next hop selection is done for pH with PPM, and Sprite. It is observed that pH outperforms the aforesaid models in terms of residual energy, proximity with BS, RSS, and overhead. Moreover, pH reduces the cumulative energy consumption, and increases the packet delivery rate. pI focuses on the pricing due to infrastructure and takes into account the different costs incurred due to building and maintenance of VMs and VSs within sensor-cloud. The analysis of pI shows how CSP incurs profit and the user satisfaction is also met, simultaneously. Finally, the complexity analysis of pH and pI are also performed and an analysis id performed to justify the

real-time processing abilities of the algorithms.

Chapter 6

Optimal Data Center Scheduling for QoS Management in Sensor-cloud

As mentioned in the previous Chapters, in sensor-cloud platforms, for every request from a particular application, a set of physical sensors is allocated. Each set of allocated physical nodes serving a particular application form a Virtual Sensor (VS). Intuitively, the set of VSs serving a particular application span across multiple regions. As the data from multiple VSs are channelized into different cloud DCs, multiple DCs which are spread geographically across the globe may get involved in the process.

In sensor-cloud, the VSs (serving a particular application) are stored inside a single VM, within a particular DC. Therefore, it becomes essential to migrate the data of different VSs (temporarily stored within geographically scattered DCs) to a single VM residing within a particular DC that would serve the application. Obviously, the need for an application specific scheduling of a particular sensor-cloud DC is realized.

For example, as shown in Figure 6.1, it is observed that n number of VSs at different regions serve an application. Data from the physical sensor nodes constituting a single

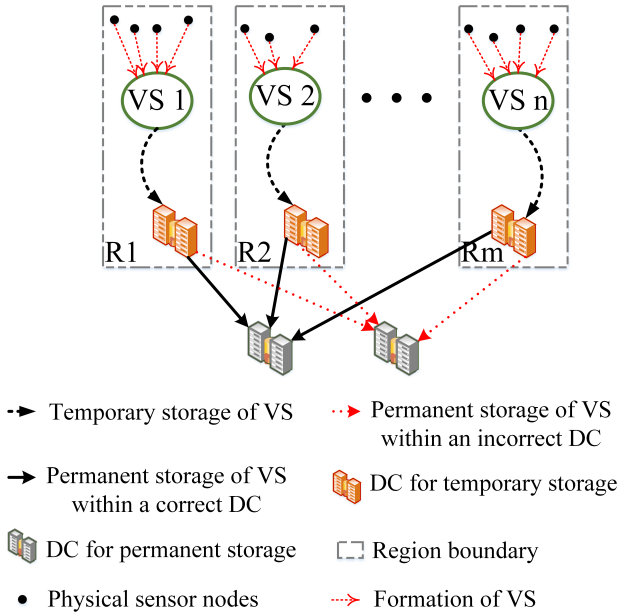


Figure 6.1: Different storage types in DCs

VS are transmitted to a DC for temporary storage. In real-life scenarios, the DCs have the maximum proximity with the physical sensor nodes. However, it is required to create a VM within a single DC to serve the application. Therefore, it is eventually required to transmit these VSs (from the DCs enabling temporary storage) to a single DC, in which the VM, serving the application, resides. In such a scenario, a random selection of a DC, as also indicated by the figure, to serve an application, might be unjust and inappropriate, as it incurs huge networking overhead from the underlying sensor networks to the cloud platform, thereby reducing the Quality of Service (QoS). It is imperative to optimize the performance of the application by analyzing and selecting the DC that provisions with the maximum QoS.

6.1 Contributions of the Chapter

This Chapter focuses on the networking dimensions of sensor-cloud, which is grossly unexplored till date. The above-mentioned problem arises from the need to select a single

DC for serving a particular application. This work focuses on a dynamic scheduling of DCs, given a particular application, and a set of geographically scattered DCs. While scheduling a particular DC, the QoS of the application is also taken into account.

Initially, the proposed work quantifies the QoS to be offered to an application by sensor-cloud in terms of the migration cost within the DCs, the delivery cost to the application from the scheduled DC, and the overall service delay of provisioning Se-aaS. User satisfaction also accounts for the effective QoS. Finally, the process for scheduling of DC is performed, and the QoS is also simultaneously maintained.

This Chapter addresses the problem by a collective decision making of various geographically distributed DCs. While arriving at a final solution, the work assumes the fallible decision making ability of the DCs; thereby, orienting the proposed problem to fit well with the real-life scenarios. The proposed solution for scheduling a DC also takes into account the four different types of asymmetry arising due to two different states of nature (good or bad), and two different alternatives of the DCs, while making a decision (yes or no), which is elaborately discussed in Section 6.4. This provides substantial credibility of the solution to be applicable de facto.

6.2 Problem Description

The problem scenario considers one or more end-user applications requesting for various types of sensor data from different regions in the form of Se-aaS. In case of Se-aaS, the end-users generally request for sensors through web templates. In return, the sensor-cloud service provider allocates physical sensor nodes and forms virtual sensors. Eventually, the sensed data is provisioned as a service to the end-users. It appears to the end-users that s/he is being served with dedicated physical sensor nodes as per requirement. The requests from the different applications are interpreted and the physical sensors are allocated accordingly, as shown in Figure 6.2. The figure clearly indicates the projection of the physical hosts within a DC and the projection of VMs within a physical

host. As mentioned in the earlier Sections, for every distinct request from the end-user, a distinct VS is formed. As the VSs are spread across multiple regions, the data from the VSs are temporarily stored into the closest sensor-cloud DC. A sensor-cloud DC is essentially a cloud DC that renders Se-aaS. However, the principle of a sensor-cloud is to serve a particular application with the data from multiple VSs residing within a single VM [129]. Therefore, to provide Se-aaS, it is required to build a VM within a single DC.

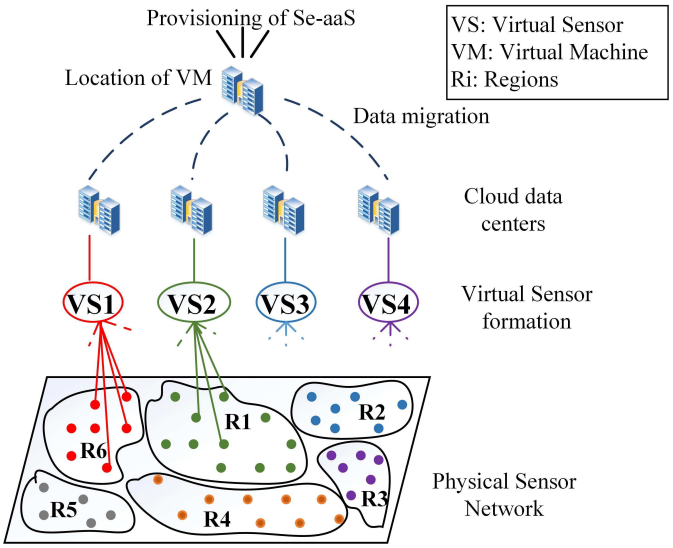


Figure 6.2: Diagrammatic representation of the problem scenario

The goal of this work is to identify the DC within which the VM will be allocated. Scheduling a DC essentially means the selection of a particular DC that will serve an end-user through its VM and VSs. The VSs, that are temporarily scattered within different DCs, are migrated to the particular VM, selected for serving a particular application. In the process of selection of the DC, the service delay of the end-user, as well as the migration cost of the DCs are optimized, thereby ensuring that the QoS is preserved.

6.3 Formal Definition of the Problem

A set $\mathcal{D}$ of ω DCs is considered within a sensor-cloud, such that, $\mathcal{D} = \mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_\omega$, located at regions $R_1, R_2, ..., R_\omega$, respectively, as shown in Figure 6.3. Table 6.1 is a notation table that illustrates the significant functions and variables used in the proposed system. An application App_i creates m different types of requests for Se-aaS. The VSs formed are represented as, $\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_m$, and are stored in temporary DCs, $\mathcal{D}^m = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_m\}, (m \leq \omega)$, respectively. Temporary DCs refer to those that temporarily store the data of the geographically scattered VSs. The data from these DCs are eventually transmitted to some other final DC where the VM of the particular user will be created. The objective of our work is to select a DC, $\mathcal{D}_w^{App_i}$, for App_i by maximizing QoS.

Table 6.1: Table of notation

Parameters	Values
$\mathcal{D}$	Set of DCs
R	Set of regions
ω	Number of DCs
$d_{i,j}$	Distance between $\mathcal{D}_i$ and $\mathcal{D}_j$
$\mathcal{V}_i$	VS stored in $\mathcal{D}_i$
$\mathcal{D}_w^{App_i}$	Winner DC for application App_i
$\mathcal{M}(\mathcal{D}_i, \mathcal{D}_j)$	Migration cost from $\mathcal{D}_i$ to $\mathcal{D}_j$
$\mathcal{L}(\cdots)$	Latency involved in delivering a packet from $\mathcal{D}_i$ to $\mathcal{D}_j$
$(l_{1,j}, l_{2,j})$	Absolute location of App_j
$(\mathcal{D}_i.x, \mathcal{D}_i.y)$	Location coordinates of $\mathcal{D}_i$
η_2	Transmission rate from a DC to an end-user
$\delta(\cdots)$	Delivery cost
$\mathcal{S}(\cdot)$	Service delay
$\mathcal{Q}(App_i)$	QoS offered to App_i
$\mathcal{Q}_{net}(App_i)$	Effective QoS offered to App_i
$\mathcal{U}(\cdot)$	User dissatisfaction-delay product
λ	Demand rate
$\mathcal{P}(\cdot, \cdot)$	Payoff for approving or disapproving DCs
$\mathcal{D}^{nom}$	Set of nominated DCs
X_{ij}	Decision outcome of $\mathcal{D}_i$ for $\mathcal{D}_j$
W	System events
Υ_i	Decision making ability of $\mathcal{D}_i$
$\mathcal{J}_{\mathcal{D}_i}$	Decision profile of $\mathcal{D}_i$

To ensure QoS, initially, a metric of QoS is designed for a particular application. The

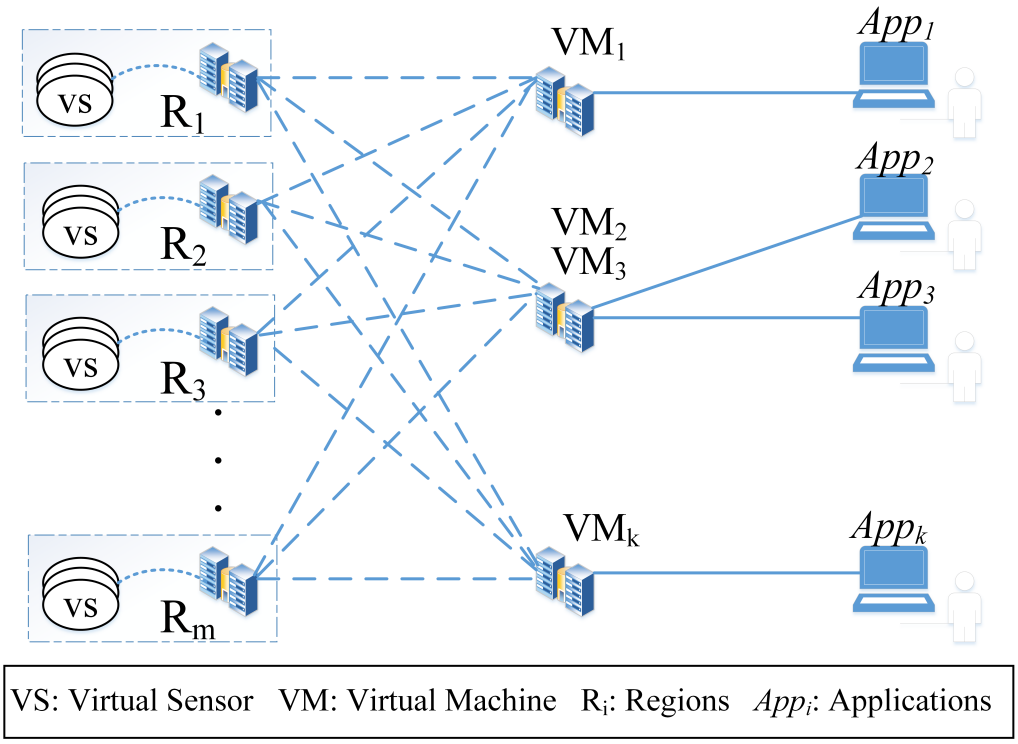


Figure 6.3: Network model of the problem scenario

QoS of Se-aaS is designed by considering several related factors, as described below.

Definition 21. *The migration cost of sensor data from DC $\mathcal{D}_i$ to DC $\mathcal{D}_j$ for transmitting p packets is denoted by $\mathcal{M}(\mathcal{D}_i, \mathcal{D}_j)$, and is defined in terms of the latency $\mathcal{L}$ involved in delivering a packet from $\mathcal{D}_i$ to $\mathcal{D}_j$. The function $\mathcal{M}(\cdot,\cdot)$ is expressed as:*

$$\mathcal{M}(\mathcal{D}_i, \mathcal{D}_j) = \mathcal{L}, \tag{6.1}$$

where,

$$\mathcal{L}(\mathcal{D}_i, \mathcal{D}_j, p) = p \times P \times d(i,j)/\eta_1. \tag{6.2}$$

The function $\mathcal{L}(\mathcal{D}_i, \mathcal{D}_j, p)$ is the latency involved in migrating p packets, (each of size P byte), from $\mathcal{D}_i$ to $\mathcal{D}_j$. The variable η_1 is the migration latency of unit byte meter per second.

Definition 22. *The delivery cost of p packets from $\mathcal{D}_i$ located at $(\mathcal{D}_i.x, \mathcal{D}_i.y)$ to an application App_j at absolute location $(l_{1,j}, l_{2,j})$ is denoted as $\delta(\mathcal{D}_i.x, \mathcal{D}_i.y, l_{1,j}, l_{2,j})$, and is expressed as:*

$$\delta(\mathcal{D}_i.x, \mathcal{D}_i.y, l_{1,j}, l_{2,j}) = \left(\sqrt{(\mathcal{D}_i.x - l_{1,j})^2 + (\mathcal{D}_i.y - l_{2,j})^2}\right)/\eta_2, \tag{6.3}$$

where η_2 is the propagation speed in meter per second through the link connecting a DC to an end-user. Therefore, δ is finally expressed in second.

Definition 23. *The service delay of App_j, $\mathcal{S}(App_j)$, is the summation of its migration cost and the delivery cost. It is defined as:*

$$\mathcal{S}(App_j) = \sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_s) + \delta(\mathcal{D}_s.x, \mathcal{D}_s.y, l_{1,j}, l_{2,j}), \tag{6.4}$$

where the data of the VSs are migrated to $\mathcal{D}_s$ from multiple DCs, $\{\mathcal{D}_i\}, 1 \leq i \leq m$.

Now, the QoS offered (in byte per second) for transmission of p packets to an application is defined to be proportional to the costs due to migration, and data delivery. Therefore, the QoS offered to App_j by a sensor-cloud is given as:

$$\mathcal{Q}(App_j) = \frac{pP}{\left[\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_s) + \delta(\mathcal{D}_s.x, \mathcal{D}_s.y, l_{1,j}, l_{2,j})\right]}. \tag{6.5}$$

However, the above metric of QoS does not consider user satisfaction into account. Therefore, $\mathcal{Q}$ is modified as $\mathcal{Q}_{net}$, as described below.

Definition 24. *The user dissatisfaction-delay product, $\mathcal{U}(\mathcal{V}_k)$, of obtaining data from each $\mathcal{V}_k$ takes into account the approximate time for processing the data of $\mathcal{V}_k$ and is defined as the product of the mean service delay in provisioning data from $\mathcal{V}_k$ and the corresponding demand rate λ_k. It is expressed as:*

$$\mathcal{U}(\mathcal{V}_k) = \lambda_k \times \frac{\mathcal{S}(App_j)}{\sum_{i=1}^{m} \lambda_k}, \tag{6.6}$$

where $\frac{\mathcal{S}(App_j)}{\sum_{i=1}^{m} \lambda_k}$ *accounts for the service delay of* App_j *for* $\mathcal{V}_k$.

Consequently, p packets are transmitted in an overall time of $\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_s) + \delta(\mathcal{D}_s, x, \mathcal{D}_s.y, l_{1,i}, l_{2,i}) + \sum_{i=1}^{m} \mathcal{U}(\mathcal{V}_k)$.

Definition 25. *The effective QoS of application* App_j *served by* $\mathcal{V}_k$ *is mathematically expressed as:*

$$\mathcal{Q}_{net}(App_j) = \frac{pP}{\mathcal{S}(App_j) + \mathcal{U}(\mathcal{V}_k)}$$

$$= \frac{pP}{\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_s) + \delta(\mathcal{D}_s.x, \mathcal{D}_s.y, l_{1,i}, l_{2,i}) + \sum_{i=1}^{m} \left(\lambda_k \times \frac{\mathcal{S}(App_j)}{\sum_{i=1}^{m} \lambda_k} \right)}. \tag{6.7}$$

Therefore, the final objective function of our work is stated as:

$$f : \mathcal{D}^m \rightarrow \mathcal{D}, \mathcal{D} = \{\mathcal{D}_i\}, 1 \leq i \leq m \leq d, \tag{6.8}$$

which maps a set of DCs to a single $\mathcal{D}_w$ and $\mathcal{D}^m$ is the set of all the m DCs as denoted previously. The function f maximizes the effective QoS, $\mathcal{Q}_{net}$, of an application. Therefore, for a particular application App_j, f belongs to the solution set of the maximization function:

$$\arg\max \left(\mathcal{Q}_{net}(App_j) \right), \tag{6.9}$$

where $\mathcal{Q}_{net}(App_j)$ is obtained from Definition 25. Having obtained the formal objective of our work in Equation (6.8) and Equation (6.42), the approach towards achieving the

solution is propounded in Section 6.4.

It is to be noted here that, the problem definition of the proposed work considers all applications of sensor-cloud, i.e., it targets all sensor-based applications (e.g., environmental monitoring, surveillance applications, multimedia applications, and so on). As different applications possess different demands (especially in terms of sensor hardware and configurations), sensor-cloud manages the inter-operability and compatibility issues which have been already discussed in Chapter 3.

6.4 System Model

Motivated by the *Optimal Decision Rule*, as illustrated in [130], the "general pairwise choice framework" is considered to be implemented over the cloud network.

6.4.1 Optimal Decision Rule

The optimal group decision-making considers the fallibility in human nature within a fixed-sized committee. Such rules are applicable for selection of investment projects in economic organizations, for design of reliable systems, and for decision making in other political or legal applications. The general pairwise choice framework of Optimal Decision Rule takes into account the four possible types of asymmetry. For example, in a n-member committee of an organization, a decision making is required for accepting/rejecting a good/bad project. Hence, the decision of the committee has four potential alternatives. The Optimal Decision Rule focuses to combine the individual opinions (assuming the fallibility of humans) and generate the rule that maximizes the payoff of the organization in terms of the profit obtained. In this work, the Optimal Decision Rule is analogously used. The rationale of the analogy is illustrated below.

6.4.2 Proposed Model

Each DC of the set $\mathcal{D}_{App_i} \subseteq \mathcal{D}$ ($\mathcal{D}_{App_i} = \{\mathcal{D}_i\}, 1 \leq i \leq m$), involved in temporary storage and processing of m VSs participate in a decision rule to elect a particular DC from a nominated set of DCs, $\mathcal{D}^{nom}$, to serve App_i. As per the *Optimal Decision Rule*, two possible states of nature are assumed for every nominated DC – good (1) and bad (-1). The payoff associated with the approval of a good DC and the disapproval of a bad DC is denoted by $\mathcal{P}(1:1)$ and $\mathcal{P}(-1:-1)$, respectively. As the model assumes four types of asymmetry, while making a pairwise choice, the payoffs associated with the approval of a bad DC and the disapproval of a good DC are also considered, and expressed as, $\mathcal{P}(1:-1)$ and $\mathcal{P}(-1:1)$, respectively.

6.4.2.1 Assumptions of the model

- Every DC, $\mathcal{D}_i \in \mathcal{D}_{App_i}$, is heterogeneous in terms of its decision making ability.
- Each DC, $\mathcal{D}_i \in \mathcal{D}_{App_i}$, possesses imperfect decision making ability.
- In the context of pairwise choice, four types of asymmetry are possible.
- The decision of $\mathcal{D}_{App_i}$ and the individual DCs have a binary interpretation.
- The likelihood of a nominated DC being good is a time variant, *apriori* probability (α_t), $\alpha_o = 1/2$.
- The load capacity of a DC is known to the other DCs of set $\mathcal{D}$.

To form $\mathcal{D}^{nom}$, the mean distance of each data centers from the location of the user application App_i ($L =< l_{i,1}, l_{i,2} >$) is initially computed, and denoted by $\xi_{avg}^{App_i}$. Therefore,

$$\xi_{avg}^{App_i} = \frac{\sum_{j=1}^{\omega} d(\mathcal{D}_j, L)}{\omega} \tag{6.10}$$

Consequently, $\forall \mathcal{D}_j \in \mathcal{D}$, $\mathcal{D}_j \in \mathcal{D}^{nom}$, if and only if:

$$d(\mathcal{D}_j, L) \leq \xi_{avg}^{App_i} \tag{6.11}$$

X_{ij} is the decision outcome of $\mathcal{D}_i$ for $\mathcal{D}_j$. Now, $X_{ij} = \{1, -1\}$ at a particular time, $\mathcal{D}_i \in \mathcal{D}_{App_i}, \mathcal{D}_j \in \mathcal{D}^{nom}$. X_{ij} is modeled as:

$$X_{ij}(t) = \begin{cases} 1, & \text{if } (d(ij) \leq d_{avg}^i) \wedge (\mathcal{C}(\mathcal{D}_j) \leq \mathcal{C}_{avg}^{nom}) \\ -1, & \text{otherwise} \end{cases} \tag{6.12}$$

where d_{avg}^i is the mean distance of $\mathcal{D}_i$ from the other DCs of the system and is expressed as:

$$d_{avg}^i = \frac{\sum_{k=1}^{\omega} d(i,k)}{\omega - 1}, \forall \mathcal{D}_k \in \forall \mathcal{D}, i \neq k, \tag{6.13}$$

and $\mathcal{C}(\mathcal{D}_j)$ computes the current load of DC $\mathcal{D}_j$ given as:

$$\mathcal{C}(\mathcal{D}_j) = \mid \mathcal{A}_j \mid \tag{6.14}$$

where $\mathcal{A}_j$ is the current set of applications served by $\mathcal{D}_j$. Therefore, $\forall \mathcal{D}_j \in \mathcal{D}^{nom}$, the following holds true:

$$\mathcal{C}_{avg}^{nom} = \frac{\sum_{j=1}^{|\mathcal{D}^{nom}|} \mathcal{C}(\mathcal{D}_j)}{\mid \mathcal{D}^{nom} \mid} \tag{6.15}$$

Definition 26. *The events of our model are:*

(i) $W_1^i = \{1, -1\}$*: Approving or disapproving a DC by* $\mathcal{D}_i$

(ii) $W_2^j = \{1, -1\}$*: A DC* $\mathcal{D}_j$ *appearing as a"good" or "bad"*

Definition 27. *The correct decision making ability of a DC* $\mathcal{D}_i$*, is denoted by* Υ_i*, and*

is expressed as:

$$\Upsilon_i^+ = P(W_1^i = 1 \mid W_1^j = 1) = P(1:1) \tag{6.16}$$

$$\Upsilon_i^- = P(W_1^i = -1 \mid W_1^j = -1) = P(-1:-1) \tag{6.17}$$

Definition 28. *The incorrect decision making ability of a DC $\mathcal{D}_i$, is denoted by Υ_i', and is expressed as:*

$$\Upsilon_i'^+ = P(W_1^i = 1 \mid W_1^j = -1) = P(1:-1) \tag{6.18}$$

$$\Upsilon_i'^- = P(W_1^i = -1 \mid W_1^j = 1) = P(-1:1) \tag{6.19}$$

However, a bias exists, that is expressed as:

$$\Upsilon_i^+ > \Upsilon_i'^+, \Upsilon_i^- > \Upsilon_i'^- \tag{6.20}$$

The probability of approving a good DC is greater than that of a bad DC and the probability of disapproving a bad DC is greater than that of a good DC. Having defined the decision making abilities of every DC, a "good" and a "bad" DC are formally defined. The "goodness" or "badness" is studied only for the elements of $\mathcal{D}^{nom}$. The criteria for "goodness" of a DC, $\mathcal{D}_j \in \mathcal{D}^{nom}$, for a particular application App_k, holds true if the total number of positive decisions for $\mathcal{D}_j$ exceeds a pre-negotiated threshold (X_{th}). That is to say that, if

$$\sum_{i=1}^{m}(X_{ij}+1)! - 1, \text{ such that } \forall \mathcal{D}_i \in \mathcal{D}_{App_k}, \tag{6.21}$$

where $(X_{ij}+1)! - 1$ returns 1 or 0 for $X_{ij} = 1$ or -1, respectively.

Definition 29. *The metric of "goodness" of $\mathcal{D}_j$, $\mathcal{G}(\mathcal{D}_j)$, is defined as the ratio of the difference of the current and the maximum load to the maximum load of $\mathcal{D}_j$ that it*

supports. Therefore,

$$\mathcal{G}(\mathcal{D}_j) = \frac{\mathcal{C}^{max}(\mathcal{D}_j) - \mathcal{C}(\mathcal{D}_j)}{\mathcal{C}^{max}(\mathcal{D}_j)}, 0 \leq \mathcal{G}(\mathcal{D}_j) \leq 1 \tag{6.22}$$

The metric of "badness" of $\mathcal{D}_j$, $\bar{\mathcal{G}}(\mathcal{D}_j)$, is obtained as the complement of the "goodness" of $\mathcal{D}_j$. Therefore,

$$\bar{\mathcal{G}}(\mathcal{D}_j) = 1 - \mathcal{G}(\mathcal{D}_j) = \frac{\mathcal{C}(\mathcal{D}_j)}{\mathcal{C}^{max}(\mathcal{D}_j)}, 0 \leq \bar{\mathcal{G}}(\mathcal{D}_j) \leq 1 \tag{6.23}$$

Consequently, every DC possesses a measure of "goodness", and "badness". Intuitively, a DC that is exactly half loaded has identical metrics for "goodness" and "badness". The proportion of good DCs is:

$$\alpha = \frac{\mathcal{D}_s^{nom}}{\mid \mathcal{D}^{nom} \mid} \text{ such that, } \mathcal{D}_s^{nom} = \{\mathcal{D}_u\}, \mathcal{G}(\mathcal{D}_u) > \bar{\mathcal{G}}(\mathcal{D}_u) \tag{6.24}$$

Now, the estimated probability of approving any DC irrespective of its being "good" or "bad", by $\mathcal{D}_i$ is its decision making ability that is influenced, and affected by the present workload of the DC [131, 132]. Here, $P(W_i^1 = 1)$ is estimated on the basis of learning its ability for the last h instants. Assuming $\mathcal{D}_i$ voted for h distinct $\mathcal{D}_j$s for the last h time instants, the following holds true:

$$\hat{P}(W_i^1 = 1)(t) = \begin{cases} \frac{1}{h} \sum_{j=1}^{h} (X_{ij} + 1)! - 1, & \text{if } t \leq h \\ \frac{1}{t} \sum_{j=1}^{t} (X_{ij} + 1)! - 1, & \text{if } t > h \end{cases} \tag{6.25}$$

where $P(W_i^1 = 1)$ at time t is estimated as the mean decision making ability of $\mathcal{D}_i$ over the last h instants if $t \leq h$. For $t > h$, the mean is computed till the current time instant. Similarly,

$$\hat{P}(W_i^1 = -1)(t) = \begin{cases} \frac{1}{h}\sum_{j=1}^{h}(1 - X_{ij})! - 1, & \text{if } t > h \\ \frac{1}{t}\sum_{j=1}^{t}(1 - X_{ij})! - 1, & \text{if } t \leq h \end{cases} \tag{6.26}$$

where $(1 - X_{ij})! - 1$ returns 0 or 1 for $X_{ij} = 1$ or -1, respectively. Having obtained α, $\hat{P}(W_i^1 = 1)$, $\hat{P}(W_i^1 = -1)$, the values of Υ_i^+ and Υ_i^- are obtained using Bayesian classification [133, 134].

$$\Upsilon_i^+ = P\left(\frac{W_1^i = 1}{W_2 = 1}\right) = \frac{P(W_1^i = 1)P\left(\frac{W_2=1}{W_1^i=1}\right)}{P(W_1^i = 1)P\left(\frac{W_2=1}{W_1^i=1}\right) + P(W_1^i = -1)P\left(\frac{W_2=1}{W_1^i=-1}\right)} \tag{6.27}$$

$$\Upsilon_i^- = P\left(\frac{W_1^i = -1}{W_2 = -1}\right) = \frac{P(W_1^i = -1)P\left(\frac{W_2=-1}{W_1^i=-1}\right)}{P(W_1^i = 1)P\left(\frac{W_2=-1}{W_1^i=1}\right) + P(W_1^i = -1)P\left(\frac{W_2=-1}{W_1^i=-1}\right)} \tag{6.28}$$

The expressions of $\Upsilon_i^{'+}$, and $\Upsilon_i^{'-}$ can be evaluated simply using Equation (6.27) and Equation (6.27), respectively. Therefore,

$$\Upsilon_i^{'+} = 1 - \Upsilon_i^- \tag{6.29}$$

$$\Upsilon_i^{'-} = 1 - \Upsilon_i^+ \tag{6.30}$$

Definition 30. *The decision profile of $\mathcal{D}_{App_i}$ for a particular $\mathcal{D}_j$, is defined as:*

$$\mathcal{J}_{\mathcal{D}_j} = \{X_{ij}\}, \forall \mathcal{D}_i \in \mathcal{D}_{App_i}, \mathcal{J}_{\mathcal{D}_j} \in \mathcal{J}, \mathcal{J} = \{1, -1\}^m \tag{6.31}$$

where $\mathcal{J}$ is the set of all of the possible decision profiles.

The outcome of the aggregation rule is $\mho : \mathcal{J} \rightarrow \{1, -1\}$. Now, the set of decision profiles can be partitioned into $\mathcal{J}_{\mathcal{D}_j}^{\mho+}$ and $\mathcal{J}_{\mathcal{D}_j}^{\mho-}$, where $\mathcal{J}_{\mathcal{D}_j}^{\mho+} = \{\mathcal{J}_{\mathcal{D}_k} \mid \mho(\mathcal{J}_{\mathcal{D}_k}) = 1\}$, and $\mathcal{J}_{\mathcal{D}_j}^{\mho-} = \{\mathcal{J}_{\mathcal{D}_k} \mid \mho(\mathcal{J}_{\mathcal{D}_k}) = -1\}$. Also, $\mathcal{J}_{\mathcal{D}_j}^{\mho+} \bigcup \mathcal{J}_{\mathcal{D}_j}^{\mho-} = \mathcal{J}$ and $\mathcal{J}_{\mathcal{D}_j}^{\mho+} \bigcap \mathcal{J}_{\mathcal{D}_j}^{\mho-} = \emptyset$.

A particular decision profile $\mathcal{J}_{\mathcal{D}_j}$ for $\mathcal{D}_j$ can be partitioned into $A(\mathcal{J}_{\mathcal{D}_j})$ and $R(\mathcal{J}_{\mathcal{D}_j})$, where $X_{ij} = 1, \forall \mathcal{D}_i \in A(\mathcal{J}_{\mathcal{D}_j})$ and $X_{ij} = -1, \forall \mathcal{D}_i \in R(\mathcal{J}_{\mathcal{D}_j})$. If $h(1:1)$, and $h(-1:-1)$ be the respective probabilities of approving or disapproving $\mathcal{D}_j$, then under decision rule $\mho$, the following can be obtained:

$$h(1:1) = \prod_{\mathcal{D}_i \in A(\mathcal{J}_{\mathcal{D}_j})} \Upsilon_i^+ \prod_{\mathcal{D}_i \in R(\mathcal{J}_{\mathcal{D}_j})} (1 - \Upsilon_i^+) \tag{6.32}$$

$$h(-1:-1) = \prod_{\mathcal{D}_i \in R(\mathcal{J}_{\mathcal{D}_j})} \Upsilon_i^- \prod_{\mathcal{D}_i \in A(\mathcal{J}_{\mathcal{D}_j})} (1 - \Upsilon_i^-) \tag{6.33}$$

Now, given a $\mathcal{D}_j$ is "good" or "bad", $\mathcal{D}_{App_i}$ approves or disapproves it for a particular decision profile $\mathcal{J}_{\mathcal{D}_j}$, under decision rule $\mho$ with probability $P_{\mathcal{D}}(\mho : 1)$ and $P_{\mathcal{D}}(\mho : -1)$, respectively. Therefore,

$$P_{\mathcal{D}}(\mho : 1) = P(\mathcal{J}_{\mathcal{D}_j} \in \mathcal{J}_{\mathcal{D}_j}^{\mho+} : 1) \tag{6.34}$$

$$P_{\mathcal{D}}(\mho : -1) = P(\mathcal{J}_{\mathcal{D}_j} \in \mathcal{J}_{\mathcal{D}_j}^{\mho-} : -1) \tag{6.35}$$

However, for Type I and Type II errors in decision making, by the DCs themselves, the following hold true:

$$P_{\mathcal{D}}^e(\mho : 1) = 1 - P_{\mathcal{D}}(\mho : 1) \tag{6.36}$$

$$P_{\mathcal{D}}^e(\mho : -1) = 1 - P_{\mathcal{D}}(\mho : -1) \tag{6.37}$$

From Equation (6.32) and Equation (6.33), it can be obtained that,

$$P_{\mathcal{D}}(\mathcal{O}:1) = \sum_{\mathcal{J}_{\mathcal{D}_k} \in \mathcal{J}_{\mathcal{D}_j}^{\mathcal{O}+}} h(1:1) \tag{6.38}$$

$$P_{\mathcal{D}}(\mathcal{O}:-1) = \sum_{\mathcal{J}_{\mathcal{D}_k} \in \mathcal{J}_{\mathcal{D}_j}^{\mathcal{O}-}} h(-1:-1) \tag{6.39}$$

The goal of our problem is to maximize the expected payoff in terms of the QoS, $\mathcal{Q}_{net}(App_i)$, for the set of all possible aggregation rules F. The payoff associated with the approval or disapproval of a $\mathcal{D}_j$ is modeled proportional to the QoS of the provisioned service to App_i. Therefore,

$$\mathcal{P}(\zeta:\zeta) \propto \mathcal{Q}_{net}(App_i)$$

$$\Rightarrow \mathcal{P}(\zeta:\zeta) = sign(\mathcal{D}_j)\Bigg\{\Big[\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_j) + \delta(\mathcal{D}_j, l_{1,i}, l_{2,i})\Big] - \sum_{i=1}^{m}\Bigg(\lambda_k \times \frac{\mathcal{S}(App_i)}{\sum\limits_{i=1}^{m} \lambda_k}\Bigg)\Bigg\} \tag{6.40}$$

where $\zeta = \{1, -1\}$ and $sign(\mathcal{D}_j)$ is defined as:

$$sign(\mathcal{D}_j) = \begin{cases} 1, & \text{if } \mathcal{G}(\mathcal{D}_j) > \bar{\mathcal{G}}(\mathcal{D}_j) \\ -1, & \text{if } \mathcal{G}(\mathcal{D}_j) \leq \bar{\mathcal{G}}(\mathcal{D}_j) \end{cases} \tag{6.41}$$

Consequently, based on the "goodness" or "badness" of a DC, a positive or negative payoff is computed. Now, following the *Optimal Decision Rule*, our goal is to maximize our expected payoff. Therefore, the goal is modified as:

$$\arg\max_{f \in F} \mathcal{E}(App_i), \tag{6.42}$$

where $\mathcal{E}(App_i)$ is expressed as:

$$\mathcal{E}(App_i) = \alpha\big[\mathcal{P}(1:1)P_{\mathcal{D}}(\mathcal{O}:1) + \mathcal{P}(-1:1)(1-P_{\mathcal{D}}(\mathcal{O}:1))\big]$$
$$+(1-\alpha)\big[\mathcal{P}(-1:-1)P_{\mathcal{D}}(\mathcal{O}:-1) + \mathcal{P}(1:-1)(1-P_{\mathcal{D}}(\mathcal{O}:-1))\big] \tag{6.43}$$

Now, the net effective payoff of for approval of a good DC is $\mathcal{P}(1) = \mathcal{P}(1:1) - \mathcal{P}(-1:1)$. Similarly, the net payoff for disapproving a bad DC is $\mathcal{P}(-1) = \mathcal{P}(-1:-1) - \mathcal{P}(1:-1)$. Therefore, simplifying Equation (6.43), the following is obtained:

$$\mathcal{E}(App_i) = \alpha P_{\mathcal{D}}(\mathcal{O}:1)\mathcal{P}(1) + (1-\alpha)P_{\mathcal{D}}(\mathcal{O}:-1)\mathcal{P}(-1) + \alpha\mathcal{P}(-1:1) + (1-\alpha)\mathcal{P}(1:-1) \tag{6.44}$$

Based on the above, the net goal function can be rewritten as:

$$\arg\max_{f\in F} \mathcal{E}(App_i) = \alpha P_{\mathcal{D}}(\mathcal{O}:1)\mathcal{P}(1) + (1-\alpha)P_{\mathcal{D}}(\mathcal{O}:-1)\mathcal{P}(-1)$$

$$= \alpha\mathcal{P}(1)\sum_{\mathcal{J}_{\mathcal{D}_k}\in\mathcal{J}^{\mathcal{O}+}_{\mathcal{D}_j}} h(1:1) + (1-\alpha)\mathcal{P}(-1)\sum_{\mathcal{J}_{\mathcal{D}_k}\in\mathcal{J}^{\mathcal{O}-}_{\mathcal{D}_j}} h(-1:-1) \tag{6.45}$$

Theorem 6.4.1. *The optimal decision rule $\hat{f}$ of our problem is denoted as:*

$$\hat{f} = \sigma(\varphi + \beta + \rho + \Psi)$$

where

$$\varphi = \ln\frac{\alpha}{1-\alpha}, \beta = \ln\frac{\mathcal{P}(1)}{\mathcal{P}(-1)}$$

$$\rho = \sum_{i=1}^{m} \left[\ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-}(1+X_{ij})! - \ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+}(1-X_{ij})! \right]$$

$$\Psi_i = \ln \frac{\Upsilon_i^+(1-\Upsilon_i^+)}{\Upsilon_i^-(1-\Upsilon_i^-)}, \Psi = \sum_{i=1}^{m} \Psi_i$$

$$\sigma(\chi) = \begin{cases} +1 & , \chi \geq 0 \\ -1 & , otherwise \end{cases}$$

Proof. For any decision profile $\mathcal{J}_{\mathcal{D}_j}$, $\hat{f}(\mathcal{J}_{\mathcal{D}_j}) = 1$ if and only if,

$$\alpha \mathcal{P}(1)h(1:1) > (1-\alpha)\mathcal{P}(-1)h(-1:-1) \tag{6.46}$$

Now, according to the *Optimal Decision Rule* [130] the sufficient condition for the optimality of $\hat{f}$ is satisfied by the partition of $\mathcal{J}$, abiding by the condition:

$$\mathcal{J}_{\mathcal{D}_j}^{\mathcal{O}+} = \left\{ \mathcal{J}_{\mathcal{D}_j} : \hat{f}(\mathcal{J}_{\mathcal{D}_j}) = 1 \right\},$$

$$= \left\{ \mathcal{J}_{\mathcal{D}_j}, \alpha \mathcal{P}(1)h(1:1) > (1-\alpha)\mathcal{P}(-1)h(-1:-1) \right\},$$

$$= \left\{ \mathcal{J}_{\mathcal{D}_j}, \frac{\alpha}{1-\alpha}\frac{\mathcal{P}(1)}{\mathcal{P}(-1)} \prod_{\mathcal{D}_i \in A(\mathcal{J}_{\mathcal{D}_j})} \Upsilon_i^+ \prod_{\mathcal{D}_i \in R(\mathcal{J}_{\mathcal{D}_j})} (1-\Upsilon_i^+) > \prod_{\mathcal{D}_i \in R(\mathcal{J}_{\mathcal{D}_j})} \Upsilon_i^- \prod_{\mathcal{D}_i \in A(\mathcal{J}_{\mathcal{D}_j})} (1-\Upsilon_i^-) \right\},$$

$$= \left\{ \mathcal{J}_{\mathcal{D}_j}, \ln \frac{\alpha}{1-\alpha} + \ln \frac{\mathcal{P}(1)}{\mathcal{P}(-1)} + \sum_{\mathcal{D}_i \in A(\mathcal{J}_{\mathcal{D}_j})} \ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-} > \sum_{\mathcal{D}_i \in R(\mathcal{J}_{\mathcal{D}_j})} \ln \frac{\Upsilon_i^-}{(1-\Upsilon_i^+)} \right\}$$

The optimality condition can be further simplified as:

$$= \ln \frac{\alpha}{1-\alpha} + \ln \frac{\mathcal{P}(1)}{\mathcal{P}(-1)} + \sum_{i=1}^{m} \ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-}((X_{ij}+1)!+1) - \sum_{i=1}^{m} \ln \frac{\Upsilon_i^-}{(1-\Upsilon_i^+)}((1-X_{ij})!-1) > 0,$$

$$= \ln \frac{\alpha}{1-\alpha} + \ln \frac{\mathcal{P}(1)}{\mathcal{P}(-1)} + \sum_{i=1}^{m} \Bigg[\ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-}(1+X_{ij})! - \ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+}(1-X_{ij})! \Bigg] + \sum_{i=1}^{m} \Bigg[\ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-} - \ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+} \Bigg] > 0,$$

$$= \ln \frac{\alpha}{1-\alpha} + \ln \frac{\mathcal{P}(1)}{\mathcal{P}(-1)} + \sum_{i=1}^{m} \Bigg[\ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-}(1+X_{ij})! - \ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+}(1-X_{ij})! \Bigg] + \sum_{i=1}^{m} \Bigg[\ln \frac{\Upsilon_i^+(1-\Upsilon_i^+)}{\Upsilon_i^-(1-\Upsilon_i^-)} \Bigg] > 0$$

Finally, it is obtained that,

$$\mathcal{J}_{\mathcal{D}_j}^{\mathcal{O}+} = \{\mathcal{J}_{\mathcal{D}_j}, \varphi + \beta + \rho + \Psi > 0\} \tag{6.47}$$

From the above, it directly follows that:

$$\mathcal{J}_{\mathcal{D}_j}^{\mathcal{O}-} = \{\mathcal{J}_{\mathcal{D}_j}, \varphi + \beta + \rho + \Psi \leq 0\} \tag{6.48}$$

Therefore, it can be inferred that, $\hat{f} = \sigma(\varphi + \beta + \rho + \Psi)$

□

Clearly, only ρ includes the choices of the individual DCs and assigns weights to the individual preferences. φ, β, and Ψ are the bias elements [130].

6.5 Analytical Results

Proposition 6.5.1. *The mean weight, M(1:-1), assigned to the decision of an individual DC, for the two states of nature, is positive.*

Proof. From Theorem 6.4.1, it can be obtained that,

$$\rho = \sum_{i=1}^{m} \left[\ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-}(1+X_{ij})! - \ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+}(1-X_{ij})! \right], \tag{6.49}$$

where $X_{ij} = \{1, -1\}$. Therefore, $1 \leq (1+X_{ij})!, (1-X_{ij})! \leq 2$. For a particular $\mathcal{D}_i$, if $X_{ij} = 1$, then Equation (7.8) can be rewritten as:

$$\rho_i = \ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-}(1+X_{ij}) - \ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+}. \tag{6.50}$$

Therefore, the weight of X_{ij} is $\ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-}$. Similarly, for $X_{ij} = -1$, weight of X_{ij} is $\ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+}$. Therefore,

$$\begin{aligned} M(1:-1) &= \frac{\ln \frac{\Upsilon_i^+}{1-\Upsilon_i^-} + \ln \frac{\Upsilon_i^-}{1-\Upsilon_i^+}}{2} \\ &= \frac{1}{2} \ln \frac{\Upsilon_i^+ \Upsilon_i^-}{(1-\Upsilon_i^+)(1-\Upsilon_i^-)} \end{aligned}$$

From Equation (6.20), it directly follows,

$$\Upsilon_i^+ \Upsilon_i^- > (1-\Upsilon_i^+)(1-\Upsilon_i^-) \text{ i.e., } M(1:-1) > 0 \tag{6.51}$$

□

Proposition 6.5.2. *The proposed optimal decision rule satisfies the Potential Pareto criterion.*

Proof. As per the *Potential Pareto criterion* (*PPC*) [135, 136], the winner compensates the losers and still remains better off. It is assumed that $\mathcal{D}_w$ is selected for App_i through decision rule $\mathcal{O}$. Therefore, $\mathcal{O}(\mathcal{J}_{\mathcal{D}_w}) = 1$ and $\mathcal{E}_{max}(App_i) = \mathcal{E}_{\mathcal{D}_w}(App_i)$:

$$\mathcal{E}_{max}(App_i) = \alpha\Big[\mathcal{P}(1:1)P_{\mathcal{D}_w}(\mathcal{O}:1) + \mathcal{P}(-1:1)(1 - P_{\mathcal{D}_w}(\mathcal{O}:1))\Big] + (1-\alpha)\Big[\mathcal{P}(-1:-1)P_{\mathcal{D}_w}(\mathcal{O}:-1) + \mathcal{P}(1:-1)(1 - P_{\mathcal{D}_w}(\mathcal{O}:-1))\Big]$$

which gives us:

$$\mathcal{E}_{\mathcal{D}_w}(App_i) > \mathcal{E}_{\mathcal{D}_j}(App_i), \forall \mathcal{D}_j \in \mathcal{D}^{nom}, \mathcal{D}_j \neq \mathcal{D}_w \tag{6.52}$$

Therefore, the compensation of any $\mathcal{D}_j$, is expressed as $\mathcal{E}_{\mathcal{D}_j}(App_i)$ and the net benefit of the winner DC is $\mathcal{E}_{\mathcal{D}_w}(App_i) - \mathcal{E}_{\mathcal{D}_j}(App_i)$. However, the winner DC must have a positive benefit. Using Equation (6.52), the following holds true:

$$\alpha P_{\mathcal{D}_w}(\mathcal{O}:1)\mathcal{P}(1) + (1-\alpha)P_{\mathcal{D}_w}(\mathcal{O}:-1)\mathcal{P}(-1) > \alpha P_{\mathcal{D}_j}(\mathcal{O}:1)\mathcal{P}(1) + (1-\alpha)P_{\mathcal{D}_j}(\mathcal{O}:-1)\mathcal{P}(-1)$$

$$\Rightarrow \alpha P_{\mathcal{D}_w}(\mathcal{O}:1)\mathcal{P}(1) + (1-\alpha)P_{\mathcal{D}_w}(\mathcal{O}:-1)\mathcal{P}(-1) + \alpha\mathcal{P}(-1:1) + (1-\alpha)\mathcal{P}(1:-1) > \alpha P_{\mathcal{D}_j}(\mathcal{O}:1)\mathcal{P}(1) + (1-\alpha)P_{\mathcal{D}_j}(\mathcal{O}:-1)\mathcal{P}(-1) + \alpha\mathcal{P}(-1:1) + (1-\alpha)\mathcal{P}(1:-1)$$

$$\Rightarrow \mathcal{E}_{\mathcal{D}_w}(App_i) = \mathcal{E}_{\mathcal{D}_j}(App_i) + c'.$$

Therefore, $\mathcal{D}_w$ compensates other DCs and also incurs a positive benefit c'; thereby, satisfying the *PPC*. □

Proposition 6.5.3. *The proposed decision rule guarantees the unanimity criterion.*

Proof. The unanimity criterion [137] is one of the most rational properties of a decision rule. For two nominated DCs $\mathcal{D}_{j_1}, \mathcal{D}_{j_2} \in \mathcal{D}^{nom}$. For a particular DC, $\mathcal{D}_i \in \mathcal{D}_{App_i}$, $\mathcal{D}_{j_1}$ is preferred to $\mathcal{D}_{j_2}$, if either:

(a) $X_{i1} = 1$, and $X_{i2} = -1$ (when $\mathcal{D}_i$ rejects $\mathcal{D}_{j_2}$ clearly),

(b) $X_{i1} = X_{i2} = 1$ (when $\mathcal{D}_{j_2}$ is not rejected , however, $\mathcal{D}_{j_1}$ is preferred).

Case a: If for every $\mathcal{D}_i \in \mathcal{D}_{App_i}$, $\mathcal{D}_{j_1} \succ_{\mathcal{D}_i} \mathcal{D}_{j_2}$ hold true, then it readily follows:

$$\mathcal{Q}_{net}^{\mathcal{D}_{j_1}}(App_i) > \mathcal{Q}_{net}^{\mathcal{D}_{j_2}}(App_i), \forall \mathcal{D}_i \in \mathcal{D}_{App_i} \tag{6.53}$$

$$\Rightarrow \frac{pP}{\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_{j_1}) + \delta(\mathcal{D}_{j_1}, l_{1,i}, l_{2,i}) + \sum_{i=1}^{m} \left(\lambda_k \times \frac{\mathcal{S}(App_i)}{\sum_{i=1}^{m} \lambda_k} \right)}$$
$$> \frac{pP}{\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_{j_2}) + \delta(\mathcal{D}_{j_1 2}, l_{1,i}, l_{2,i}) + \sum_{i=1}^{m} \left(\lambda_k \times \frac{\mathcal{S}(App_i)}{\sum_{i=1}^{m} \lambda_k} \right)},$$

Consequently, $\mathcal{E}_{\mathcal{D}_{j_1}}(App_i) > \mathcal{E}_{\mathcal{D}_{j_2}}(App_i)$; thereby, inferring $\mathcal{O}(\mathcal{J}_{\mathcal{D}_{j_1}}) = 1$.

Case b: In this case, for every $\mathcal{D}_i \in \mathcal{D}_{App_i}$, it is true that $\mathcal{D}_{j_1} \succeq_{\mathcal{D}_i} \mathcal{D}_{j_2}$. This implies that:

$$\frac{pP}{\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_{j_1}) + \delta(\mathcal{D}_{j_1}, l_{1,i}, l_{2,i}) + \sum_{i=1}^{m} \left(\lambda_k \times \frac{\mathcal{S}(App_i)}{\sum_{i=1}^{m} \lambda_k} \right)}$$

$$\geq \frac{pP}{\sum_{i=1}^{m} \mathcal{M}(\mathcal{D}_i, \mathcal{D}_{j_2}) + \delta(\mathcal{D}_{j_1 2}, l_{1,i}, l_{2,i}) + \sum_{i=1}^{m} \left(\lambda_k \times \frac{\mathcal{S}(App_i)}{\sum_{i=1}^{m} \lambda_k} \right)}$$

$$\Rightarrow \mathcal{P}_{\mathcal{D}_{j_1}}(1) \geq \mathcal{P}_{\mathcal{D}_{j_2}}(1), \text{ and } \mathcal{P}_{\mathcal{D}_{j_1}}(-1) \leq \mathcal{P}_{\mathcal{D}_{j_2}}(-1) \tag{6.54}$$

$$\Rightarrow \mathcal{E}_{\mathcal{D}_{j_1}}(App_i) > \mathcal{E}_{\mathcal{D}_{j_2}}(App_i) \Rightarrow \mathcal{O}(\mathcal{J}_{\mathcal{D}_{j_1}}) = 1 \tag{6.55}$$

□

6.6 Performance Evaluation

This Section presents and analyzes the performance of the proposed system of scheduling a DC for serving a particular application App_i. The details of the testbed information are provided in Table 6.2.

Table 6.2: Testbed information

Parameters	**Values**
Processor	Intel(R) Core(TM) i5-2400 CPU @ 3.10 GHz
RAM	4GB, DDR3
Disk space	320 GB
Operating system	Ubuntu 14.04 LTS
Application software	MATLAB R2013a

6.6.1 Explanation of Parameters

The parameters used for this work have been selected on the basis of prior related works [104–106]. The values of these parameters are set as per the related works in this domain that are most cited [107–111].

6.6.2 Single Application Scenario

The experiments were initially performed for a single application that demands for 10 distinct VSs. The underlying sensor network was simulated over a uniform, random deployment of 100 physical sensor nodes over a region of 500m ×500m for 500 simulation seconds. The experimental setup is illustrated in Table 6.3. The experiments were performed to select 10 temporary DCs forming $\mathcal{D}_{App_i}$. Based on the proximity of the DCs with the application center $l_1 = 28$ and $l_2 = 196$, $\mathcal{D}^{nom}$ was built. The parameters of the component DCs of $\mathcal{D}^{nom}$ are illustrated in Table 6.4. For the purpose of selection and analysis of the optimal decision rule, the set of decision rules considered is $F = \{f_i(\cdot)\}$, such that, $\forall f_i() \in F$:

$$f_i(\mathcal{J}_{\mathcal{D}_i}) = 1, f_i(\mathcal{J}_{\mathcal{D}_j}) = -1, \forall \mathcal{D}_i, \mathcal{D}_j \in \mathcal{D}^{nom}, \mathcal{D}_i \neq \mathcal{D}_j \tag{6.56}$$

The performance of the rules in F are studied and analyzed, as shown in Figure 6.4.

As per Equation (6.56), every decision rule in F schedules a unique DC from the set $\mathcal{D}^{nom}$. Following ever $f_i \in F$, the cumulative migration cost of all the DCs of $\mathcal{D}_{App_i}$ to the DCs of $\mathcal{D}^{nom}$ are analyzed over time in Figure 6.4(a). It is observed that $\mathcal{D}_5$ has the lowest migration cost followed by $\mathcal{D}_3, \mathcal{D}_4, \mathcal{D}_1$, and $\mathcal{D}_2$. Figure 6.4(b) depicts the cumulative delivery cost of the DCs, in which $\mathcal{D}_5$ performs poorly. On the other hand, $\mathcal{D}_2$ bears a low cost for delivering the packets to the application center. Figure 6.4(c) highlights the closeness of $\mathcal{D}_3, \mathcal{D}_2$, and $\mathcal{D}_1$ in terms of the overall service delay. However, in Figure 6.4(d), the basic QoS is evaluated in which the outcome of the

Table 6.3: Experimental setup for single application

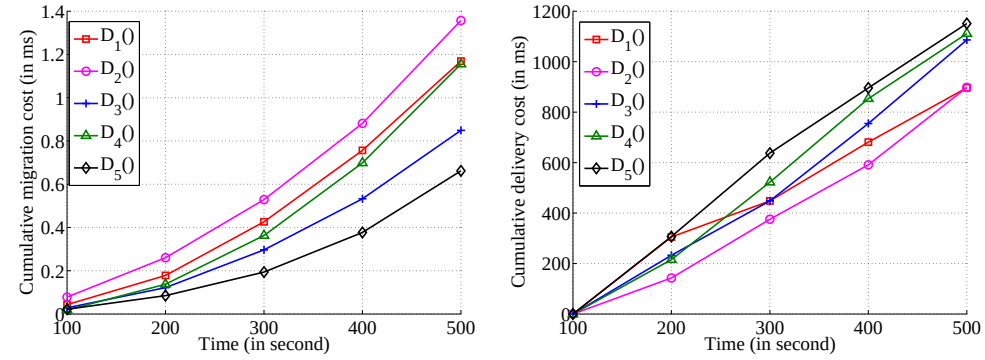
Parameters	**Values**
Sensor deployment area	500 m × 500 m
Deployment	Uniform, random
Number of nodes	100
Number of VSs	10
Number of temporary DCs (| $\mathcal{D}_{App_i}$ |)	10
Number of nominated DCs (| $\mathcal{D}^{nom}$ |)	5
Size of each packet	2 byte
Transmission speed	100 m/s
Distribution of demand rate	Poisson

Table 6.4: Parameters of the set of nominated DCs ($\mathcal{D}^{nom}$)

	$\mathcal{D}_1^{nom}$	$\mathcal{D}_2^{nom}$	$\mathcal{D}_3^{nom}$	$\mathcal{D}_4^{nom}$	$\mathcal{D}_5^{nom}$
Abscissa	38	224	48	273	453
Ordinate	296	464	188	56	317
Migration latency (Bps)	330.93	456.58	236.24	86.54	398.73
Delivery latency (m/s)	248.19	305.37	142.86	142.86	215.61

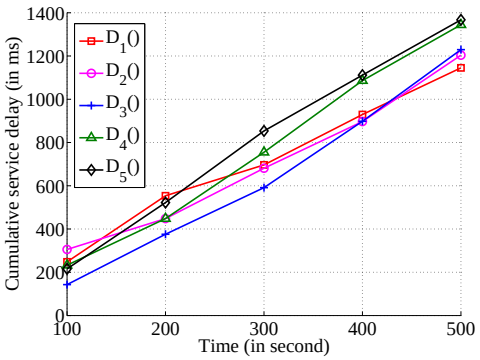
decision rule $f_3()$, i.e., $\mathcal{D}_3$ obtains the maximum QoS. The user-dissatisfaction delay product is evaluated and analyzed in Figure 6.4(e). It is observed that $\mathcal{D}_3$ provides the minimum dissatisfaction to App_i. The effective QoS is shown in Figure 6.4(f), in which $f_3()$ emerges as the optimum decision rule.

Now, the correctness of the decision rule $f_3()$ is analyzed in terms of the heterogeneous, and fallible decision making ability of $\mathcal{D}_{App_i}$ and the "goodness" or "badness" of the components of $\mathcal{D}^{nom}$. As shown in Figure 6.5, the decision making abilities of the DCs are studied by learning the behavior of each DC for last k instants of time. Here, it is assumed that $k = 100$. In Figure 6.5(a), it is found that the mean probability of a correct decision is 0.6615; whereas, that of a wrong decision is 0.3385. Moreover, the mean probability of rejecting a "good" DC is 0.684; whereas, that of a "bad" DC is 0.639. The probabilities of the Type I and Type II error are depicted in Figure 6.5(b),

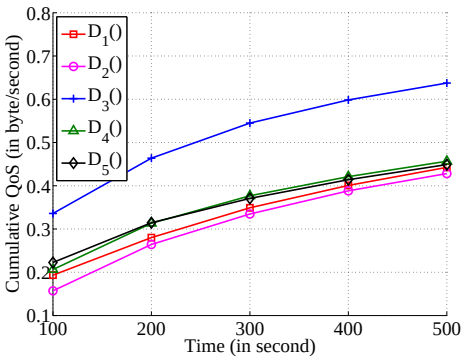


(a) Analysis of migration cost

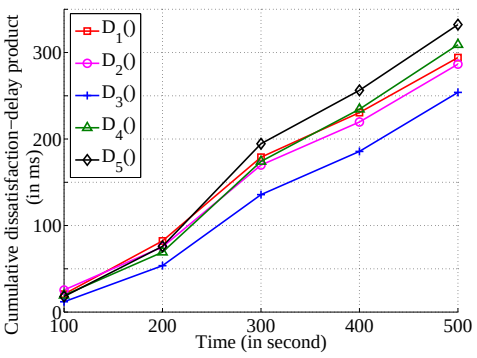
(b) Analysis of delivery cost



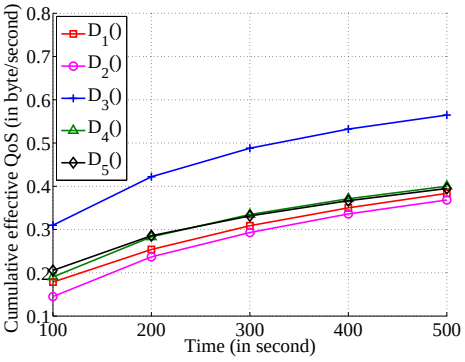
(c) Analysis of service delay



(d) Analysis of QoS



(e) Analysis of user dissatisfaction

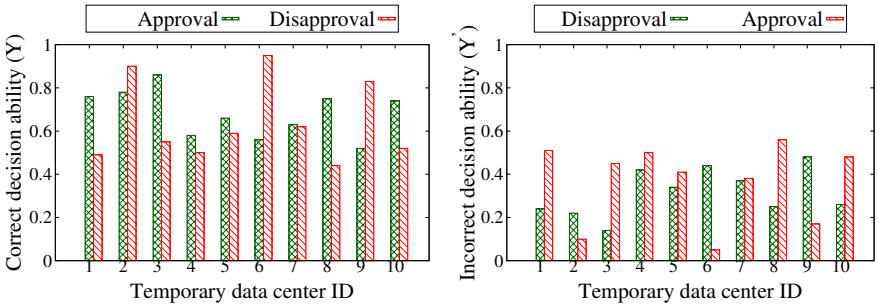


(f) Analysis of effective QoS

Figure 6.4: Performance evaluation of the set of decision rules (F)

the mean probability being 0.316, and 0.361, respectively. With this fallible decision making ability of $\mathcal{D}_{App_i}$, $f_3()$ eventually schedules the optimal DC.

Figure 6.6 illustrates the capacity, "goodness" and "badness" of every $\mathcal{D}_j \in \mathcal{D}^{nom}$.



(a) Analysis of $\Upsilon_i^+, \Upsilon_i^-$ (b) Analysis of $\Upsilon_i'^-, \Upsilon_i'^+$

Figure 6.5: Analysis of the decision making abilities of the set of the temporary DCs ($\mathcal{D}_{App_i}$)

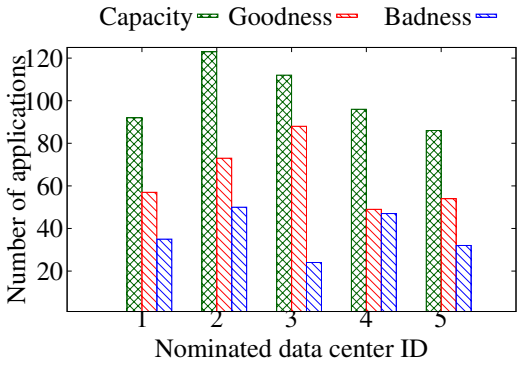


Figure 6.6: Analysis of the "goodness" and "badness" of the nominated data centers

For the sake of simplicity, Equation (6.22) and Equation (6.23) are simplified to obtain the capacity $\mathcal{C}$, "goodness" and "badness" of a DC, $\mathcal{C}^{max}(\mathcal{D}_j)$, and $\mathcal{C}(\mathcal{D}_j)$ are the total number of applications that can be supported by, and that are running withing $\mathcal{D}_j$, respectively. Therefore:

$$\mathcal{G}(\mathcal{D}_j) = \mathcal{C}^{max}(\mathcal{D}_j) - \mathcal{C}(\mathcal{D}_j), \bar{\mathcal{G}}(\mathcal{D}_j) = \mathcal{C}^{max}(\mathcal{D}_j) - \mathcal{G}(\mathcal{D}_j) \tag{6.57}$$

Now, the outcome of $f_3()$ is clearer, as $\mathcal{D}_3$ exhibits the maximum "goodness", and minimum "badness", although $\mathcal{D}_2$ has a higher capacity to support applications.

6.6.3 Multiple Application Scenario

Having studied the performance evaluation of the proposed system for a single application, now the system performance is examined and discussed for a multiple applications. The experimental setup is depicted in Table 6.5. The performance metrics that have been considered are:

- Mean turnaround time;
- Mean throughput;
- Channel utilization; and
- Mean data center utilization.

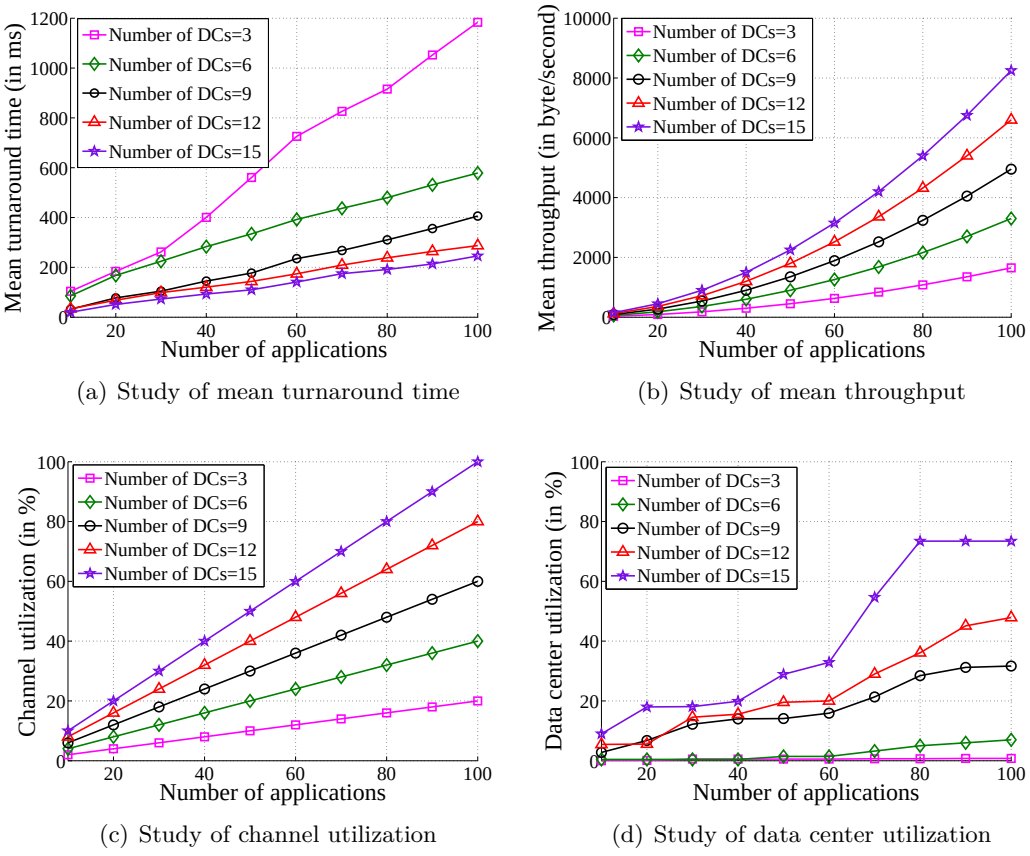
Based on the above metrics, the system performance is evaluated by varying the number of applications and the number of DCs, as shown in Figure 6.7.

Table 6.5: Experimental setup for multiple applications

Parameters	**Values**
Sensor deployment area	500 m × 500 m
Number of applications	$\{10, 20, ..., 100\}$
Number of temporary data centers ($\mid \mathcal{D}_{App_i} \mid$)	50
Number of nominated data centers ($\mid \mathcal{D}^{nom} \mid$)	$\{3, 6, 9, 12, 15\}$
Intra data center bandwidth	Very high, $>$ 8 Mbps
Channel bandwidth	Moderate, $\leq$ 8 Mbps
Size of each packet	2 byte
Transmission latency	100 m/s
Distribution of demand rate	Poisson
Maximum number of demand requests	$1 - 10$

Mean turnaround time

For the purpose of examination of the system performance in terms of the mean turnaround time τ_{mean} for every application, the metric is defined as the mean of the turnaround



(a) Study of mean turnaround time

(b) Study of mean throughput

(c) Study of channel utilization

(d) Study of data center utilization

Figure 6.7: Performance evaluation for multiple applications

time of all the applications running in the system. The turnaround time, τ, of a particular application is computed as the service delay incurred to serve a particular demand of the application. τ is expressed as:

$$\tau(App_i) = \sum_{j=1}^{g(i)} \lambda_j \times \frac{\mathcal{S}_{tot}}{\sum_{p=1}^{n} \sum_{k=1}^{g(p)} \lambda_k} \tag{6.58}$$

$$\tau_{mean}(App_i) = \sum_{q=1}^{|\mathcal{D}^{nom}|} \tau_q \tag{6.59}$$

where $\mathcal{S}_{tot}$ is the total service delay incurred in processing n applications (per DC), each requesting for $g(p)$ distinct services, and λ being the demand rate. As indicated in Figure

6.7(a), it can be seen that when the number of DCs are low, the value of τ_{mean} increases with the increase in the number of applications. With the increase in the number of nominated DCs, the mean turnaround time falls reasonably, as is the case of $12 - 15$ DCs.

Mean throughput

The mean throughput, $\mathcal{T}_{mean}$, of the system is defined as the average outflow of information in bits per unit time from every nominated DC. If the p_i packets are generated from a $\mathcal{D}_i \in \mathcal{D}^{nom}$ per Δt unit of time, then the $\mathcal{T}_{mean}$ is expressed as:

$$\mathcal{T}_{mean} = \left(\sum_{q=1}^{|\mathcal{D}^{nom}|} \frac{P \times p_q}{\Delta t} \right) / \mid \mathcal{D}^{nom} \mid \tag{6.60}$$

where P being the size of each packet. Now Figure 6.7(b) clearly shows the variation of the mean throughput with the variation of the number of applications being served and the total number of nominated DCs. It is observed that the mean throughput increases significantly with both the increase in the count of the applications and the DCs.

Channel utilization

For the purpose of evaluating the channel utilization χ_B, the metric is defined as the ratio of the amount of channel utilized in unit time to the total bandwidth available χ_B^{max}, expressed as percentage. Mathematically, χ is expressed as:

$$\chi_B = \frac{\sum_{q=1}^{|\mathcal{D}^{nom}|} P \times p_q}{\chi_B^{max}} \times 100\% \tag{6.61}$$

Figure 6.7(c) indicates that percentage of channel utilization is initially low for lower number of nominated DCs serving the applications. As the count of DCs increase with the utilization percentage increases reasonably with the increase in the number of currently running applications.

Mean data center utilization

The mean data center utilization is defined as the ratio of the number of DCs required to handle the current set of running applications to the total number of DCs that are available in the system, expressed as percentage. If every $\mathcal{D}_j \in \mathcal{D}^{nom}$ has a capacity to handle n_i number of applications, and $n = \{10, 20, ..., 100\}$ is assumed to be the total number of applications then χ_D is expressed as:

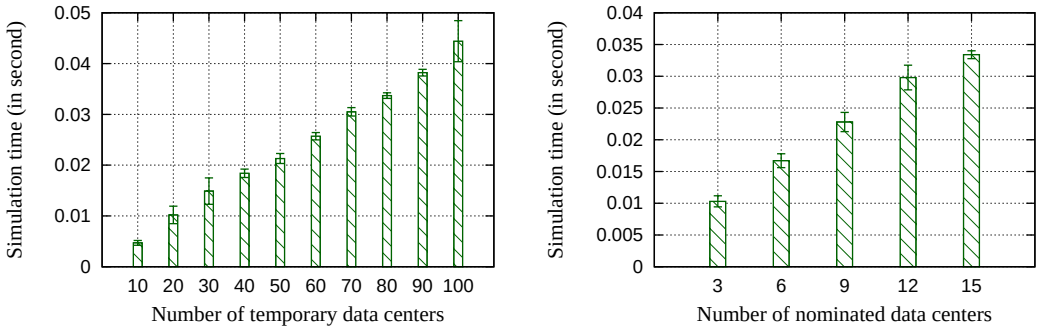
$$\chi_D = \frac{\left[\left(\sum_{p=1}^{n}\sum_{k=1}^{g(p)}\lambda_k\right) \bmod (\mid \mathcal{D}^{nom}\right] \mid) + 1}{\mid \mathcal{D}^{nom} \mid} \times 100\% \tag{6.62}$$

From Figure 6.7(d), it is observed that the utilization percentage is negligible with $3-6$ nominated DCs, and is almost independent to the number of applications being served. As the count of DCs increase from 9 onwards, the utilization percentage rises significantly with the increase in the application count.

6.6.4 Complexity Analysis

In this subsection, the asymptotic computational complexity of the proposed work is discussed and analyzed to justify its applicability for real-time processing. The metric for computational complexity has been measured in terms of the simulation time by varying the parameters of simulation, as shown in Figure 6.8. Initially, the setup for performing the experiment assumes a fixed number of applications ($n = 100$). The variation of the simulation time are recorded by varying the number of DCs in different iterations.

As shown in Figure 6.8(a), the number of temporary DCs are varied from 10 to 100 with a step of 10, and the number of nominated DCs are kept fixed, $\mathcal{D}^{nom} = 9$. For a particular number of temporary DC, values are recorded for 50 iterations and are plotted with 95% confidence. It was observed that the simulation time averages at 0.024 within interval $[0.022, 0.026]$ with 95% confidence.



(a) Plot of number of temporary DCs versus simulation time

(b) Plot of number of nominated DCs versus simulation time

Figure 6.8: Complexity analysis by varying system components

Figure 6.8(b) indicates the variation in the computational complexity as the number of nominated DCs vary. For this experiment, the number of temporary DCs is kept constant, $\mathcal{D}_{App_i} = 50$, and the count of the nominated DCs are varied from 3 to 15, with a step size of 3. It is observed that the simulation time centers at 0.0226 within the interval $[0.0214, 0.0238]$, with 95% confidence.

The mean variance of complexity with the increase in DCs (for Figures 6.8(a), and 6.8(b)) is found to be 8.9×10^{-6} second, and 5×10^{-6} second, respectively. Therefore, the increase of the computational complexity of the proposed work with the increased number of DCs is negligible.

6.7 Summary

The proposed work focused on the networking of a sensor-cloud infrastructure. In a sensor-cloud scenario, data from various physical sensors are grouped to form a VS. An application requests for multiple such VSs spanning across multiple geographical regions. Therefore, data from the VSs are collected by several geo-spatial DCs. However, sensor-cloud involves collection of these VSs to a single VM within a single DC for further analysis and collective processing. This necessitates scheduling of a single DC to serve

each application. For the purpose of scheduling, the work propounds a QoS based optimal decision rule.

Chapter 7

Development of a Working Prototype of Sensor-cloud Infrastructure

Having discussed about the different technical aspects of sensor-cloud in the previous Chapters, this Chapter presents the implementation details of a working prototype of sensor-cloud infrastructure. In the endeavor of prototyping sensor-cloud, some of its limitations also became evident.

7.1 Limitations of Sensor-cloud

Sensor-cloud infrastructure is envisioned to support multiple end-user organizations with real-time sensor services. However, existing WSNs typically generate data with enormous *volume*, *velocity*, and *variety*, i.e., the generated data is *big* in size and are, hence, to be processed differently. In sensor-cloud platforms, the data are handled using traditional data processing techniques, which are incapable of managing heterogeneous and voluminous data in real-time. Thus, with the increasing growth in the velocity, variety,

and variability of data, management becomes a serious concern and difficulty. Additionally, when multiple user-organizations are required to be simultaneously provisioned with sensor services, the existing sensor-cloud infrastructure is likely to be overwhelmed with a huge number of data requests, thereby, potentially creating a "bottleneck" within the sensor-cloud platform.

Evidently, the existing sensor-cloud systems are unable to capture, analyze, and control the present data efficiently, in real-time. Another problem that is typically encountered with the traditional systems is that, as the generated physical data is highly unstructured, the systems fail to correlate, connect, and process the huge volumes of data in real-time. This becomes a real difficulty for users or organizations with a large number of queries to be processed over big-data in real-time.

7.2 Contributions of this Chapter

The goal of this Chapter is to address the aforesaid limitations of sensor-cloud and introduce an enhanced version in terms of performance of data (or query) processing, storage, and management. Thus, this Chapter proposes a modified form of sensor-cloud infrastructure – the *Big-Sensor-Cloud Infrastructure* (*BSCI*). BSCI is a platform for big-data storage, processing, leveraging, and efficient remote management. Similar to sensor-cloud, BSCI also thrives on virtualization of sensor devices. The major difference between the two infrastructures lies in the data capture, analyzing, and control mechanisms. Unlike sensor-cloud, BSCI manages "big" unstructured sensor data from varied data sources and arranges, correlates, and connects the data using "sophisticated" big-data management techniques. This enables the user organizations to execute the computationally intensive queries over large data sets in less time.

The proposed BSCI has its own novelty from a *business* perspective as well. Sensor-cloud is already being viewed as a potential substitute of conventional WSNs. With the implementation of BSCI – the proposed technology – the end-users of this technology

will rapidly evolve because of its efficiency and usefulness. In this context, the work bears its own relevance as it hugely improves the restricted access of sensor networks and their resource-constrained nature. The proposed prototype contributes immensely in effective data management, storage, real-time processing, and retrieval of big sensor data. The tangibility of BSCI can be measured in terms of its ability to render Se-aaS even in situations involving a tsunami of data. This positively affects the financial aspects of the end-user organizations. The CSP also benefits from the model with the widespread dissemination and effective management of the sensor devices.

7.3 Design of Big-Sensor-Cloud Infrastructure

This Section presents and describes the design details of the proposed BSCI. Initially, the use-case diagram of the system is presented in Figure 7.1 in which it is observed that BSCI comprises of five distinct types of actors:

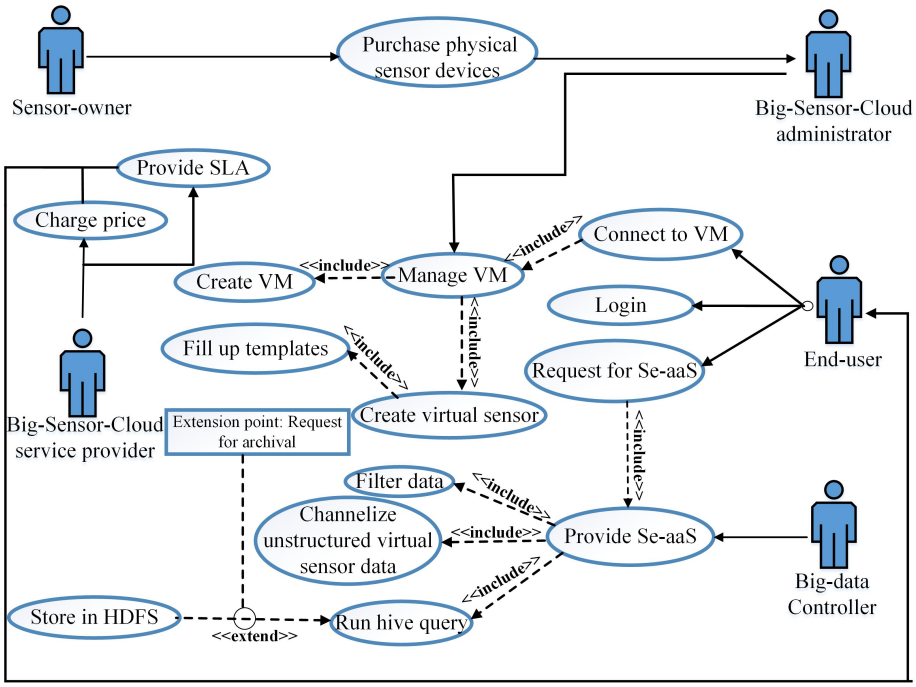


Figure 7.1: Use case diagram for Big-Sensor-Cloud Infrastructure

(i) *End-user*: The end-users (person/organization) possess their own applications, which are to be fed with big sensor-data from the physical sensor networks. As the type and amount of the demand changes with time, the end-users enjoy scalability of Se-aaS, provided by the Big-Sensor-Cloud Service Provider (BSCSP), Thus, the end-users are privileged to demand different sensor services at different time instants from heterogeneous sensor devices, and the services are offered instantaneously by the BSCSP. In return, the end-users are liable to pay as per their usage of Se-aaS to the BSCSP.

(ii) *Sensor-owner*: The sensor-owners bear a role from a business perspective. They purchase physical sensor devices and lend them to the BSCSP. The sensor-owners earn a monthly monetary profit as per the usage of their respective sensor devices.

(iii) *Big-Sensor-Cloud administrator*: The Big-Sensor-Cloud administrator primarily manages and controls the entire cloud processing activities involving virtualization of the physical sensor devices into distinct Virtual Sensors (VSs), maintenance and monitoring of the physical sensor devices, organization of the unstructured data, executing computationally intensive queries over the big-data sets, and real-time service provisioning of Se-aaS. However, the administrator plays a significant role in virtualization of the big sensor data, and quantifies the data usage by the individual end-users. Big sensor data segregation and filtration are also handled by the administrator.

(iv) *Big-data Controller*: The Big-data Controller operates within a Virtual Machine (VM). The controller is primarily responsible for the tabular structurization and modular organization of big sensor data with each VM in a distributed manner. The controller is also responsible for the structured storage and provisioning of huge data volumes in real-time.

(v) *Big-Sensor-Cloud Service Provider (BSCSP)*: The BSCSP is a business actor of

BSCI. The BSCSP maintains a log of the quantified usage of the end-users, and charges price from the end-users, as per their usage of Se-aaS. The BSCSP maintains a pricing policy and offers a Service Level Agreement (SLA) to the end-users at the time of login.

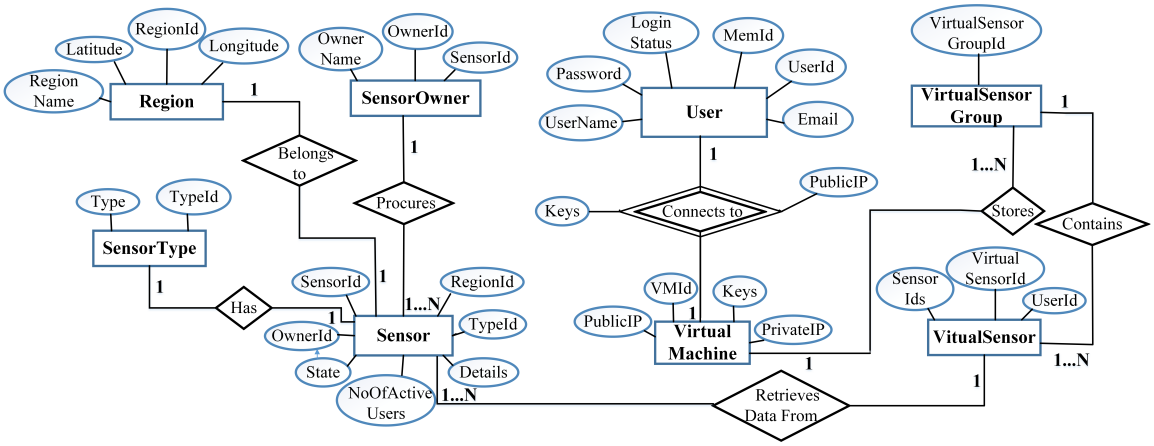


Figure 7.2: Entity Relationship Diagram for Big-Sensor-Cloud Infrastructure

The entities of the proposed BSCI are identified and the relationship among the entities are depicted in Figure 7.2. In Chapter 3, the entities – region, sensor and sensor type and their inter-relationship were modeled. In this Chapter, the entity modeling of the user, VM, and Virtual Sensor Group (VSG) is introduced, as depicted in the figure. A temporary VM is allocated on behalf of every active end-user and is employed to serve the VSs (or groups) in terms of processing and storage. A VM remains active unless an end-user voluntarily chooses to kill it.

7.4 Architecture of Big-Sensor-Cloud Infrastructure

This Section presents the architectural details of BSCI. Primarily, it is a four-layered architecture, as shown in Figure 7.3. The several end-user organizations request for the sensed data to be fetched into their application from the various application-dependent physical sensor nodes. The user-organization gets connected to the Big-Sensor-Cloud

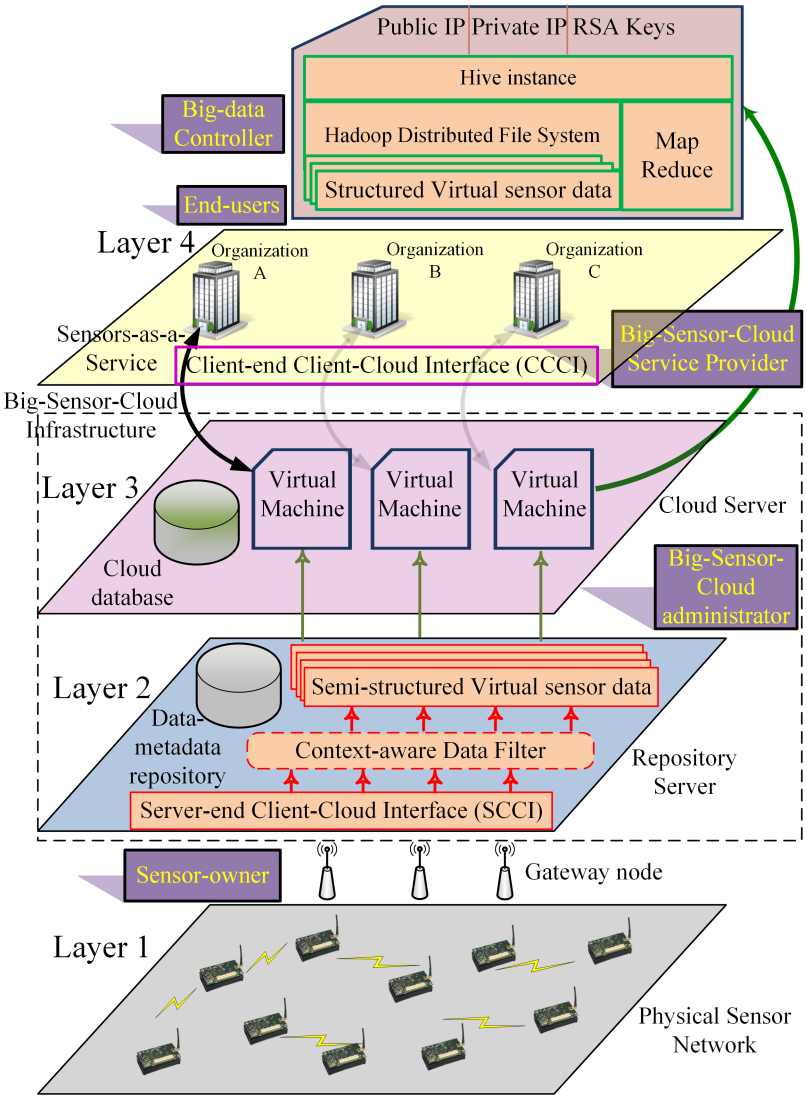


Figure 7.3: Architecture of Big-Sensor-Cloud Infrastructure

service provider (BSCSP) over the client-cloud interface at the client-end (CCCI). Initially, the BSCSP provides a specific template to be filled by the end-user organization comprising of all the information relevant to the application-dependent data. These templates enable the BSCSP in mapping with the information stored in the Data-metadata repository, which in turn, helps in activating the specific physical sensor nodes spread across the physical sensor network. The raw sensed data from the various activated physical sensor nodes are transmitted to their respective nearest base stations. The raw

sensed data constantly moves in large volume to the repository server of the BSCI, where they are segregated through a *Context-Aware Data Filter*, based on the application dependent data. The outcome of the Data Filter is the unstructured VS data accumulating a group of specific application dependent data to be fetched into their respective requesting VM. The unstructured data cannot be efficiently handled with the help of existing traditional technologies, because the data arrive in large volumes with huge variety and speed, thus resembling the three Vs characterizing big-data, i.e., *volume*, *variety*, and *velocity* [138, 139].

Big-data within a VM can be handled using the Hadoop [140] open source software, which consists of two important layers – (a) the execution engine, known as *Map Reduce*, and (b) the file system known as *Hadoop Distributed File System* (*HDFS*). The unstructured sensor data is maintained structurally within the HDFS using the programming paradigm, Map Reduce. Map Reduce performs two basic tasks – *Map Task* and *Reduce Task*. The Map Task takes the unstructured VS data as the input, thereby producing a sequence of key-value pairs, which is sorted and shuffled by the intermediate sorting algorithm implemented between the Map and Reduce tasks. Finally, the sorted data are fed into the *Reduce Task*, which combines all the values related to a specific key and are stored within the HDFS.

7.5 Implementation of Big-Sensor-Cloud Infrastructure

This Section presents the layer-wise detailed structure of BSCI. The functional components of every layer are described as follows and the implementation details associated with the components are described in Table 7.1.

Layer 1: Physical sensor network layer

The bottommost layer corresponds to physical wireless sensor devices that communicate with one another using the standard multi-hop routing protocols. The physical

Table 7.1: Implementation details of Big-Sensor-Cloud Infrastructure

Components	Activity	Require material	Time (hrs)	Major Skills
Physical sensor nodes	Deploy at different regions Store definition and metadata in Data-metadata Repository Configure routing	1. Microcontroller development board with Zigbee (IEEE 802.15.4) 2. Sensors	50	Embedded programming in C
Gateway node	Configure routing Configure duty cycling Data channelization through Universal Serial Bus	1. Microcontroller development board with Zigbee (IEEE 802.15.4) 2. USB port	50	Serial communication Protocol, Embedded programming in C
Client-Cloud interface	Build a web interface to manage the high level requirements of the user in the form of templates	Computer server	10	Web technologies, Graphics handling in Java
Repository Server	Data channelization into unstructured Virtual Sensors Data filtering Virtualization Data direction to Cloud server	Computer server	50	Remote Method Invocation, Java programming
VS	Creation and management Activation and deactivation	Computer server	3	Database skills
Cloud Server	Create VMs dynamically Execute structurization of data Redirect queries to VMs	Computer server, Eucalyptus	100	Shell scripting, Eucalyptus management
VM	Run customized queries Analysis of data based on query	Computer server	100	Hive query handling, Hadoop Distributed File System

devices transmit the raw sensed data to the Big-Sensor-Cloud infrastructure through the Gateway node.

Layer 2: Repository Server of Big-Sensor-Cloud

2.1 Repository Server: The raw sensed heterogeneous data (with high volume, velocity, and variability) are dumped into the repository server, within which the data are further processed to generate semi-structured VS data. The individual components of the Repository Server are discussed.

2.1.1 **Client-Cloud Interface** (**CCI**): CCI is one of the components of Layer 2, which

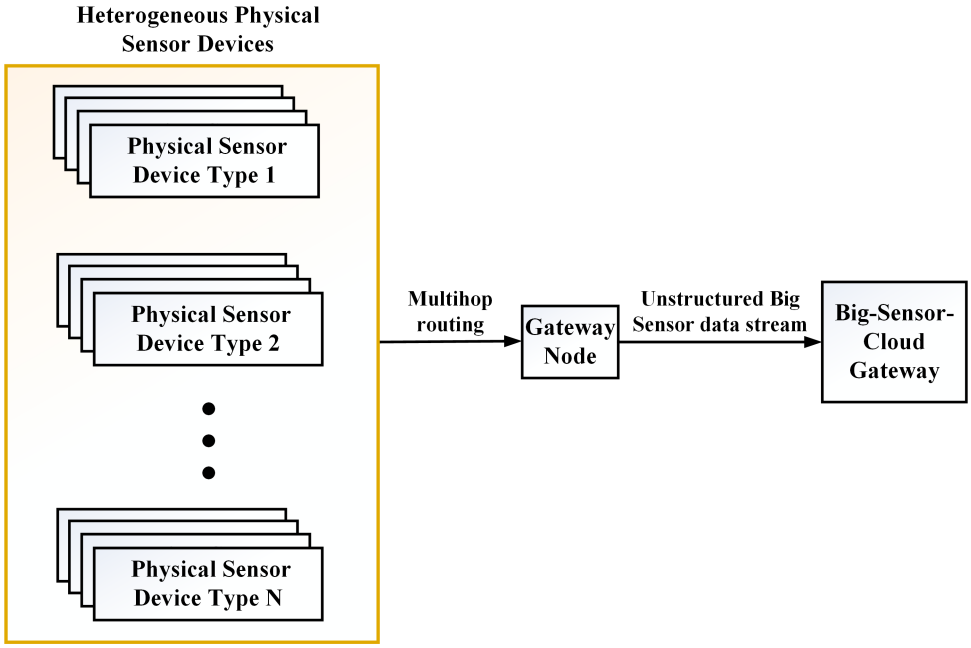


Figure 7.4: Block diagram of layer 1

is further divided into *Server-side Client-Cloud Interface* (*SCCI*), and *Client-side Client-Cloud Interface* (*CCCI*). CCCI resides in Layer 4. This interface connects the end-users to the Big-Sensor-Cloud end through the user login functionality. CCI interacts with the end-users, and collects the high-level demand requirements. The requirements are interpreted in terms of resource allocation within the cloud-end by SCCI.

2.1.2 **Context-aware Data Filter**: The incoming big-data stream of raw sensed data is subjected to specialized filters that segregate the data as per the application demand. The filters are equipped with the ability to handle voluminous data with enormous heterogeneous data volumes (in zettabyte) generated with tremendous velocity.

2.1.3 **Semi-structured VSs**: The filtered data are grouped and aggregated into VSs. Several VSs form a VSG. The data within the VSs are semi-structured in nature, and are channelized to the respective VMs for further processing.

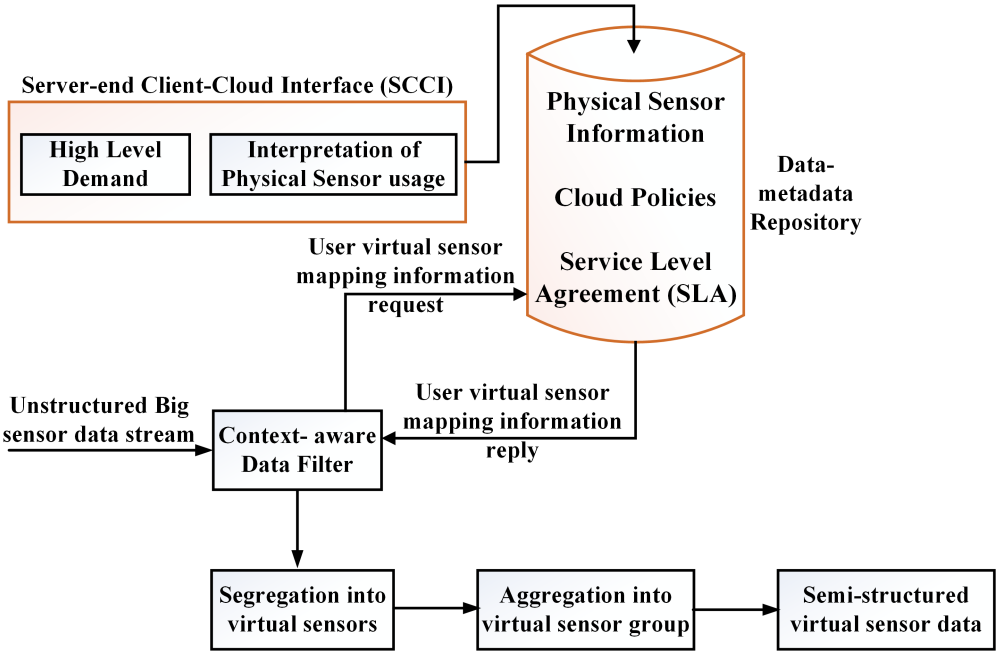


Figure 7.5: Block diagram of layer 2

2.1.4 **Data-metadata Repository**: The information about the physical sensor devices and the configurations are stored in the data-metadata repository. The policies, SLAs, and the mapping of VSs with the application demand are also maintained here.

Layer 3: Cloud Server of Big-Sensor-Cloud

3.1 Cloud Server: The cloud server obtains the semi-structured VS data and routes those to the respective VMs of the respective end-users.

3.1.1 **Virtual Machines (VMs)**: The VMs are created dynamically based on the user-demand. The end-users connect to the respective VMs using the Public IP, and the encrypted RSA keys that are provided to them prior the connection setup phase. Once the VMs are created, the end-users obtain data from the VMs and archive within them, as per requirement.

3.1.1.1 **Hive Instances**: Within every VM, a Hive instance is executed for efficient

processing and management of the data starting from loading of the data, to the execution of Data Definition Language (DDL), Data Manipulation Language (DML), and Data Control language (DCL) scripts.

3.1.1.2 **Structured VS Data**: The output of Hive is obtained in the form of structured VS data, which are stored with HDFS. The future queries on the big virtual-sensor data volumes are executed over the structured data sets to achieve efficiency in processing with minimum delay.

3.1.1.3 **Hadoop Distributed File System** (**HDFS**): The result of the Hive queries are stored within HDFS from where it is transferred to the disk storage of VMs.

3.1.2 **Cloud database**: The cloud database stores the necessary information of the VMs, the public and private IPs of the VMs, the keys to connect with the VMs, and the mapping of the VMs with the respective end-users. The database also maintains the metadata of the structured VS information of the different VMs.

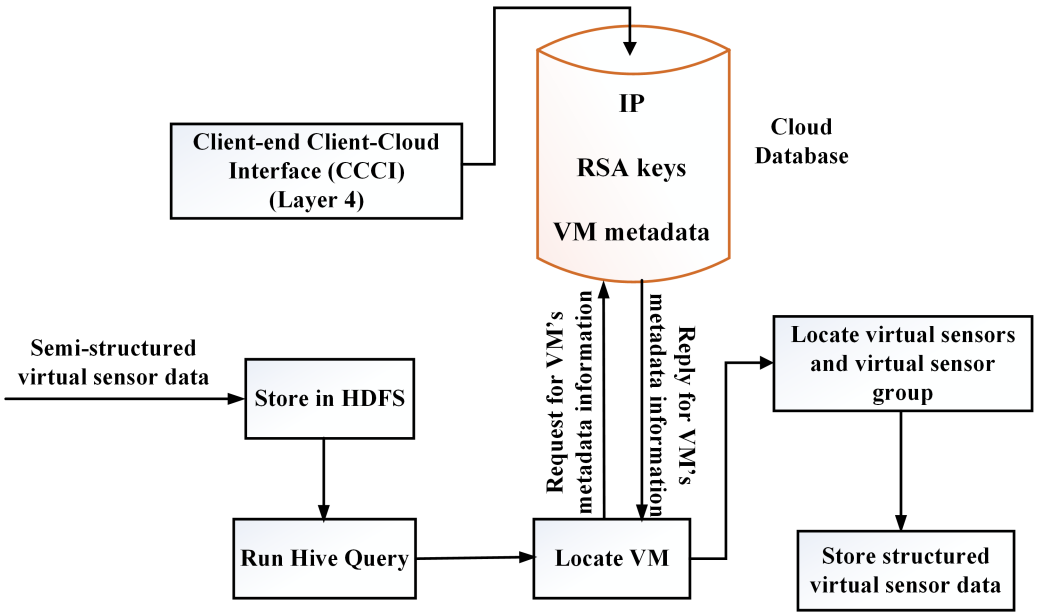


Figure 7.6: Block diagram of layer 3

Layer 4: Provisioning Se-aaS to the end-users

The topmost layer of BSCI is the organizational layer, in which, multiple organizations request for Se-aaS from the BSCSP. The organizations are connected to the Big-Sensor-Cloud through the CCCI. Followed by the user login operation, the end-users are allowed to get connected to the VM and access it.

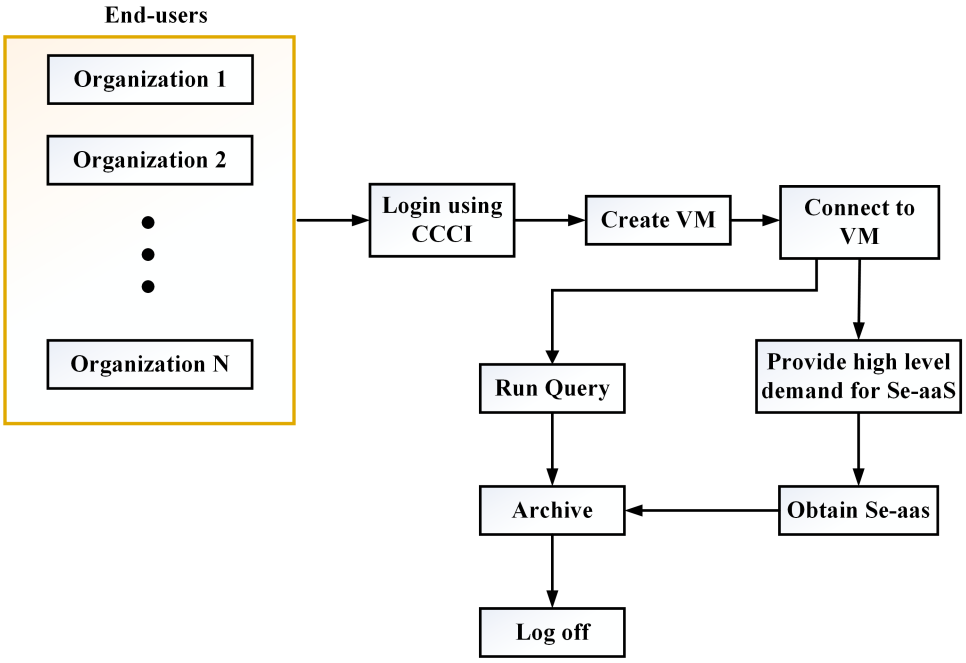


Figure 7.7: Block diagram of layer 4

7.6 Performance Evaluation

In this Section, the performance of the proposed BSCI platform is analyzed. The analysis is covered in two distinct subsections. In the first subsection, the existing sensor-cloud infrastructure is compared to the proposed BSCI and the afore-mentioned bottleneck of sensor-cloud platforms are studied, discussed, and analyzed. Followed by this, a comparative study is also performed to investigate the sustainability of the two platforms. The experimental setup is illustrated in Table 7.2.

Table 7.2: Experimental setup

Parameters	Values
Processor	Intel(R) Core(TM) $i3-2105$ CPU @ 3.10 GHz
RAM	2 GB, DDR3
Disk space	320 GB
Operating system	Ubuntu 14.04 LTS
Query types	DDL, DML, Retrieval
No. end-users	20
No. of sensor-owner	10
Type of sensor	5
Record count	$[5, 10, 15] \times 10^6$
Time	5 years (60 months)
Nodes registered by sensor-owner (n_1)	1000
Number of nodes in the WSN (n_2)	1000
Number of faulty nodes (n_3)	100
Nodes used monthly (n_4)	50
Unit cost price of a node (C_s)	20 currency unit
Unit cost due to deployment (C_{deploy})	3 currency unit
Unit cost due to maintenance ($C_{maintain}$)	10 currency unit/month
Unit cost due to rent (C_{rent})	10 currency unit/month
Cost per unit usage of Se-aaS (C_{Se-aaS})	10 currency unit/month

7.6.1 Explanation of Parameters

The parameters used for this work have been selected on the basis of prior related works [104–106]. The values of these parameters are set as per the related works in this domain that are most cited [107–111].

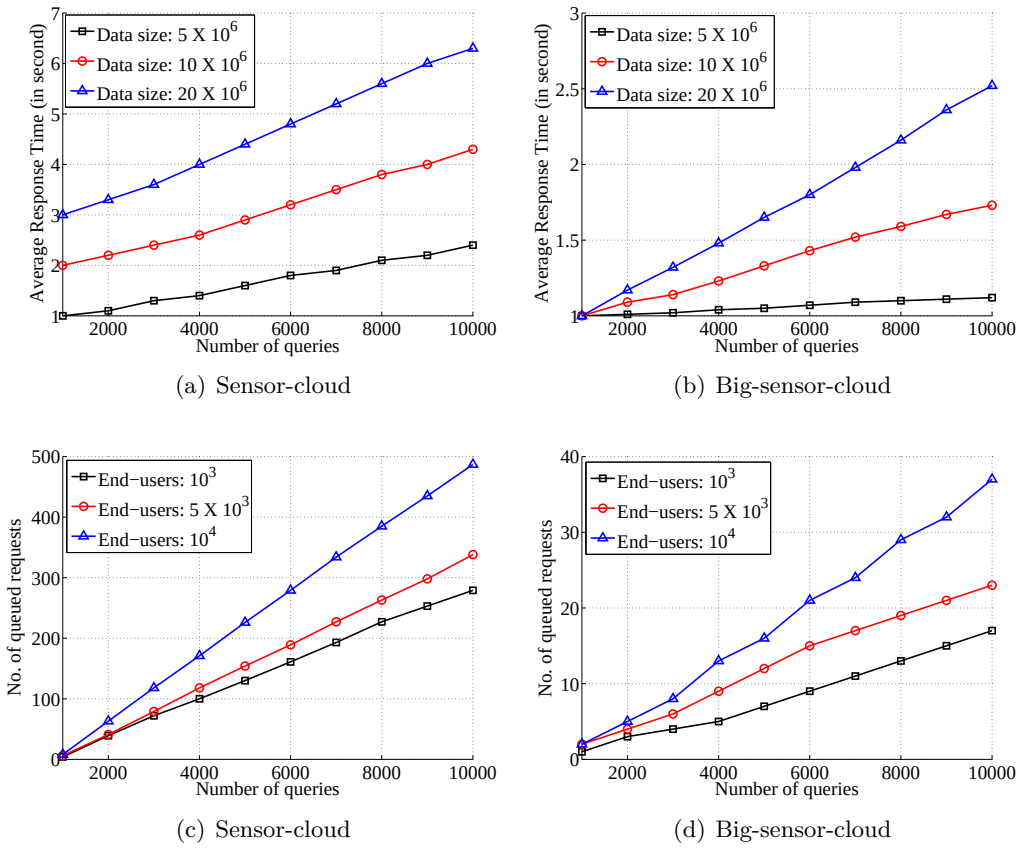
7.6.2 Bottleneck Analysis of Existing Sensor-cloud

Figure 7.8 studies and analyzes the of BSCI in comparison to traditional sensor-cloud. To compare the performance of both the platforms, two different metrics are considered – average response time, and the number of queued requests, which are defined below.

Definition 31. *The average response time ($\mathcal{R}_{q_i}$) of a query q_i, triggered at time t_i, is the time difference between the time instant (t_{si}) when the processing for q_i commenced*

and the time instant when the query was triggered.

$$\mathcal{R}_{q_i} = t_{si} - t_i \tag{7.1}$$



(a) Sensor-cloud (b) Big-sensor-cloud

(c) Sensor-cloud (d) Big-sensor-cloud

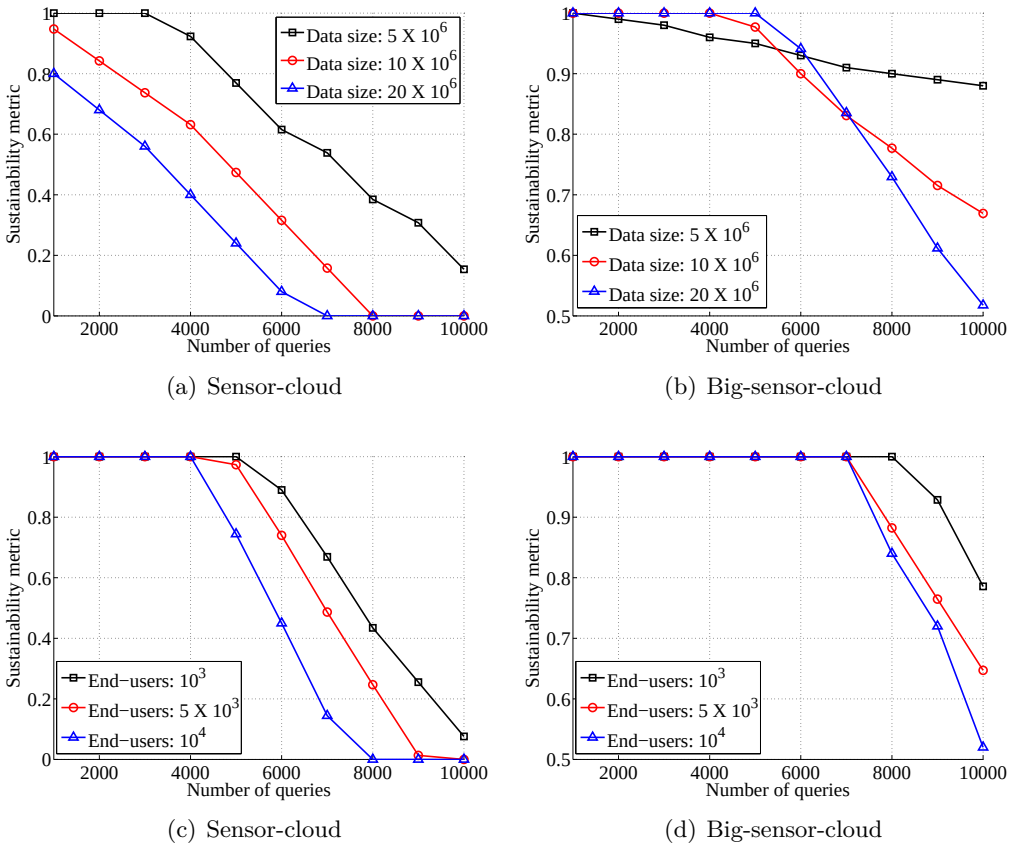
Figure 7.8: Comparative analysis of performance in sensor-cloud and big-sensor-cloud platforms

Therefore, for q number of queries, the average response time, $\hat{\mathcal{R}}_q$, is obtained as,

$$\hat{\mathcal{R}}_q = \sum_{i=1}^{q} \mathcal{R}_{q_i} \tag{7.2}$$

To study the average response time, an experimentation is performed by varying the number of queries from 2000 to 10000 and the average response time for every query is plotted by varying the data set of every query from 5 million to 20 million. Figure

7.8(a) reflects that with the increase in the query count and the data set size, the average response time is increased from 2.5 to 7 second averaging at 3.5 second whereas, Figure 7.8(b) indicates that even with large query count and data set sizes, the average response time varies from 1.2 to 2.5 second with the mean being at 1.5 second. Therefore, the response time is in general faster in BSCI than in sensor-cloud platforms.



(a) Sensor-cloud (b) Big-sensor-cloud

(c) Sensor-cloud (d) Big-sensor-cloud

Figure 7.9: Comparative analysis of sustainability

The second experimentation is done to analyze the number of requests that may remain queued in a heavy-traffic scenario, i.e., when the query count and the number of end-users increase rapidly. For a total of q application requests, the number of queued applications $(q', q' \leq q)$ is given as:

$$q' = q - q_{opt} \tag{7.3}$$

where,

$$q_{opt} \in \{1, q\} \text{ and, } \mathcal{R}_{q_{opt}} \gg \mathcal{R}_{q_{th}}, \mathcal{R}_{q_{opt}} > \mathcal{R}_{q_{opt}-1} \tag{7.4}$$

where q_{opt} is the optimal query count beyond which the average response time of a query exceeds the threshold response time, $\mathcal{R}_{q_{th}}$, and thus, creates the bottleneck.

In Figure 7.8(c), it is observed that as the query count increases, more and more application requests are being queued and the situation is worse when the count of end-users reaches up to 10^4. It can be observed that around 500 applications are left unserved maximally in such a situation. On the contrary, in a heavy traffic scenario, BSCI maintains a moderate length of application queue varying from 18 to 38 with the mean count being close to 20.

To examine the sustainability of the two platforms, the metric of sustainability is defined as follows:

Definition 32. *Sustainability ($\mathcal{S}$) is expressed within a scale of* 0 *to* 1 *and is defined as the proportion of queries that can be responded within the average response time threshold and with the number of queued requests within the permissible limit. Therefore,*

$$\mathcal{S} = \begin{cases} 0.5\left(\frac{q-q_{opt}}{q} + \frac{\sum_{i=1}^{q_{opt}} \mathcal{R}_{q_i}}{\sum_{i=1}^{q} \mathcal{R}_{q_i}}\right), & q' < q_{opt}, \hat{\mathcal{R}}_q < \mathcal{R}_{q_{th}} \\ 0.5\left(\frac{q-q'}{q} + \frac{\sum_{i=1}^{q'} \mathcal{R}_{q_i}}{\sum_{i=1}^{q} \mathcal{R}_{q_i}}\right), & otherwise \end{cases} \tag{7.5}$$

Figure 7.9 reflects the sustainability of BSCI and sensor-cloud by varying the two metrics – average response time, and the number of queued requests. From Figure 7.9(a), the sustainability reduces and eventually dies off in sensor-cloud with the increase in the average response time. However, it is better for BSCI where it sustains much longer

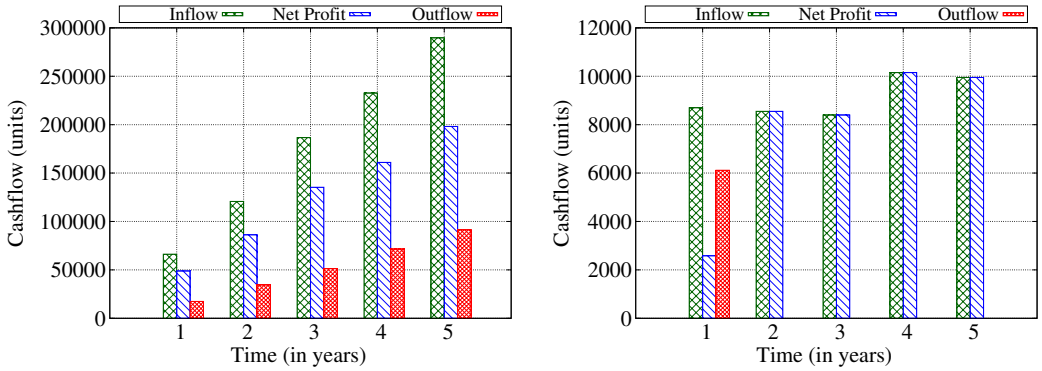
for data size of 5 and 10 million. The sustainability reduces with larger data sets of 20 million. As observed from Figure 7.9(b), the sustainability is temporarily high with higher data size even when the number of queries is low. From the proposed definition of sustainability (Definition 32), it can be observed that, one of the factors on which $\mathcal{S}$ depends on is the the number of queued requests within the permissible limit. With a very large number of requests, the queue overflows thereby pushing the remaining requests in an un-queued state, i.e., the requests are not admitted to be processed. This reduces the sustainability. However, when the number of queries is low, it is likely to to accomodate all the requests into the queue and hence, there are no requests that are un-queued. Thus, even with higher data size, the sustainability increases. Figure 7.9(c) depicts that sensor-cloud loses sustainability with the increase in the number of queued requests. However, for BSCI, as shown in Figure 7.9(d), the value indicated by the sustainability metric is improved and retained for a longer time.

7.6.3 Performance Analysis of BSCI

For the purpose of testing the performance of the proposed system, another set of different experimental results are illustrated. Initially, a cash flow analysis is performed to investigate the profitability of BSCI. Thereafter, experiments are performed to comparatively analyze the performance of sensor-cloud and BSCI in terms of the query execution time.

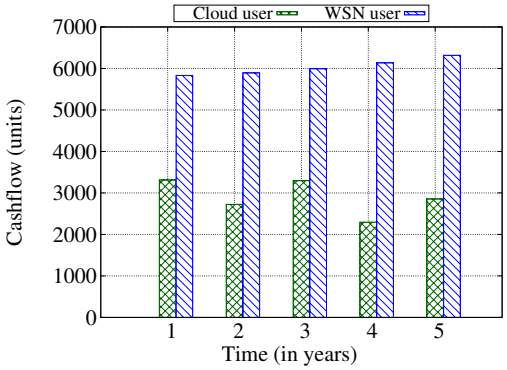
The theoretical analysis of the cash flow for the different actors of sensor-cloud are already obtained in Chapter 3. Now the Equations and the plots are re-validated through the proposed BSCI prototype.

The experimental results, as shown in Figure 7.10(a), demonstrate the annual cash inflow, outflow, and net profit of the BSCSP for a period of 5 years. The BSCSP serves 20 end-users, and pays rental fees to 10 sensor owners. The net profit is computed by subtracting the cash outflow from the cash inflow, and is found to be positively



(a) Analysis for Big-Sensor-Cloud Service Provider

(b) Analysis for sensor-owners



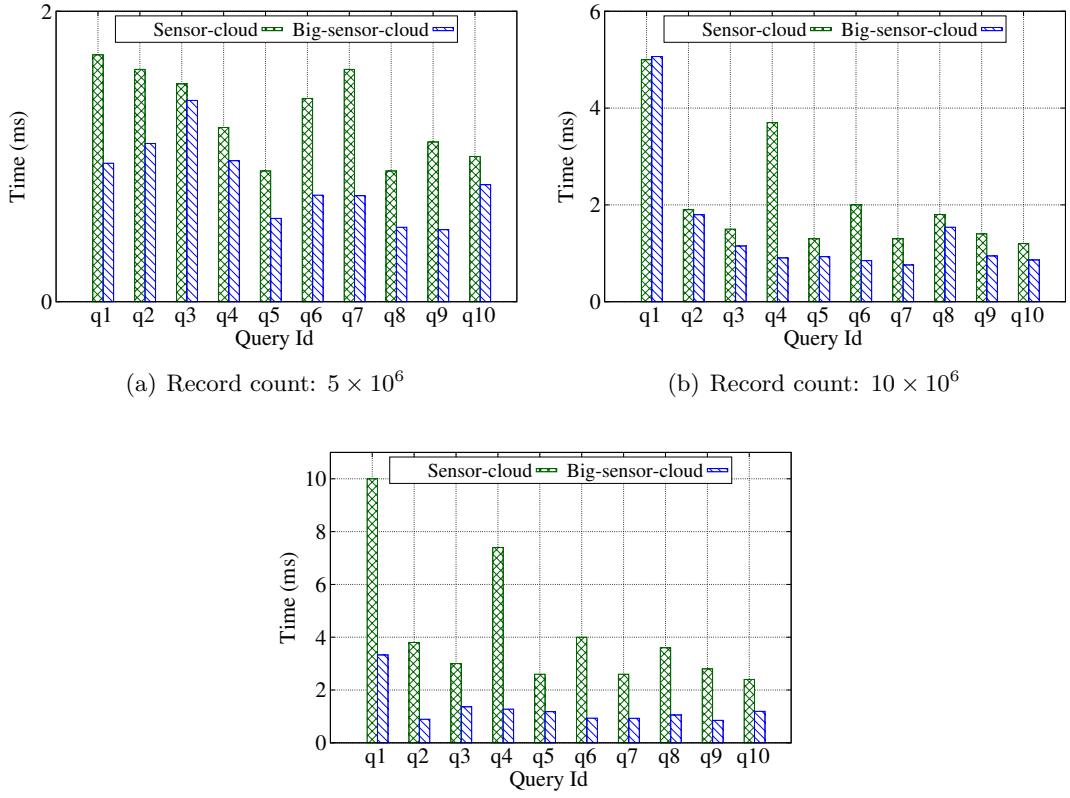
(c) Comparison between WSN user and Big-Sensor-Cloud user

Figure 7.10: Cumulative cash flow analysis for the various actors of BSCI

increasing over time. Figure 7.10(b) shows the average cash flows for a sensor-owner with variable usage of his/her sensor devices. As shown in the Figure, the sensor-owner experiences a cash outflow only in the first year in terms of investments in procuring the sensor nodes. The outflow is nullified in the subsequent years and the inflow is based on the rental fee for the corresponding sensor usage, thereby incurring a positive net profit. In order to examine the cash flows of the end-user (in case of both normal WSNs and Big-Sensor-Cloud), the inflows are not directly measurable as such, in terms of the usage of Se-aaS. A comparative study of the cash outflow is analyzed for both the cases in Figure 7.10(c). The cash outflow of a WSN user access is due to several reasons – *deployment*, *maintenance*, and *overhead*, whereas the end-users of Big-Sensor-Cloud pay

on a per-usage basis only for the units of Se-aaS that they consume. The obtained results from Figure 7.10 have similar trends to Figure 3.2 of Chapter 3, thereby supporting the validation of the proposed prototype of BSCI.

For the sake of comparison of performance of sensor-cloud and BSCI, several query types involving DML, DDL, and data retrieval are executed on varied data volumes. A comparative study is performed in terms of the execution time of the queries both in sensor-cloud and BSCI.



(a) Record count: 5×10^6

(b) Record count: 10×10^6

(c) Record count: 20×10^6

Figure 7.11: Analysis of DML query execution time

Figure 7.11 presents a comparative study of DML query execution time for different data sets with varied size of data. Initially, experiments were performed on a data set with 5×10^6 number of records or data entries, as shown in Figure 7.11(a). For both of

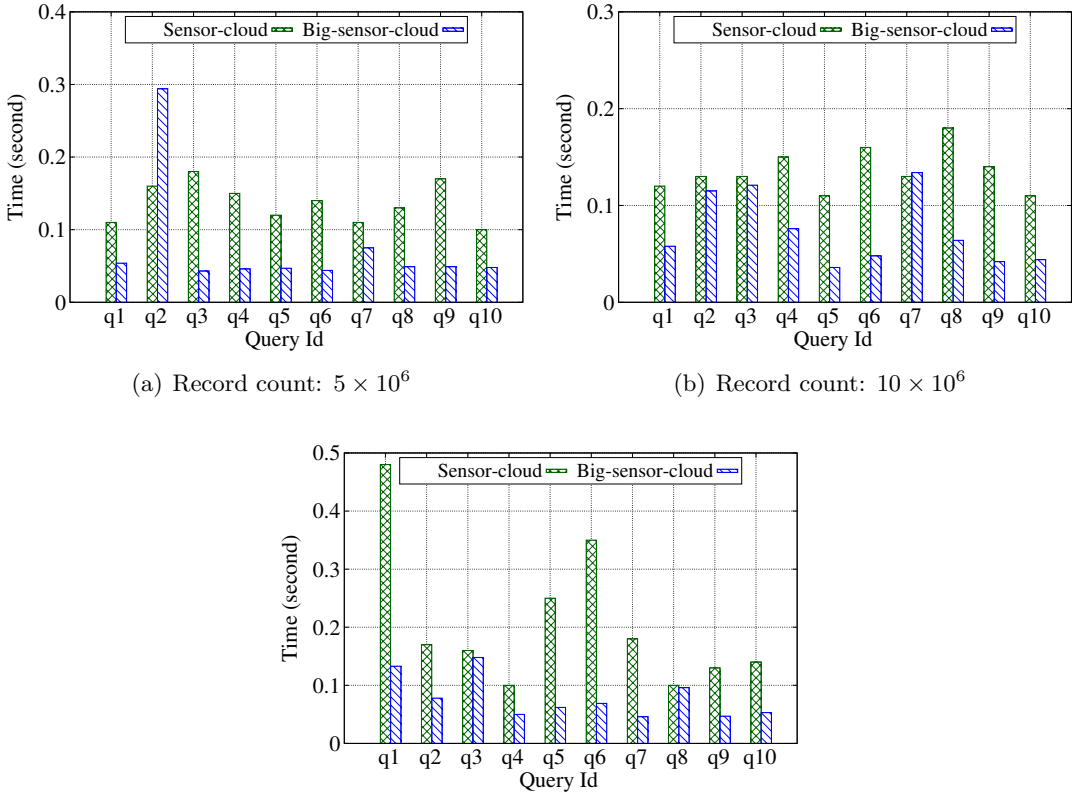
the paradigms, the query execution time varies insignificantly. However, the execution time is consistently low in BSCI, compared to that in sensor-cloud. This is primarily because of the fact that the entire data set undergoes thorough structurization within the Hive instance, whereas in sensor-cloud, it requires manipulation of every data entry within a table of a database. For the data-set with 10×10^6 number of entries, as shown in Figure 7.11(b), the execution time is marginally higher for BSCI, in case of query 1 ($q1$). The principal reason behind this is the fact that the first query might require some additional operations related to initial setup and configuration. For subsequent queries ($q2$ to $q10$), BSCI outperforms sensor-cloud significantly. In Figure 7.11(c), the results of experimentation with data sets comprising of 20×10^6 entries (or records) are shown. A huge improvement of the query execution time is observed when BSCI is used. Therefore, DML query execution time is improved using BSCI compared to sensor-cloud.

Similar experiments are performed for DDL data queries, the results of which are summarized in Figure 7.12. For the data-set with 5×10^6 number of records, the query execution time is marginally lower for BSCI, as illustrated in Figure 7.12(a). As the number of data entries increases to 10×10^6 (see Figure 7.12(b)), there is substantial reduction in the query execution time. Finally, in Figure 7.12(c), the improvement is maximum for BSCI, compared to that in sensor-cloud.

Figure 7.13 highlights the analysis of query execution time for retrieval of varied data-sets. For the retrieval type of queries, the reduction in query execution time is substantially lower in most of the queries. In fact, as shown in Figure 7.13(c), BSCI outperforms sensor-cloud remarkably.

7.7 Summary

This Chapter discusses the development of a prototype implementation of BSCI for realizing Se-aaS. The work addresses the problems of existing sensor-cloud infrastructure in terms of its processing ability. Unlike sensor-cloud, BSCI is a platform that handles



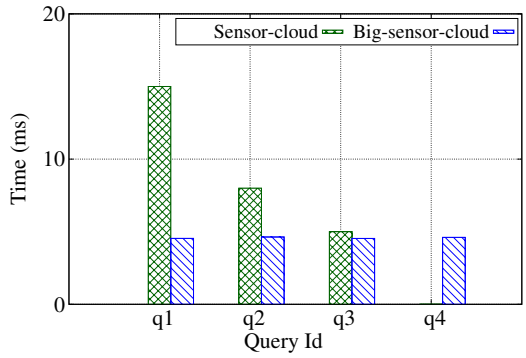
(a) Record count: 5×10^6

(b) Record count: 10×10^6

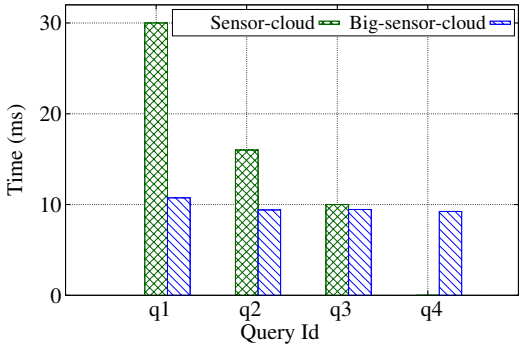
(c) Record count: 20×10^6

Figure 7.12: Analysis of DDL query execution time

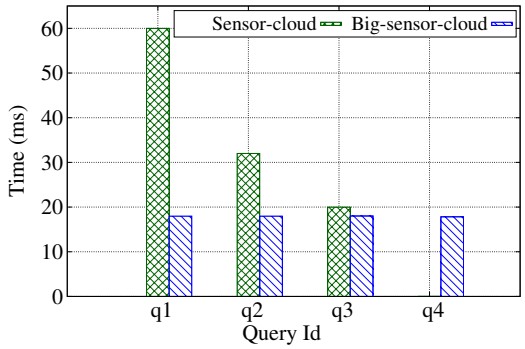
the processing, structuring, and orientation of big-data generated from multiple organizations, simultaneously. Within each VM of BSCI, an HDFS and a Hive instance are installed to enable the distributed processing of the voluminous and heterogeneous data. The work illustrates the implementation details of the infrastructure. The experimental results (for DDL, DML, and data retrieval queries) highlight the enhancement achieved through BSCI over sensor-cloud platforms in terms of the query execution time. The cash flow analysis, done for various actors, justifies the prospect of BSCI from a business perspective.



(a) Record count: 5×10^6



(b) Record count: 10×10^6



(c) Record count: 20×10^6

Figure 7.13: Analysis of retrieval query execution time

Chapter 8

Application Specific Analysis of Sensor-cloud Infrastructure: Target Tracking

The previous Chapters of the dissertation discusses the various challenges involved in conceptualizing sensor-cloud infrastructure and the dissertation eventually presents a working prototype of sensor-cloud in Chapter 7. After establishing the prototype implementation of sensor-cloud, this dissertation focuses to validate it by executing a single WSN-based application through it. This Chapter mounts a very popular WSN-based application to a sensor-cloud platform and examines the consequent performance of the application.

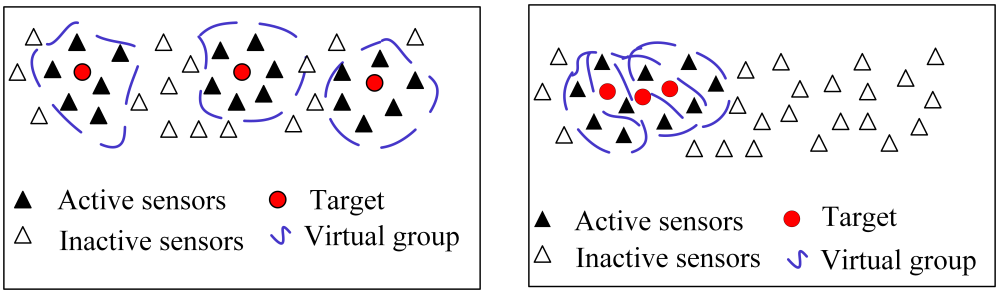
Target tracking in WSNs is already explored in the recent past and is one of the most popular WSN-based applications. This Chapter focuses to mount a common algorithm of target tracking application within sensor-cloud and analyze the uncertainties that might come up while executing the traditional sensor-based applications using sensor-cloud.

As mentioned earlier, sensor-cloud is a computational platform that may involve multiple organizations with heterogeneous demands. In such a multi-organizational sce-

nario, conventional target tracking applications are anticipated to encounter a difficulty. The difficulty considers a scenario of tracking multiple targets when the mobile targets, belonging to different organizations, may come so close that they might fall under the sensing range of one or more physical sensor nodes, thereby leading to overlapping coverage of the targets by the sensor nodes. It is important to generate distinct clusters (VSs) of sensor-nodes corresponding to each of the targets. The difficulty in addressing the problem is that in a conventional WSN, the sensor nodes are typically equipped with some basic analytical and decision-making abilities, and algorithmic processing of data occurs within each sensor node followed by transmission of the target-specific aggregated data. However, in a sensor-cloud environment, sensor nodes are treated as mere sensing units with minimal network management and end user supervision. The raw sensed data are directly transmitted to the sensor-cloud environment where it as aggregated in a target specific manner, and then transmitted to the end-users. This introduces the challenge to manage distinct sensor clusters for each target. In such a scenario, the problem of sensor-target mapping induces research interest.

8.1 Contribution of the Chapter

The proposed work is not a trivial extension of the existing works, as prior related works are implemented on conventional WSNs. The contribution of this work is to address the above-mentioned issues within sensor-cloud by correctly mapping sensors to their corresponding targets, assuming that a sensor node covers one (Figure 8.1(a)) or more (Figure 8.1(b)) targets, at a particular time instant. It is required to perform a *primary mapping* of the physical sensor nodes to targets followed by a *secondary mapping* of physical sensor nodes to virtual sensor groups. The work performs a *utility-based* primary mapping within the sensor-cloud environment, simply from the raw sensed data and the previous knowledge about the targets.



(a) Distinct cluster (b) Non-distinct cluster

Figure 8.1: Local cluster formation

8.2 S-DMA: Social choice based Dynamic Mapping Algorithm

It is assumed that every sensor node is capable of estimating the target coordinates from its sensor reading. There are two symmetric distance matrices $X(1..N)$ and $Y(1..N)$, which contain the global coordinates of every node, i, represented as $C(i) = (X[i], Y[i])$. It is required to detect the presence of overlapping sensor coverage area involving two or more targets.

8.2.1 Detection of Overlapping Coverage

It is assumed that n_t is the total number of targets tracked within a sensor-cloud environment. The location coordinates of a detected target t_i at time t are denoted by (x_{t_i},y_{t_i}). The sensor-cloud infrastructure uses the sensed data (x_{t_i},y_{t_i}) at time $t-1$ and predicts (x'_{t_i},y'_{t_i}) at time t using standard location estimation algorithms. Nodes that are within d distance from the target t_i are activated, where the value of the distance parameter d is predetermined. Thus, $\xi((x'_{t_i}, y'_{t_i}), C(j)) \leq d$, for each j, where ξ denotes the Euclidean metric in a 2D plane. The metric $\xi(p, q)$ between two points p and q is expressed as $\xi(p, q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$. Thus, an array of active sensor nodes is obtained from the above relation. To determine the formation of overlapping regions, it

is needed to examine, for every selected node s_i, such that $\xi(C(s_i), t_k) > d,\ k = n_t - 1$. In other words, only for a single target t_j, the inequality must hold true to ensure non-overlapping sensor coverage. However, if an overlap is detected, the scheduling of sensor nodes needs to be governed and steered, accordingly. It is assumed that a total of n_s overlapped sensor nodes are detected for n_t targets.

8.2.2 Calculation of 'Eligibility' Factor of a Sensor Node

Initially, for every possible sensor-target combination, a boolean parameter, named 'eligibility' factor is defined. It is the output of a binary function $u(\cdot, \cdot)$, referred to as the 'eligibility' function. The function is expressed as a mapping $u : S \text{ X } T \rightarrow [0,1]$, where S is the set of overlapping sensor nodes selected for the current set of targets, and T is the current set of targets to be tracked. If ρ_{s_i} is the sensing radius of a node, the mapping u is defined as:

$$u(s_i, t_j) = \begin{cases} 0, & \xi(s_i, t_j) > \rho_{s_i} \\ 1, & \text{otherwise} \end{cases} \tag{8.1}$$

8.2.3 Computation of Nodal Preference

Prior to computing the nodal preferences, a new metric termed Coverage Contraction Factor (CCF) is introduced.

Definition 33. *Coverage Contraction Factor (α_{s_i}) is introduced to examine a node's energy content and it computes the residual battery status of a sensor node. CCF is expressed as:*

$$\alpha_{s_i} = (E_{act,s_i} - E_{cur,s_i})/E_{act,s_i} \tag{8.2}$$

where E_{cur} and E_{act} are the current and initial energy levels of a node and $0 \leq \alpha_{s_i} \leq 1$.

As the approach of the proposed work is based on the *Theory of Social Choice* [141], every active sensor node has its own preference of targets, articulated by means of a linear ordering. Preference P_i and indifference I_i of a node i are the symmetric and

asymmetric components of the relation, respectively [141]. Thus, $t_a P_i t_b \not\Rightarrow t_b P_i t_a$. Also, $t_a I_i t_b \Rightarrow t_b I_i t_a \Rightarrow t_a \equiv t_b$.

Now, the focus is on evaluating a node's ordering of preferences for each target of interest at time t. After the data from the physical sensor nodes are transmitted, nodal preferences are evaluated on servers of the sensor-cloud infrastructure. A utility function Ψ is designed for every sensor-target pair. Thus,

$$\Psi(s_i, t_j) = \begin{cases} \frac{\lambda_1}{\alpha_{s_i}} + \lambda_2 \frac{\rho_{s_i}}{\xi(s_i, t_j)} & , \alpha_{s_i} \neq 0 \\ \mathbb{B} + \lambda_2 \frac{\rho_{s_i}}{\xi(s_i, t_j)} & , \alpha_{s_i} = 0 \end{cases} \tag{8.3}$$

where $\mathbb{B}$ is a large integral value, and λ_1, λ_2 (when $\lambda_1 < \lambda_2$) are the weighted system-modeled coefficients. Nodal ordering of preferences of targets are based on a *preference* value Θ, which is defined as:

$$\Theta(s_i, t_j) = \Psi(s_i, t_j) \times u(s_i, t_j) \tag{8.4}$$

If a node is not *eligible* for tracking a particular target, the *preference* value is zero. Having calculated the *preference* value for every sensor-target pair, each sensor node then creates its own ordering of choices. For a sensor node s_i, $\Theta(s_i, t_a) > \Theta(s_i, t_b) \Rightarrow t_a P_{s_i} t_b, \Theta(s_i, t_a) = \Theta(s_i, t_b) \Rightarrow t_a I_{s_i} t_b$. However, the preference ordering for every sensor node should be *complete* and *transitive* [141]. Hence, it implies,

$$(t_j P_{s_i} t_k) \vee (t_j I_{s_i} t_k),\ \forall j, k \in T \tag{8.5}$$

$$(t_j X_{s_i} t_k) \wedge (t_k X_{s_i} t_l) \Rightarrow t_j X_{s_i} t_l,\ \forall j, k, l \in T \tag{8.6}$$

where $X_{s_i} = \{P_{s_i}, I_{s_i}\}$. Equations (8.5) and (8.6) ensure the *completeness* and *transitivity* axioms, respectively. After obtaining the preferences of every node at time t, a matrix $\Theta_{net}[1..n_s][1..n_t]$ is obtained for the entire network. Now, some relevant terms

are defined as follows:

Definition 34. *A preference ordering R_{t_i} for a particular target t_i is the set of preference values of the different sensor nodes casted for t_i, i.e., $R_{t_i} = \{\Theta_{s_1,t_i}, \Theta_{s_2,t_i}, .., \Theta_{s_{n_s},t_i}\}$.*

Definition 35. *A preference profile $\mathcal{P}$ is the set of potential preferences, i.e., $\mathcal{P} = \{R_{t_1}, R_{t_2}, .., R_{t_{n_t}}\}$.*

8.2.4 Social Choice Aggregation

Once the preference profile is established, the *Social Aggregation Function* (*SAF*) and *Social Choice Function* (*SCF*) are embarked on. The SAF is defined as a mapping $F : \mathcal{P}^{n_s!} \rightarrow \mathbb{R}^{n_t \times n_{th}}$, i.e., F maps a preference domain to a mapping matrix M, $M[i][j]$ denotes the allocation of the j^{th} sensor node to target t_i, n_{th} is the threshold value for the maximum number of sensor nodes that can be allocated to a target. SCF f is defined as a mapping $f : \mathcal{P} \times T \rightarrow S$, i.e., given a preference profile and a particular target, a particular sensor node or the social choice winner s_{win} can be mapped to the target based on the choice of the society.

$$F(\mathcal{P}) = F(R_{t_1}, R_{t_2}, .., R_{t_{n_t}}) = M, f(R_{t_i}) = s_{win} \tag{8.7}$$

In this work, it is assumed that $n_t \ll n_s$. Multiple iterations are performed on random ordering of targets till all sensor nodes are allocated. For the selection of a 'fair' winner, as per *Arrow's Impossibility Theorem* [142], the positive effects of *Plurality Voting* [141] and *Borda's algorithm* [141] are merged with the proposed algorithm, as presented in Algorithm 1. For t_i, the society mean μ_{soc} is formulated and expressed as,

$$\mu_{soc} = \frac{\sum_{\forall s_j \in S} \beta_{s_j} \times \Theta(s_j, t_i)}{\sum_{\forall s_j \in S} \beta_{s_j}} \tag{8.8}$$

where $\{\beta_{s_j} \times \Theta(s_j, t_i)\}$ denotes the is the social preference order. $r(s_j, R_{t_i})$ is the posi-

tional value[1] of a voter in the profile of a target, expressed as, $\beta_{s_j} = n_s - r(s_j, R_{t_i})$. The winner node s_{win} is obtained by,

$$s_{win} = M[i][win] = \min_{\forall s_j \in S} |(\mu_{soc} - \Theta_{s_j,t_i})| \tag{8.9}$$

The winner node s_{win} can be considered as the *Plurality* winner, as its score for t_i is the closest to the society mean, thereby earning the highest ability to win the target.

Input:

1. Set of mobile targets: T.

2. Set of sensor nodes with overlapping coverage: S.

Output:

A mapping matrix $M[1..n_t][1..n_{th}]$.

```
for s_i ∈ S do
    for t_j ∈ T do
        Compute Θ(s_i, t_j)
    end
end
while (∃s_i ∈ S) ∧ (s_i ≠ M[p][q]), ∀p ∈ T, q ∈ S do
    Generate a random ordering of targets T̂
    for j = 1 to n_t do
        M[j][win] = s_win = f(R_{T̂_j})
        Remove s_win from S, 𝒫
    end
end
```

Algorithm 6: S-DMA algorithm

[1]Positional significance of a node can be viewed as its Borda score. However, it is not explicitly termed as 'Borda' score, as the Borda score is ideally applicable to candidates instead of voters.

8.3 Analytical Results

Proposition 8.3.1. *The worst case asymptotic computational complexity of S-DMA and communicational complexity for a single target are $O(n_s \times n_t)$ and $O(n_s^2)$, respectively.*

Proof. For n_t number of mobile targets, step 1 of *S-DMA* is computed in $n_s \times n_t$ time. Steps 3, 5, and 6 of Algorithm 1 is executed in constant time c_1. Steps 4 through 7 takes $O(n_t)$. Thus,

$$T(n_t) = n_s n_t + T(n_s - 1) + c_1 n_t, \;\; T(1) = c_2$$

Simplifying, it can be obtained that, $T(n_t) = O(n_s \times n_t)$. For communication load, every node, s_i, communicates with the sensor-cloud through multi-hop route. The number of hops is $O(i-1)$. Communication load C for a single target involving n_s overlapping sensors can be expressed as $C(1) = \Theta(1), C(n_s) = \sum_{i=2}^{n_s} O(i-1) \simeq O(n_s^2)$. This completes the proof. □

Lemma 8.3.2. *S-DMA satisfies non-dictatorship.*

Proof. An SCF is dictatorial if $\exists s_i : t_a P_{s_i} t_b \Rightarrow t_a P_{s_j} t_b, \forall s_j \in S$ [142]. But in *S-DMA*, $\forall s_i \in S$, $\Theta_{s_i,..}$. Also, $\nexists s_i : t_a P_{s_i} t_b \Rightarrow t_a P_{s_j} t_b$, $\forall s_j \in S$. For any target t_i, if $f(R_{t_i}) = s_{win}$, $\nexists s_i$, such that, $f(R_{t_i}) = s_j$, where $S = S - \{s_i\}$, or $S = S + \{s_i\}, s_i \neq s_j$. Thus, *S-DMA* is non-dictatorial. □

Lemma 8.3.3. *S-DMA satisfies the Independence of Irrelevant Alternatives.*

Proof. Independence of Irrelevant Alternatives (*IIA*) claims that the internal ranking between two alternatives is independent of a third alternative [142]. In *S-DMA*, let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two preference sub-profiles containing $\hat{S}$ sensor nodes, and let t_i an t_j be two target alternatives, such that $\hat{S} \subset S, t_i \succeq_{\mathcal{P}_1} t_j$ and, $t_i \succeq_{\mathcal{P}_2} t_j$. Then, *S-DMA* concludes that $\forall s_i \in \hat{S}$. Thus, $\Theta_{s_i,t_i} \geq \Theta_{s_i,t_j} \Rightarrow \sum_{\forall s_i \in \hat{S}} \Theta_{s_i,t_i} \geq \sum_{\forall s_i \in \hat{S}} \Theta_{s_i,t_j} \Rightarrow t_i \succeq_{\hat{S}} t_j$. This concludes the proof. □

Corollary 8.3.4. *S-DMA dissatisfies the Pareto Axiom (P).*

Explanation: From Arrow's Impossibility Theorem [142], it can be obtained that no aggregation function can simultaneously satisfy non-dictatorship, IIA and P. Thus, from Lemma III.1 and III.2 it can be inferred that *S-DMA* dissatisfies P.

Theorem 8.3.5. *S-DMA tends to select the Condorcet winner.*

Proof. The sensor node preferences are assumed as (t_a, t_b, t_c), (t_b, t_a, t_c), (t_b, t_c, t_a), (t_c, t_b, t_a), (t_c, t_a, t_b), and (t_a, t_c, t_b). Let t_a be the Condorcet winner (through pairwise voting) for some sensor node s_i. Therefore,, $\Theta_{s_2} + \Theta_{s_3} + \Theta_{s_4} + \Theta_{s_5} \leq \Theta_{s_1} + \Theta_{s_6}$. Assuming the correctness of a Condorcet winner, a hyper-plane is considered and divided into distinct sensor node regions. The normal to the correct side is obtained as $N_1 = (1, -1, -1, -1, -1, 1)$. In *S-DMA* let the positional significance be (2,γ,0). From Equation 8.9, it is obtained that, $\gamma\Theta_{s_2} + \gamma\Theta_{s_5} \leq 2\Theta_{s_1} + 2\Theta_{s_6}$. Thus, $N_2 = (2k, -\gamma k, 0, 0, -\gamma k, 2k)$, assuming $\Theta_{s_i,t_a} = k, \forall s_i \in S$. If ϕ is the angle between N_1 and N_2, $\cos(\phi) = \frac{N_1.N_2}{|N_1||N_2|} = \frac{2\gamma+4}{\sqrt{6}\sqrt{2\gamma^2+16}} = h(\gamma)$. Thus,

$$\frac{dh(\gamma)}{d\gamma} = \frac{1}{\sqrt{6}} \frac{2\sqrt{2\gamma^2+16} - 4(2\gamma^2+16)^{-\frac{1}{2}}(2\gamma+4)}{(2\gamma^2+16)} \tag{8.10}$$

The angle between the normals should be minimized to obey Condorcet criterion. Evaluating, $\frac{dh(\gamma)}{d\gamma} = 0$, it is found that $\gamma \to 1$. Thus, correct positional value is assigned by following *S-DMA*. This concludes the proof. □

Proposition 8.3.6. *S-DMA respects Plurality voting.*

Proof. Plurality voting selects a winner agent, which obtains the highest score of the society. From Equation 8.9, it is found that the winner node s_{win} satisfies $|(\mu_{soc} - \Theta_{s_{win},t_i})| \to 0 \Rightarrow \Theta_{s_{win},t_i} \to \mu_{soc}$. Thus, $\Theta_{s_{win}}$ respects μ_{soc}, which is a society parameter. Hence, s_{win} is also the *Plurality* winner. □

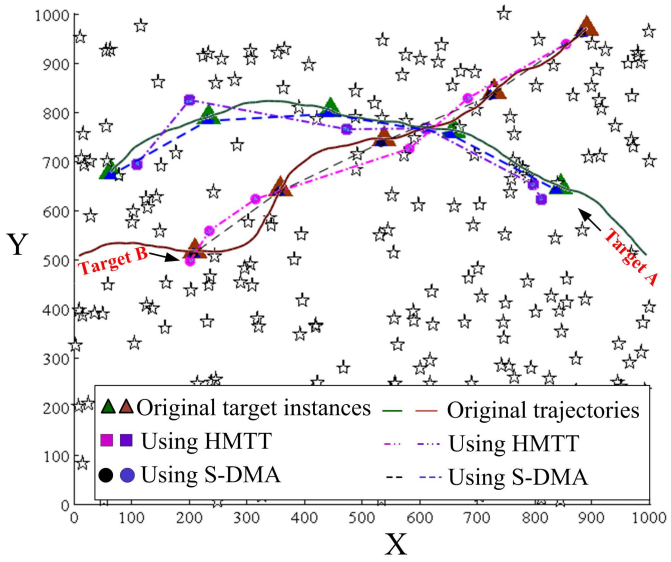


Figure 8.2: Projection of S-DMA against HMTT

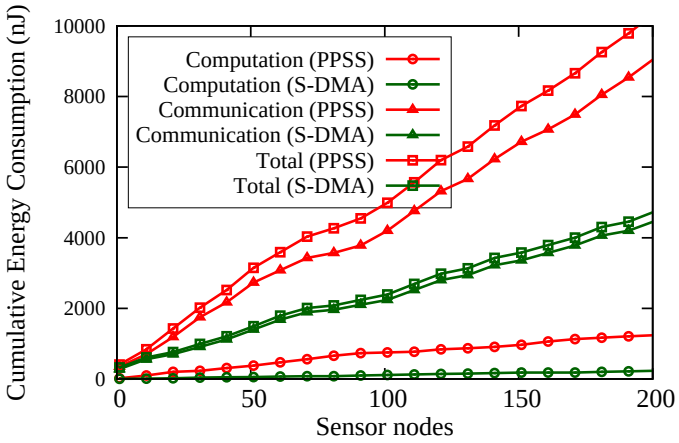
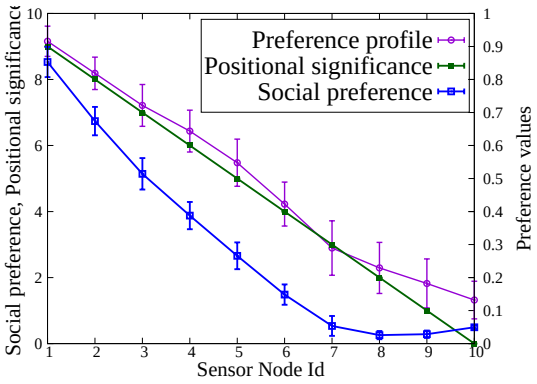
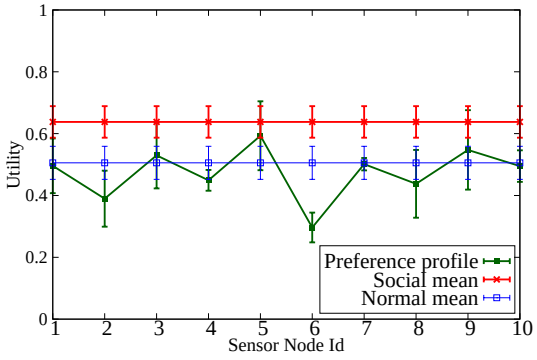


Figure 8.3: Comparison of energy consumption

Now some of the experimental results are presented. To validate the correctness of *S-DMA*, a *uniform random deployment* of 250 sensor nodes (ρ_{s_i} = 200 m, $C_{s_i} \geq 2\rho_{s_i}$) is assumed in an area of 1 km x 1 km, C_{s_i} being the communication range. Figure 8.2 shows that two targets enter the zone, and move close to each other. *S-DMA* clearly outperforms the existing algorithm —Hierarchical Markov Decision Process (HMDP) for target tracking (HMTT) [143], in terms of tracking accuracy. Unlike [143], *S-DMA* proposes a “fair" sensor-target mapping, which improves the tracking accuracy especially



(a) Target specific nodal preferences



(b) Analysis of R_{t_i}

Figure 8.4: Values related to the preference profile

in situations of overlapping coverage of sensors. Assuming the communication and processing energy as 40 nJ/bit and 10 nJ/bit, respectively, and subjecting the algorithms to identical sensing phenomenon, Figure 8.3 clearly shows that, unlike Probability-based Prediction and Sleep Scheduling protocol (PPSS) [144], S-DMA exhibits a low energy consumption for computation as the processing and evaluation is mainly executed at the sensor-cloud end. Further, in PPSS, multi-hop communication within the network and data transmission to a data center contributes for the overall communication energy. On the contrary, inter node communication is negligible in S-DMA, multi-hop transmission being the main component for energy consumption. This conserves the total energy appreciably. To enhance the understandability, Figure 8.4(a) depicts the combined im-

pact of positional significance and the preference values on the collective preferences of a target for 10 sensor nodes. The preference profile curve is aligned to the secondary y axis. Figure 8.4(b) demonstrates the difference of magnitude of the normal mean from the social mean. The experiment is executed to compute the standard deviation of the summation of preference values for the sensors assigned to each target. The mean of the standard deviations over 100 iterations was found close to 0.71 with a 99% confidence interval. This suggests that the proposed algorithm is unbiased to targets and maintains uniformity while mapping.

8.4 Conclusion

The proposed algorithm, *S-DMA*, ensures the best possible allocation of sensors to targets. However, there can be challenges if two adjacent sensor nodes are heterogeneous with respect to their sensing types, as multi-hop communication in such scenario will require protocol standardization. Our future work will focus on extending the current problem in the context of such heterogeneous sensor nodes.

Chapter 9

Summary and Conclusion

The dissertation focuses on the development of sensor-cloud infrastructure. The proposed research was initiated in 2013, when all of the existing research works focused primarily on the dogma, the principles, and the conceptualization of sensor-cloud. However, research work that provide scientific and technological concreteness to the concept of sensor-cloud infrastructure were scarce. From an implementation viewpoint, the scope of technical and theoretical research in this domain is identified to mathematically suggest the shift of paradigm from traditional WSNs. The goal of this research is to resolve the technical challenges and eventually build a fully-functional prototype of sensor-cloud. The proposed research has succeeded in building a partially-functional prototype of sensor-cloud infrastructure, which is currently the only available infrastructure rendering Se-aaS, to the best of my knowledge.

9.1 Summary of the dissertation

The problem that is aimed to be solved is that with the existing state-of-the-art (i.e. traditional WSNs), the common mass of people cannot enjoy the very emerging sensor technology without being directly involved with the purchase, deployment, maintenance, and management of the sensor nodes. The problem is intense and nontrivial as most

of the end-users are naive and overheads associated with a WSN are expensive. To address the afore-mentioned problem, the proposed research focuses on building a holistic prototype of sensor-cloud infrastructure, rendering Se-aaS – the first attempts of its kind. While building the prototype, innumerable challenges and difficulties are identified as potential research objectives of the dissertation.

Chapter 2 presents the literature survey on the domain of sensor-cloud platform. The Chapter is broken down into distinct sections comprising of the evolution, pricing, and the networking aspects in sensor-cloud platforms. In the end, those works, that have practical contributions from a developer's point of view, are elaborately studied and analyzed.

Chapter 3 presents a thorough theoretical characterization of virtualization within sensor-cloud infrastructure. Some interesting features of sensor-cloud platforms are also investigated and studied. Followed by the theoretical characterization, the Chapter experimentally compares the network performance for both traditional WSNs and sensor-cloud platforms. Further, from an economic viewpoint, a cash flow analysis is also performed to determine the profit for all the actors of sensor-cloud.

In Chapter 4, the motivation behind the need for a data caching mechanism within sensor-cloud is presented. Subsequently, the Chapter presents a dynamic and adaptive data aching policy that simultaneously maintains the information accuracy and reduces the network overhead.

The pricing policy to be followed within sensor-cloud is investigated and studied in Chapter 5. The Chapter clearly divides the aspect of pricing into two distinct categories – pricing due to hardware and pricing due to infrastructure. The Chapter propounds two different algorithms for both hardware and infrastructure and eventually shows that the proposed pricing scheme optimizes the profit of the CSP, thereby maintaining the end-user satisfaction.

Chapter 6 focuses on the networking of multiple DCs involved in a sensor-cloud plat-

form. Considering the fact that multiple VSs serving an application may be temporarily formed at geo-spatially distributed DCs, however, eventually, it is required to choose a single DC that handles all data processing, management, aggregation for the particular application. To address this problem, the Chapter proposes a DC scheduling algorithm that chooses the correct DC, thereby optimizing the network overhead associated with it.

In Chapter 7, the holistic prototype building of sensor-cloud is thoroughly discussed and analyzed. Some of the limitations of sensor-cloud platforms are identified and a modified platform is constructed. Finally, experiments are performed to examine the performance of the existing and the modified sensor-cloud prototypes.

Once the prototype is built, Chapter 8 focuses to validate it by executing a common WSN based application using it. In this specific work, the chosen application is target tracking. It is observed that the application had to be modified because of the difficulty encountered due to this paradigm shift. However, the difficulty was resolved by proposing a new algorithm.

9.2 Contribution of Our Work

The proposed research has focused on building a holistic prototype of sensor-cloud infrastructure, rendering Se-aaS. The primary contributions of the dissertation have been as follows:

- *Theoretical characterization of sensor-cloud and justification for a paradigm shift from conventional WSNs*: Initially, the work has focused on the theoretical modeling of virtualization of physical sensor nodes. The necessity for a paradigm shift for all WSN-based applications to a sensor-cloud platform has been experimentally justified. The proposed work has suggested a framework for performance analysis of sensor-cloud based on few chosen metrics such as fault-tolerance, lifetime of a

sensor node, and energy consumption, in contrast to that of a WSN. Finally, this work has endeavored to conceive the idea of using physical Se-aaS.

- *Data caching policies with the infrastructure*: An optimal caching mechanism within sensor-cloud has been proposed to obtain resource efficiency in terms of energy and network lifetime. The proposed data caching mechanism is dynamic, and is adaptive to the change of the physical environment. Thereby, it preserves the accuracy of information and conserves the network resources, simultaneously.

- *Designing of a dynamic and optimal pricing scheme, specifically for Se-aaS*: As a cloud computing platform generally conforms with a pay-per-use model, within sensor-cloud platforms, the end-users utilize the physical sensors and the cloud infrastructure as per their demand and pay as per their use, to the CSP. Thus, a pricing scheme has been developed for Se-aaS to quantify the use by the end-users and charge them accordingly.

- *Optimal DC scheduling and networking withing DCs for QoS management*: The work has proposed a DC scheduling algorithm for routing and channelization of the data of the VSs originating from multiple regions, to geographically distributed sensor-cloud data centers (DCs). The work has also focused to choose a single DC serving a particular application by reducing the network overhead simultaneously.

- *Functional prototype development for Se-aaS*: This work has dealt with the development of a prototype of sensor-cloud infrastructure using real sensor hardware and cloud platform. The work has identified the limitations of the basic sensor-cloud infrastructure and has proposed a modified infrastructure.

- *Application specific analysis of sensor-cloud infrastructure*: This work has considered the challenges and uncertainties that might arise while executing the traditional sensor-based applications using sensor-cloud. The problem has been inves-

tigated in a multiple target tracking application scenario, using the sensor-cloud platform.

9.3 Future Scope of Work

In future, sensor-cloud platforms can induce significant research interests in the following topics:

- Future works may include details of design issues, and standardization of communication protocols for sensor-cloud infrastructure.
- Schemes for optimization of sharing and coherence of resources can also be proposed. Additionally, each type of application can be analyzed for understanding the distinctness of its behavior within a sensor-cloud environment.
- It is also motivating to explore additional issues associated with the QoS parameters of sensor-cloud infrastructure. The examination of cost-effectiveness due to caching also induces research attention. Other aspects of sensor virtualization can also be considered as a relevant direction of future research.
- Further, another interesting area is extending the problem for dynamic shifting of VMs among DCs for serving applications, thereby ensuring localized load sharing and balancing among the DCs. Revisiting the problem for a mobile sensor-cloud scenario also induces research interest.

References

[1] S. Misra and S. Singh, "Localized Policy-Based Target Tracking Using Wireless Sensor Networks," *ACM Transactions on Sensor Networks*, vol. 8, no. 3, July 2012.

[2] N. Watthanawisuth, A. Tuantranont, and T. Kerdcharoen, "Design for the next generation of wireless sensor networks in battlefield based on ZigBee," in *Defense Science Research Conference and Expo (DSR)*, Singapore, August 2011, pp. 1–4.

[3] H. Yang, Y. Qin, G. Feng, and H. Ci, "Online Monitoring of Geological CO_2 Storage and Leakage Based on Wireless Sensor Networks," *IEEE Sensors Journal*, vol. 13, no. 2, pp. 556 – 562, February 2013.

[4] S. Jimenez-Fernandez, P. de Toledo, and F. del Pozo, "Usability and Interoperability in Wireless Sensor Networks for Patient Telemonitoring in Chronic Disease Management," *IEEE Transactions on Bio-Medical Engineering*, vol. 60, no. 12, pp. 3331 – 3339, November 2013.

[5] C. Bachmann, M. Ashouei, V. Pop, and M. Vidojkovic, "Low-power wireless sensor nodes for ubiquitous long-term biomedical signal monitoring," *IEEE Comm. Magazine*, vol. 50, no. 1, pp. 20 – 27, Jan 2012.

[6] Y. Yuan, J. Zhao, and C. Qiu, "Estimating Crowd Density in an RF-Based Dynamic Environment," *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3837 – 3845, Oct 2013.

[7] X. Wang, M. Chen, T. Kwon, and H.-C. Chao, "Multiple Mobile Agents Itinerary Planning in Wireless Sensor Networks: Survey and Evaluation," *IET Communications*, vol. 5, no. 12, pp. 769–1776, 2011.

[8] Y. Rachlin, R. Negi, and P. K. Khosla, "The Sensing Capacity of Sensor Networks," *IEEE Transactions on Information Theory*, vol. 57, pp. 1675–1691, March 2011.

[9] Y. Liu, K. Liu, and M. Li, "Passive Diagnosis for Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, vol. 18, pp. 1132–1144, August 2010.

[10] R. Shorey, A. Ananda, M. C. Chan, and W. T. Ooi, *Mobile, Wireless, and Sensor Networks: Technology, Applications, and Future Directions.* Wiley-IEEE Press, 2006.

[11] S. S. Iyengar, N. Parameshwaran, V. V. Phoha, N. Balakrishnan, and C. D. Okoye, *Fundamentals of Sensor Network Programming: Applications and Technology.* Wiley-IEEE Press, 2010.

[12] A. Marchiori and Q. Han, "A Two-Stage Bootloader to Support Multi-application Deployment and Switching in Wireless Sensor Networks," in *International Conference on Computational Science and Engineering*, 2009, pp. 71–78.

[13] J. L. Hill, "System Architecture for Wireless Sensor Networks," Ph.D. dissertation, University of California, Berkeley, 2003.

[14] 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*, vol. 2013, Nov 2013.

[15] K.-L. Tan, "What's NExT?: Sensor + Cloud!?" in *DMSN*, ser. ACM International Conference Proceeding Series, D. Zeinalipour-Yazti and W.-C. Lee, Eds. ACM, 2010.

[16] L. Ferretti, F. Pierazzi, M. Colajanni, and M. Marchetti, "Scalable Architecture for Multi-User Encrypted SQL Operations on Cloud Database Services," *IEEE Transactions on Cloud Computing*, vol. 2, pp. 448–458, Oct 2014.

[17] X. Zhang, L. Yang, C. Liu, and J. Chen, "A Scalable Two-Phase Top-Down Specialization Approach for Data Anonymization Using MapReduce on Cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 2, pp. 363–373, Feb 2014.

[18] K.-W. Park, J. Han, J. Chung, and K. H. Park, "THEMIS: A mutually verifiable billing system for the cloud computing environment." *IEEE Transactions on Services Computing*, vol. 6, no. 3, pp. 300–313, 2013.

[19] 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 (NBiS)*, Sept 2010, pp. 1–8.

[20] Open Geospatial Consortium. http://www.opengeospatial.org/.

[21] M. Botts, Ed., *Sensor Model Language (SensorML) for In-situ and Remote Sensors*, Open Geospatial Consortium Inc., 2004.

[22] C.-F. Lai, H. Wang, H.-C. Chao, and G. Nan, "A Network and Device Aware QoS Approach for Cloud-Based Mobile Streaming," *IEEE Transactions on Multimedia*, vol. 15, no. 4, pp. 747–757, June 2013.

[23] C.-F. Lai, Y.-X. Lai, H.-C. Chao, and J. Wan, "Cloud-assisted Real-time Transrating for HTTP Live Streaming," *IEEE Wireless Communications Magazine*, vol. 20, no. 3, pp. 62–70, June 2013.

[24] S. E. Plummer, "The GLOBCARBON Cloud Detection System for the Along-Track Scanning Radiometer (ATSR) Sensor Series," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 6, pp. 1718–1727, June 2008.

[25] M. M. E. A. Mahmoud and X. Shen, "A Cloud-Based Scheme for Protecting Source-Location Privacy against Hotspot-Locating Attack in Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 10, pp. 1805–1818, Oct 2012.

[26] S. Misra, S. Bera, A. Mondal, R. Tirkey, H.-C. Chao, and S. Chattopadhyay, "Optimal gateway selection in sensor-cloud framework for health monitoring," *IET Wireless Sensor Systems*, vol. 3, no. 4, December 2013.

[27] C. Zhu, Z. Sheng, V. C. M. Leung, L. Shu, and L. T. Yang, "Toward Offering More Useful Data Reliably to Mobile Cloud From Wireless Sensor Network," *IEEE Transactions on Emerging Topics in Computing*, vol. 3, no. 1, pp. 84–94, March 2015.

[28] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM Computing Surveys*, vol. 35, no. 3, pp. 114–131, 2003.

[29] M. M. Hassan, B. Song, and E.-N. Huh, "A Framework of Sensor-Cloud Integration Opportunities and Challenges," in *International Conference on Ubiquitous Information Management and Communication*, 2009.

[30] M. Eggert, R. Haubling, M. Henze, L. Hermerschmidt, R. Hummen, D. Kerpen, A. N. Perez, B. Rumpe, D. Thiben, and K. Wehrle, "SensorCloud: Towards the

Interdisciplinary Development of a Trustworthy Platform for Globally Interconnected Sensors and Actuators," RWTH Aachen University, Germany, Tech. Rep., 2013.

[31] L. D. Kumar, S. Grace, A. Krishnan, V. M. Manikandan, R. Chinraj, and M. R. Sumalatha, "Data Filtering in Wireless Sensor Networks Using Neural Networks for Storage in Cloud," in *International Conference on Recent Trends In Information Technology (ICRTIT)*, 2012.

[32] 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*, vol. 2013, November 2012.

[33] S. Olariu, A. Wada, and L. Wilson, "Wireless sensor networks: Leveraging the virtual infrastructure," *IEEE Network*, vol. 18, no. 4, pp. 51–56, July 2004.

[34] T. Ojha, M. Khatua, and S. Misra, "Tic-Tac-Toe-Arch: a self-organising virtual architecture for Underwater Sensor Networks," *IET Wireless Sensor Systems*, vol. 3, no. 4, pp. 307–316, April 2013.

[35] C. Zhu, H. Nicanfar, V. C. M. Leung, and L. T. Yang, "An Authenticated Trust and Reputation Calculation and Management System for Cloud and Sensor Networks Integration," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 118–131, Jan 2015.

[36] C. Zhu, V. C. M. Leung, L. T. Yang, and L. Shu, "Collaborative Location-Based Sleep Scheduling for Wireless Sensor Networks Integratedwith Mobile Cloud Computing," *IEEE Transactions on Computers*, vol. 64, no. 7, pp. 1844–1856, July 2015.

[37] Y. Xu and A. Helal, "Scalable Cloud-Sensor Architecture for the Internet of Things," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 285–298, June 2016.

[38] M. V. Nguyen and E.-N. Huh, "An Efficient Key Management for Secure Multicast in Sensor-Cloud," in *First ACIS/JNU International Conference on Computers, Netw, Systems and Industrial Engineering (CNSI)*, May 2011, pp. 3–9.

[39] A. Chandra, Y. Lee, B. M. Kim, S. Y. Maeng, S. H. Park, and S. R. Lee, "Review on Sensor Cloud and Its Integration with Arduino Based Sensor Network," in *International Conference on IT Convergence and Security (ICITCS)*, Dec 2013, pp. 1–4.

[40] S. Bhunia, J. Pal, and N. Mukherjee, "Fuzzy Assisted Event Driven Data Collection from Sensor Nodes in Sensor-Cloud Infrastructure," in 14^{th} *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, May 2014, pp. 635–640.

[41] S. Z. Yale and S. Zhong, "Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks," in *Proceedings of IEEE INFOCOM*, 2002, pp. 1987–1997.

[42] P. Chavali and A. Nehorai, "Managing Multi-Modal Sensor Networks Using Price Theory," *IEEE Transactions on Signal Processing*, vol. 60, no. 9, pp. 4874–4887, Sept 2012.

[43] J. Elias, F. Martignon, L. Chen, and E. Altman, "Joint Operator Pricing and Network Selection Game in Cognitive Radio Networks: Equilibrium, System Dynamics and Price of Anarchy," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 9, pp. 4576–4589, Nov 2013.

[44] S. Li and J. Huang, "Price Differentiation for Communication Networks," *IEEE/ACM Transactions on Networking*, vol. 22, pp. 703–716, June 2014.

[45] S.-K. Ng and W.-G. Seah, "Game-Theoretic Approach for Improving Cooperation in Wireless Multihop Networks," *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics*, vol. 40, pp. 559–574, June 2010.

[46] L. Buttyan and J.-P. Hubaux, "Enforcing Service Availability in Mobile Ad-Hoc WANs," in *First Annual Workshop on Mobile and Ad Hoc Networking and Computing (MobiHOC)*, August 2000, pp. 87–96.

[47] R. Kaewpuang, D. Niyato, P. Wang, and E. Hossain, "A Framework for Cooperative Resource Management in Mobile Cloud Computing," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 12, pp. 2685–2700, December 2013.

[48] I. A. Kash and P. B. Key, "Pricing the Cloud," *IEEE Internet Computing*, vol. 20, no. 1, pp. 36–43, Jan 2016.

[49] S. Arevalos, F. Lopez-Pires, and B. Baran, "A Comparative Evaluation of Algorithms for Auction-Based Cloud Pricing Prediction," in *IEEE International Conference on Cloud Engineering (IC2E)*, April 2016, pp. 99–108.

[50] Z. Li and M. Li, "A Hierarchical Cloud Pricing System," in *IEEE* 9^{th} *World Congress on Services (SERVICES)*, June 2013, pp. 403–411.

[51] H. Xu and B. Li, "Dynamic Cloud Pricing for Revenue Maximization," *IEEE Transactions on Cloud Computing*, vol. 1, pp. 158–171, July 2013.

[52] W. Wang, B. Liang, and B. Li, "Revenue maximization with dynamic auctions in IaaS cloud markets," in *IEEE/ACM* 21^{st} *International Symposium on Quality of Service*, June 2013, pp. 1–6.

[53] H. Xu and B. Li, "Maximizing revenue with dynamic cloud pricing: The infinite horizon case," in *IEEE International Conference on Communications (ICC)*, June 2012, pp. 2929–2933.

[54] B. Sharma, R. Thulasiram, P. Thulasiraman, S. Garg, and R. Buyya, "Pricing Cloud Compute Commodities: A Novel Financial Economic Model," in 12^{th} *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, May 2012, pp. 451–457.

[55] S. Son and K. M. Sim, "A Price-and-Time-Slot-Negotiation Mechanism for Cloud Service Reservations," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 3, pp. 713 – 728, 2012.

[56] N. Nasiriani, C. Wang, G. Kesidis, B. Urgaonkar, L. Y. Chen, and R. Birke, "On Fair Attribution of Costs under Peak-Based Pricing to Cloud Tenants," in *IEEE* 23^{rd} *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Oct 2015, pp. 51–60.

[57] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Exploiting Task Elasticity and Price Heterogeneity for Maximizing Cloud Computing Profits," *IEEE Transactions on Emerging Topics in Computing*, vol. PP, no. 99, pp. 1–1, 2015.

[58] Y. Chi, X. Li, X. Wang, V. C. M. Leung, and A. Shami, "A Fairness-Aware Pricing Methodology for Revenue Enhancement in Service Cloud Infrastructure," *IEEE Systems Journal*, vol. PP, no. 99, pp. 1–12, 2015.

[59] A. Prasad and S. Rao, "A Mechanism Design Approach to Resource Procurement in Cloud Computing," *IEEE Transactions on Computers*, vol. 63, no. 1, pp. 17–30, Jan 2014.

[60] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimization of Resource Provisioning Cost in Cloud Computing," *IEEE Transactions on Services Computing*, vol. 5, no. 2, pp. 164–177, April 2012.

[61] S. Ren and M. van der Schaar, "Joint design of Dynamic Scheduling and Pricing in wireless cloud computing," in *Proceedings of IEEE INFOCOM*, April 2013, pp. 185–189.

[62] H. Roh, C. Jung, W. Lee, and D.-Z. Du, "Resource pricing game in geo-distributed clouds," in *IEEE INFOCOM*, April 2013.

[63] H. Qin, X. Wu, J. Hou, H. Wang, W. Zhang, and W. Dou, "Self-Adaptive Cloud Pricing Strategies with Markov Prediction and Data Mining Method," in *International Conference on Cloud and Service Computing (CSC)*, Nov 2012, pp. 219–226.

[64] I. Jangjaimon and N. Tzeng, "Effective Cost Reduction for Elastic Clouds under Spot Instance Pricing through Adaptive Checkpointing," *IEEE Transactions on Computers*, vol. PP, no. 99, pp. 1–1, 2013.

[65] V. Kantere, D. Dash, G. Francois, S. Kyriakopoulou, and A. Ailamaki, "Optimal Service Pricing for a Cloud Cache," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, pp. 1345–1358, Sept 2011.

[66] N. Edalat, W. Xiao, C.-K. Tham, E. Keikha, and L.-L. Ong, "A price-based adaptive task allocation for Wireless Sensor Network," in *IEEE* 6^{th} *International Conference on Mobile Adhoc and Sensor Systems*, Oct 2009, pp. 888–893.

[67] Z. Hui, Q. Zhi-hong, S. Da-yang, and Z. Ding-guo, "Study on Price-Driven Load Balance in Wireless Sensor Network," in *International Conference on Information Engineering (ICIE)*, vol. 1, July 2009, pp. 355–358.

[68] R. Greenwell, X. Liu, and K. Chalmers, "Pricing Intelligence as a Service for Cloud Computing," in *IEEE* 5^{th} *International Conference on Cloud Computing Technology and Science*, vol. 2, Dec 2013, pp. 244–247.

[69] A. Gohad, N. C. Narendra, and P. Ramachandran, "Monetizing the Cloud: Pricing Model Governance," in *IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, Oct 2013, pp. 1–6.

[70] K. Nagothu, B. Kelley, M. Jamshidi, and A. Rajaee, "Persistent Net-AMI for Microgrid Infrastructure Using Cognitive Radio on Cloud Data Centers," *IEEE Systems Journal*, vol. 6, no. 1, pp. 4–15, March 2012.

[71] X. Xu, J. Wu, G. Yang, and R. Wang, "Low-power task scheduling algorithm for large-scale cloud data centers," *Journal of Systems Engineering and Electronics*, vol. 24, no. 5, pp. 870–878, Oct 2013.

[72] H. Wu, A. Tantawi, Y. Diao, and W. Wang, "Adaptive memory load management in cloud data centers," *IBM Journal of Research and Development*, vol. 55, no. 6, pp. 5:1–5:10, Nov 2011.

[73] Z. Zheng, J. Wang, J. Ren, W. Hou, and J. Wang, "Least Maintenance Batch Scheduling in Cloud Data Center Networks," *IEEE Communications Letters*, vol. 18, no. 6, pp. 901–904, June 2014.

[74] X. Yuchi and S. Shetty, "Hierarchical Random Graph Based Network Diversity Modeling for the Cloud," in *2016 IEEE World Congress on Services (SERVICES)*, June 2016, pp. 35–38.

[75] F. Larumbe and B. Sanso, "A Tabu Search Algorithm for the Location of Data Centers and Software Components in Green Cloud Computing Networks," *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, pp. 22–35, Jan 2013.

[76] C. Mastroianni, M. Meo, and G. Papuzzo, "Probabilistic Consolidation of Virtual Machines in Self-Organizing Cloud Data Centers," *IEEE Transactions on Cloud Computing*, vol. 1, no. 2, pp. 215–228, July 2013.

[77] A. Beloglazov and R. Buyya, "Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers under Quality of Service Constraints," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1366–1379, July 2013.

[78] D. Bruneo, "A Stochastic Model to Investigate Data Center Performance and QoS in IaaS Cloud Computing Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, pp. 560–569, March 2014.

[79] Y. Hua, X. Liu, and H. Jiang, "ANTELOPE: A Semantic-Aware Data Cube Scheme for Cloud Data Center Networks," *IEEE Transactions on Computers*, vol. 63, no. 9, pp. 2146–2159, Sept 2014.

[80] C. Assi, S. Ayoubi, S. Sebbah, and K. Shaban, "Towards Scalable Traffic Management in Cloud Data Centers," *IEEE Transactions on Communications*, vol. 62, no. 3, pp. 1033–1045, March 2014.

[81] W. Zhang, Y. Wen, and H.-H. Chen, "Toward transcoding as a service: energy-efficient offloading policy for green mobile cloud," *IEEE Network*, vol. 28, no. 6, pp. 67–73, Nov 2014.

[82] H. Liang, L. Cai, D. Huang, X. Shen, and D. Peng, "An SMDP-Based Service Model for Interdomain Resource Allocation in Mobile Cloud Networks," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 5, pp. 2222–2232, Jun 2012.

[83] H. Li, M. Dong, K. Ota, and M. Guo, "Pricing and Repurchasing for Big Data Processing in Multi-Clouds," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 266–277, April 2016.

[84] B. Sharma, R. K. Thulasiram, P. Thulasiraman, and R. Buyya, "Clabacus: A Risk-Adjusted Cloud Resources Pricing Model Using Financial Option Theory," *IEEE Transactions on Cloud Computing*, vol. 3, no. 3, pp. 332–344, July 2015.

[85] M. Aazam, E. N. Huh, M. St-Hilaire, C. H. Lung, and I. Lambadaris, "Cloud Customer's Historical Record Based Resource Pricing," *IEEE Transactions on Parallel and Distributed Systems*, no. 7, pp. 1929–1940, July 2016.

[86] P. Massonet, S. Dupont, A. Michot, A. Levin, and M. Villari, "An architecture for securing federated cloud networks with Service Function Chaining," in *IEEE Symposium on Computers and Communication (ISCC)*, June 2016, pp. 38–43.

[87] M. Filer, J. Gaudette, M. Ghobadi, R. Mahajan, T. Issenhuth, B. Klinkers, and J. Cox, "Elastic optical networking in the microsoft cloud," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 8, no. 7, pp. A45–A54, July 2016.

[88] L. Jiao, A. Tulino, J. Llorca, Y. Jin, and A. Sala, "Smoothed online resource allocation in multi-tier distributed cloud networks," in *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2016, pp. 333–342.

[89] I. Demydov, O. Lavriv, Z. Kharkhalis, and M. M. El-Hatri, "Concept of the migrating firewall to scalable cloud networks," in 13^{th} *International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET)*, Feb 2016, pp. 643–645.

[90] S. Murugesan and I. Bojanova, *Cloud Network and I/O Virtualization.* Wiley-IEEE Press, 2016, ch. S. Murugesan and I. Bojanova, p. 744. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7493824

[91] F. Salim, M. D. Pena, Y. Petrov, N. Sony, B. Wu, and A. A. Saad, "EnviS Tag, Scan, View: A Location-Based App for Visualizing Spatio-temporal Data from Sensor Cloud," in *IEEE* 15^{th} *International Conference on Mobile Data Management*, vol. 1, July 2014, pp. 329–332.

[92] A. A. Chandra, Y. Lee, B. M. Kim, S. Y. Maeng, S. H. Park, and S. R. Lee, "Review on Sensor Cloud and Its Integration with Arduino Based Sensor Network," in *International Conference on IT Convergence and Security (ICITCS)*, Dec 2013, pp. 1–4.

[93] B. K. Sen, S. Khatua, and R. K. Das, "Target coverage using a collaborative platform for sensor cloud," in *IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS)*, Dec 2015, pp. 1–6.

[94] A. Sen and S. Madria, "A Risk Assessment Framework for Wireless Sensor Networks in a Sensor Cloud," in *IEEE* 16^{th} *International Conference on Mobile Data Management*, vol. 2, June 2015, pp. 38–41.

[95] S. Saha, "Secure sensor data management model in a sensor - cloud integration environment," in *Applications and Innovations in Mobile Computing (AIMoC)*, Feb 2015, pp. 158–163.

[96] N. Kedia, "Water quality monitoring for rural areas- a Sensor Cloud based economical project," in 1^{st} *International Conference on Next Generation Computing Technologies (NGCT)*, Sept 2015, pp. 50–54.

[97] L. Neto, J. Reis, D. Guimaraes, and G. Goncalves, "Sensor cloud: SmartComponent framework for reconfigurable diagnostics in intelligent manufacturing environments," in *IEEE* 13^{th} *International Conference on Industrial Informatics (INDIN)*, July 2015, pp. 1706–1711.

[98] M. Hirafuji, H. Yoichi, T. Kiura, K. Matsumoto, T. Fukatsu, K. Tanaka, Y. Shibuya, A. Itoh, H. Nesumi, N. Hoshi, S. Ninomiya, J. Adinarayana, D. Sudharsan, Y. Saito, K. Kobayashi, and T. Suzuki, "Creating high-performance/low-cost ambient sensor cloud system using OpenFS (Open Field Server) for high-throughput phenotyping," in *Proceedings of SICE Annual Conference (SICE)*, Sept 2011, pp. 2090–2092.

[99] C. Srimathi, S.-H. Park, and N. Rajesh, "Proposed framework for underwater sensor cloud for environmental monitoring," in 5^{th} *International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2013, pp. 104–109.

[100] A. Kothari, V. Boddula, L. Ramaswamy, and N. Abolhassani, "DQS-cloud: A data quality-aware autonomic cloud for sensor services," in *International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, Oct 2014, pp. 295–303.

[101] H. Dong, Q. Hao, T. Zhang, and B. Zhang, "Formal Discussion on Relationship between Virtualization and Cloud Computing," in 11^{th} *International Conference on Parallel and Distributed Computing, Applications and Technologies*, 2010.

[102] S. Heitz, P. Matgen, G. Schumann, and L. Pfister, "Active and Passive Microwave Sensors as a Tool to Monitor Soil Moisture Over Winter," in *IEEE International Geoscience and Remote Sensing Symposium*, vol. 2, July 2008, pp. II–773–II–776.

[103] X. Tian and C. Ji, "Bounding the Performance of Dynamic Channel Allocation with QoS Provisioning for Distributed Admission Control in Wireless Networks," *IEEE Transactions on Vehicular Technology*, vol. 50, no. 2, pp. 388–397, March 2001.

[104] S. Bera, S. Misra, and D. Chatterjee, "C2C: Community-Based Cooperative Energy Consumption in Smart Grid," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, pp. 1–1, 2017.

[105] S. Bera, S. Misra, S. K. Roy, and M. S. Obaidat, "Soft-WSN: Software-Defined WSN Management System for IoT Applications," *IEEE Systems Journal*, vol. PP, no. 99, pp. 1–8, 2016.

[106] A. Mondal, S. Misra, and M. S. Obaidat, "Distributed Home Energy Management System With Storage in Smart Grid Using Game Theory," *IEEE Systems Journal*, vol. PP, no. 99, pp. 1–10, 2015.

[107] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, Oct 2004.

[108] Y. C. Liang, Y. Zeng, E. C. Y. Peh, and A. T. Hoang, "Sensing-Throughput Tradeoff for Cognitive Radio Networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 4, pp. 1326–1337, April 2008.

[109] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 493–506, June 2004.

[110] M. D. King, W. P. Menzel, Y. J. Kaufman, D. Tanre, B.-C. Gao, S. Platnick, S. A. Ackerman, L. A. Remer, R. Pincus, and P. A. Hubanks, "Cloud and aerosol properties, precipitable water, and profiles of temperature and water vapor from

MODIS," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 2, pp. 442–458, Feb 2003.

[111] M. D. King, Y. J. Kaufman, W. P. Menzel, and D. Tanre, "Remote sensing of cloud, aerosol, and water vapor properties from the moderate resolution imaging spectrometer (MODIS)," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 30, no. 1, pp. 2–27, Jan 1992.

[112] R. K. Lam, D.-M. Chiu, and J. C. Lui, "On the Access Pricing and Network Scaling Issues of Wireless Mesh Networks," *IEEE Transactions on Computers*, vol. 56, no. 11, pp. 1456–1469, 2007.

[113] W. Liu, L. Cui, and X. Niu, "EasiTPQ: QoS-Based Topology Control in Wireless Sensor Network." *Signal Processing Systems*, vol. 51, no. 2, pp. 173–181, 2008.

[114] Y.-L. Lai and C. J., "A Cloud-Storage RFID Location Tracking System," *IEEE Transactions on Magnetics*, vol. 50, no. 7, pp. 1–4, July 2014.

[115] A. Zhou and B. He, "Transformation-Based Monetary Cost Optimizations for Workflows in the Cloud," *IEEE Transactions on Cloud Computing*, vol. 2, no. 1, pp. 85–98, March 2014.

[116] J. Shin, M. Jo, J. Lee, and D. Lee, "Strategic Management of Cloud Computing Services: Focusing on Consumer Adoption Behavior," *IEEE Transactions on Engineering Management*, vol. PP, pp. 1–9, 2014.

[117] Y. Feng, B. Li, and B. Li, "Price Competition in an Oligopoly Market with Multiple IaaS Cloud Providers," *IEEE Transactions on Computers*, vol. 63, no. 1, pp. 59–73, 2014.

[118] H. Lu, X. Wu, W. Zhang, and J. Liu, "Optimal Pricing of Multi-model Hybrid System for PaaS Cloud Computing," in *International Conference on Cloud and Service Computing (CSC)*, Nov 2012, pp. 227–231.

[119] D. Ardagna, B. Panicucci, and M. Passacantando, "Generalized Nash Equilibria for the Service Provisioning Problem in Cloud Systems," *IEEE Transactions on Services Computing*, vol. 6, no. 4, pp. 429–442, 2013.

[120] J. Musacchio and J. Walrand, "WiFi access point pricing as a dynamic game," *IEEE/ACM Trans. Networking*, vol. 2, pp. 289–301, 2006.

[121] K. Cheung, H. So, W.-K. Ma, and Y. Chan, "Received signal strength based mobile positioning via constrained weighted least squares," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP).*, vol. 5, April 2003.

[122] M. Hossain, P. Atrey, and A. Saddik, "Context-aware QoI computation in multi-sensor systems," in 5^{th} *IEEE Intl Conf on Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008.*, Sept 2008, pp. 736–741.

[123] E. Ciftcioglu, A. Yener, and M. Neely, "Maximizing Quality of Information From Multiple Sensor Devices: The Exploration vs Exploitation Tradeoff," *IEEE Journal on Selected Topics in Signal Processing*, vol. 7, no. 5, pp. 883–894, Oct 2013.

[124] S. Sheng and R. Gao, "Structural dynamics-based sensor placement strategy for high quality sensing," in *Proceedings of IEEE Sensors, 2004.*, Oct 2004, pp. 642–645 vol.2.

[125] D. Fudenberg and J. Tirole, *Game Theory, Chapter 8.* MIT Press, 1991.

[126] R. Wolff, *Stochastic Modelling and the Theory of Queues.* Prentice-Hall, 1989.

[127] X. Yu, "Distributed cache updating for the dynamic source routing protocol," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 609–626, June 2006.

[128] S. Adibi and G. Agnew, "Multilayer flavoured dynamic source routing in mobile ad-hoc networks," *IET Comms*, vol. 2, May 2008.

[129] S. Madria, V. Kumar, and R. Dalvi, "Sensor Cloud: A Cloud of Virtual Sensors," *Software, IEEE*, vol. 31, no. 2, pp. 70–77, Mar 2014.

[130] R. C. Ben-Yashar and S. I. Nitzan, "The Optimal Decision Rule for Fixed-Size Committees in Dichotomous Choice Situations: The General Result," *International Economic Review*, vol. 38, no. 1, pp. pp. 175–186, 1997. [Online]. Available: http://www.jstor.org/stable/2527413

[131] (2012) Cultivating a Crystal Ball for Data Center Availability and Performance. Emerson Network Power. [Online]. Available: http://www.emersonnetworkpower.com/documenta-tion/en-us/solutions/cio-topics/documents/cultivating-a-crystal-ball-for-data-center-playbook.pdf

[132] G. Leopold. (2014, April) Survey Finds Disconnect Between Big Data and Decision-Making. [Online]. Available: http://www.datanami.com/2014/04/02/survey-finds-disconnect-between-big-data-and-decision-making/

[133] E. Baralis, L. Cagliero, and P. Garza, "EnBay: A Novel Pattern-Based Bayesian Classifier," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 12, pp. 2780 – 2795, December 2013.

[134] C. Ordonez and S. K. Pitchaimalai, "Bayesian Classifiers Programmed in SQL," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 1, pp. 139 – 144, January 2010.

[135] B. P. Brownstein, "Pareto Optimality, External Benefits and Public Goods: A Subjectivist Approach," *The Journal of Libertarian Studies*, vol. 4, no. 1, 1980.

[136] Z. He, G. Yen, and J. Zhang, "Fuzzy-Based Pareto Optimality for Many-Objective Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 269–285, April 2014.

[137] C. Borgers, *Mathematics of Social Choice: Voting, Compensation, and Division.* Society for Industrial and Applied Mathematcs, 2010.

[138] X. Zhang, W. Dou, J. Pei, S. Nepal, C. Yang, C. Liu, and J. Chen, "Proximity-Aware Local-Recoding Anonymization with MapReduce for Scalable Big Data Privacy Preservation in Cloud," *IEEE Transactions on Computers*, vol. 64, no. 8, pp. 2293–2307, Aug 2015.

[139] L. Gu, D. Zeng, P. Li, and S. Guo, "Cost Minimization for Big Data Processing in Geo-Distributed Data Centers," *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 3, pp. 314–323, Sept 2014.

[140] W. Yu, Y. Wang, and X. Que, "Design and Evaluation of Network-Levitated Merge for Hadoop Acceleration," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 3, pp. 602–611, March 2014.

[141] D. Bouyssou, T. Marchant, and P. Perny, *Decision-making Process: Concepts and Methods.* Wiley Online Library, Jan 2010.

[142] J. Geanakoplos, "Three Brief Proofs of ARROW'S IMPOSSIBILITY THEOREM," *Cowles Foundation For Research In Economics*, Yale University 2001.

[143] W. L. Yeow, C. K. Tham, and W. C. L. Wong, "Energy Efficient Multiple Target Tracking in Wireless Sensor Networks," *IEEE Transactions on Vehicular Technology*, vol. 56, pp. 918 – 928, 2007.

[144] 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*, vol. 12, no. 4, pp. 735–747, Apr 2013.

Publications out of this work

Patents

- **S. Chatterjee**, A. Roy, S. K. Roy, S. Misra, M. S. Bhogal, and R. Daga, "Sensory network for persuasive and pervasive virtualization of physical sensors into renderable time service", Indian patent filed in November 2014 (Ref: 1145/KOL/2014).

Journal

- **S. Chatterjee**, S. Misra, and S. U. Khan, "Optimal Data Center Scheduling for Quality of Service Management in Sensor-cloud," *IEEE Transactions on Cloud Computing*, 2015.
- **S. Chatterjee**, R. Ladia, and S. Misra, "A Dynamic Optimal Pricing Scheme for Heterogeneous Service-Oriented Architecture for Sensor-cloud Infrastructure," *IEEE Transactions on Services Computing*, 2015.
- S. Misra, A. Singh, **S. Chatterjee**, and A. K. Mandal, "QoS-Aware Sensor Allocation for Target Tracking in Sensor-Cloud," *Ad Hoc Networks*, Elsevier, 2015.
- **S. Chatterjee**, and S. Misra, "Sensor-Target Mapping in Presence of Overlapping Coverage: Tracking Using Sensor-Cloud," *IEEE Communication Letters*, 2014.
- S. Misra, **S. Chatterjee**, and M. S. Obaidat, "On Theoretical Modeling of Sensor-Cloud: A Paradigm Shift From Wireless Sensor Network," *IEEE Systems Journal*, 2014.

Conference

- **S. Chatterjee**, and S. Misra, "Adaptive Data Caching for Provisioning Sensors-As-A-Service," *IEEE BlackSeaCom*, 2016.
- **S. Chatterjee**, and S. Misra, "QoS Estimation and Selection of CSP in Oligopoly Environment for Internet of Things," *IEEE WCNC*, 2016.
- **S. Chatterjee**, and S. Misra, "Optimal Composition of a Virtual Sensor for Efficient Virtualization Within Sensor-cloud," *IEEE ICC*, 2015.

- **S. Chatterjee**, S. Sarkar, and S. Misra, "Energy-Efficient Data Transmission in Sensor-Cloud," International Conference on Applications and Innovations in Mobile Computing (AIMoC), 2015.
- **S. Chatterjee**, and S. Misra, "Dynamic and Adaptive Data Caching Mechanism for Virtualization within Sensor-Cloud," *IEEE ANTS*, 2014.

BIO-DATA

1. **Personal Details**

   - *Name*: Subarna Chatterjee
   - *Roll No.*: 13IT91P01
   - *Father's Name*: Mr. Subhash Chatterjee
   - *Date of Birth*: February 27, 1990
   - *Permanent Address*: S-13, Cluster - III, Purbachal, Salt Lake, Kolkata - 700097, West Bengal, India

2. **Academic Qualification**:

   - *Jul. 2013 – Present*: Ph.D. Research Scholar at Indian Institute of Technology Kharagpur, India.
   - *Aug. 2008 – Jul. 2012*: B. Tech. in Computer Science and Engineering from Institute of Engineering and Management, Kolkata, India

3. **Research Experience**:

   - *Dec. 2015 – Present*: Tata Consultancy Services Research Scholar, Indian Institute of Technology Kharagpur, India.
   - *Apr. 2013 – Dec. 2014*: Junior Research Fellow (JRF), Indian Institute of Technology Kharagpur, India.

4. **Patents**:

   (a) **S. Chatterjee**, A. Roy, S. K. Roy, S. Misra, M. S. Bhogal, and R. Daga, "Sensory network for persuasive and pervasive virtualization of physical sensors into renderable time service", Indian patent filed in November 2014 (Ref: 1145/KOL/2014).

5. **Journal Publications**:

(a) S. Sarkar, **S. Chatterjee**, and S. Misra, "Analysis of Fog Computing: The Convergence of Cloud Computing to Green Computing," *IEEE Transactions on Cloud Computing*, 2015.

(b) **S. Chatterjee**, S. Misra, and S. U. Khan, "Optimal Data Center Scheduling for Quality of Service Management in Sensor-cloud," *IEEE Transactions on Cloud Computing*, 2015.

(c) **S. Chatterjee**, R. Ladia, and S. Misra, "A Dynamic Optimal Pricing Scheme for Heterogeneous Service-Oriented Architecture for Sensor-cloud Infrastructure," *IEEE Transactions on Services Computing*, 2015.

(d) S. Misra, A. Singh, **S. Chatterjee**, and A. K. Mandal, "QoS-Aware Sensor Allocation for Target Tracking in Sensor-Cloud," *Ad Hoc Networks*, Elsevier, 2015.

(e) S. Sarkar, **S. Chatterjee**, and S. Misra, "Evacuation and Emergency Management Using a Federated Cloud," *IEEE Cloud Computing Magazine*, 2015.

(f) S. Misra, and **S. Chatterjee**, "Social Choice Considerations in Cloud-Assisted WBAN Architecture for Post-Disaster Healthcare: Data Aggregation and Channelization," *Information Sciences, Elsevier*, 2014.

(g) **S. Chatterjee**, and S. Misra, "Sensor-Target Mapping in Presence of Overlapping Coverage: Tracking Using Sensor-Cloud," *IEEE Communication Letters*, 2014.

(h) S. Misra, A. Singh, **S. Chatterjee**, and M. S. Obaidat, "Mils-Cloud: A Sensor-Cloud Based Architecture for the Integration of Military Tri-Services Operations and Decision Making," *IEEE Systems Journal*, 2014.

(i) S. Misra, **S. Chatterjee**, and M. S. Obaidat, "On Theoretical Modeling of Sensor-Cloud: A Paradigm Shift From Wireless Sensor Network," *IEEE Systems Journal*, 2014.

6. **Conference Publications**:

(a) **S. Chatterjee**, and S. Misra, "Adaptive Data Caching for Provisioning Sensors-As-A-Service," *IEEE BlackSeaCom*, Varna, Bulgaria, 2016.

(b) **S. Chatterjee**, and S. Misra, "QoS Estimation and Selection of CSP in Oligopoly Environment for Internet of Things," *IEEE WCNC*, Doha, Qatar, 2016.

(c) **S. Chatterjee**, and S. Misra, "Optimal Composition of a Virtual Sensor for Efficient Virtualization Within Sensor-cloud," *IEEE ICC*, London, U.K., 2015.

(d) **S. Chatterjee**, and S. Misra, "Quantification of Node Misbehavior in Wireless Sensor Networks: A Social Choice-Based Approach," *IEEE ICC Workshop*, London, U.K., 2015.

(e) **S. Chatterjee**, S. Sarkar, and S. Misra, "Energy-Efficient Data Transmission in Sensor-Cloud," International Conference on Applications and Innovations in Mobile Computing (AIMoC), Kolkata, India, 2015.

(f) P. V. S. R. Teja, **S. Chatterjee**, S. N. Das, and S. Misra, "Two-Level Mapping to Mitigate Congestion in Machine to Machine (M2M) Cloud," International Conference on Applications and Innovations in Mobile Computing (AIMoC), Kolkata, India, 2015.

(g) **S. Chatterjee**, and S. Misra, "Dynamic and Adaptive Data Caching Mechanism for Virtualization within Sensor-Cloud," *IEEE ANTS*, New Delhi, India, 2014.

7. **Fellowships and Scholarships**:

(a) Awarded the **Facebook Grace Hopper Scholarship** (*one of the* 50 *recipients worldwide*), 2016

(b) Awarded the **N2 Women Young Researcher Fellowship** through **ACM SIGMOBILE** program to attend IEEE WCNC, 2016

(c) Awarded the **Student Scholarship** to attend the **Grace Hopper Celebration of Women in Computing India (GHCI), 2015** conference

(d) Awarded **Google Anita Borg Fellowship, Asia Pacific**, (2015).

(e) Awarded **Tata Consultancy Service (TCS) Research Fellowship** (2014-2017).

(f) Awarded **Fellowship** from **ISIRD, IIT Kharagpur** (2013-2014).

(g) Awarded **Merit scholarship** by **Government of India** for exemplary performance in Higher Secondary Examination.

(h) Awarded **Merit Scholarship** from **South Eastern Railway** for exemplary performance in Secondary and Higher Secondary examinations.

(i) Nominated for **Merit scholarship** by **Government of India Ministry of Human Resource Development Department** of Higher Education for exemplary performance in Secondary Examination.

8. **Awards and Certifications**:

(a) Selected as one of the **Most Qualified Young Scientist** to attend the **Heidelberg Laureate Forum, 2016**.

(b) Selected as the **Leader of the Google Anita Borg Scholarship Community** (*solitary from the country and one of the seven across the world*) and invited to **attend Google I/O 2016**.

(c) Awarded as the **Winner** in **Hackathon in Robotics** at **Google Shanghai**, 2015.

(d) Awarded a **fully-sponsored Travel Grant** by **Ministry of Human Resource** (**MHRD**) for presenting in a **listed conference - IEEE ICC**, 2015.

(e) Awarded certificate of appreciation at **ComSoc Student Competition** "Communications Technology Changing the World", 2014 for being ranked among the **top 9** projects.

(f) Awarded **Second Runners Up** in **Samsung Innovation Award**, 2014.

(g) Awarded **certificate of Merit** and **certificate of participation** from **West Bengal Renewable Energy Development Agency** (**WBREDA**) in association with **Vivekananda Institute of Environment and Management**, Kolkata for being **Second Runners Up** in a model making competition.

(h) Awarded **Merit Award** from **United Bank of India Employees' Co-operative Credit Society Ltd** for unparallel performance in Secondary and Higher Secondary Examination.

9. **Professional Affiliations**

- Graduate student member, IEEE
- Graduate student member, ComSoc
- Graduate student member, IEEE Women in Engineering