Title
Addressing Privacy, Fairness, and Scalability Challenges for Context-Aware Applications in Smart Environments

Author
Shin, Eun-Jeong

Publication Date
2019

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA,
IRVINE

Addressing Privacy, Fairness, and Scalability Challenges for Context-Aware Applications in Smart Environments

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Eun-Jeong Shin

Dissertation Committee:
Professor Sharad Mehrota, Chair
Professor Nalini Venkatasubramanian
Professor Alfred Kobsa
Professor Ramesh Jain

2019

DEDICATION

To the scientists who inspired me to stay passionately curious

"Imagination is the highest form of research"
- Albert Einstein

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I am deeply grateful for my advisor Professor Sharad Mehrotra for his continuous support during my five years Ph.D. He has been supportive in my research and I could ask for advice whenever possible. His vision led me to work on interesting problems such as fairness, scalability, and user-privacy.

I would also like to thank Professor Nalini Venkatasubramanian, Professor Alfred Kobsa, and Professor Ramesh Jain for joining my doctoral committee. These professor's vision and care inspired me to think of how important it is having a mentor during Ph.D. I would like to add special thanks to Professor Kyu-Young Whang, who is my advisor at Korea Advanced Institute of Science and Technology. I am lucky to have him as my Master's advisor. He introduced me to fundamentals of data management systems and taught me how to become a serious system builder.

I am grateful to Dr. Roberto Yus for his invaluable suggestions and discussions during my research. His idea directly inspired the work in Chapter 5. I would like to thank Dr. Shantanu Sharma for proof-reading the draft version of the thesis and providing invaluable suggestions.

I would like to thank Dhrubajyoti Ghosh for working together to design and implement context-aware messaging system and many other collaborative works. His diligence and capability to articulate and communicate problems influenced me as a stronger researcher.

I would like to thank all other TIPPERS team members and alumnus: Daokun Jiang, Dhrubajyoti Ghosh, Georgios Bouloukakis, Guoxi Wang, Peeyush Gupta, Primal Pappachan, Roberto Yus, Sameera Ghayyur, Sumaya Almanee, and Yiming Lin for working hard together to make system stable and reliable. I would also like to thank the Honeywell team including Dr. Phillip Lee, Dr. Valerie Guralnik, and Dr. Raj Rajagopalan to provide me valuable feedback throughout the internship and led me to a more enlighted career path.

Lastly, but most importantly, I would like to thank my husband Rex for understanding through my Ph.D. journey. Endless nights spent in the lab, I could not imagine how I manage Ph.D. without his support. I am grateful for my daughter, Sophie, to be the greatest joy of my life. My parents Myung-Ock Park and Suk-Hyun Shin, have been there for me and have encouraged me throughout the good and bad times. They have been my motivation and inspiration to move forward and reach my full potential. I owe my sincerest appreciation to sister Hyung-Jeong Shin and brother Hee-Won Shin for endless help. My parents-in-law Jung-Ihn Lee and Jai-Myung Yang, have put their effort staying with us as a family and their support made everything easier.

I would like to acknowledge DARPA to support my research under agreement number FA8750-16-2-0021. Information and Computer Science department had been generously supported my academic years – I would like to thank the school for giving me an opportunity to grow as a scholar.

The text of this dissertation is a reprint of the material as it appears in "Exploring Privacy Breaches and Mitigation Strategies of Occupancy Sensors in Smart Buildings", "Exploring Fairness in Participatory Thermal Comfort Control in Smart Buildings", and "SCARF: A Scalable Data Management Framework for Context-Aware Applications in Smart Environments". The co-authors listed in these publications directed and supervised research which forms the basis for the dissertation.

CURRICULUM VITAE

Eun-Jeong Shin

EDUCATION

Doctor of Philosophy in Computer Science **2019**
University of California, Irvine *Irvine, California*

Master of Science in Computer Sciences **2007**
Korea Advanced Institute of Science and Technology *Daejeon, Korea*

Bachelor of Science in Computer Sciences **2005**
Ewha Womens University *Seoul, Korea*

RESEARCH EXPERIENCE

Graduate Student Researcher **2014–2019**
University of California, Irvine *Irvine, California*

Research Intern **Summer 2017**
Honeywell Laboratory *Golden Valley, Minnesota*

Senior Research Engineer **2007–2014**
LG Electronics, CTO *Seoul, Korea*

Graduate Student Researcher **2005–2006**
Korea Advanced Institute of Science and Technology *Daejeon, Korea*

SOFTWARE ENGINEERING EXPERIENCE

Software Engineering Intern **2018**
Obsidian Security *Newport Beach, California*

LEADERSHIP ROLES

Deputy Leader, Junior Board (JB) **2012**
LG Electronics, CTO *Seoul, Korea*

Board Member, Junior Board (JB) **2011–2012**
LG Electronics, CTO *Seoul, Korea*

TEACHING EXPERIENCE

Teaching Assistant **2014-2019**
University of California, Irvine *Irvine, California*

Next Generation Search Systems *Winter 2019*

Introduction to Data Management *Fall 2018*

Introduction to Data Management *Fall 2017*

Principles of Data Management *Fall 2016*

Programming in Java as a Second Language *Fall 2015*

Introduction to Data Management *Spring 2015*

Projects in Databases and Web Applications *Winter 2015*

Intermediate Programming *Fall 2014*

REFEREED JOURNAL PUBLICATIONS

Implementation of a Parallel Web Crawler for the Odysseus Large-Scale Search Engine **2008**
Journal of KIISE: Computing Practices and Letters 14 (6), 567-581

REFEREED CONFERENCE PUBLICATIONS

SCARF: A Scalable Data Management Framework for Context-Aware Applications in Smart Environments **Nov 2019**
Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous)

Exploring Privacy Breaches and Mitigation Strategies of Occupancy Sensors in Smart Buildings **Nov 2019**
The 1st ACM International Workshop on TechnologyEnablers and Innovative Applications for Smart Cities and Communities (TESCA)

SemIoTic: Bridging the Semantic Gap in IoT Spaces **Nov 2019**
Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Built Environments (BuildSys)

Exploring Fairness in Participatory Thermal Comfort Control in Smart Buildings **Nov 2017**
Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environments (BuildSys)

Message of Interest: A Framework of Location-Aware Messaging for an Indoor Environment **Mar 2016**
IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)

ABC3D: Using an Augmented Reality Mobile Game to Enhance Literacy in Early Childhood **Mar 2016**
IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)

SOFTWARE

TIP `https://github.com/ucisharadlab/tippers-main`
A Framework to Provision and Deploy Interoperable IoT Applications.

ABSTRACT OF THE DISSERTATION

Addressing Privacy, Fairness, and Scalability Challenges for Context-Aware Applications in Smart Environments

By

Eun-Jeong Shin

Doctor of Philosophy in Computer Science

University of California, Irvine, 2019

Professor Sharad Mehrota, Chair

Various in-situ and mobile sensors are deployed in the smart environments for the purpose of providing personalized services to the end users. Large amount of data is being collected, stored and analyzed from a variety of sensors in a real-time manner. Such data introduces several challenges to the end users. These challenges include privacy and security of the users and system development challenges such as scalability, data management, energy efficiency, and data analytics.

The functionality of a smart environment depends heavily on the context. For example, the HVAC (Heating, Ventilation, and Air Conditioning) system of a building takes into account the occupancy of various regions of the building and determines the appropriate temperature of the regions. Such decision requires determination of accurate contexts such as occupancy, air pressure, outside air temperature, and water pressure. However, the state of a motion detector used to derive occupancy can reveal user-privacy, due to revealing the absence or presence of a user at a given space. Similarly, participatory thermal comfort control systems take users thermal comfort votes into account to control the temperature of a building. These systems require fairness in decision making. For example, if the majority of the votes for their comfort level is good even though one person is uncomfortable, the IoT-enabled building

management systems will keep ignoring the minors opinion. When applying decision-making methods to aggregate user opinions that is collected through sensor data, the result can be unfair to certain groups of users. Likewise, context-aware messages can benefit from taking user and environmental context into account to deliver messages in real time/near real time manner. Such a messaging application requires scalable context collection and evaluation of context predicates. However, enrichment techniques which convert low-level sensor readings into meaningful context cannot be applied to the context data on the resource-constrained mobile devices due to the high resource requirements of these algorithms.

We identify aforementioned context reasoning challenges within two different IoT testbeds: first, TIPPERS (Testbed for IoT-based Privacy-preserving PERvasive Spaces) is an experimental six-story smart building testbed in UC Irvine designed to study the numerous privacy challenges due to fine-grained monitoring of building occupants and visitors using a diverse set of sensors. Second, Honeywell testbed consists of total 7687 *Enlighted* sensors which are capable of recording occupancy, light, power, and temperature.

This thesis is organized as follows. First, we study occupancy sensors and propose a privacy attack in which the adversary associates an individual with the occupancy sensor by combining the occupancy data with other public information that could easily be obtained online, and that breaches the user-privacy. Second, we present the first study, up to the authors' knowledge, on fairness in the aggregation of user thermal comfort in Participatory Thermal Comfort Control (PCC) systems. Third, we propose a context-aware messaging framework, SCARF that empowers senders and receivers to control message delivery through policies defined over context collected from a variety of mobile and in-situ sensors.

Chapter 1

Introduction

1.1 IoT Environment

We are living in the era of IoT where a large number of heterogeneous sensors (e.g. mobile phones, smart light sensors, smart meters, smart vehicles, and office equipment) are connected to the Internet to provide ubiquitous services. IoT is not only a technical revolution that influences our daily lives, but also enabled physical space to be transformed into smart spaces. Examples of such space are smart buildings where it can be further categorized into smart hospital, community, and shopping malls. Consider smart university example where student and employee can be identified through face recognition without a burden to carry physical IDs. Tracking of the last location of person and equipment is also feasible through location tracking system built upon sensor data. Smart attendance maintenance and equipment tracking applications transformed cumbersome manual jobs into automated task.

Sensors play a critical role for capturing the state of the environment accurately. Capturing sensor data is an initial step for smart applications to perform actions and/or expose situation-aware services within the physical space. There are two types of sensors in IoT environment: in-situ sensors and mobile sensors. In-situ sensors are used to create a dynamic representation of the state of buildings, by collecting data that is used to actuate or control building management systems. HVAC sensors are one of the biggest component of in-situ sensors. For example, thermostat is capable of monitoring the temperature of the rooms. Occupancy sensors can measure motion changes. Indoor air quality measurement sensors are becoming increasingly important as it detects carbon levels in the air. WiFi access points can capture the device connected to a certain location inside the buildings. As smartphones become integral part of residents, building management systems have better opportunity to utilize smartphone sensors including accelerometer, gyroscope, magnetometer, GPS, proximity, microphone, touchscreen, fingerprint, and light sensors.

The functionality of a smart environment depends heavily on the context. For example, HVAC system of a building depends on an environmental context such as occupancy to determine the temperature of the various regions of building. However, such decision can lead to the loss of *user-privacy*. An adversary can efficiently identify a set of sensors that corresponds to the given interest (with a relatively small amount of additional information). Furthermore, several inference attacks are feasible by figuring out potentially sensitive information including the daily routine of the users: commute patterns, work habits, break patterns, and smoking habits. Similarly, to control the environmental factors (e.g., room temperature, air, and light), participatory decision making process takes users thermal comfort into account. While deciding group thermal comfort to actuate temperature changes, these systems can be improved by considering *fairness*. Finally, context-aware applications play an important role in these environments. These applications require real-time/near real time response times. Collection and evaluation of context data in a *scalable* manner to support such context-aware applications is a very relevant direction of research.

We address three situations where context influence decision making in smart environments, specifically, privacy, fairness, and scalability. First, we study occupancy sensors and propose a privacy attack in which the state of a motion sensor can reveal user-privacy, due to revealing the presence of a user at a given space. Second, we present fair decision making methods to aggregate user opinions that is collected through sensor data. Third, We propose *SCARF* to support scalable data collection and evaluation for smart environments. We focused particularly on scalability challenges with context-aware messaging applications which we still see the improvement can be made in terms of delivery considering both senders and receivers context. To accomplish this goal, we designed a context data collection and management system called *SCARF* .

1.2 Context-Awareness

Context consists of a variety of observable phenomena including time, location, activity (e.g. driving/walking/running/still), sensor state (e.g. active, inactive), sensor value (e.g. occupancy, temperature, air quality, water pressure), and user preference on surroundings (e.g. thermal comfort level, lighting preference). Context information is generated by processing raw sensor data. Further, it is checked for consistency and metadata is added [87]. Collected sensor data can be enriched or processed further to be used for smart applications. We consider two types of context in this thesis:

- **User Context.** User context consists of the properties of users inside a building such as the location, activity, or preference on surroundings of the user between two timestamps.
- **Environmental Context.** Environmental context consists of a number of environnemtal properties such as the temperature, occupancy, and water pressure between two timestamps.

Context-awareness in the messaging domain has also been explored [52, 19, 20]. For instance, the authors in [52] considered adapting the message body based on the recipient's context (e.g., language of user, interest of users, devices used by the user, etc.) in a distributed pub-sub system. Recently, Gmail has introduced a new feature [2] that empowers senders a control over the message translation by providing a temporal context.

Consider the following two different architectures to build context-aware applications:

- **Server-based Centralized Architecture** Once context data is collected from mobile phones, it will be transmitted to server. Transmitted sensor data is stored and evaluated on server side.

- **Edge-based Architecture** Context data acquisition, context data storage, and context evaluation take place in edge (mobile phones).

In this thesis, we leverage two architectures: a server-based centralized architecture and an edge-based architecture to address the problem of scalability of context collection and evaluation. SCARF collects the context data efficiently from resource constrained mobile/edge and multiple in-situ sensors (not resource constrained) deployed in smart environments. In particular, SCARF takes a set of contextual queries from applications and optimizes the rate at which the context data needs to be collected and/or transmitted from the sensor to the server. Furthermore, SCARF decides whether the code for evaluating context should be executed on the server or on the edge. Execution on the edge will result in reduced communication between the server and the edge but can result in low performance due to resource constraints of the edge. On the other hand, evaluation of context on the server, will require SCARF to transmit sensor data from the sensor to the server but can result in high performance due to high resource availability of the server. SCARF is an adaptive framework which generates an optimal context acquisition plan and a context evaluation plan such that the overall latency is minimized.

1.3 Motivation: Context Reasoning Challenges in IoT Environment

1.3.1 Privacy Challenge

There is an increasing legislative support for user privacy (e.g. the European general Data Protection Regulation - GDPR - and California Consumer Privacy Act - CCPA -) that requires applications to deal with user-defined policies. General Data Protection Regulation (GDPR) was approved by European Union (EU) and it is enforced starting on 25th May 2018. The aim of the GDPR[1] is to protect all EU citizens from privacy and data breaches in an increasingly data-driven world. GDPR applies to the processing of personal data by controllers and processors in the EUs, regardless of whether the processing takes place in the EU or not. In a smart building context, the new concept of 'privacy-by-design' requires organizations to consider how they ensure compliance with GDPR.

In Chapter 4, we present exploratory study on privacy breaches and mitigation strategies of occupancy sensors in smart buildings. Occupancy sensors are an integral part of many smart building applications, including energy and space optimizations. However, as more occupancy data is collected at an increasingly fine grained level, possibilities of potential privacy breaches also increase.

We show an attack in which the adversary associates an individual with the occupancy sensor by combining the occupancy data with other public information that could easily be obtained online. We conduct an experiment using real-world data to demonstrate that the proposed attack is not only feasible, but the adversary can efficiently identify a small set of sensor IDs that contains the ID of interest given a relatively small amount of additional information. We also summarize a number of mitigation strategies against the proposed attack.

[1] `https://www.eugdpr.org/eugdpr.org.html`

1.3.2 Fairness Challenge

As IoT-enabled smart buildings made engagement with building residents easier, many approaches have been presented to address the challenges of capturing the user thermal comfort and preference and implementing actuation into the HVAC system of the smart building. To increase occupants comfort and convenience, the research community as well as industry have been actively working on the issue of participatory thermal comfort in the last years. For examples, Honeywell introduced Vector[2] Occupant app that allows users to submit occupant temperature feedback to optimize comfort settings. Comfy[3] is another workspace application enables personalization of space by offering temperature controls.

When buildings do not have access to residents feedback, actuation system relies on predefined rules to control the temperature in rooms disregarding their residents' thermal comfort. By enabling occupants to express their feedback using their mobile devices, actuation based on user comfort level became feasible. In general setting, to reach consensus among group members, we apply decision making methods (e.g., majority, mean, trimmed mean, and median) to aggregate the diverse expected comfort levels of the inhabitants. However, such methods might be *unfair* to some participants.

Chapter 5 presents the first study of the issue of fairness in participatory thermal comfort in smart buildings. Inspired by the traditional definitions in scheduling, we introduce a definition of fairness that is suitably adapted to the particularities of our scenario. We then present our design of an aggregation method that ensures fairness. Finally, we show how our algorithm behaves compared with traditional aggregation methods in diverse simulated scenarios.

[2] `https://buildingsolutions.honeywell.com/en-US/solutions/Vector/Pages/default.aspx`
[3] `https://www.comfyapp.com/the-app/`

1.3.3 Scalable Context Collection and Evaluation Challenge

We aim for a holistic framework which allows context collection and evaluation in scalable manner. Clearly, a truly holistic solution to the challenge of context collection and evaluation must address following issues:

- **Scalability.** Scalability is an important issue while supporting large number of context-aware applications. Several architectures have been proposed in the past to address the scalability issues such as processing speed-up, reduced communication bandwidth, and increased reliability [6]. Among different programming paradigms of context-aware systems, service-oriented and agent-oriented [72] programming model have addressed the scalability issues up to some extent but they do not consider the scalability of context collection and evaluation from a large number of heterogeneous sources.

- **Latency.** The context-aware applications need to provide real time/near real time responses to the end users. In this scenario, latency becomes a significant issue while detecting/extracting contextual features from collected context data [80, 76, 75].

- **Energy Consumption.** Most of the devices in smart environments are battery powered. Energy consumption becomes a critical issue as context-aware applications consume large amount of energy from user's devices. Several authors [93, 108, 28, 94] have pointed out this issue for context-aware applications.

- **Security.** Some of the major security concerns of context-aware applications include context data access by malicious users, gaining access to user's mobile sensors by attacker and inferring sensitive user information from the output of these applications. Systems that support context-aware applications need to provide security guarantees against these types of attackers.

- **Privacy.** Context-aware applications can disclose behavior and social phenomena such as a user's location, their friends, associates, the timing of their interactions which can lead to serious privacy concerns [60].

However, we leave these other challenges out of scope for this thesis and maintain a narrow focus on the scalable framework for context data collection and evaluation. The approach outlined below can be leveraged as one solution to the plethora of IoT context data management challenges. Such a framework should perform under a variety of challenging conditions (e.g. context evaluation on limited resource environment, effective communication in emergency response) by achieving minimum latency and ensuring bounded amount of energy consumption.

1.4 Thesis Contributions and Organization

This thesis aims to address some of the above challenges through scalable context collection and evaluation framework.

The following is the overall organization and research contributions of the thesis:

- Chapter 2 surveys related work.
- Chapter 3 introduces two IoT testbeds where the experimental evaluations took place.
- Chapter 4 introduces privacy challenge in reasoning occupancy in IoT environment. We study privacy breaches and mitigation strategies of occupancy sensors.
- Chapter 5 introduces fairness challenge in determining group thermal comfort in IoT environment. We study an opportunity for fairness when applying decision making methods to aggregate user opinions in participatory thermal comfort system.
- Chapter 6 introduces scalable context collection and evaluation challenge and proposes a framework named SCARF as a solution.

- Chapter 7 concludes the dissertation with lessons learned, a holistic view of the challenges, and a future research directions of context reasoning challenges.

Chapter 2

Related Work

We now survey relevant work to provide an appropriate background for this thesis. We start with an overview of context-aware services related to IoT environment. We then explore systems for context-aware applications and provide a brief overview of existing systems by surveying existing systems. Lastly, we review the work in terms of energy-efficient processing. These emerging technologies and concepts can help enable scalable context collection and evaluation solutions in IoT environment, and so we build upon these in designing our proposed framework.

2.1 Context-Awareness in IoT Environment

Several survey papers [42, 71, 27, 92] have discussed different aspects of IoT applications. In [71], the authors categorize the IoT applications into three groups: (1) monitoring and control (where the smart home is known to be at the forefront of innovation regarding IoT monitoring and control systems); (2) big data and business analytics (to discover changes in customer behaviors and market conditions, to increase customer satisfaction, and to provide value-added services to customers); (3) information sharing and collaboration. Context-aware applications have also been studied by many authors in the past [82, 29, 9, 65]. Yurur *et al.* presented a detailed survey of context-aware applications in mobile sensing environments [115]. The authors introduced a model which captures the essential components of context-aware applications. The study topic varies from interface design [15], context modeling [23, 11] to system design [4].

The context data adds more meaningful and value to the sensor data [117, 22]. Ramesh and Pinaki [54] pointed out that content without context is meaningless. Context data which were popularly used by context-aware applications were location, activity, timestamps, schedule and a combination of them. As location became a prominent condition while delivering messages, location-based applications became a norm. Location-based messaging was introduced in the recent applications such as Snapchat [50], Drop Message [49], Yikyak [114], and Notificare [1]. However, these applications either consider a limited number of contexts or it relies on the context data from a limited number of mobile sensors located in the user's mobile device.

2.2 Context Reasoning Challenges in IoT Environment

Context-aware layered architecture is represented in [14]. In this layered architecture, the pre-processing layer is responsible for reasoning and interpreting contextual information. The sensors queried in the underlying layer most often return technical data that are not appropriate for end applications to use by application developers. Context reasoners abstract higher level concepts from raw sensor readings. Context provisioning middleware [64] has been studied by Knappmeyer *et al.* This paper addresses an issue of context reasoning as the main criteria required for the middleware.

2.2.1 Occupancy Reasoning

Accurate occupancy detection has been an active area of research both in academia as well as in industry [5, 79, 59, 62, 86]. The use-case of such research has been focused into two main categories such as, reducing energy footprint of the buildings [5] and optimizing the space utilization of the users inside the building [62]. As the use of occupancy data is being widely implemented, identifying privacy leakage and developing potential mitigation strategies against the potential privacy breaches have been gaining increasing attention [59, 109].

In [59], an optimization framework was proposed where the goal is to minimize the information regarding the location traces of individuals while guaranteeing performance of the HVAC system. However, the chapter starts with the assumption that the mapping between the locations and individuals are already available to the adversary.

Under certain circumstances, smart buildings equipped with Wi-Fi APs are capable of capturing Wi-Fi Mac addresses and converting them to occupancy has been explored. For examples, TIPPERS [79] is such an example which captures individual's location based on Wi-Fi Mac addresses. In this approach, there exists a sensor-to-individual, which is an one to one, mapping. However, for most commercial buildings, it is often not the case that unique mapping between sensor and individual exists.

A differentially private mechanism for streaming data was studied in [25], and the proposed mechanism was implemented on the building occupancy data in [40]. In this work, differential privacy ensures that an occupancy pattern in a particular zone is statistically similar whether an individual is in the zone or not. In this chapter, we study the problem of linking an individual and the sensor ID at each presence sensor level.

2.2.2 Participatory Thermal Comfort Reasoning

There have been multiple contributions in the literature towards participatory comfort control in buildings [68, 57, 44, 33]. However, up to the authors' knowledge, we are presenting the first approach that takes into account fairness in this context. In the following, we review works in the area of participatory thermal comfort control. In particular, we describe approaches to obtain user thermal comfort preferences and to implement such preferences into the HVAC system. Also, we review works considering fairness in other aspects of computer science.

Works on Participatory Comfort Control

Thermal comfort is defined by American Society of Heating, Refrigerating and Air-conditioning Engineers (ASHRAE) Standard 552013 as the condition of mind that expresses satisfaction with the thermal environment. The *Predicted Mean Vote (PMV)* scale is a common mechanism to measure how comfortable a person is with room temperature [36]. The PMV scale consist of seven points of thermal sensation varying from cold (-3) to hot (3), with other points on the scale corresponding to *cool, slightly cool, neutral, slightly warm* and *warm.* Predicted Percentage of Dissatisfied (PPD) predicts the percentage of occupants that will be dissatisfied with thermal conditions. PPD is a function of PMV, given that as PMV moves further from neutral (0), PPD increases.

There are two main issues to be addressed in order to provide participatory thermal comfort: 1) Capture and aggregate user thermal preferences, and 2) Implement appropriate thermal comfort into the HVAC system. Several approaches for PCC systems that take into account both issues have been presented in the literature [55, 13, 69, 110]. In [3], authors presented the survey results of occupants satisfaction with indoor environmental quality including satisfaction with the temperature, air quality, lighting, and acoustics in buildings.

Industry is also paying attention to the development of PCC systems. For instance, Comfy[1] and Honeywell's Vector[2] systems collect user thermal feedback which is communicated to the building administrator. However, all these systems use traditional decision making methods (i.e., majority, mean, trimmed mean, and median) to aggregate user thermal preferences which, as we will demonstrate in Section 5.3.2, are not fair to the participants.

Some works have focused on obtaining the user thermal comfort and could serve as input for our aggregation algorithm. In [68], the authors developed a Temperature Comfort Correlation (TCC) model that builds a profile for each occupant based on three parameters: 1) personal information such as age, gender, height, weight, etc, 2) indoor and outdoor temperatures, 3) trainable parameters such as users votes. Thermal comfort is a function of heat gain and heat loss, which are primarily settled by metabolic rates and parameters are plugged into this function. In [95], the authors introduced a framework for building occupants serving as participatory sensors wherein occupants provide their feedback of thermal comfort and dynamic temperature control is applied [34, 13, 56, 110, 98] based on the feedback.With increasing levels of instrumentation in buildings, e.g., audiovisual sensing, methods to extract thermal comfort of occupants using visual signals and features along with specialized image processing have been studied [58].

The aggregated thermal comfort of the different users in a zone computed by our algorithm can be translated into HVAC control. Different strategies have been proposed in the literature for such a task. In [34], authors proposed a real-time and learned strategy for temperature control. [32] proposed an HVAC control strategy based on occupancy prediction and real time occupancy monitoring via a sensor network of cameras. In [37], signals measured directly on the body were used to infer user comfort and control the air-conditioning system to direct air flow where it was needed. Techniques to estimate occupancy in buildings via sensors in conventional building management systems have been shown to be effective for

[1] `https://www.comfyapp.com/the-app/`

[2] `https://buildingsolutions.honeywell.com/en-US/solutions/Vector/Pages/default.aspx`

zone-based HVAC scheduling and managing energy usage effectively while ensuring thermal comfort [12].

Fairness in Scheduling

The notion of fairness in scheduling has been addressed along multiple dimensions and has addressed a range of trade-offs - including fair share scheduling in OS, delay scheduling for clustered systems [111, 116], wireless networks [24, 46], shared memory architectures [81, 48, 47], and cloud computing [41, 90]. We focus on the treatment of fairness in scheduling algorithms, in particular in the context of a carpooling problem, which is related to our scenario.

The carpooling problem supposes that N people have decided to form a carpool to commute to office and a decision has to be made regarding who has to drive on a given day. Being selected as a driver is a burden and thus, the driver role has to be shared by the carpool members. In this context, to be fair, each person should be driving approximately $1/k$ of the time that she rides with $k-1$ others. For example, if the carpool consists of A, B, and C, then A might be expected to drive $1/2$ of the times that she rides with only B or only C and $1/3$ of the times that she rides with both B and C.

Fagin and Williams [35] generalized this definition. Assume that at time t, A has participated in the carpool on b_2 days when exactly 2 persons participated in the carpool, on b_3 when exactly 3 persons participated, and so on. They defined *A's ideal number of drives* as the number $(1/2)b_2 + (1/3)b_3 + ... + (1/N)b_N$. Then, their formal definition of fairness considers a carpool scheduling algorithm as *fair* if *"for each N (where N is the number of member of the carpool), there is a number P such that whatever the schedule of arrivals, it is the case that at each time t and for each carpool member A, the number of times that A has actually driven differs from his ideal number of drives in absolute value by no more than P"*.

In Fagin and Williams scheduling algorithm, U is considered to be a value that represents the total cost of a trip. U is a least common multiple of $1, 2, ..., m$ where m is the largest number of people who ever ride together at a time in the carpool. The ledger maintained by the algorithm contain one row for the date and one column for each participant. It starts with initial record with all 0's. If there are k participants on the given day, and A is the driver, then A's entry is increased by $U(k-1)/k$ units. For the rest of the users who do not drive, entry is decreased by U/k units. To decide the next driver, the algorithm chooses the person with the lowest entry in the table. [35] shows the above algorithm maintains fairness with a bounded deviation from the ideal number of drives.

2.2.3 Scalable Context Collection and Evaluation

A number of authors have identified the requirements, implementation, and major research challenges of systems that support context-aware applications [16, 77, 115]. According to the authors in [77], existing implementations can be categorized according to the functionalities provided by these systems such as the capability of collection, storage, query, prediction, and context representation.

Context-aware approach is widely used for searching in IoT. In [10], the authors proposed a context-aware service framework on top of IoT controlled systems, which is applied on the fault management process in electric power distribution networks. Their proposal takes automate actions depending on contextual information sensed from the IoT environment and received by the framework through its controlled systems. In [97], a context-aware and multi-service trust management system fitting the new requirements of the IoT is designed. To the best of our knowledge, we could not find any work that addresses the problem of minimizing the latency of the context-aware applications while maintaining a bound on the energy consumption cost.

2.3 Energy Efficiency of Context-Aware Applications

Several authors [61, 7, 8] in the past highlighted the importance of energy efficiency of context-aware mobile applications. Iqbal *et al.* presented a generic architecture [51] for context-aware mobile applications using a four stage energy saving techniques introduced in [118] which is *Suppression, Substitution, Adaptation and Piggybacking.* They proposed a state machine consisting of states and transitions, through which energy consumption of an application is optimized. Nailah *et al.* suggested context-aware energy optimization for perpetual IoT-based safe communities, which uses a semantic approach that utilizes context of extracted activities of daily living (ADLs) from device data to drive energy optimized sensor activations [7]. CASSARAM [88] proposes to select the sensor from large set of sensors with capabilities and parallel functionalities. Based on the user requirements the context will be identified and relevant sensors are used to the context data. From the large set of sensors, CASSARAM finds out which sensor is containing more energy.

Piggyback Crowdsensing (PCS) [70, 112] is a framework designed to reduce the energy overhead of smartphone-based crowd-sensing. Using prior knowledge about user habits and CPU/network profiles of applications, PCS determines when to trigger on-board sensors to minimize energy consumption. Data upload only happens at night when the smartphone is connected to WiFi and power adapter at users homes. When and how to upload the collected information is not a significant concern. In [99], authors have proposed an efficient context data processing pipeline where an early feature detection stage can eliminate the usage of expensive context recognition. Although such an approach can minimize the overall energy consumption cost of mobile devices, but it can incur high latency depending on the complexity of the context evaluation function. In contrast, our hybrid architecture chooses to collect, transmit and evaluate context data appropriately either on the mobile side or on the server side to overcome the latency barrier of the end applications.

Chapter 3

IoT Testbeds

For chapter 4 and 5, we use TIPPERS dataset for experiments. For chapter 5, we experimented on Honeywell Testbed dataset. We explain the dataset in detail in the following.

3.1 TIPPERS

TIPPERS (Testbed for IoT-based Privacy-preserving PERvasive Spaces) [79] is an experimental six-story smart building testbed designed to study the numerous privacy challenges due to fine-grained monitoring of building occupants and visitors using a diverse set of sensors [9]. To date, TIPPERS uses the data from pre-installed 40 cameras, 64 WiFi APs, over six thousand Heating, Ventilation, and Air Conditioning (HVAC) sensors measuring airflow and ventilation as well as temperature at different parts of the building, and a large number of light and motion sensors. Also, several hundred of bluetooth beacons covering all major regions in the building, over a hundred smart plug meters to monitor energy consumption of connected devices are installed. Data from these sensors flows through the TIPPERS system that fuses the underlying sensor data to produce mainly two higher-level data streams PRESENCE, which monitors location of all individuals who are inside the building as a function of time, and ENERGY, which monitors energy usage at different spatial resolutions. The information managed by the TIPPERS database system is used to build a variety of applications from real-time awareness of resources, people, and events, to mechanisms to perform analytics on historical data.

Dataset. In TIPPERS, the location of individuals is determined dynamically based on several lower level data sources including connectivity to WiFi APs, presence of an individual in a video feed, WiFi fingerprinting, as well as connectivity to different bluetooth beacons. Of the different methods for localizing individuals within the building, using WiFi APs to track client connections is the most useful because (a) it is ubiquitous, since wireless network covers the entire building, and (b) it does not require active participation of, or any software to run on the client machine. One of the key challenges, however, in using such dataset is the relatively coarse granularity (region level) compared to the much finer granularity that can be achieved using beacons and/or cameras. While coarse granularity data suffices for certain applications/analysis tasks (e.g., understanding region level occupancy of the building), for other tasks needing finer granularity (e.g., at the room level) additional mechanisms for localization need to be designed. One of the mechanisms explored in TIPPERS is to postulate the finer granularity localization, viz., room level, as a data cleaning challenge. Additional information such as location of occupants office, calendar entries, data collected over time to observe patterns in the location of individuals, as well as fine granularity location data collected sporadically using other sensors such as beacons placed in some locations, is used within TIPPERS to develop models for fine grained localization using WiFi AP datasets. *Hedwig*is built on top of PRESENCE and OCCUPANCY which is derived from PRESENCE.

3.2 Honeywell Testbed

In Honeywell Golden Valley laboratory, total 7687 Enlighted[1] sensors (see figure 4.2) which are capable of recording occupancy, light, power, and temperature are placed for sensing and actuating purposes. These sensors are used for energy and space management.

Dataset. Enlighted is continuously recording occupancy data every 5 minute interval. We have gathered data from Honeywell Golden Valley laboratory, which is a three story commercial building, but focused on data collected from second floor, which is a main floor where most employees' work spaces are located. Total 2493 fixtures were collected, in particular, 1719 fixtures on the second floor for 68 days. Along with occupancy data, we used sensor dataset which stores sensor ID, name, floor ID, area ID, area name, area description, and the x, y coordinates of sensors.

[1] `http://www.enlightedinc.com/`

Chapter 4

Exploring Privacy Breaches and Mitigation Strategies of Occupancy Sensors in Smart Buildings

Smart buildings are becoming increasingly prevalent as an integral part of our society. Equipped with various sensors and actuators, smart buildings enable automated building controls including light, HVAC, and space optimization [101]. This results in personalized comfort controls for the building occupants as well as economic savings for the owners of buildings. However, as more fine-grained data at individual levels are collected for building automation, dangers of potential privacy breach also increase [78]. New technologies are putting us at risk from increasingly sophisticated privacy attacks. Evidently, IoT sensors process and produce massive amount of data and potentially, these data could be a potential source for attackers to exploit privacy of users.

One type of sensor that has been identified as a potential privacy risk is the occupancy sensor [59]. Occupancy sensors detect presence of an individual which enables dynamic control of light and HVAC systems based on the detected occupancy information. However, the collected occupancy data could potentially be used for inference of individual's occupancy patterns and tracking individuals [109].

In this chapter, we study an occupancy reasoner, occupancy as an environmental context, has privacy implications in smart buildings. In particular, we are interested in a setting where each individual is given a primary location where the person is expected to be throughout the day, such as cubicles or offices.

4.1 Background on Occupancy Sensors

In many cases, the automation control is indifferent to the identities of building occupants. For example, detection of a presence is sufficient to trigger turning on the light at the sensed location. The automated building control does not require the knowledge of *who* is triggering the occupancy sensor, but only requires the information that presence is detected at a particular location.

For this reason, the database containing user-identifying information is typically kept separately from the database storing occupancy information, and there is no communication between those databases. Because of this separation, it is easy to believe that unless the adversary compromises both databases, it is extremely difficult to infer the mapping between the set of sensors that are located at primary locations of individuals and the identities of individuals. The findings of this chapter demonstrate that maintaining separate databases is insufficient for mitigating privacy leakage from occupancy sensors. Specifically, we make the following set of contributions.

- We propose a new privacy attack in smart buildings when the gathered occupancy information does not contain unique identifiers of individuals. In the proposed attack, the adversary combines the sensed occupancy data with auxiliary information which

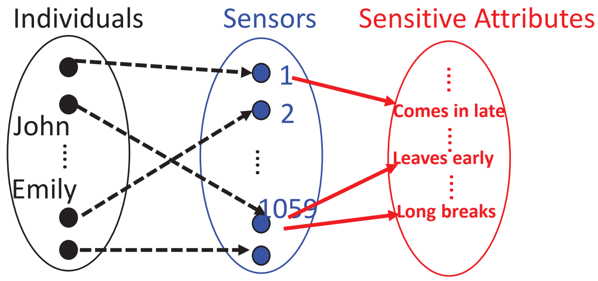


Figure 4.1: Illustrating the privacy implications of the presence data. Once the adversary creates a mapping between the set of individuals and the set of presence sensors, the adversary can infer sensitive attributes that pertain to specific individuals.

is often public information. This attack enables the adversary to identify the sensor ID whose location is associated with the individual, which in turn, leads to leakage of sensitive information (Figure 4.1).

- We perform an experiment using real-world data to validate the feasibility of the proposed attack. The experiment shows that even with a small set of auxiliary information, the adversary can obtain a significant amount of information regarding the sensor ID at the primary location of the targeted individual.
- We propose a set of mitigation strategies against the proposed attack including the use of differential privacy, time-based encryption and deletion of the stored occupancy data.

The chapter is organized as follows. Section 4.2 contains our assumptions of the building and the adversary. In Section 4.3, we describe the proposed privacy breaching attack in detail. Section 4.4 describes the experimental study. Section 4.5 summarizes potential mitigation strategies against the proposed attack.

4.2 Model and Preliminaries

In this section, we present data processing flow, adversary and building models. We also define the notations that will be used throughout the chapter.

Figure 4.2: Enlighted sensor.

4.2.1 Data Source

We have used Enligted dataset which is continuously storing occupancy data every 5 minute interval. We have gathered data from Honeywell Golden Valley laboratory, which is a three story commercial building, but focused on data collected from second floor, which is a main floor where most employees' work spaces are located. Example dataset is shown in Table 4.1.

Capture Time	Sensor ID	Occupancy (Hex)	Power (Watt)	Temp. (F)
8/1/2019 12:05:00 AM	5314	1152921504606840000	0	71.16
8/1/2019 12:05:00 AM	238	0	1.2	76.92
8/1/2019 12:05:00 AM	904	0	1.2	72.78

Table 4.1: Occupancy raw dataset.

4.2.2 Data Processing Flow

We have developed a location inferencer to find out office/cubicle candidates. A location inferencer takes occupancy readings and sensor locations as inputs and produces location candidates as outputs. It outputs location candidates as a png file. Figure 4.3 illustrates data processing flow of location inferencer.

4.2.3 Building Model

We consider a building that is divided into multiple locations where each location is equipped with a presence sensor. We assume that a presence sensor is capable of inferring whether or

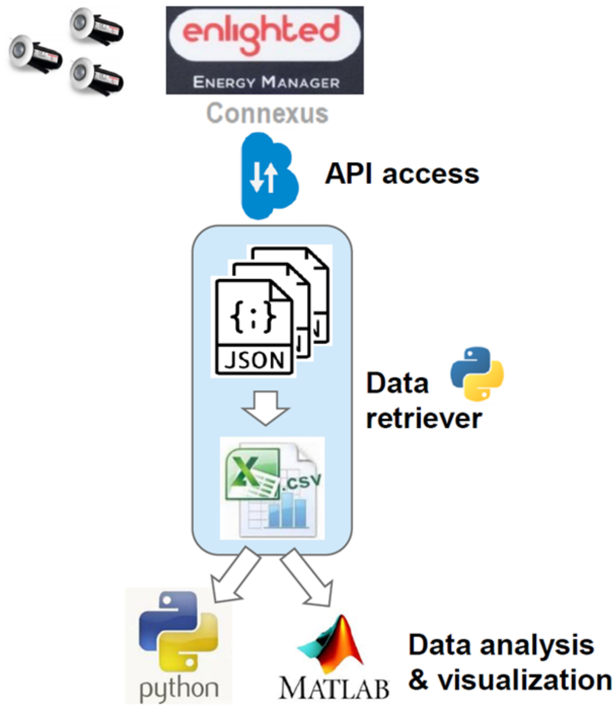


Figure 4.3: Data processing flow.

not the location is currently occupied, but is not capable of inferring which specific individual is triggering the occupancy. An example of such a sensor would be a motion detector which uses ultrasonic or microwave technologies.

Each individual is given a primary location where the individual will be spending most of the time when the individual is in the building. For example, a primary location might be a designated office, a particular room, or a particular cubicle.

The set of presence sensors are denoted as $\mathcal{S}$. Each sensor periodically detects presence in its neighborhood. The occupancy pattern of sensor $s_j \in \mathcal{S}$ at time t is denoted as o_j^t and is defined as

$$o_j^t = \begin{cases} 1, & \text{if sensor } s_j \text{ detects presence at time } t \\ 0, & \text{else.} \end{cases}$$

Similarly, we define $o_j^{[t_0,t_f]} = 1$ if occupancy is continuously detected during the time interval

$[t_0, t_f]$ and $o_j^{[t_0,t_f]} = 0$ otherwise.

4.2.4 Adversary Model

An application that consumes the occupancy data could potentially use the data other than its intended purposes. For example, an adversary might be interested in finding the behavioral patterns of the users, friendships, or floor maps of the building. The goal of the adversary is to infer the sensor ID that is installed at the primary location of the targeted individual i denoted as s_i. It is assumed that the adversary can query any sensor ID to observe the presence information at any given time t.

The adversary also has a set of auxiliary information of the individual i, which is denoted as $\mathcal{A}_i$. Auxiliary information is defined as occupancy-pattern information that the adversary has about the individual. An example of a piece of auxiliary information is a travel schedule of the individual from which the adversary can infer that the individual was not in the building for a given time interval. Such information is often public information that can easily be obtained from social network sites including LinkedIn, where postings may include an individual receiving an award or giving a presentation at a conference. Such information would indicate that the individual was not present in the building on the days when this was posted.

For each auxiliary information $a_i^{[t_0,t_f]} \in \mathcal{A}_i$, we denote $a_i^{[t_0,t_f]} = 1$ if the individual i was known to be in the building during the time interval $[t_0, t_f]$ and $a_i^{[t_0,t_f]} = 0$ if the individual i was known to be absent from the building during the time interval $[t_0, t_f]$.

4.3 Proposed Attack

In this section, we propose an attack in which the adversary infers the sensor ID of the targeted individual i with high probability by combining the auxiliary information with the queried occupancy patterns. The main idea of the attack is to exploit the fact that the sensor s_i has to satisfy all the conditions presented by the auxiliary information $\mathcal{A}_i$. An illustration of the proposed attack is shown in Figure 4.4. In this particular attack, the adversary computes the intersection of four sets (out of office in April, out of office in May, in the building in August, out of office in June) that are consistent with the auxiliary information such as "out of office" or "in the building" to uniquely identify the sensor ID that is associated with the targeted individual.

4.3.1 Formal Description of the Attack

Without loss of generality, we define the set $\mathcal{A}_i$ as

$$\mathcal{A}_i = \{a_i^{[t_0,t_1]}, \dots, a_i^{[t_{n-1},t_n]}\} \tag{4.1}$$

$\mathcal{A}_i$ represents the set of auxiliary information obtained by the adversary for individual i. For each item of auxiliary information $a_i^{[t_{m-1},t_m]} \in \mathcal{A}_i$, define the set $\mathcal{S}_m^i \subset \mathcal{S}$ as

$$\mathcal{S}_m^i = \{s_j : o_j^{[t_{m-1},t_m]} = a_i^{[t_{m-1},t_m]}\} \tag{4.2}$$

In other words, the set $\mathcal{S}_m^i$ is the set of sensors such that the corresponding occupancy patterns are consistent with the auxiliary information $a_i^{[t_{m-1},t_m]}$. Given n auxiliary information, the adversary can construct n such sets, and the sensor s_i has to be in all of the sets $\mathcal{S}_1^i, \dots, \mathcal{S}_n^i$. Assuming the adversary does not have any a priori information regarding s_i, the probability of correctly guessing s_i given $\mathcal{A}_i$ is written as

$$\mathbf{P}(\hat{s}_i = s_i | \mathcal{A}_i) = \frac{1}{|\bigcap_{m=1}^{n} \mathcal{S}_m^i|} \tag{4.3}$$

where $\hat{s}_i$ is the adversary's estimate of s_i.

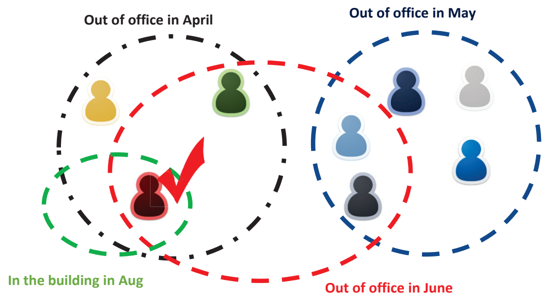


Figure 4.4: Figure illustrating the proposed attack. The attacker continuously intersects sets of sensors that are consistent with the auxiliary information.

4.3.2 Area Type Inference

The adversary infers area types without having a floor map. We explore the idea that the adversary infers area types (office, cubicle, hallway) from occupancy data alone.

The adversary starts from the intuition that sensors which behave similarly are more likely to be the same area type (temporal aspect). Also, sensors that are close to each other are more likely to belong to the same area type (spatial aspect). Here, the adversary explores temporal aspects of group sensors. In particular, the adversary finds features of group sensors in three area types: office, cubicle, and hallway.

Table 4.2 summarizes our occupancy data exploration results. We test out 7 different hypotheses and accepted 5 among 7. Two hypotheses which turned out to be unuseful were as follows: 1) H5: Average number of changes ('01' and '10') carries enough information to decide whether the sensor is located in office/cubicle/hallway, 2) H7: Summation of occupancy data with fast sampling rate (5 seconds) carries enough information to decide whether the sensor is located in office/cubicle/hallway. The main reason for these negative results was due to occupancy sensor's fidgety behavior. Occupancy reading in 5-second intervals is unreliable to use as a feature. However, by combining H3, H4, and H6 and calculating average sparsity, maximum un-occupancy, and occupancy variance as features, the adversary can infer a user's location with high probability.

4.4 Experimental Evaluation

We conducted an experimental study using real-world occupancy sensor data from an office building. The goal of this study was to experimentally evaluate the effectiveness of the attack described in Section 4.3.

We selected one zone of the office building for the experimental study which contained 1663 motion detectors. Each motion detector records a binary value of 1 if it senses presence via motion and 0 otherwise. The presence value is recorded every 5 seconds and transmitted to a centralized database.

The auxiliary information we considered was daily travel schedules of individuals. The travel schedules used for this study were obtained either with user consent or through public data from LinkedIn. The set of auxiliary information was gathered within the period of 5 months.

For each day, we assumed that an individual was present in the building if person stayed in his or her primary location for more than an hour, which is equivalent to having more than 720 number of 1's in the presence data for one day. The assumption was made to account for triggering of the motion detector by bypassing people or patrolling security guards at night time. If the presence data for a sensor contained less than 720 number of 1's for one day, then we assume that the area covered by the sensor was unoccupied for that day.

We conducted our experiment on four employees. For the first case, we used three pieces of auxiliary information.

1. The person traveled for two days during weekdays and therefore was not present in the building during the travel.
2. The person has not worked during the weekends for a particular month.

ID	Hypothesis	Finding	Conclusion
H1	Occupancy data with several days of occupied/unoccupied metadata will increase the probability of inferring a user's location.	Accepted; combining occupied/unoccupied days of occupancy data will significantly reduce number of candidate locations.	A combination of occupied/unoccupied days is meaningful to infer one's location
H2	Part of occupancy sensor readings are not reliable	Accepted; Some sensors have no readings over the whole period, some of them have very low occupancy value	Data cleaning before processing is required
H3	Average sparsity of occupancy carries enough information to decide whether the sensor is located in office/cubicle/hallway	Accepted; using this as a feature to apply clustering sensors will find good area type clusters	K-Means clustering algorithm with average sparsity, maximum unoccupancy, and occupancy variance find clusters with three different area types
H4	Maximum un-occupancy carries enough information to decide whether the sensor is located in office/cubicle/hallway	Accepted; using this as a feature to apply clustering sensors will find good area type clusters.	K-Means clustering algorithm with average sparsity, maximum unoccupancy, and occupancy variance finds good clusters with three different area types
H5	Average number of changes ('01' and '10') carries enough information to decide whether the sensor is located in office/cubicle/hallway	Rejected; using this as a feature to apply clustering sensors will not find good area type clusters	Average number of changes is not reliable to use as a feature
H6	Occupancy variance carries enough information to decide whether the sensor is located in office/cubicle/hallway	Accepted; using this as a feature to apply clustering sensors will find good area type clusters.	K-Means clustering algorithm with average sparsity, maximum unoccupancy, and occupancy variance finds good clusters with three different area types
H7	Summation of occupancy data with fast sampling rate carries enough information to decide whether the sensor is located in office/cubicle/hallway	Rejected; summation of occupancy data is not enough to differentiate area type	Summation of real time occupancy is not reliable

Table 4.2: Occupancy data exploration summary.

3. The person returned to work after two days of travel and was present in the building on the third day.

Without any additional information, the probability of correctly guessing the sensor ID that is located at the person's primary location is $\frac{1}{1663}$. After identifying the set $\mathcal{S}_1$ that satisfied the first auxiliary information 1), the cardinality of the set was reduced to 14 from 1663. In addition, after obtaining the intersection $\mathcal{S}_1 \bigcap \mathcal{S}_2 \bigcap \mathcal{S}_3$, where $\mathcal{S}_2$ and $\mathcal{S}_3$ are the sets that satisfied the auxiliary information 2) and 3) respectively, the cardinality of the new set was reduced to 9 from 1663. We have verified that the person's primary location (an office space) was indeed one of the nine candidate locations that we identified.

Figure 4.5 shows how probabilities of correctly identifying primary locations (Equation (4.3)) change with auxiliary information for all four cases. The horizontal axis is the relative time of the year when the auxiliary information was gathered within the period of five months. For cases 2,3, and 4, the *off* label shown in Figure 4.5 indicates that the person was not present in the building during weekdays and the *on* label indicates that the person was present in the building. In all four cases, we observed that with two to three pieces of auxiliary information, the cardinalities of candidate sets will reduce to approximately 10, resulting in significant increase of probabilities of correctly identifying the primary locations for all four individuals to approximately 0.1 from $\frac{1}{1663}$.

In all four cases, it was observed that a person's absence in the building during weekdays significantly reduces the size of the candidate set initially. This is due to two reasons. First, identifying the candidate set of locations that are occupied during weekdays will eliminate almost all sensors that are located in hallways, which are usually occupied during weekdays. Second, since the building chosen for this study does not allow working from home, being absent during weekdays is a rare event.

However, as can be seen in cases 1,2 and 4, having additional auxiliary information of

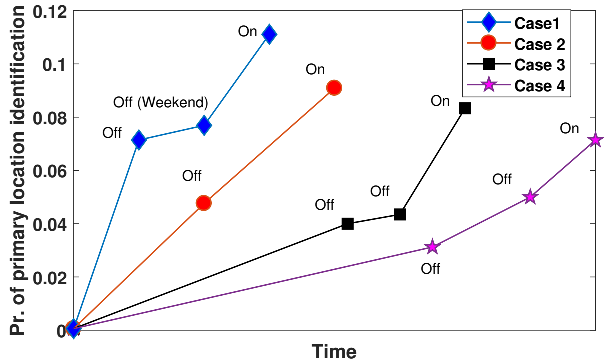


Figure 4.5: Figure illustrating the increase of probability of correctly identifying the primary locations given auxiliary information. Horizontal axis is the relative time of the year when auxiliary information are collected. The *off* label indicates that the person was not present in the building during weekdays for cases 2,3 and 4. The *on* label indicates the person was present in the building.

being absent in the building does not provide much additional information. This is because the candidate set with the first item of auxiliary information contained locations that are unoccupied for a long period of time. Therefore, unless significant changes have occurred in such empty locations including new people being assigned to the locations in a short period of time, the same locations will be counted even after refining the candidate set with an additional auxiliary information of the individual being absent in the building.

To eliminate the empty locations from the candidate set, it is required to combine an additional piece of auxiliary information of the candidate being *present* in the building. As seen in all four cases, the probability of correctly identifying the sensor ID (Equation (4.3)) significantly increases when the targeted individual's presence in the building is accounted for.

To further reduce the number of sensors to explore, we conduct an experimental study on area type inference. We apply k-means clustering algorithm with three features:

- Average occupancy sparsity: an average of distances between chunks of 1s. For example, if we have binary readings of '11111111111111000001000011000000000000000110001 0000001100101' as o_t, an average occupancy sparsity's value is $(5 + 4 + 15 + 3 + 6 +$

2 + 1)/7 * 5 = 25.

- Maximum un-occupancy: a maximum value of chunks of 0s. For the same binary number given above, maximum un-occupancy's value is a *maximum* of (5, 4, 15, 3, 6, 2, 1) = 15. Offices will be likely to have longer consecutive un-occupancy than hallways and cubicles.

- Occupancy variance: variance of 0s and 1s. Hallways are likely to have bigger variance due to combinations of multiple people moving around. However, offices or cubicles will have less dynamic movement patterns since they are occupied by single persons for the most of the time.

Figure 4.6 shows an experimental result of applying k-means clustering algorithm with k = 3 to the set of sensors installed on the second floor. Blue dots are candidates for cubicles, red dots for hallways, and yellow dots for offices. For the subset (150) of sensors installed on the second floor, error rates are: hallways (18%), office (12%), and cubicle (30%). Sensors installed in hallways and cubicles show higher error rate than sensors installed in offices as sensors installed on top of cubicles observe similar patterns to the sensors installed in hallways.

We filter out raw sensor observations based on a user's presence information in the building. We apply set intersection for the dates he/she was present/absent. By applying set intersection for 4 different known absence/presence dates, we increase the probability of correctly guessing an employee's office candidates. Once we have reduced the number of candidate sensor locations, we apply area type inference algorithm introduced in 4.3.2 to further reduce the candidate locations. By combining set intersection approach for known travel dates and k-means clustering algorithm to determine space types, we were able to achieve more or less than 10 candidate points for three candidates.

Based on our findings, we ran the experiment on one of executive member's data. By applying

Figure 4.6: Figure illustrating area type can be inferenced based on k-means clustering algorithm. The blue dot indicates cubicle. The red dot indicates hallway. The yellow dot indicates office.

set intersection for 4 different known absence/presence dates and area inference algorithm, we narrowed down the probability of his locations to 1/7.

4.5 Potential Mitigation Strategies

In this section, we propose a set of potential mitigation strategies against the proposed attack.

4.5.1 Principle of Least Privilege for Privacy

Principle of least privilege has been widely accepted as one of the governing principles in designing mitigation strategies in security. The principle of least privilege states that the minimum amount of information and resources should be given to a subject in order for the subject to perform required duties. The same principle can be used for privacy enhancement as stated in [73].

For building applications, we need to ensure that only the minimum, necessary level of occupancy information is disclosed. For example, for occupancy-driven HVAC control, the controller only requires occupancy information at the zone level, but does not require the information to be consumed at the individual sensor level [31]. Specifically, this implies that the HVAC controller will be given the number of people present in a zone, but not presence information at each individual sensor.

The adversary could still potentially perform the attack using the zone level occupancy information to at least identify which zone includes the targeted individual's primary location. On the days when the individual is not present in the building, the adversary would expect to observe a decrease in the number of occupants in the zone. However, unlike the attack performed on the individual sensor level, the behaviors of other occupants in the zone would affect the accuracy of the identified sets in Equation 4.3. It is possible that the decrease in the number of occupants could be due to other individuals not present in the building when the targeted individual is present. Similarly, even when the targeted individual is not present in the building, other individuals that were not previously in the zone could be present, resulting in no change or even an increase in the number of occupants in the zone. Moreover, even if the adversary were able to correctly identify the zone where the targeted individual resides in, k-anonymity [103] will be provided where k is the number of sensors in the zone.

4.5.2 Differential Privacy

When the occupancy information at the zone level is released, it is possible to implement additional privacy enhancing mechanisms that will provide differential privacy. In this setting, probabilistic noise is added to the released occupancy information to ensure that answers to queries regarding the occupancy level in a zone would be statistically similar even when a single individual is present or not present in the zone [30].

We have implemented PeGaSus [25] on the collected occupancy data to examine how the utility of the occupancy data would change as we vary the parameter ϵ. However, initial results suggest that even for a relatively large value of ϵ, the utility of the gathered occupancy data decreases significantly. Finding the optimal trade-off between the utility and privacy for the differential privacy based mitigation would be part of future work.

4.5.3 Deletion of Past Information

The proposed attack does not rely on the particular timeframes when the presence data are collected. Whether the presence data was collected in the past or present, any data that matches the auxiliary information would be equally valuable in narrowing down the set of candidate sensor IDs of the targeted individual.

While the past data could be valuable in many applications that rely on predicting future occupancy levels as well as energy consumptions through machine learning techniques, keeping a large amount of past data will provide additional opportunities for the adversary. Setting an expiration time on the gathered occupancy data and either deleting or encrypting it automatically at the time of expiration will provide additional privacy against the proposed attack by eliminating the occupancy data that correspond to auxiliary information in the past.

The adversary can gather the auxiliary information of an individual from various sources including social networks, news, or other offline sources. However, the building cannot control the leakage of such auxiliary information, and more importantly, does not know which individual may be targeted by the adversary.

From our experience, we identified the most common and likely kind of auxiliary information as the day to day travel schedules since such information is often published online. Automatic reply e-mails set up during vacations or travels also reveal specific dates when the targeted individual is not in the building.

Assuming that the set of auxiliary information gathered by the adversary is the set of specific dates when the individual was in the building or not in the building, releasing the daily occupancy data is likely to lead to privacy breach with high probability.

4.5.4 Analysis of the Mitigation Strategy

It is assumed that the parameter p is known to the adversary. In addition, without loss of generality, we assume that $p < 0.5$. Since for any $p > 0.5$, the adversary can flip all the elements in $\tilde{\mathbf{o}}_j$ and regard each element being flipped with probability $1 - p$.

Theorem 4.1. *The probability of s_i being included in $\bigcap_{m=1}^{n} S_m^i$ under the proposed mitigation strategy is given as*

$$\mathbf{P}(s_i \in \bigcap_{m=1}^{n} S_m^i) = (1-p)^n \tag{4.4}$$

Proof. The proof is trivial and omitted for brevity. □

Theorem 4.1 states that if the adversary were to naively conduct the attack described in Section 4.3, then the probability of correctly deducing a set that includes s_i is a monotonically decreasing function in n. Therefore, additional auxiliary information would in fact increase the probability of deducing an incorrect set instead of correctly narrowing down the possible set of candidates for s_i as shown in (4.3).

The adversary, however, can perform a more sophisticated attack by computing the probability

$$\mathbf{P}(s_j = s_i|\tilde{\mathbf{o}}_j) \tag{4.5}$$

for each sensor s_j and then construct a set that is highly likely to include the target sensor s_i. The effectiveness of the mitigation strategy can be quantified by the size of the computed set such that the probability of s_i being included in the set is comparable to that of (4.3).

Chapter 5

Exploring Fairness in Participatory Thermal Comfort Control in Smart Buildings

5.1 Background on Fairness and Thermal Comfort Control

The control of the temperature in most of our current buildings (e.g., offices) is still based on rules that try to match the comfort of average people. For instance, Federal Occupational Safety and health Administration (OSHA) regulations usually recommend that the temperature in the different rooms be in the 20-25^oC range[1] which is considered to be comfortable. The reality is that people have diverse comfort preferences depending upon different factors such as their gender, age, weight, and nationality [113, 63]. Such prespecified control tends not to be comfortable for many of the inhabitants of the building [69].

[1] `https://www.osha.gov/pls/oshaweb/owadisp.show_document?p_table=INTERPRETATIONS&p_id=24602`

Smart buildings equipped with a range of plug-and-play IoT devices that provide sensing/actuation capabilities are becoming the norm; these technologies allow users to interact with spaces they live/work in on a fine grained basis and participate in the control and management of these spaces. Participatory engagement with building environments in a more integrated manner enables a new level of personalized services including customized capture and exchange of information. Such participation can be used in the context of thermal comfort control. In this context, Participatory Comfort Control (PCC) systems adapt the temperature of buildings to increase the comfort of their residents.

However, adapting the temperature to the desired comfort level of the inhabitants of the building presents several challenges. On the one hand, there is a need for mechanisms to allow people to express their feedback regarding the temperature in the room they are. On the other hand, the feedback of the users have to be implemented in the HVAC (Heating, ventilation and air conditioning) system of the building. Multiple approaches have been presented in the literature to address these challenges. For instance, systems have been proposed to obtain feedback from people automatically (e.g., by determining their thermal profiles using information about their gender, age, height and weight [21] or manually (e.g., by enabling them to vote using their smartphones) [68, 44, 67]. Also, different mechanisms have been presented to translate the preferences of the users into the appropriate HVAC settings [37, 32, 34].

Nevertheless, there is an intrinsic challenge in such PCC systems: people inside a same room usually have different thermal comforts that have to be somehow aggregated to decide the temperature for the room (unfortunately, having different temperature zones in the same room might not be always possible). Works in the literature dealing with this problem often leverage traditional decision making methods to aggregate the possibly diverse opinions of people. For example, group decision making processes, such as majority or mean, have been extensively studied to reach consensus [104]. However, these methods might not be *fair* with the participants. For example, think about a group of people that meet daily in the same room (e.g., coworkers sharing a meeting room). If a majority of the people prefer cold temperature and a minority prefer the temperature to be hot, the previously mentioned PCC systems might determine that the temperature in the meeting room should be cold always. In the long term, the minority will be clearly uncomfortable.

In this chapter, we study longitudinal fairness in the aggregation of user thermal comfort in PCC systems, user thermal comfort as a user context in smart buildings. We present the first study, up to the authors' knowledge, on this issue. In particular, the contributions of this chapter are as follows:

- We introduce a definition of fairness, inspired by the traditional definitions in scheduling, adapted to the particularities of our scenario. Under this definition we prove that systems that use traditional group decision making processes, such as majority, mean, trimmed mean, and median, to aggregate people thermal comfort are not fair.
- We present our design of an aggregation algorithm that ensures fairness with the participants. We prove mathematically that the proposed algorithm is fair regardless of the number of people and iterations.
- We present a tool to simulate scenarios to evaluate different aggregation algorithms. The tool enables us to define the thermal profile of the inhabitants of a building, the frequency of the feedback obtained from them, and the desired aggregation algorithm, and executes a simulation that computes its fairness and comfort.
- We show experimental results of our approach and the traditional aggregation group decision making methods used in the literature (i.e., majority, mean, trimmed mean, and median) using the simulation tool with two scenarios: a weekly group meeting and a building with hundreds of participants and 64 zones. For the latter, we drive the simulation by the real occupancy data captured in a building.

The experimental results show the fairness of our algorithm in contrast to the rest. They also show that, although there is a trade off between fairness and comfort, the overall discomfort of the participants with our algorithm remains similar to the discomfort that results from other techniques that target minimizing discomfort without considering fairness.

The rest of the content is structured as follows. In Section 5.2, we introduce a motivating scenario of a real building with different rooms and people. In Section 5.3, we present our modeling of fairness and comfort in occupant-participatory thermal comfort approaches. In Section 5.4, we show our fair algorithm for the aggregation of user thermal preferences. Finally, the experimental evaluation is presented in Section 5.5.

5.2 Motivating Scenario

Consider a future commercial office building where the HVAC (Heating, ventilation and air conditioning) system has been instrumented to capture resident feedback on thermal comfort; furthermore, let us assume that the system is capable of incorporating this input when adjusting temperatures for different rooms in the building. Note that any existing system can be used as a starting point for this instrumentation [44, 67, 34, 98, 89, 66].

Person	Room	Comfort
Alison	2011	+1
Eleanor	2059	-3
Erin	2059	-3
James	2011	+1
Jason	2059	-3
Karl	2059	+3
Katie	2059	-3
Katherine	2059	-3
Lisa	2011	+1

Figure 5.1 & Table 5.1: Distribution of the building residents in our sample scenario (left) and their thermal comfort (right)

Let us also suppose that the distribution of people at a given time in the rooms of the our scenario is as shown in Figure 5.1 and that their thermal comfort at a given time is given in Table 5.1.

In the case of rooms occupied by a single person, the individual's preferred temperature setting can be applied. When multiple residents of a space have identical thermal preferences (e.g., room 2011), the selection of a comfortable setting for all is also simple. When several residents in a room have varying thermal preferences (e.g., room 2059 where 5 users are cold (-3) and 1 of them feels hot (3)), one can apply any commonly used aggregation method to represent a group decision, such as: 1) *majority* (which selects the comfort supported by more than half of the participants), 2) *mean* (which selects the average value among votes), 3) *trimmed mean* (which selects the average value after trimming a percentage of the largest and smallest values), or 4) *median* (which selects the value, in an ordered set of values, below and above which there are an equal number of values). As an example, if we were to use the system presented in [91], the mean method would be applied to situations with several participant thermal comforts in the same room. A setting of "cold (-3)" (the mean of the comfort values) will result in discomfort for the single person that feels hot (3) at all times – such a smart building is *unfair* to some of its residents.

5.3 Modeling Fairness and Comfort

In this section, we first explain the modeling of the participatory comfort control scenario. Then, we present a definition of fairness for such scenario that is inspired by the definition of fairness in the carpooling problem described earlier.

5.3.1 Modeling Thermal Comfort

Typically, PCC systems are slot-based control systems where decisions are made for a fixed interval of time, which we will refer in the following as a "round". We consider that at the beginning of each round, thermal comfort collection happens. We adopt the PMV scale (see Section 2.2.2) to model participant comfort. Thus, thermal comfort of a participant P_i at round r is a value from the PMV scale and can be defined as $TC_r[P_i] = [-3, 3]$ (see Table 5.1 for an example).

Let $P[z] = \{P_1, ..., P_n\}$ be the set of participants in a specific zone z, and let $TC_r[P[z]]$ denote the aggregated comfort of zone z set by the aggregate thermal control algorithm in round r. Note that $TC_r[P[z]] = [-3, 3]$ (i.e., it is also a value from the PMV scale). For example, considering the set of participants in room 2059 in our running example (see Figure 5.1) and their thermal comfort (see Table 5.1), if traditional aggregation methods such as majority, mean, trimmed mean, or median were used the value of $TC_r[P[2059]]$ would be -3,-2,-2, and -3, respectively.

Moreover, we can measure the disagreement between a participant's thermal comfort $TC_r[P_i]$ and the aggregated thermal comfort of the participants in the zone (viz., $TC_r[P[z]]$) as the difference in absolute value of these two terms $TCD_r[P_i] = |TC_r[P_i] - TC_r[P[z]]|$. In our running example, the disagreement per participant for the different approaches to aggregate thermal comfort discussed earlier is shown in Table 5.2. A measure of the total disagreement of the participants in the zone can be obtained by summing up their respective disagreements:

$$TCD_r[P[z]] = \sum_{i=0}^{r} TCD_r[P_i] \tag{5.1}$$

We can aggregate the disagreement of a user over time to obtain a value that represents how uncomfortable the user has been with the decisions made until round r as follows:

$$ATCD_r[P_i] = \sum_{k=0}^{r} TCD_k[P_i] \tag{5.2}$$

Finally, we generalize the scenario to buildings with m thermal zones where the temperature

Participant	Majority	Mean	Trimmed Mean	Median
Eleanor	0	1	0	0
Erin	0	1	0	0
Jason	0	1	0	0
Karl	6	5	6	6
Katie	0	1	0	0
Katherine	0	1	0	0

Table 5.2: Disagreement between the participant thermal comfort and the zone thermal comfort for traditional approaches in our running example.

can be controlled. We define the set of zones as $Z = \{Z_1, ..., Z_m\}$. Also, we define the set of participants of the building at a round r as $P = \{P_1, ..., P_n\}$. P includes all the participants that have been in the building until round r.

5.3.2 Modeling Fairness

The definition of fairness in our context is based on the definition of fairness in the carpool scenario explained in Section 2.2.2. In the carpool scenario every time that users are involved in driving, they split the "cost" of the ride. A driver should ideally have driven at any point of time her share in the rides she took part in.

In our scenario, the equivalent of a ride is a round, which is the moment when the thermal comfort of the zone(s) is calculated using the thermal comfort of the N inhabitants within. During round r, each user P_i "loses" the value represented by her thermal comfort disagreement, which we refer to as *real loss* $L_r[P_i] = TCD_r[P_i]$. We define the "total loss" (or cost) of the round as $L_r = \sum_{i=0}^{N} L_r[P_i]$, which is equal to the thermal comfort disagreement of the zone $TCD_r[P[z]]$ in Equation 5.1. Then, we split the loss L_r among the N users to define their *ideal loss*. We define the *extra loss* of a user P_i in round r, $EL_r[P_i] = L_r[P_i] - L_r/N$ as the difference between her ideal loss and her real loss. Finally, we define the *accumulated extra loss* of a user P_i at round r as:

$$AEL_r[P_i] = \sum_{k=0}^{r} EL_k[P_i] \tag{5.3}$$

We define that an algorithm to select the thermal comfort of a zone is "fair" if for the N users in the zone and at the round r, their accumulated extra loss in absolute value is bounded by some constant M which is independent of r.

Our definition (and subsequently algorithm) of fairness is based on the ones of the carpool scenario but is different. The main difference between the carpooling and the thermal comfort scenarios is that in the latter only one person is affected (the chosen driver) at the time of making a decision, whereas the rest of the people are benefited always and in the same way. In the thermal comfort scenario, users might have the same or similar preference, which would mean that selecting the temperature preferred by one user might also satisfy others. Therefore, straightforward algorithms (like round robin selection of the driver) or the greedy algorithm presented by Fagin and Williams [35] would not satisfy fairness in our context.

Under our definition of fairness it is possible to show that the traditional methods to aggregate participant thermal comfort are not fair.

Majority selection is not fair. Assume the scenario for room 2059 in Table 5.1 where there are six participants P_1, P_2, P_3, P_4, P_5, and P_6 whose thermal comfort are -3, -3, -3, -3, -3, and 3, respectively. In this situation, selecting the value supported by the majority would make $TC_0[P[2059]] = -3$, the real loss per user will be 0, 0, 0, 0, 0, and 6 and the ideal loss per user will be 1. Thus, the extra loss will be -1, -1, -1, -1, -1, and 5. If this exact situation keeps happening for r rounds, the extra loss for participants P_1, P_2, P_3, P_4, and P_5 will be $-1r$ and for participant P_6 will be $5r$, which are not bounded by any fixed number M (as r gets large).

Mean selection is not fair. Let's consider the same scenario described before. In that case, the mean value would make $TC_0[P[2059]] = -2$, the real loss will be 1, 1, 1, 1, 1, and 5, the ideal loss per user will be 5/3, the extra loss will be $-2/3$, $-2/3$, $-2/3$, $-2/3$, $-2/3$, and 10/3. Therefore, for r iterations the extra loss is again not bounded.

Trimmed mean selection is not fair. We can find an scenario similar to the previous for which this method is not fair regardless of the percentage removed from the smallest and largest user thermal comforts. For instance, using 20% (the value used in our experiments in Section 5.5) and the same scenario used for majority, the value selected will make $TC_0[P[2059]] = -3$. Therefore, the extra loss is the same than in the majority example and not bounded for r rounds.

Median selection is not fair. As in the case of the trimmed mean, median will make $TC_0[P[2059]] = -3$ and thus the extra loss of the participants is not bounded for r rounds.

5.4 A Fair Aggregation Algorithm

In this section, we present our algorithm to calculate the thermal comfort of a zone using the thermal comfort of the participants within. Then, we prove that our algorithm is fair according to the definition of fairness explained in the previous section.

As explained earlier, both the thermal comfort of the participants and the zones take values in the range $[-3, 3]$. The goal of our algorithm is to evaluate from these 7 possible values which one(s) will minimize the aggregated loss of the participants defined in Equation 5.3. Therefore, for each of the 7 possible values (v) of the thermal comfort of the zone, we compute the aggregated extra loss of each participant as $AEL_r^{(v)}[P_i] = EL_r^{(v)}[P_i] + AEL_{r-1}[P_i]$. We, then, take the $Lmax$ distance of the vectors $< AEL_r^{(v)}[P_1], ..., AEL_r^{(v)}[P_N] >$ and the vector $< 0, 0, .., 0 >$ for each value (v). Then, we choose the value (v) that minimizes the $Lmax$ distance.

Let's show how the algorithm will work for one round of our running example of Table 5.1 and the room 2059. First, our algorithm generates Table 5.3 which contains the information used to determine the zone thermal comfort at round r. Notice that $AEL_0[P_i] = 0$ for all the participants as this is the first round ($r = 1$). Then, each block in the table contains the values for the loss ($L_1^{(v)}[P_i]$), extra loss ($EL_1^{(v)}[P_i]$), and accumulated extra loss ($AEL_1^{(v)}[P_i]$) for all the possible values of (v). For example, $(v) = (-3)$ generates a loss of 6 (the maximum loss possible) for every participant whose thermal comfort is 3 ($TC_1[P_i] = 3, i = 1, 2, 3, 4, 5$), whereas it generates no loss for the participant whose thermal comfort is -3 ($TC_1[P_6] = -3$). This means that the ideal loss for $(v) = (-3)$ is $\frac{6+6+6+6+6+0}{6} = 5$. So, participants with $TC_1[P_i] = 3, i = 1, 2, 3, 4, 5$ have an extra loss of $EL_1^{(-3)}[P_i] = 1, i = 1, 2, 3, 4, 5$ and the participant with $TC_1[P_6] = -3$ has an extra loss of $EL_1^{(-3)}[P_6] = -5$. Finally, as $AEL_0[P_i] = 0$ then $AEL_1^{(v)}[P_i] = EL_1^{(v)}[P_i]$.

	P_1	P_2	P_3	P_4	P_5	P_6
$TC_1[P_i]$	3	3	3	3	3	-3
$AEL_0[P_i]$	0	0	0	0	0	0
$L_1^{(-3)}[P_i]$	6	6	6	6	6	0
$EL_1^{(-3)}[P_i]$	1	1	1	1	1	-5
$AEL_1^{(-3)}[P_i]$	1	1	1	1	1	-5
$L_1^{(-2)}[P_i]$	5	5	5	5	5	1
$EL_1^{(-2)}[P_i]$	2/3	2/3	2/3	2/3	2/3	-10/3
$AEL_1^{(-2)}[P_i]$	2/3	2/3	2/3	2/3	2/3	-10/3
$L_1^{(-1)}[P_i]$	4	4	4	4	4	2
$EL_1^{(-1)}[P_i]$	1/3	1/3	1/3	1/3	1/3	-5/3
$AEL_1^{(-1)}[P_i]$	1/3	1/3	1/3	1/3	1/3	-5/3
$L_1^{(0)}[P_i]$	3	3	3	3	3	3
$EL_1^{(0)}[P_i]$	0	0	0	0	0	0
$AEL_1^{(0)}[P_i]$	0	0	0	0	0	0
$L_1^{(1)}[P_i]$	2	2	2	2	2	4
$EL_1^{(1)}[P_i]$	-1/3	-1/3	-1/3	-1/3	-1/3	5/3
$AEL_1^{(1)}[P_i]$	-1/3	-1/3	-1/3	-1/3	-1/3	5/3
$L_1^{(2)}[P_i]$	1	1	1	1	1	5
$EL_1^{(2)}[P_i]$	-2/3	-2/3	-2/3	-2/3	-2/3	10/3
$AEL_1^{(2)}[P_i]$	-2/3	-2/3	-2/3	-2/3	-2/3	10/3
$L_1^{(3)}[P_i]$	0	0	0	0	0	6
$EL_1^{(3)}[P_i]$	-1	-1	-1	-1	-1	5
$AEL_1^{(3)}[P_i]$	-1	-1	-1	-1	-1	5

Table 5.3: Information generated by our algorithm for the running example.

With the information in Table 5.3, our algorithm computes the *Lmax* distance to be 5 for

$(v) = -3$ and $(v) = 3$ (corresponding to $AEL_1^{(-3)}[P_6]$ and $AEL_1^{(3)}[P_6]$, respectively), 10/3 for $(v) = -2$ and $(v) = 2$, 5/3 for $(v) = -1$ and $(v) = 1$, and 0 for $(v) = 0$. Then, the algorithm chooses $(v) = 0$ which minimizes the $Lmax$ distance. Notice that, in the next round $r = 2$, $AEL_1[P_i] = 0 \quad \forall \quad i$. This way, if the thermal comfort of the users remains the same, the values on Table 5.3 will be the same for $r = 2$. Thus, in this scenario, where majority and mean are not fair, our algorithm is fair as $AEL_r[P_i] = 0 \quad \forall \quad i, r$.

In the following, we prove that our algorithm is fair for any scenario, number of participants and rounds, under the fairness definition given in Section 5.3.2.

First, we will prove that three propositions that we will use to prove the fairness of our algorithm.

$L_r^{(v)}[P_i]$ is bounded by $[0, 6]$.

Proof. $L_r^{(v)}[P_i]$ represents the loss for participant P_i when the thermal comfort of the zone selected is (v) that is, the difference between the participant's thermal comfort $TC_r[P_i]$ and (v). As both $TC_r[P_i]$ and (v) take values ranging between $[-3, 3]$, $L_r^{(v)}[P_i]$ is bounded by $[0, 6]$. □

Theorem 5.1. *$AEL_r[P_i] < 7N$ for all i in N*

Proof. For that we prove it by contradiction. We assume that at time t it was the first time that a participant crossed the bound. Therefore,

$$AEL_t[P_i] > 7N \tag{5.4}$$

Because it was the first time, it means that in this round his thermal comfort was not chosen.

Because of our algorithm this can only happen if

$$AEL_{t-1}[P_i] <= AEL_{t-1}[P_j] for some j. \tag{5.5}$$

Also, we know that if you don't get chosen, your increment of AEL is at max 6. Therefore,

$$7N > AEL_{t-1}[P_i] > 7N - 7. \tag{5.6}$$

by (5.5) and (5.6), then

$$7N > AEL_{t-1}[P_j] >= AEL_{t-1}[P_i]. \tag{5.7}$$

Because $AEL_0[P_i] = 0$ for all i, both P_i and P_j had to increase their AEL at some point to reach their AEL_{t-1}.

a) Let's consider that both of them increased at the same time $t - i$. If that's the case, then at that round our algorithm picked another P_k with the highest AEL in $t - i - 1$. Therefore,

$$\begin{aligned} &AEL_{t-(i+1)}[P_k] > AEL_{t-(i+1)}[P_j] and \\ &\quad AEL_{t-(i+1)}[P_k] > AEL_{t-(i+1)}[P_i] \end{aligned} \tag{5.8}$$

In that situation, both P_i and P_k increased, but the max increase is 6 and by (5.6) and (5.7),

$$AEL_{t-i}[P_i] > AEL_{t-(i+1)}[P_i] > AEL_{t-i}[P_i] - 7 \tag{5.9}$$

or by (5.6)

$$AEL_{t-i}[P_i] > AEL_{t-(i+1)}[P_i] > 7N - 7 - 7 \tag{5.10}$$

also

$$AEL_{t-i}[P_j] > AEL_{t-(i+1)}[P_j] > AEL_{t-i}[P_j] - 6 \tag{5.11}$$

or by (5.7) and (5.6)

$$AEL_{t-i}[P_j] > AEL_{t-(i+1)}[P_j] >= 7N - 7 - 7 \tag{5.12}$$

by (5.8), (5.9) and (5.10),

$$AEL_{t-i}[P_j] <= AEL_{t-(i+1)}[P_k] >= 7N - 7 - 7 \tag{5.13}$$

Similarly to the previous situation, because $AEL_0[P] = 0$, P_k had to increase his AEL at some point to reach his $AEL_{t-(i+1)}$. At that point, someone else P_z should had been chosen, and so, he had the highest $AEL_{t-(i+2)}$. Then, following the same logic,

$$\begin{aligned} AEL_{t-(i+1)}[P_i] > AEL_{t-(i+2)}[P_i] > 7N - 7 - 7 - 7 \\ AEL_{t-(i+1)}[P_j] > AEL_{t-(i+2)}[P_j] >= 7N - 7 - 7 - 7 \\ AEL_{t-(i+1)}[P_k] <= AEL_{t-(i+1)}[P_z] >= 7N - 7 - 7 - 7 \end{aligned} \tag{5.14}$$

If we apply it recursively we will reach P_n which is the last participant at time t' and will have

$$AEL_{t'}[P_n] >= 7(N - (N - 1)) \tag{5.15}$$

Which means that at that point all of them will be greater than $7(N - (N - 1))$ and then contradict the checksum property.

□

5.5 Experimental Evaluation

In this section, we first present the simulator tool developed to perform the experiments. Then, we explain the experimental set up with the scenarios. Finally, we show the results of the experiments to compare traditional aggregating methods and our approach in terms of fairness and comfort.

5.5.1 Simulator Tool

We have developed a simulator to test various building occupants scenarios. The simulator takes input parameters such as a set of zones, a set of participants with their corresponding thermal profiles (e.g. hot/neutral/cold preferred) and movement between rooms, a set of initial temperatures per room, simulation period, etc. Also, it enables the user to select the algorithms to use to compute thermal comfort of the rooms (it supports the five algorithms considered along the chapter –majority, mean, trimmed mean, median, and our algorithm–). Also, we included a variant of our algorithm which bases the selection of the participant whose thermal comfort should be selected at a given round on the *Lmax* distance of the accumulated extra loses of the participants. We included this algorithm because it presents an interesting trade off fairness vs. comfort. Nevertheless, such algorithm is not fair according to our definition as we have found a pathological situation where the accumulated extra loss is unbounded[2].

Once the settings have been defined, the tool starts the simulation of the movement of people through the different rooms and periodically uses the defined algorithm to compute the thermal comfort of the zone/room based on the thermal comfort of the participants in it. Finally, the tool simulates a change of temperature in such rooms according to the computed thermal comfort.

The simulator is composed of four main modules:

- Participant manager, which generates participants and their thermal comfort based on the profiles and ratio given as input (e.g., if the input ratio is 50:30:20, 50% of cold, 30% of neutral, and 20% of hot preferred participants will be generated). Also, it manages the movement of the participants according to the input trajectories.
- Zone thermal comfort manager, which computes the thermal comfort for each zone using the thermal comfort of the participants in the room and the selected algorithms.
- Fairness and thermal comfort calculator, which computes the accumulated loss and discomfort of each participant at each round.
- Temperature manager, which simulates a change of temperature according to the computed thermal comfort for each zone. HVAC control is out of the scope of the chapter but we implemented a simple mechanism in the simulator that changes the temperature by 1^oC every 30, 45, and 60 minutes if the computed thermal comfort is -3/3, -2/2, and -1/1, respectively.

After each simulation, the tool outputs CSV files with the results for each algorithm in terms of accumulated loss and discomfort per user and per round.

[2] Due to the space restrictions we included more information about such situation and the complete algorithm at `tippersweb.ics.uci.edu/fairness`.

5.5.2 Experimental Setup

We used the simulator to empirically test the behavior of our approach compared with traditional aggregating methods. We tested two scenarios:

1. A simulation of a weekly group meeting with regular participants. We assume that 10 people meet in the same room every week at 8am. Meeting lasts for about 30 minutes and the comfort of the room is computed at the beginning of the meeting. The scenario tries to simulate short meetings, which are common in buildings like the one described in our experiments. Due to the time needed to implement the temperature changes, we compute the comfort at the beginning in order to change the temperature for the duration of the whole meeting. Also, we introduce some randomness for each meeting guaranteeing that at least 4 of the participants vote in each round.

2. A complete building with hundreds of participants. The TIPPERS [79] dataset consists of a trace of distinct devices connected to 64 Wi-Fi access points located in the 6 story Bren Hall building at UC Irvine collected over a 20 month period from January 2016 to September 2017. We used a subset of the TIPPERS dataset for the period of May 29th to June 2nd 2017 from 8am to 8pm. To filter out passerby devices (some of the WiFi AP ranges cover a few meters outside of the building), we discarded MAC addresses that had less than 50 events registered in the log for the period. After the filter, we obtained 18,837 events and 1065 unique MAC addresses. WiFi AP ranges cover several rooms in most of the cases so, for simplicity, we considered each WiFi AP as a zone. Therefore, all the MAC addresses connected to a WiFi AP at a given time represent participants in the same zone.

We defined three types of thermal profiles for the two scenarios: hot preferred, neutral, and cold preferred (see Figure 5.2). These profiles are based on the temperature in the zone.

For example, as Figure 5.2 shows, for a room temperature of 23^oC the thermal comfort of a hot preferred, neutral, and cold preferred users will be -1 (slightly cold in the PMV scale), 0 (neutral), and 2 (warm), respectively. For each scenario, we performed two tests with two different thermal profile settings. One setting represents a scenario with a clear majority of thermal profiles, where 90% of people prefer cold temperature and 10% of people prefer hot temperature (in the following we refer to it as 90/10). The second setting represents a balanced scenario where 50% of people prefer cold temperature and 50% of people prefer hot temperature (50/50).

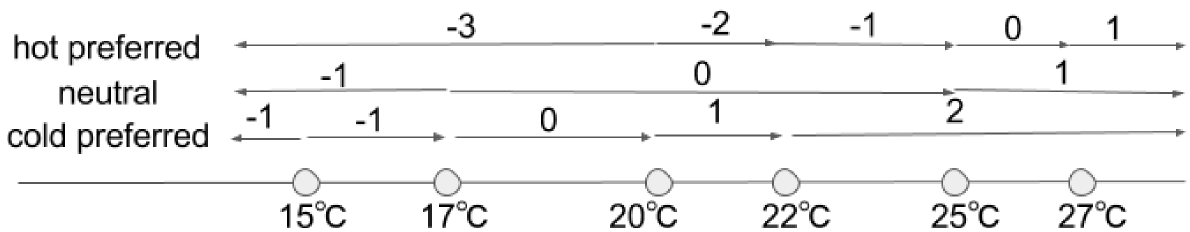


Figure 5.2: Thermal comfort for different thermal profiles in our experiments.

5.5.3 Results for the Meeting Simulation

In the following, we show the results of fairness and comfort achieved by each method for the first scenario.

Fairness

Figure 5.3 shows a comparison of the fairness of the different algorithms in terms of accumulated loss per round. For each algorithm we show two graphs (for the 90/10 and 50/50 settings) and a measure of the average accumulated loss per participant (represented by the dots in the graph) and the standard deviation from it. Notice that for each graph, the value at the last round represents the final accumulated loss at the end of the simulation.

The first implication of the results is that the accumulated loss for the majority (Figure 5.3(a)), mean (Figure 5.3(b)), trimmed mean (Figure 5.3(c)), and median (Figure 5.3(d))

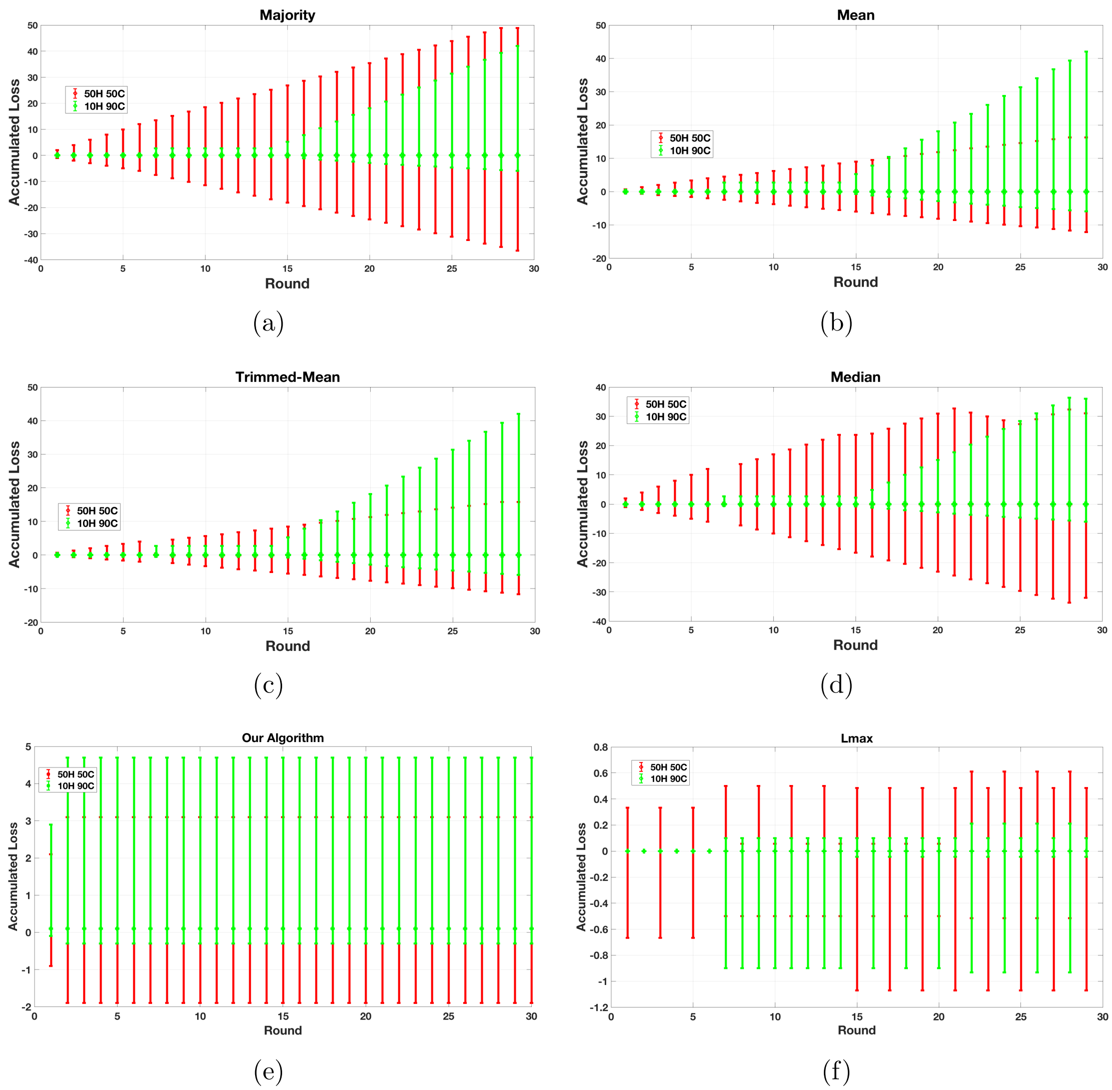


Figure 5.3: Comparison of fairness for the first scenario in terms of accumulated loss for traditional methods: majority (a), mean (b), trimmed mean (c), and median (d), our approach (e), and a variant (f).

are not bounded. The different graphs show that the accumulated loss keeps increasing for each round for both settings. It is interesting to notice that the behavior is similar for both settings even when one of them represents a clear majority vs minority situation. This is caused by the randomness introduced in terms of the participants at the meeting at each round. This means that even in the 50/50 setting, there might be rounds with a clear majority. If we focus on the results for our algorithm (see Figure 5.3(e)), we can see that, in contrast with the other algorithms, it keeps the accumulated loss bounded as expected.

Additionally, Figure 5.3(f) shows the variant of our algorithm based on *Lmax* distance which also keeps the loss bounded in this test (although it is not bounded in general as commented before).

Thermal Comfort

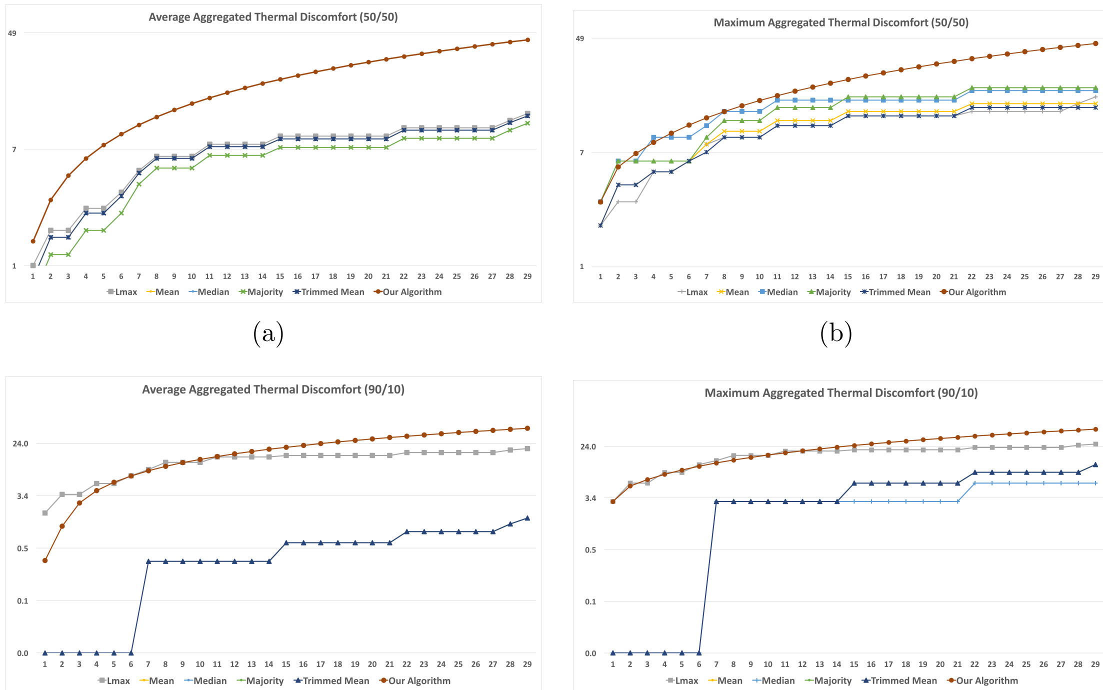
Figure 5.4 shows the comparison of the algorithms in terms of comfort (measured as aggregated thermal discomfort of the users per round). Figure 5.4(a) shows the comparison in terms of average discomfort per user for the setting 50/50 and we can see that as expected, the majority algorithm achieves the best comfort as it selects the most comfortable temperature for the majority of the participants. Our algorithm, increases the average discomfort in comparison with the traditional approaches. This happens because by being fair, sometimes it selects the thermal comfort of the minority and the majority of users thus, feel uncomfortable. In terms of the maximum value of discomfort (see Figure 5.4(b)), our algorithms achieve the worst comparing to the majority and median algorithms. For the 90/10 setting, our algorithm increases the discomfort greatly in comparison with others because every time the comfort of the minority gets selected, most of the participants feel uncomfortable. Interestingly, the variant of our algorithm using *Lmax* behaves similarly to the rest of algorithms in terms of comfort for the 50/50 scenario.

5.5.4 Results for the Building Simulation

In the following, we show the results for the second scenario.

Fairness

Due to the complexity of the scenario and the high number of participants in it (1065), we show test results achieved in terms of accumulated loss at the final round of the simulation. Table 5.4 shows accumulated loss for each method in both 90/10 and 50/50 settings. As in previous scenario, our algorithm achieves better results in terms of accumulated loss than



(a) (b)

(c) (d)

Figure 5.4: Comparison of comfort for the first scenario in terms of aggregated thermal comfort.

most of the traditional approaches. Notice that for some specific cases, our algorithm obtains a slightly higher value than the mean/trimmed mean approach. We analyzed the data and discovered that the occupancy data extracted from the real building shows a situation where only in very specific moments the zones contain a significant amount of participants (e.g., more than 4) and in most of the rounds, most of the participants are in one or two zones (remember that the zones correspond to WiFi APs). This corresponds to areas of high concentration of people in the building and to specific events.

Scenario	Loss	Majority	Mean	Median	Trim. Mean	Our Algo	Lmax Algo
90/10	Max	7.5	7.5	7.5	2.5	3	0.5
90/10	Min	-3	-1.5	-3	-1	-5.33	-0.83
50/50	Max	8.5	2.83	6	2.83	3	0.83
50/50	Min	-6.5	-2.16	6	-2.16	-5.33	-0.66

Table 5.4: Accumulated loss at the last round for each algorithm in the building scenario.

Thermal Comfort

In terms of comfort, we again looked at the accumulated discomfort at the final round of the simulation. Comparing the average discomfort per user, the results show that for the 50/50 setting, our algorithm achieves close discomfort compared to the majority algorithm which obtains the lowest discomfort (0.25 vs 0.2). In case of the 90/10 setting, the difference is 0.05, which is similar to the previous scenario (0.11 vs 0.06). Notice that as we pointed out in the previous scenario, the variants of our algorithm using $Lmax$ keep the discomfort similar to the rest of algorithms. For the 50/50 setting, $Lmax$ obtain an average discomfort of 0.48 and for the 90/10 setting, they obtain an average discomfort 0.8.

In summary, the experiments of the two scenarios and the different settings show that, as expected, our algorithm obtains the lowest accumulated loss per user while increasing the accumulated discomfort. An interesting conclusion of the experiments is that the $Lmax$ variant of our algorithm (which is not fair under our definition) maintains a low accumulated loss while maintaining the discomfort in similar levels than the traditional algorithms. Therefore, it seems that this variant could present a good trade off between fairness and comfort. However, it is not straightforward to design an algorithm that can be proven fair for any scenario while including the goal of maximizing the comfort of the participants.

Chapter 6

SCARF: A Scalable Data Management Framework for Context-Aware Applications in Smart Environments

6.1 Background on Context-Awareness

We are living in the era of smart environments where we are surrounded by billions of devices collecting contextual information from us. The rapid growth of sensors has revolutionized smart environments such as smart homes, smart shopping malls, and intelligent office buildings. With ever increasing sensor deployments, understanding "context" and subsequently making sense of environment, situation, sensors, along with providing real time responses have become fundamental research challenges [100].

Context-aware applications play an important role in smart systems, starting from smart hospital [18, 45, 26, 102], smart surveillance [83, 85], smart community [17, 105], smart notification [84, 106, 107, 53], smart shopping [74, 38], to smart building [96, 43] applications. Let us consider the following examples of context-aware applications in smart environments. In a smart shopping mall, when a user steps in, the system recognizes the user based on the cameras installed in the mall. The system retrieves previously purchased items by the user and creates a recommendation list of items for the user to purchase. Suppose a user with a health problem, such as allergies, enters into a smart building. The system detects the current level of pollen in the air at the user's location and sends a warning to the user that the pollen level inside that location is very high. Based on this warning, the user takes out or buys an asthma pump and/or avoid that particular location inside the building.

Context acquired from user devices is often very low-level for the end applications to interpret. Such information must be enriched/tagged appropriately before it can be effectively used by the applications. Advancement of data processing techniques, such as machine learning and crowd-powered techniques, have led to a variety of sophisticated analysis algorithms being developed in past few years specifically targeting Internet of Things (IoT) systems. However, these techniques can not be applied to the collected context data on the resource constrained mobile devices due to high resource requirements of these algorithms.

Systems that support context-aware applications face with several interesting challenges. Some of these challenges are as follows:

- **Scalability.** Scalability is an important issue while supporting large number of context-aware applications. Several architectures have been proposed in the past to address the scalability issues such as processing speed-up, reduced communication bandwidth, and increased reliability [6]. Among different programming paradigms of context-aware systems, service-oriented and agent-oriented [72] programming model have addressed the scalability issues up to some extent but they do not consider the scalability of context collection and evaluation from a large number of heterogeneous sources.

- **Latency.** The context-aware applications need to provide real time/near real time responses to the end users. In this scenario, latency becomes a significant issue while detecting/extracting contextual features from collected context data [80, 76, 75].

- **Energy Consumption.** Most of the devices in smart environments are battery powered. Energy consumption becomes a critical issue as context-aware applications consume large amount of energy from user's devices. Several authors [93, 108, 28, 94] have pointed out this issue for context-aware applications.

- **Security.** Some of the major security concerns of context-aware applications include context data access by malicious users, gaining access to user's mobile sensors by attacker and inferring sensitive user information from the output of these applications. Systems that support context-aware applications need to provide security guarantees against these types of attackers.

- **Privacy.** Context-aware applications can disclose behavior and social phenomena such as a user's location, their friends, associates, the timing of their interactions which can lead to serious privacy concerns [60].

In this chapter, we focus on the scalability challenge, while delivering a message based on a combination of user context such as location, activity and environmental context such as temperature, occupancy in smart buildings: in particular, achieving minimum latency and ensuring a bounded amount of energy consumption. We propose a context collection/management framework entitled SCARF which resides between the sensors and the applications to support the needs of the context-aware applications. In particular, SCARF takes a set of contextual queries from applications and optimizes the rate at which the context data needs to be collected and/or transmitted to the server in order to minimize the latency. Furthermore, SCARF decides whether the code for extracting contextual features should be executed on the server or on the edge. If the code is executed on the edge, it results in reduced communication between the server and the edge. However, it can result in low performance due to low resource availability of the edge. On the other hand, if it is executed on the server, the context data needs to be transmitted from the sensor to the server. However, it can result in better performance due to high resource availability of the server.

The proposed framework SCARF is an adaptive framework which changes the acquisition plan and contextual feature extraction plan based on several factors. It considers the dynamic state of sensors (e.g., sensor acquisition cost can change based on the battery level, network connectivity, optimizations performed by the underlying OS etc.), cost of network transfer at different points of time, list of contextual features that are concurrently being evaluated by the applications, to determine an optimized plan. We use SCARF to develop a novel context-aware messaging application called *Hedwig* that allows users to send and receive contextualized messages.

In summary, our contribution in this section is as follows:

1. We introduce a framework named SCARF, an efficient and scalable framework for collecting and managing context data from a variety of mobile and in-situ sensors for context-aware applications.

2. We have developed a novel context-aware messaging application named Hedwig where the delivery of the message is dictated by the conditions on complex contexts set by the senders and receivers.

3. We provide a survey of existing context-aware applications and identify the data management, energy management, and security issues of the context-aware applications.

4. We have experimentally evaluated SCARF using different types of real world contexts mimicking the needs of context-aware applications in smart environments.

The rest of the section is organized as follows. Section 6.2 introduces novel context-aware messaging application Hedwig. Section 6.3 describes how we model context in our framework SCARF. Section 6.4 describes the architecture of SCARF. Section 6.5 provides experimental evaluations of context in scalable manner.

6.2 *Hedwig*: Context-Aware Messaging Application

A *context-aware message* differs from a traditional message in a number of ways. Sender and receiver policies are specified to dictate the delivery of the message. Such policies are defined as conditions on the context. Sender policy specifies when or how the sender wishes the message to be delivered to the receiver. The receiver specifies the policies under which the receiver wishes to receive messages (e.g., after 5 p.m.). Context consists of a variety of observable phenomena such as time, location, activity (e.g. driving/walking/running/still), environmental conditions (e.g., increased allergens in an area, bad weather) and the message properties (e.g. sender, groups in which the sender belongs etc.).

Context-aware messaging is motivated by the scenarios listed below:

- Consider a scenario of emergency response where fire fighters are evacuating residents from a building during a major fire incident. One of the challenges during fire events is effective communication between the fire fighters. For instance, fire fighters need to communicate about the regions that they already evacuated, number of people still left to be rescued in a certain region, hazardous situations (e.g., potential for explosion, high air pressure, water/gas leakage, etc.) they may have observed at a particular region etc. Context-based messaging based on location, occupancy, room properties such as temperature, air pressure can be extremely beneficial in such scenarios. For instance, a context aware message may inform fire fighters about the number of people trapped at a location or the locations of water leakage and/or evacuation status of the region that they may be entering.
- Consider a scenario of an office space wherein co-workers send messages to each other

that are delivered based on context (e.g., invitation of lunch may only be sent to those people who are at their desks anytime before 12 noon, and whose status is "not busy"). The delivery context may use the GPS sensor to determine if a person is in the building, and furthermore, use a Wi-Fi access point connectivity data or BLE beacon data to determine if a person is on his/her desk. It may further view the calendar entry of the person to check their availability.

- Consider another scenario of a mobile application on user's mobile device that communicates with the infrastructure of a store (e.g., in-situ sensors) to determine context (e.g., the store aisle number where a particular product on the user's shopping list is shelved). Note that, although this scenario is similar to the newly added location based reminders supported in Google Home[1], the ability to use richer context information from a variety of sensors is a novel feature of *Hedwig*.

We have implemented *Hedwig* using a context data management system entitled *SCARF*. We explain the detailed architecture of SCARF in the following section.

6.3 Modeling Context in SCARF

We consider two types of context-aware applications: *user* and *environmental* context-aware applications. User and environmental context are defined as follows:

User Context. We define a user context as a context which consists of the properties of users inside a building. We denote a user context using the notation of U_i.

In the following, we provide examples of user context that can be specified in SCARF:

- $\langle user, start\_time, end\_time \rangle$

[1]https://www.slashgear.com/google-home-can-now-set-location-based-reminders-on-your-phone-14523229/

- $\langle user, location, start\_time, end\_time \rangle$

- $\langle user, activity, start\_time, end\_time \rangle$

- $\langle user, location, occupancy, start\_time, end\_time \rangle$

- $\langle user, location, activity, start\_time, end\_time \rangle$

Environmental Context. An environmental context is defined as any context which consists of a number of building properties such as temperature, occupancy, air volume, and water pressure. We denote an environmental context using the notation of B_i. Example environmental contexts can be as follows: ⟨ Temperature of room A is higher than 85° F⟩ , ⟨ Occupancy of rooms higher than 100 ⟩, and ⟨ List of rooms after 7 p.m. with occupancy higher than 5⟩. Such contexts can be specified by the building administrators to monitor the building resources such as rooms and sensors at different points of time of a day. Examples of environmental contexts that can be specified in SCARF:

- $\langle sensor, threshold, start\_time, end\_time \rangle$

- $\langle sensor, sensor\_health, rate, start\_time, end\_time \rangle$

- $\langle sensor, sensor\_anomaly, value, start\_time, end\_time \rangle$

Context Evaluation. Both environmental and user context can be derived from the data captured by the sensors that are part of the infrastructure. The sensors, in general, can be in-situ (such as temperature sensors and occupancy sensors) or mobile sensors such as mobile devices, cell phones, and smart watches that are carried by the user. Data from the sensors is collected based on the need of applications and such data is used to create contextual information relevant to the context-aware application. For instance, a context-aware message in Hedwig may specify that a user should be delivered a message when the user enters his/her

Notation	Definition
C_i	Context
s_i	An in-situ sensor
m_i	A mobile sensor
$\mathcal{E}_i$	Energy consumption cost of collecting one observation from m_i
$\mathcal{S}_i$	Data storage cost of storing one observation of m_i
$\mathcal{Y}_i$	Data transmission cost of transmitting observations of m_i
ν_i^a	Acquisition frequency of sensor m_i
ν_i^t	Transmission frequency of sensor m_i
$\mathcal{A}(m_i, \nu_i^a, \nu_i^t)$	Total acquisition cost of sensor m_i with acquisition frequency of ν_i^a and the transmission frequency of ν_i^t
f_i	Evaluation function
$w_s(f_i)$	Running time of the function f_i in the server
$w_m(f_i)$	Running time of the function f_i in the mobile
b_i	A bit vector signifying whether the function should be evaluated in the server or on the mobile

Table 6.1: Frequently used notations.

workspace. Sensor data, e.g., the GPS device on the mobile phone carried by the user and/or data about connectivity of the user's device to the in-situ WiFi access points can be used to determine the above context (viz., a user enters his/her office). Note that often context required by the end application may not be directly captured by the sensor. Instead, such a context is evaluated using (a sequence of) possibly expensive functions on top of data from one or more sensors. To support context-aware applications, SCARF continuously computes the contextual features from raw sensor data at a given periodicity between the timestamps when the application needing the specific context is active.

Cost of Context Acquisition. We assume that the cost of context acquisition from a mobile sensor is significantly higher than an in-situ sensor. We therefore focus on the data acquisition cost from the mobile sensors in the following part of the section. The cost of context acquisition from a mobile sensor m_i is dependent on the following factors: energy consumption in capturing data for a given sensor associated with the mobile device, storage cost of the collected data and the energy consumption cost of transmitting the data to the server.

We denote the energy consumption cost of one probe from a mobile sensor by the notation of $\mathcal{E}$. For example, the cost of a sensor m_i when it is probed once is denoted by $\mathcal{E}_i$. We associate an **acquisition frequency** (denoted by ν_i^a) with each of the mobile sensors. This frequency signifies the number of times the sensor is probed per second. For example, if the acquisition frequency of a GPS sensor is set to five then the GPS sensor is probed five times per second. Similarly, we denote the storage cost by the notation of $\mathcal{S}$ and the data transmission cost by the notation of $\mathcal{Y}$. In the previous example, the storage cost of m_i is denoted by $\mathcal{S}_i$ and the data transmission cost is denoted by $\mathcal{Y}_i$. Both $\mathcal{S}_i$ and $\mathcal{Y}_i$ are measured in terms of energy consumption cost (battery level drop) of the mobile device.

We associate a **transmission frequency** with each sensor and it is denoted by the notation of ν_i^t. This is defined as the number of times context data is transmitted from the mobile sensor to the server per second. For example, if $\nu_i^t = 2$, then the sensor data is transmitted from the mobile sensor to the server twice per second. This implies that the context data is collected for a duration of $1/2$ seconds $= 0.5$ seconds and then transmitted to the server. On the other hand, if $\nu_i^t = 0.2$ then sensor data is collected for five seconds and transmitted to the server at the end of each five seconds time interval.

Definition 6.1. ***Context Acquisition Cost.*** *Suppose a mobile sensor m_i is collecting context data with an acquisition frequency of ν_i^a and a transmission frequency of ν_i^t between time t_0 and t_1. Then the total cost of context acquisition is as follows.*

$$\mathcal{A}(m_i, \nu_i^a, \nu_i^t) = (t_1 - t_0)\nu_i^a \mathcal{E}_i + \mathcal{S}_i \frac{\nu_i^a}{\nu_i^t} + (t_1 - t_0)\nu_i^t \mathcal{Y}_i(\frac{\nu_i^a}{\nu_i^t}) \tag{6.1}$$

where $\mathcal{E}_i$ is the energy consumption cost of one probe from m_i, $\mathcal{S}_i$ is the energy consumption *cost of storing one record of m_i, and $\mathcal{Y}_i(n)$ is the energy consumption cost of transmitting n records from the mobile sensor to the server.*

Evaluation Function. Input to an evaluation function is a list of context records (e.g. GPS coordinates) between time t_0 and t_1. Output of the function is a boolean value signifying whether the context condition (e.g. a user is in the building) is true or not. For example, a sample evaluation function for GPS data can be as follows: given a list of GPS locations of a user and a target GPS location, a function which determines the geodesic distance between the list and the target location and outputs *true* if the distance is below a certain threshold. An evaluation function is denoted by the notation of f_i.

Cost of Context Evaluation. The cost of an evaluation function f_i is defined as the average execution time of the function on a single tuple of the context data. Each evaluation function has two costs: *server-running cost* and *mobile-running cost.* Server running cost, denoted by $w_s(f_i)$, refers to the execution time of the function on a single tuple on the server. Mobile running cost, denoted by $w_m(f_i)$ refers to the execution time on the mobile device.

Latency. We define the latency as the delay in detecting an event by the system from the time of occurrence of the event. Let us consider an example context ⟨Location(P_1,R_1) = true⟩. Suppose user P_1 reached location R_1 at 10:00:00 a.m. and SCARF detected the event at 10:02:00 a.m., then the latency will be two minutes.

Definition 6.2. ***Plan.*** *Let $\mathcal{M} = \{m_1, m_2, ..., m_n\}$ be a set of mobile sensors from which context data needs to be collected and evaluated. A plan consists of the following items.*

- *List of acquisition frequency values: $\nu^a = \langle\ \nu^a_1,\ \nu^a_2,\ \cdots,\ \nu^a_n\ \rangle$.*
- *List of transmission frequency values*[2] *: $\nu^t = \langle \nu^t_1, \nu^t_2, \cdots, \nu^t_n \rangle$.*

[2]We will henceforth assume that all ν^t_i values are equal to each other and we denote it by ν^t_0. The reason

- *List of function evaluation costs on the server:* $w_s = \langle\ w_s(f_1),\ w_s(f_2),\ \cdots,\ w_s(f_n)\ \rangle$.

- *List of function evaluation costs on the mobile:* $w_m = \langle\ w_m(f_1),\ w_m(f_2),\ \cdots,\ w_m(f_n)\ \rangle$.

- *A bit vector signifying whether the function is evaluated on the server side or on the mobile side.* $B = \langle\ b_1,\ b_2,\ \cdots,\ b_n\ \rangle$. *If* $b_1 = 1$ *then the evaluation function is evaluated on the server side where as if* $b_1 = 0$ *then it is evaluated on the mobile side.*

We denote a plan by the notation of $P(\nu^a, \nu^t, B)$.

Definition 6.3. ***Latency of a Plan.*** *Given a plan defined as above, we define the latency of a plan* $P(\nu^a, \nu^t, B)$ *as follows:*

$$Latency(P) = \sum_{i=1}^{n} \frac{\nu_i^a}{\nu_0^t} \cdot (b_i \cdot w_s(f_i) + (1 - b_i) \cdot w_m(f_i)) \tag{6.2}$$

In the following we provide an example of a plan and the latency of the plan.

Example 6.1. *Let* m_1, m_2, m_3 *be three sensors from which context data needs to be collected and evaluated. Let us consider a plan where the acquisition frequency values are 2/sec, 3/sec and 4/sec respectively. Let the values of the transmission frequencies are as follows:* $\nu_1^t = \nu_2^t = \nu_3^t = 1/30$. *This implies that at the end of each 30 seconds time interval, collected context data is transmitted to the server. At the end of each thirty seconds time interval, the number of records of* m_1 *that needs to be evaluated are 30*2 = 60. Similarly, the number of records of* m_2 *and* m_3 *are 90 and 120 respectively.*

Suppose the cost of functions f_1, f_2, f_3 *when they are executed on the server are 0.01 sec, 0.02 sec and 0.03 seconds respectively. The cost of the functions* f_1, f_2, f_3 *on the mobile*

behind this choice is that we only evaluate the context data at the end of each transmission cycle. If we transmit context data earlier and not evaluate it until the end of a transmission cycle, it will always increase the latency.

are 0.04, 0.015 *and* 0.05 *respectively. Suppose the context data from* m_1 *and* m_3 *needs to be evaluated on the server and the data from* m_2 *needs to be evaluated on mobile, then the latency of the plan is calculated as follows: 60*0.01 + 90*0.015 + 120*0.03 = 5.5 seconds.*

Problem Statement. Let $\mathcal{M} = \{m_1, m_2, ..., m_n\}$ be a set of mobile sensors from which context data needs to be collected and evaluated. Let $\mathcal{E}^{max}$ be the maximum cost (in terms of energy consumption) that can be incurred for the context data acquisition and evaluation from $\mathcal{M}$. Our objective is to generate a plan which minimizes the latency given that the total cost of context data acquisition, storage and transmission from all the sensors is bounded by $\mathcal{E}_i^{max}$. Our optimization problem is formally defined as follows:

$$\begin{aligned} \underset{\nu^a, \nu^t, B}{\text{minimize}} \quad & Latency(P(\nu^a, \nu^t, B)) \\ \text{subject to} \quad & \sum_{i=1}^{n} (t_1 - t_0)\nu_i^a \mathcal{E}_i + \mathcal{S}_i \frac{\nu_i^a}{\nu_0^t} + (t_1 - t_0)\nu^t \mathcal{Y}_i(\frac{\nu_i^a}{\nu_0^t}) \leq \mathcal{E}^{max}, \\ & \nu_i^a \geq \nu^{min} \end{aligned} \tag{6.3}$$

where ν^{min} is the minimum frequency value which ensures that there is no missing event due to the acquisition of data being far apart on the time-line. We assume that if a context condition is true at a particular time instant, then it will remain true for at least a certain amount of time interval called $event^{min}$. If $event^{min}$ is two seconds, then we set the value of ν^{min} to be 0.5. This ensures that there exists at least one data point in every two seconds. Our goal is to choose the frequency values ν^a, ν^t and the bit vector B in such a way that minimizes the latency.

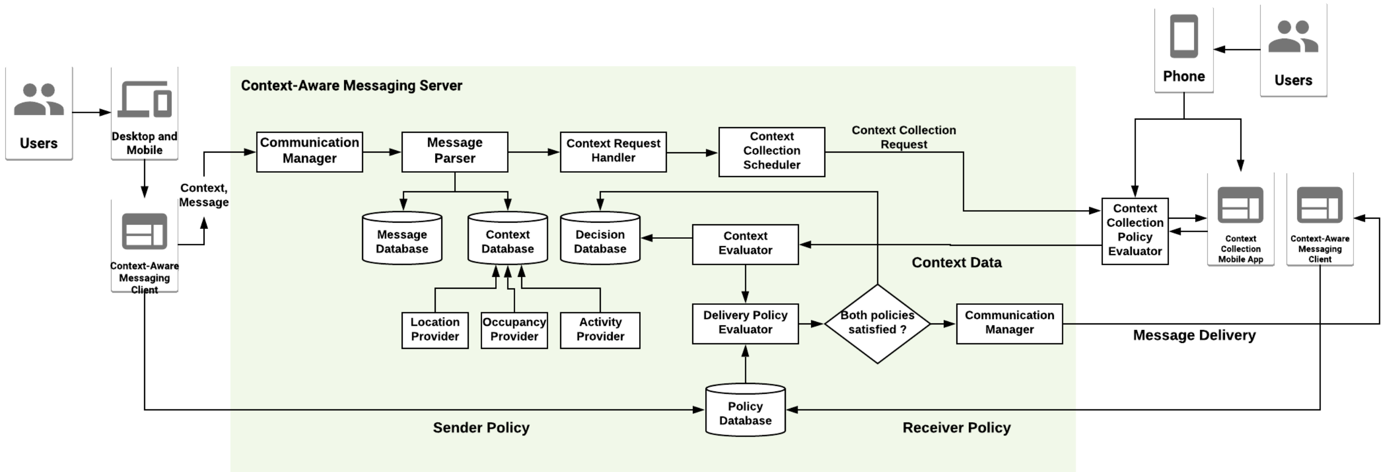


Figure 6.1: Diagram of the Server-based Architecture

6.4 Description of SCARF

6.4.1 Overview

SCARF divides the entire execution time into multiple smaller time intervals called epochs. In each epoch, we generate a plan as described in Definition 6.2 by solving the optimization problem shown in Equation 6.3. This plan is generated at the server. Depending on the outcome of the optimal solution (i.e., the bit vector B), the evaluation is either performed at the server or on the mobile. Each epoch consists of two phases: a plan generation phase and a plan execution phase. We assume that the cost of the sensors (i.e., $\mathcal{E}_i$) and the context evaluation costs (i.e., w_s and w_m) does not change within an epoch. Algorithm 1 presents a high-level overview of how SCARF works.

The input to SCARF is a list of mobile sensors, a list of context evaluation functions, a start time and end time between which the context has to be evaluated. The variables $totalExecutionTime$ and $curEpochTime$ maintain the overall execution time and the execution time within an epoch respectively. We maintain two hashmaps for storing the cost values of the mobile sensors w_s and w_m. The energy consumption cost $\mathcal{E}$ is updated at the end of each epoch. In line 8, we generate the plan using the updated cost values w_s and w_m.

Plan is executed until the epoch time is exhausted.

Algorithm 1 Adaptive Approach.

procedure ADAPTIVE APPROACH($\mathcal{M}$, $\{f_1, f_2, \dots f_n\}$, t_0, t_1)
  $totalExecutionTime \leftarrow 0$
  $contextResult \leftarrow False$
  INITIALIZE-DATA-STRUCTURE(w_s, w_m) // Initialize the data structures to store the running costs of functions on server and mobile
  while $totalExecutionTime < (t_1 - t_0)$ **do**
    $curEpochTime \leftarrow 0$
    $\mathcal{E} \leftarrow$ GET-UPDATED-COST($\mathcal{M}, f_1, f_2, \dots, f_n$)
    $\langle P, t_p \rangle \leftarrow$ GENERATE-PLAN($\mathcal{E}$, w_s, w_m, Δ) // t_p: plan-generation time
    while $curEpochTime < (epochTime - t_p)$ **do**
      $t_1 \leftarrow time()$
      $contextResult \leftarrow$ EXECUTE-PLAN(P)
      if $contextResult \leftarrow True$ **then**
        return True
      end if
      $t_2 \leftarrow time()$
      $curEpochTime \leftarrow curEpochTime + (t_2 - t_1)$
    end while
  end while
  return $False$
end procedure

In SCARF, we have implemented a hybrid architecture that leverages server-based centralized system with an edge-based distributed component. In the following, we describe the life-cycle of context data acquisition, context data storage and context data evaluation in the different components of SCARF.

As a first step, given a condition on context from an application, SCARF checks the sensor data from the in-situ sensors. If a particular context can be evaluated just by using the data from the in-situ sensors, then SCARF does not have to generate any plan for that particular context. If a particular context can not be evaluated by the data from in-situ sensors, then SCARF generates a plan for the mobile sensors according to Equation 6.3. Depending on the output of the bit vector B of the optimization problem, SCARF uses the following strategies. In a certain epoch, if $b_i = 1$ then SCARF collects the sensor data at the mobile device for a certain duration of time and then transmit it to the server. Server stores the data in a local database and runs the context evaluation function on that data. If $b_i = 0$, then the mobile device collects the sensor data, stores it in a local database and also executes the evaluation function on the collected data. The process of context evaluation on the server side is shown in Figure 6.1 whereas the process of context evaluation on the mobile side is shown in Figure 6.2.

6.4.2 Plan Generation

At the beginning of each epoch, SCARF performs this step. In this step, SCARF generates an acquisition plan of the sensors, transmission plan and an evaluation plan whether the context will be evaluated in the server side or on the mobile side. It solves the optimization problem as described in Equation 6.3 and determines the above plans. Note that, the constraint of the equation has following coefficients: $\mathcal{E}_i$, $\mathcal{S}_i$ and $\mathcal{Y}_i$ which need to be updated in every epoch.

The first step of plan generation phase of a certain epoch k, is to update the cost values $\mathcal{E}_i$, $\mathcal{S}_i$ and $\mathcal{Y}_i$ respectively. We determine the average energy consumption costs incurred in the previous epoch (i.e., epoch $k-1$) and set it as the value of $\mathcal{E}_i$ in epoch k. For example, suppose in a certain epoch $k-1$, the total energy consumed by 100 probes of a certain mobile sensor is 0.02. Then the value of $\mathcal{E}_i$ in epoch k will be as follows: $\mathcal{E}_i = 0.02/100 = 0.0002$. Similarly, let the values of $\mathcal{S}_i$ and $\mathcal{Y}_i$ are 0.0001 and 0.004 respectively.

Once the cost values are updated we solve the optimization problem as shown in Equation 6.3. In the above example, if we add the values of the $\mathcal{E}_i$, $\mathcal{S}_i$, $\mathcal{Y}_i$ in Equation 6.3, we get the following optimization problem:

$$
\begin{aligned}
\underset{\nu^a,\nu^t,B}{\text{minimize}} \quad & \sum_{i=1}^{n} \frac{\nu_i^a}{\nu_0^t} \cdot (b_i \cdot w_s(f_i) + (1-b_i) \cdot w_m(f_i)) \\
\text{subject to} \quad & (t_1 - t_0)\nu_i^a \cdot 2 \cdot 10^{-4} + \frac{\nu_i^a}{\nu_0^t} \cdot 10^{-4} \\
& + (t_1 - t_0)\nu_0^t \cdot 4 \cdot 10^{-3} \leq \mathcal{E}_i^{max} \\
& \nu_i^a \in R, \nu_0^t \in R, b_i \in \{0,1\} \\
& \nu^{max} \geq \nu_i^a \geq \nu^{min}, \nu^{max} \geq \nu_0^t \geq \nu^{min}
\end{aligned}
\tag{6.4}
$$

The frequency value ν^{max} denotes the maximum frequency of data collection that is supported by the operating system of the mobile device. The above optimization problem is a mixed integer non-linear programming problem where the values of ν_i^a and ν_0^t are real numbers and the value b_i is an integer which can take two values 0 or 1. Since the domains of ν_i^a and ν_0^t are bounded, we solve the above optimization problem numerically. We discretize the domain of ν_i^a and ν_0^t from the values of ν^{min} to ν^{max} using a width of 0.01. For each pairs, we check the constraint condition to find out if the pair is a feasible solution of the optimization problem. If the solution is feasible, then we calculate the objective function and store the value and the solution in a minimum priority queue. After iterating over the space of ν_i^a and ν_0^t, we retrieve the optimal solution from the priority queue.

6.4.3 Context Data Collection

Given a set of mobile sensors $m_1, m_2, ..., m_n$ from which context data have to be collected, SCARF generates an acquisition plan (i.e., ν_i^a) for each sensors. Note that the energy consumption cost of a sensor in a mobile device is dependent on several factors such as the operating system installed, underlying hardware, and type of the network to which the device is connected to.

In a certain epoch, if the context data needs to be evaluated on the server side, then

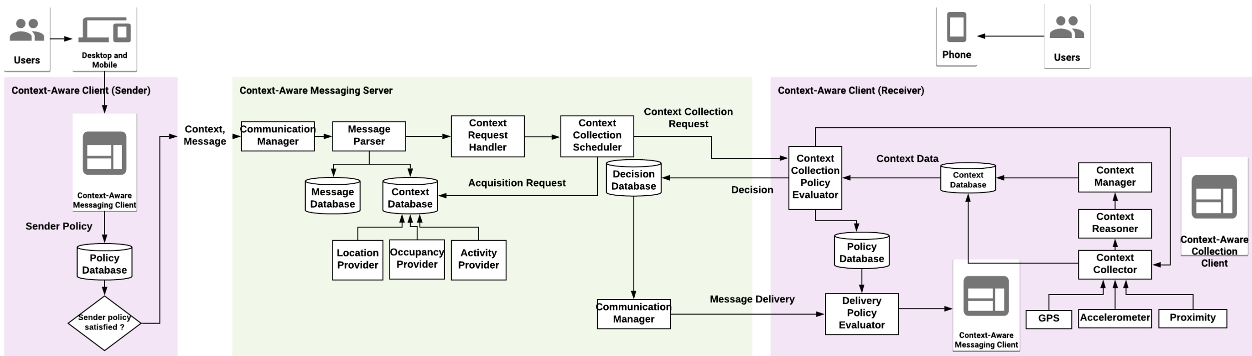


Figure 6.2: Diagram of Edge-based Architecture

SCARF uses a push-based mechanism to collect the data. The sensor data gets collected at the mobile device at a frequency of ν_i^a till the end of the epoch. At the end of the epoch, the mobile device pushes the collected data to the server side. Mobile device uses appropriate RESTful API to transmit the data to the server. The server stores the collected data in an internal database.

In an epoch, if the context data needs to be evaluated on the mobile side, then the data collection takes place on the mobile device itself. The sensor data is collected and stored at a frequency of ν_i^a at the mobile device till the end of the epoch. Since the context data is evaluated on the mobile device itself, the collected data is not transmitted from the mobile device to the server in this epoch. The context data is stored in a SQLite database of the mobile device.

6.4.4 Context Data Management

Context data collected is stored in a database called $ContextDB$. In the following, we explain how this database is managed both on the server-side as well as on the mobile side. Context Reasoner (as explained in the next section) queries this database and evaluate the required context. Query performance of ContextDB affects the latency of the end applications.

In the server-side, we maintain a centralized database, which stores the context information collected from all the mobile devices. The schema of the database is shown as follows:

- OBSERVATION (id INT PRIMARY KEY, sensor_id INT, payload VARCHAR(255), observation_type_id INT, timestamp datetime)
- USER (user_id INT PRIMARY KEY , name VARCHAR(255), email VARCHAR(255), office VARCHAR(255), mobile_number VARCHAR(20), preferred_communication VARCHAR(255))
- SENSOR (sensor_id INT PRIMARY KEY, description VARCHAR(255), name VARCHAR(255), platform_id INT)
- PLATFORM (platform_id INT PRIMARY KEY, description VARCHAR(255), name VARCHAR(255))
- PLATFORM_SENSOR (platform_id INT, sensor_id INT)

In the above schema, the OBSERVATION table contains the collected context data from the mobile sensors. In this table, sensor_id contains the id of the mobile sensor, observation_type_id contains the type of the context data such as GPS, accelerometer, gyroscope, light, etc. Payload attribute contains the actual value of the sensor and it is stored as a JSON object. For example, for a GPS sensor, a sample payload will be as follows: {"Latitude": "33.6431", "Longitude": "-117.84"}. The timestamp attribute captures the time at which the observation was collected by the mobile sensor. Context reasoner queries this table based on the sensor_id, timestamp and observation_type_id attributes. We created indexes on these attributes of OBSERVATION table for better query performances by the reasoner.

In the mobile-side implementation, the schema of the ContextDB is as follows.

- GPS (id INTEGER PRIMARY KEY, latitude FLOAT, longitude FLOAT, accuracy FLOAT, provider VARCHAR(255), timestamp DATETIME)

- ACCELEROMETER (id INTEGER PRIMARY KEY, x_coordinate FLOAT, y_coordinate FLOAT, z_coordinate FLOAT, timestamp DATETIME)
- GYROSCOPE (id INTEGER PRIMARY KEY, x FLOAT, y FLOAT, z FLOAT, timestamp DATETIME)
- LIGHT (id INTEGER PRIMARY KEY, light FLOAT, timestamp DATETIME)

In an epoch, if a context is evaluated on the mobile side, then after the context evaluation step, we remove all the records from the tables. Since mobile devices have a limited storage capacity, this ensures that none of the tables become very large. Furthermore, we create indexes on the timestamp attribute of the above tables for faster retrieval of the records based on the timestamp attribute.

6.4.5 Context Data Evaluation

The process of context data evaluation consists of multiple steps. In the first step, it fetches the required context data from ContextDB. The context data collected only in the current epoch is fetched from the database. For example, if the length of an epoch is one minute, then the data collected in last one minute is fetched from the ContextDB. In the following, we define the specifications of a context reasoner, and then we explain how the context evaluation takes place on both the server side and on the edge-side.

Context Reasoner. The input to a context reasoner is the context condition and a list of context data of a particular type. For example, if the context is location, then it takes an input as a set of GPS records. The output of the context reasoner is a boolean value of $True$ or $False$. If the context reasoner outputs true at any point, then it is not further evaluated in the future.

In the epochs, where the context data is evaluated on the server side, context reasoner fetches the data of the last epoch from the ContextDB and checks whether the context condition is satisfied or not. We deploy the reasoner at the same host where ContextDB is hosted to reduce the network I/O of fetching the data from the database to the server. Context Evaluator on the server is multi-threaded.

In an epoch, if the context data needs to be evaluated on the mobile side, the reasoner takes the following steps. It fetches the records from the local ContextDB and executes the reasoner on top of the fetched data. If the evaluation of the context becomes true, then it sends a *success* message to the server signifying that the context evaluation was successful and further context evaluations should be stopped.

Benefits and Limitations of the Server and Edge Based Approaches. In the server-based approach, we perform the context evaluation in the same server in which we maintain the Context DB. The benefits of the context evaluation step on the server side are as follows:

- SCARF can execute complex context reasoners such as machine learning based algorithms, complex rule-based algorithms on the central server. SCARF can use deep learning based algorithms which require special hardwares such as GPGPUs to execute.
- The server-side component does not have any limitations on the storage size. The size of the Context DB can potentially be very large without causing any performance issue.
- ContextDB in the server-side can be replicated appropriately to support context-aware applications during node failures.

The advantages of the edge-based component are as follows:

- Since ContextDB is maintained on the user's mobile devices, sensitive context data (such as location and activity) is very less prone to attack by the attackers.

- The context evaluation step is performed by the mobile device itself, which implies that the server only gets to receive the output of the context evaluation step. Malicious users on the server will not be able to perform any inference attacks on the context data.

6.5 Experimental Evaluation

In this section, we experimentally evaluate our framework using real world dataset. In the following, we first explain the experimental setups, then present experimental results.

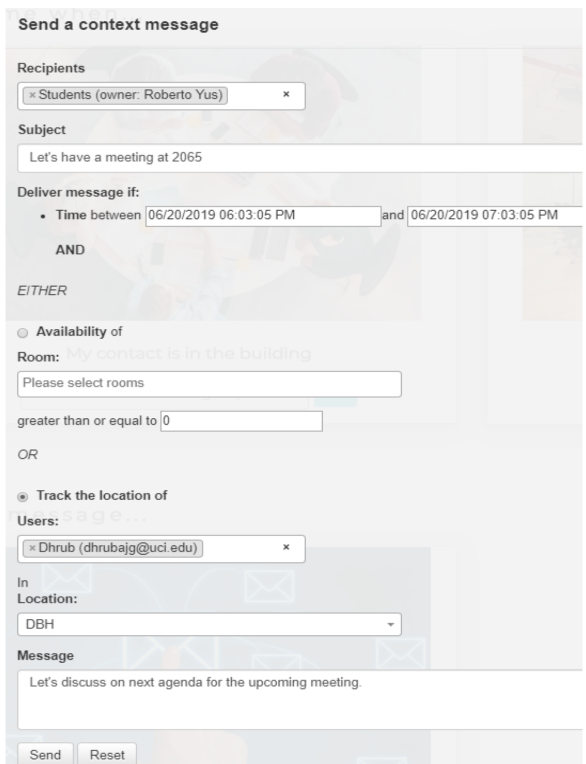


Figure 6.3: Figure illustrating the context-aware message application.

Implementation and Deployment Specifications. We implemented both the components (server and edge) of SCARF in Java. In the server-side implementation, we use the MySQL database for managing Context DB. In the edge-side implementation, we use SQLite database. Our server has the following specifications: Intel(R) Xeon(R) CPU E5-46400, 132

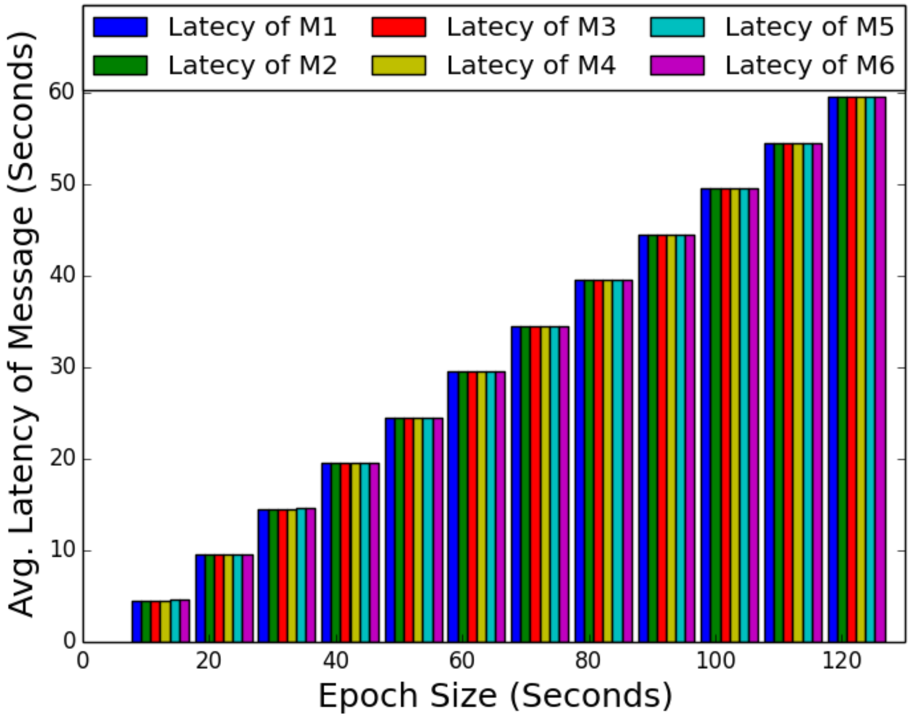


Figure 6.4: Figure illustrates latency of messages.

GB RAM, 80 TB of disk space and runs CentOS 7.1. Mobile devices have the following specifications: i) Motorola Moto X (2nd Generation): 2.5GHz Qualcomm Snapdragon 801 processor, 16 GB internal storage, 2GB RAM, 2300 mAh battery capacity, running Android 6.0 and ii) Samsung Galaxy 5: 2.1 GHz quad-core Cortex-A15 and 1.5 GHz quad-core Cortex-A7 CPU, internal storage of 16GB, 2 GB LPDDR3 RAM, 2800 mAh battery capacity, running Android 5.0.

Sensors. We consider five different types of mobile sensors: accelerometer, location (gps and network), gyroscope, light, and proximity. These data is stored on Android SQLite database as described in section 6.4.4. For accelerometer sensor, we store the accelerometer values along the directions of x-, y-, and z-axis. For GPS, we store latitude, longitude, accuracy, and timestamp values. For proximity sensors, we store the proximity values measured in centimetres. For light sensors, we store the values of light levels measured in terms of luminous flux (i.e., lux). For gyroscope sensors, we store the rate of rotation values along the directions of x-, y-, and z-axis. For in-situ sensors, we have used the sensors deployed as part of a live IoT testbed named TIPPERS [79]. This testbed is collecting real time sensor data from 64 WiFi access point sensors, 40 video cameras, 50 beacons and 6000 HVAC sensors located in our department building since January 2016.

Context-Aware Messages Used. We use the following context-aware messages to evaluate our framework.

1. $\mathcal{M}_1$: **If** $Location(P_1, \text{“}R_1\text{”}) = true$ **then** send a message.

2. $\mathcal{M}_2$: **If** $Activity(P_1, \text{“}still\text{”}) = true$ **then** send a message.

3. $\mathcal{M}_3$: **If** $Location(P_1, \text{“}R_1\text{”}) = true$ **AND** $Activity(P_1, \text{“}still\text{”}) = true$ **then** send a message.

4. $\mathcal{M}_4$: **If** $Occupancy(\text{“}R_1\text{”}) = 10$ **AND** $Location(P_1, \text{“}DBH\text{”}) = true$ **then** send a message.

5. $\mathcal{M}_5$: **If** $Temperature(\text{“}R_1\text{”}) > 75$ **AND** $Temperature(\text{“}R_1\text{”}) < 80$ **AND** $Occupancy(\text{“}R_1\text{”}) < Occupancy(\text{“}R_1\text{”}).capacity/3$ **then** send a message.

6. $\mathcal{M}_6$: **If** $Occupancy(\text{“}R_1\text{”}) < Occupancy(\text{“}R_1\text{”}).capacity/3$ **AND** $Location(P_1, \text{“}R_1\text{”}) = true$ **AND** $Temperature(\text{“}R_1\text{”}) > 75$ **AND** $Temperature(\text{“}R_1\text{”}) < 80$ **then** send a message.

In the above messages, first and second messages each contains a single user context. The third message consists of two user contexts and the fourth message consists of an environmental and a user context. The fifth message consists of two environmental contexts. The sixth message consists of two environmental contexts and one user context. Although we have chosen the above six representative contexts, it can be specified for any environmental contexts and user contexts. Figure 6.3 shows the interface for creating these context-aware messages.

6.5.1 Experimental Result.

Experiment 1. Latency of SCARF at Different Event Times. In our experiments we set the value of $event^{min}$ to be 120 seconds, i.e., when a particular context condition becomes true, it remains true for the duration of next 120 seconds. This implies that SCARF needs to set the transmission frequency to be at least once in 120 seconds in order to ensure that there is no missing event. Since latency is sensitive to the actual time of event occurrence, we measure the latency of SCARF for a given value of epoch size (i.e., $1/\nu_0^t$) as follows.

Let us consider that the epoch size for a certain message is 10 seconds, i.e., after every ten seconds, the context data is transmitted and evaluated. An event can take place at T_{event} = 1 second, 2 seconds, $\cdots$, or at 10 seconds. We measure the latency of SCARF in each of these ten possibilities and derive the average latency. We report this value as the latency of SCARF at the epoch size of 10 seconds. Similarly, we calculate the average latency for the other epoch sizes as shown in Figure 6.4. The acquisition frequency value for this experiment was set up at 10.

The results for all the messages are shown in Figure 6.4. From this table, we conclude that SCARF performs very well in terms of the latency of these messages across different event times. For $\mathcal{M}_1$, the average latency of SCARF varies from 4.5134 seconds to 59.5033 seconds depending on the time at which the context condition was true. Furthermore, we observe that the latency of the messages increase when the epoch size is high (i.e., transmission frequency is low) but at the same time the cost reduces. Depending on the value of $\mathcal{E}^{max}$, SCARF chooses appropriate transmission frequency.

Experiment 2. Latency Comparison with the Baseline Approaches. In this experiment, we measure the latency of SCARF with two baseline approaches. For baseline 1 approach, we perform the context acquisition, context storage, and context evaluation entirely on the mobile side. In baseline 2 approach, we perform all the three steps (i.e., context acquisition, context storage, and context evaluation) completely on the server side.

We choose different values of $\mathcal{E}^{max}$ for each message and compare the latency (measured in milliseconds) of our approach with the baseline approaches. $\mathcal{E}^{max}$ is the maximum energy consumption cost set up for each messages. For all the messages, the latency of SCARF is lower than both *Baseline 1* and *Baseline 2*. The latency values of different approaches for $\mathcal{M}_1$ are shown in Table 6.2 and a comparison plot is drawn in Figure 6.5. For $\mathcal{M}_1$ and $\mathcal{M}_2$ that utilizes user contexts such as Location and/or Activity, the latency of *Baseline 2* is higher than *Baseline 1*. The reason behind this observation is that in *Baseline 2*, the collected context data need to be transmitted to the server from the mobile devices. Since the message only contains user context that is collected at the mobile device, it is beneficial to perform the context evaluation on the mobile device. Hence the *Baseline 1* approach outperforms the *Baseline 2* approach. Similar trends follow for message $\mathcal{M}_3$ and $\mathcal{M}_4$. On the other hand, $\mathcal{M}_5$ which utilizes two environmental contexts such as Occupancy and Temperature, the latency of *Baseline 1* is higher than *Baseline 2* since the data required for both the environmental contexts are local to the server, the *Baseline 2* incurs less latency as compared to the *Baseline 1* approach. For $\mathcal{M}_6$ which utilizes two environmental contexts and one user context such as Occupancy, Temperature and Location, *Baseline 2* shows higher latency due to the transmission cost of sending GPS data from mobile to the server. The reason behind the performance gap of our approach and the baseline approaches is as follows: our approach chooses an optimal acquisition frequency, transmission frequency and bit vector B specifying if the context should be evaluated on the server or on the mobile side in order to minimize the latency.

Experiment 3. Energy Consumption Cost. In this experiment, we provide the details

$\mathcal{E}^{max}$ **(Battery Level Drop)**	**Baseline 1 (ms)**	**Baseline 2 (ms)**	**Our Approach (ms)**
24 mAh	341.66	378.33	287.33
47 mAh	251.66	258.66	200.66
70 mAh	186	231	152
93 mAh	162	234	146
117 mAh	87	147	82.66

Table 6.2: Average Latency comparison of $\mathcal{M}_1$

Sensor	**Device**	**Probe Cost**	**Transmission Cost**	**Evaluation Cost**
GPS	Device 1	4.6	0.092	0.76
Accelerometer	Device 1	0.023	0.46	0.012
Gyroscope	Device 1	0.383	0.18	0.0046
GPS	Device 2	1.6	0.168	0.93
Accelerometer	Device 2	0.028	0.056	0.014
Gyroscope	Device 2	0.56	0.168	0.0056

Table 6.3: Details of Cost in *Device 1* and *Device 2* measured in mAh.

of the energy consumption cost from different types of sensors. The results are shown in Table 6.3. For each type of sensors, we measure the probe cost of one sensor probe and report it as the probe cost in Table 6.3. We measure the cost of transmitting one record of a particular sensor type from the mobile device to the server and report it as the transmission cost. We performed this step for 10,000 iterations and measure the average transmission cost from the total cost. In the evaluation cost column, we provide the energy consumption cost of evaluating the context reasoner on a single record. From this experiment, we draw the following conclusions: the probe cost, transmission cost and evaluation cost vary depending on the type of the device, type of hardware, operating system and the type of network. Furthermore, this implies that the cost model of SCARF as shown in the constraint of Equation 6.3 is an accurate and practical model for measuring the cost of context acquisition, transmission, and evaluation.

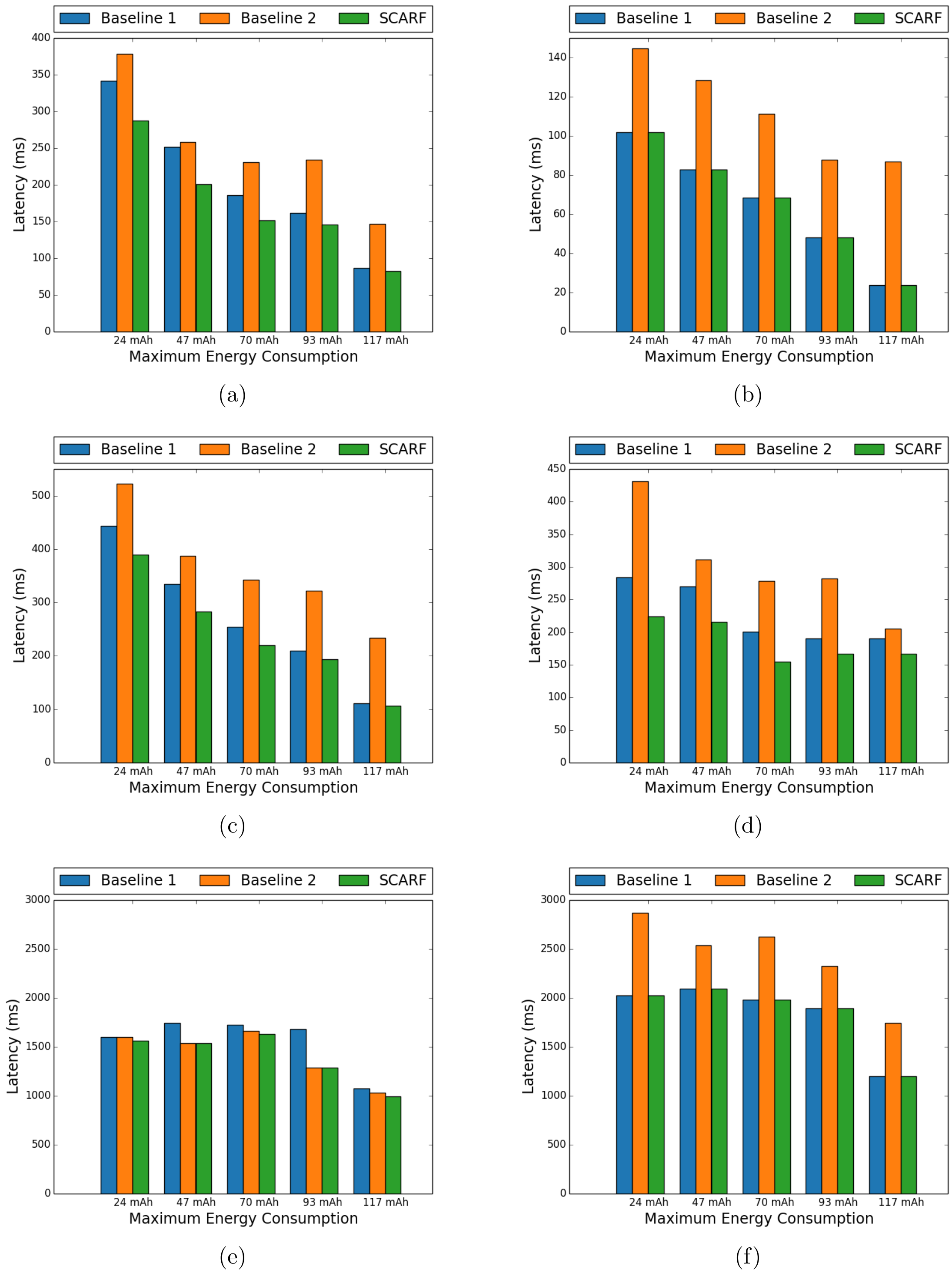


Figure 6.5: Comparison of SCARF with baseline approaches: $\mathcal{M}_1$ (a), $\mathcal{M}_2$ (b), $\mathcal{M}_3$ (c), $\mathcal{M}_4$ (d), $\mathcal{M}_5$ (e), and $\mathcal{M}_6$ (f).

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we have presented context reasoning challenges in IoT environments along with their solutions: feasible location attack from occupancy sensors in smart buildings, a fair decision making algorithm for participatory thermal comfort control systems, and a context-aware messaging framework for smart environments, $SCARF$.

In chapter 4, we studied privacy implications of occupancy sensors in smart buildings. We have proposed a new privacy breaching attack in which the adversary combines the gathered occupancy data with auxiliary information, and identifies the sensor ID that is installed at the primary location of the targeted individual. We have validated the feasibility of the proposed attack using real-world data where we have demonstrated that even a small amount of auxiliary information is sufficient to greatly reduce the set of sensors that the targeted individual's sensor ID is located in. We have also proposed mitigation strategies against the proposed attack. The principle of least privilege can be used as a governing principle in providing the right level of information to various applications. Differential privacy and deletion of past occupancy data could also be used to enhance the privacy of the building occupants.

In chapter 5, we have presented the first analysis, up to the authors' knowledge, of the fairness of the different methods used to aggregate residents' thermal comforts. Specifically, we defined fairness in the context of thermal comfort control and proved that traditional group decision making processes, such as majority, mean, trimmed mean, and median, to aggregate people thermal comfort are not fair. We also explained an aggregation algorithm that ensures fairness with the participants. Our algorithm takes the thermal comfort of the participants as input but is agnostic about how this information is determined; it could be obtained from personal information (such as age, gender, height, and weight) [68], from stochastic thermal profiles related to room conditions [39], or even asked directly to the participants [95]. We presented experimental results performed using a simulator that we implemented to recreate scenarios driven by real occupancy data. The experimental results show how the accumulated loss for participants in our simulated scenarios increased with traditional decision making algorithms whereas it remains bounded for our algorithm. Also, they show, even when there exists a trade off between fairness and comfort, different variants of our algorithm can achieve a similar comfort than the traditional approaches.

In chapter 6, we proposed a context-aware messaging application *Hedwig* which allows both senders and receivers to specify policies in which the message must be delivered. The framework leverages a scalable system of context data collection and management system named *SCARF* . In *Hedwig*, we can support much generalized notions of context than the existing context-aware messaging frameworks. Furthermore the process of context detection utilizes context data from user's mobile devices as well as the context data collected from a variety of in-situ sensors deployed over the infrastructure.

7.2 Future Work

Our work on privacy, fairness, and context-aware messaging has identified a number of opportunities on follow-up work.

Privacy and Security Another direction that we plan to investigate is how to enhance privacy and security of users using context-aware applications. Several works exist on how to perform activity detection, location detection in a privacy-preserving manner, but a generalized framework which is capable of evaluating any type of context in a privacy-preserving manner is an interesting direction of future work. From the study of feasible privacy attacks, the exact details of implementing the mitigation strategies will differ from application to application. A systematic approach of determining the exact parameters of the mitigation strategies will be part of future work. Finally, it would be interesting to study how to design generalized privacy-preserving framework for other types of contexts such as activity, occupancy, mood, comfort, and/or temperature in smart buildings.

Fairness We plan to continue our study about the trade off between fairness and comfort using other variants of our algorithm. The challenge is to prove that these variants, which achieved good results in terms of fairness and comfort in our experiments, are fair for any scenario. Another aspect which was out of the scope of the study but in our plan is to study whether these algorithms can be gamed by participants. Achieving energy efficiency while obtaining thermal comfort has been addressed in many works. We also plan to study how to combine energy efficiency into our fairness vs. comfort trade off. Finally, it would be interesting to study how to generalize the concept of fairness for other participatory services in the smart buildings (e.g., time professors spent with students in tutoring hours).

Scalability We would like to continue our research towards improving the scalability of context-aware messaging frameworks. We are investigating how to evaluate more complex contexts such as contexts including large number of disjunctive and conjunctive conditions.

We are also looking into the problem where the context evaluation is inaccurate in nature (i.e., the evaluation function can make mistakes while evaluating a context). One possibility would be to use multiple context evaluation functions to evaluate a single context and then decide whether the context condition is true or not. Finally, it would be interesting to study how to achieve scalability when there are multiple senders and recipients of context-aware messages and/or several applications are running at the same time.

Bibliography

[1] Notificare application. http://notifica.re.s3-website-eu-west-1.amazonaws.com/features/contextual-messaging/.

[2] Write now, send later with schedule send in gmail. https://gsuiteupdates.googleblog.com/2019/04/write-now-send-later-with-schedule-send.html.

[3] S. Abbaszadeh, L. Zagreus, D. Lehrer, and C. Huizenga. Occupant satisfaction with indoor environmental quality in green buildings. 2006.

[4] A. P. Achilleos et al. An automation component for cross-platform, context-aware applications development. In *CONTEXT 2015, Lanarca, Cyprus*, 2015.

[5] Y. Agarwal, B. Balaji, R. Gupta, J. Lyles, M. Wei, and T. Weng. Occupancy-driven energy management for smart building automation. In *Proceedings of the 2nd ACM workshop on embedded sensing systems for energy-efficiency in building*, pages 1–6. ACM, 2010.

[6] U. Alegre, J. C. Augusto, and T. Clark. Engineering context-aware systems and applications: A survey. *Journal of Systems and Software*, 117:55–83, 2016.

[7] N. S. Alhassoun et al. Context-aware energy optimization for perpetual iot-based safe communities. *Sustainable Computing: Informatics and Systems*, 2019.

[8] A. Alkhresheh et al. Context-aware automatic access policy specification for iot environments. In *IWCMC*. IEEE, 2018.

[9] A. Almagrabi et al. Towards context-aware messaging for emergency situations. In *ISCIT, 2012*.

[10] S. A. Amin, A. Ali-Eldin, and H. A. Ali. A context-aware dispatcher for the internet of things: The case of electric power distribution systems. *Computers & Electrical Engineering*, 52:183–198, 2016.

[11] C. Anderson et al. An ontology-based reasoning framework for context-aware applications. In *CONTEXT 2015, Lanarca, Cyprus*, 2015.

[12] O. Ardakanian, A. Bhattacharya, and D. Culler. Non-intrusive techniques for establishing occupancy related energy savings in commercial buildings. In *3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, BuildSys'16, pages 21–30, 2016.

[13] B. Balaji, H. Teraoka, R. Gupta, and Y. Agarwal. Zonepac: Zonal power estimation and control via hvac metering and occupant feedback. In *5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys'13, pages 18:1–18:8, 2013.

[14] M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.

[15] F. A. Batarseh and J. Pithadia. Context-aware user interfaces for intelligent emergency applications. In *Modeling and Using Context*, 2017.

[16] P. Bellavista et al. A survey of context data distribution for mobile ubiquitous systems. *ACM Comput. Surv.*, 44(4):24:1–24:45, Sept. 2012.

[17] K. Benson, C. Fracchia, G. Wang, Q. Zhu, S. Almomen, J. Cohn, L. Darcy, D. Hoffman, M. Makai, J. Stamatakis, et al. Scale: Safe community awareness and alerting leveraging the internet of things. *IEEE Communications Magazine*, 53(12):27–34, 2015.

[18] J. Black et al. Pervasive computing in health care: Smart spaces and enterprise information systems. In *MobiSys 2004 Workshop on Context Awareness*, 2004.

[19] L. Briesemeister et al. Disseminating messages among highly mobile hosts based on inter-vehicle communication. In *IEEE Intelligent Vehicles Symposium*, Oct 2000.

[20] L. Briesemeister and G. Hommel. Role-based multicast in highly mobile but sparsely connected ad hoc networks. MobiHoc '00. IEEE Press, 2000.

[21] N. M. Byrne, A. P. Hills, G. R. Hunter, R. L. Weinsier, and Y. Schutz. Metabolic equivalent: one size does not fit all. *Journal of Applied physiology*, 99(3):1112–1119, 2005.

[22] J.-P. Calbimonte, H. Jeung, O. Corcho, and K. Aberer. Semantic sensor data search in a large-scale federated sensor network. 2011.

[23] S. Chabridon et al. Unified modeling of quality of context and quality of situation for context-aware applications in the internet of things. In *Modeling and Using Context*, 2017.

[24] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar. Throughput and fairness guarantees through maximal scheduling in wireless networks. *IEEE Transactions on Information Theory*, 54(2):572–594, 2008.

[25] Y. Chen, A. Machanavajjhala, M. Hay, and G. Miklau. PeGaSus: Data-adaptive differentially private stream processing. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1375–1388. ACM, 2017.

[26] A. Coronato et al. Towards an implementation of smart hospital: A localization system for mobile users and devices. In *2008 International Conference on Pervasive Computing and Communications (PerCom)*.

[27] L. Da Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on industrial informatics*, 10(4):2233–2243, 2014.

[28] S. K. Datta, C. Bonnet, and N. Nikaein. Self-adaptive battery and context aware mobile application development. In *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 761–766. IEEE, 2014.

[29] A. K. Dey and G. D. Abowd. Cybreminder: A context-aware system for supporting reminders. In *Handheld and Ubiquitous Computing*. Springer, 2000.

[30] C. Dwork. Differential privacy. *Encyclopedia of Cryptography and Security*, pages 338–340, 2011.

[31] V. L. Erickson, M. Á. Carreira-Perpiñán, and A. E. Cerpa. Observe: Occupancy-based system for efficient reduction of hvac energy. In *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 258–269. IEEE, 2011.

[32] V. L. Erickson and A. E. Cerpa. Occupancy based demand response hvac control strategy. In *2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*, BuildSys'10, pages 7–12, 2010.

[33] V. L. Erickson and A. E. Cerpa. Thermovote: participatory sensing for efficient building hvac conditioning. In *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, pages 9–16. ACM, 2012.

[34] V. L. Erickson and A. E. Cerpa. Thermovote: Participatory sensing for efficient building hvac conditioning. In *4th ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, BuildSys'12, pages 9–16, 2012.

[35] R. Fagin and J. H. Williams. A fair carpool scheduling algorithm. *IBM Journal of Research and Development*, 27(2):133–139, 1983.

[36] P. O. Fanger. *Thermal comfort. Analysis and applications in environmental engineering.* McGraw-Hill, 1972.

[37] M. Feldmeier and J. A. Paradiso. Personalized hvac control system. In *2010 Internet of Things (IOT)*, pages 1–8, 2010.

[38] G. Garcia-Perate et al. Ambient recommendations in the pop-up shop. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 773–776. ACM, 2013.

[39] A. Ghahramani, C. Tang, and B. Becerik-Gerber. An online learning approach for quantifying personalized thermal comfort via adaptive stochastic modeling. *Building and Environment*, 92:86–96, 2015.

[40] S. Ghayyur, Y. Chen, R. Yus, A. Machanavajjhala, M. Hay, G. Miklau, and S. Mehrotra. Iot-detective: Analyzing iot data under differential privacy. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1725–1728. ACM, 2018.

[41] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica. Dominant resource fairness: Fair allocation of multiple resource types. In *8th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'16, pages 24–24, 2011.

[42] D. Gil, A. Ferrández, H. Mora-Mora, and J. Peral. Internet of things: A review of surveys based on context aware intelligent services. *Sensors*, 16(7):1069, 2016.

[43] S. N. Han et al. Semantic context-aware service composition for building automation system. *IEEE Transactions on industrial informatics*, 10(1), 2013.

[44] L. A. Hang-yat and D. Wang. Carrying my environment with me: A participatory-sensing approach to enhance thermal comfort. In *5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, pages 1–8, 2013.

[45] A. Holzinger et al. From smart health to smart hospitals. In *Smart health*. Springer, 2015.

[46] X. L. Huang and B. Bensaou. On max-min fairness and scheduling in wireless ad-hoc networks: analytical framework and implementation. In *2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, MobiHoc'01, pages 221–231, 2001.

[47] S. Im, J. Kulkarni, and B. Moseley. Temporal fairness of round robin: competitive analysis for lk-norms of flow time. In *27th ACM symposium on Parallelism in Algorithms and Architectures*, (SPAA'15), pages 155–160, 2015.

[48] S. Im, J. Kulkarni, and K. Munagala. Competitive algorithms from competitive equilibria: Non-clairvoyant scheduling under polyhedral constraints. In *46th Annual ACM Symposium on Theory of Computing*, STOC'14, pages 313–322, 2014.

[49] A. Inc. Drop message. `http://www.dropmessages.com/`.

[50] S. Inc. Snapchat. `https://www.snapchat.com/`.

[51] M. U. Iqbal et al. A generic approach for energy efficient context recognition using smart phones. In *Intelligent Environments (IE), 2015.*

[52] H. Jafarpour et al. Ccd: A distributed publish/subscribe framework for rich content formats. *IEEE Trans. Parallel Distrib. Syst.*, 23(5):844–852, May 2012.

[53] H. Jafarpour, B. Hore, S. Mehrotra, and N. Venkatasubramanian. Ccd: A distributed publish/subscribe framework for rich content formats. *IEEE Transactions on Parallel and Distributed Systems*, 23(5):844–852, May 2012.

[54] R. Jain and P. Sinha. Content without context is meaningless. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, pages 1259–1268, New York, NY, USA, 2010. ACM.

[55] F. Jazizadeh and B. Becerik-Gerber. Toward adaptive comfort management in office buildings using participatory sensing for end user driven control. In *4th ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, BuildSys'12, pages 1–8, 2012.

[56] F. Jazizadeh and B. Becerik-Gerber. Toward adaptive comfort management in office buildings using participatory sensing for end user driven control. In *4th ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, BuildSys'12, pages 1–8, 2012.

[57] F. Jazizadeh, F. M. Marin, and B. Becerik-Gerber. A thermal preference scale for personalized comfort profile identification via participatory sensing. *Building and Environment*, 68:140–149, 2013.

[58] F. Jazizadeh and S. Pradeep. Can computers visually quantify human thermal comfort?: Short paper. In *3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, BuildSys'16, pages 95–98, 2016.

[59] R. Jia, R. Dong, S. S. Sastry, and C. J. Sapnos. Privacy-enhanced architecture for occupancy-based HVAC control. In *ACM/IEEE 8th International Conference on Cyber-Physical Systems (ICCPS)*, pages 177–186. IEEE, 2017.

[60] S. Jones et al. efriend: an ethical framework for intelligent environments development. *Ethics and Information Technology*, 2015.

[61] V. Jovanovikj et al. Security policy model for ubiquitous social systems. In *CONTEXT 2017*.

[62] H. Khurana and D. Kirschner. Space utilization and building management system analysis, Sept. 27 2018. US Patent App. 15/467,824.

[63] B. Kingma and W. van Marken Lichtenbelt. Energy consumption in buildings and female thermal demand. *Nature Climate Change*, 5(12):1054–1056, 2015.

[64] M. Knappmeyer et al. Survey of context provisioning middleware. *Communications Surveys Tutorials, IEEE*, 15(3):1492–1519, Third 2013.

[65] S. Knox et al. Strategy-proof thermal comfort voting in buildings. *Revue d'Intelligence Artificielle*, 22:pp 549–568, June 2008.

[66] K. Konis and M. Annavaram. The occupant mobile gateway: a participatory sensing and machine-learning approach for occupant-aware energy management. *Building and Environment*, 118:1–13, 2017.

[67] A. Krioukov and D. Culler. Personal building controls. In *11th International Conference on Information Processing in Sensor Networks*, IPSN'12, pages 157–158, 2012.

[68] A. H.-y. Lam, Y. Yuan, and D. Wang. An occupant-participatory approach for thermal comfort enhancement and energy conservation in buildings. In *5th International Conference on Future energy systems*, e-Energy'14, pages 133–143, 2014.

[69] A. H.-y. Lam, Y. Yuan, and D. Wang. An occupant-participatory approach for thermal comfort enhancement and energy conservation in buildings. In *5th International Conference on Future Energy Systems*, e-Energy'14, pages 133–143, 2014.

[70] N. D. Lane, Y. Chon, L. Zhou, Y. Zhang, F. Li, D. Kim, G. Ding, F. Zhao, and H. Cha. Piggyback crowdsensing (pcs): energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, page 7. ACM, 2013.

[71] I. Lee and K. Lee. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, 58(4):431–440, 2015.

[72] V. R. Lesser. Cooperative multiagent systems: A personal view of the state of the art. *IEEE Transactions on knowledge and data engineering*, 11(1), 1999.

[73] L. Liu, E. Yu, and J. Mylopoulos. Security and privacy requirements analysis within a social setting. In *Proceedings. 11th IEEE International Requirements Engineering Conference, 2003.*, pages 151–161. IEEE, 2003.

[74] S. Longo et al. Enriching shopping experiences with pervasive displays and smart things. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, UbiComp '13 Adjunct, New York, NY, USA, 2013. ACM.

[75] A. I. Maarala, X. Su, and J. Riekki. Semantic reasoning for context-aware internet of things applications. *IEEE Internet of Things Journal*, 4(2):461–473, 2016.

[76] R. Mahmud, K. Ramamohanarao, and R. Buyya. Latency-aware application module management for fog computing environments. *ACM Transactions on Internet Technology (TOIT)*, 19(1):9, 2018.

[77] P. Makris et al. A survey on context-aware mobile and wireless networking: On networking and computing environments' integration. *Communications Surveys Tutorials, IEEE*, 15(1):362–386, First 2013.

[78] A. Martínez-Ballesté, P. A. Pérez-Martínez, and A. Solanas. The pursuit of citizens' privacy: a privacy-aware smart city is possible. *IEEE Communications Magazine*, 51(6):136–141, 2013.

[79] S. Mehrotra et al. TIPPERS: A privacy cognizant iot environment. In *PerCom Workshops*, 2016.

[80] P. Moore et al. Fog computing and low latency context-aware health monitoring in smart interconnected environments. In L. Barolli, F. Xhafa, N. Javaid, E. Spaho, and V. Kolici, editors, *Advances in Internet, Data & Web Technologies*, pages 29–40, Cham, 2018. Springer International Publishing.

[81] O. Mutlu and T. Moscibroda. Parallelism-aware batch scheduling: Enhancing both performance and fairness of shared dram systems. *ACM SIGARCH Computer Architecture News*, 36(3):63–74, 2008.

[82] Y. Nakanishi et al. Context aware messaging service: A dynamical messaging delivery using location information and schedule information. *Personal Technologies*, 4(4):221–224, 2000.

[83] Y. Nam et al. Intelligent video surveillance system: 3-tier context-aware surveillance system with metadata. *Multimedia Tools and Applications*, 57(2), 2012.

[84] H. Nguyen, M. Y. S. Uddin, and N. Venkatasubramanian. Multistage adaptive load balancing for big active data publish subscribe systems. In *Proceedings of the 13th ACM International Conference on Distributed and Event-based Systems*, pages 43–54. ACM, 2019.

[85] V. Onut et al. Smart surveillance system applications. In *Proceedings of the 2010 conference of the center for advanced studies on collaborative research*. IBM Corp., 2010.

[86] P. Pappachan, M. Degeling, R. Yus, A. Das, S. Bhagavatula, W. Melicher, P. E. Naeini, S. Zhang, L. Bauer, A. Kobsa, et al. Towards privacy-aware smart buildings: Capturing, communicating, and enforcing privacy policies and preferences. In *Distributed Computing Systems Workshops (ICDCSW), 2017 IEEE 37th International Conference on*, pages 193–198. IEEE, 2017.

[87] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE Communications Surveys Tutorials*, 16(1):414–454, First 2014.

[88] C. Perera, A. Zaslavsky, C. H. Liu, M. Compton, P. Christen, and D. Georgakopoulos. Sensor search techniques for sensing as a service architecture for the internet of things. *IEEE Sensors Journal*, 14(2):406–420, 2013.

[89] M. Pritoni, K. Salmon, A. Sanguinetti, J. Morejohn, and M. Modera. Occupant thermal feedback for improved efficiency in university buildings. *Energy and Buildings*, 144:241–250, 2017.

[90] Q. Pu, H. Li, M. Zaharia, A. Ghodsi, and I. Stoica. Fairride: Near-optimal, fair cache sharing. In *8th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'16, pages 393–406, 2016.

[91] S. Purdon, B. Kusy, R. Jurdak, and G. Challen. Model-free hvac control using occupant feedback. In *IEEE 38th Conference on Local Computer Networks Workshops (LCN Workshops)*, pages 84–92, 2013.

[92] Y. Qin, Q. Z. Sheng, N. J. Falkner, S. Dustdar, H. Wang, and A. V. Vasilakos. When things matter: A survey on data-centric internet of things. *Journal of Network and Computer Applications*, 64:137–153, 2016.

[93] N. Ravi, J. Scott, L. Han, and L. Iftode. Context-aware battery management for mobile phones. In *2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 224–233. IEEE, 2008.

[94] N. Ravi, J. Scott, and L. Iftode. Context-aware battery management for mobile phones: A feasibility study. In *Sixth Annual IEEE International Conference on Pervasive Computing and Communications, 2008. PerCom 2008*, 2006.

[95] S. Reddy, D. Estrin, and M. Srivastava. Recruitment framework for participatory sensing data collections. In *8th International Conference on Pervasive Computing*, Pervasive'10, pages 138–155, 2010.

[96] R. J. Robles and T.-h. Kim. Context aware tools for smart home development. *International Journal of Smart Home*, 4(1), 2010.

[97] Y. B. Saied, A. Olivereau, D. Zeghlache, and M. Laurent. Trust management system design for the internet of things: A context-aware and multi-service approach. *Computers & Security*, 39:351–365, 2013.

[98] A. Sanguinetti, M. Pritoni, K. Salmon, A. Meier, and J. Morejohn. Upscaling participatory thermal sensing: Lessons from an interdisciplinary case study at university of california for improving campus efficiency and comfort. *Energy Research & Social Science*, 2017.

[99] Seungwoo et al. Seemon: Scalable and energy-efficient context monitoring framework for sensor-rich mobile environments. *Personal Technologies*, 4(4), 2008.

[100] O. B. Sezer et al. Context-aware computing, learning, and big data in internet of things: a survey. *IEEE Internet of Things Journal*, 5(1):1–27, 2017.

[101] D. Snoonian. Smart buildings. *IEEE spectrum*, 40(8):18–23, 2003.

[102] A. Solanas et al. Smart health: a context-aware health paradigm within smart cities. *IEEE Communications Magazine*, 52(8), 2014.

[103] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[104] E. Taylor, K. Hewitt, and W. Lawless. Group decision-making: Consensus rule versus majority rule. *Procedia Technology*, 9:498–504, 2013.

[105] M. Y. S. Uddin, A. Nelson, K. Benson, G. Wang, Q. Zhu, Q. Han, N. Alhassoun, P. Chakravarthi, J. Stamatakis, D. Hoffman, et al. The scale2 multi-network architecture for iot-based resilient communities. In *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 1–8. IEEE, 2016.

[106] M. Y. S. Uddin, V. Setty, Y. Zhao, R. Vitenberg, and N. Venkatasubramanian. Richnote: Adaptive selection and delivery of rich media notifications to mobile users. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pages 159–168. IEEE, 2016.

[107] M. Y. S. Uddin and N. Venkatasubramanian. Edge caching for enriched notifications delivery in big active data. In *38th IEEE International Conference on Distributed Computing Systems, ICDCS 2018, Vienna, Austria, July 2-6, 2018*, pages 696–705, 2018.

[108] N. Vallina-Rodriguez, P. Hui, J. Crowcroft, and A. Rice. Exhausting battery statistics: understanding the energy demands on mobile handsets. In *Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds*, pages 9–14. ACM, 2010.

[109] X. Wang and P. Tague. Non-invasive user tracking via passive sensing: Privacy risks of time-series occupancy measurement. In *Proceedings of the Workshop on Artificial Intelligent and Security Workshop*, pages 113–124. ACM, 2014.

[110] D. A. Winkler, A. Beltran, N. P. Esfahani, P. P. Maglio, and A. E. Cerpa. FORCES: feedback and control for occupants to refine comfort and energy savings. In *2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp'16, pages 1188–1199, 2016.

[111] C. S. Wong, I. Tan, R. D. Kumari, and F. Wey. Towards achieving fairness in the linux scheduler. *ACM SIGOPS Operating Systems Review*, 42(5):34–43, 2008.

[112] H. Xiong, D. Zhang, G. Chen, L. Wang, and V. Gauthier. Crowdtasker: Maximizing coverage quality in piggyback crowdsensing under budget constraint. In *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 55–62. IEEE, 2015.

[113] N. Yamtraipat, J. Khedari, and J. Hirunlabh. Thermal comfort standards for air conditioned buildings in hot and humid thailand considering additional factors of acclimatization and education level. *Solar Energy*, 78(4):504–517, 2005.

[114] Yikyak. Yikyak. `http://www.yikyakapp.com/features/`.

[115] O. Yurur et al. Context-awareness for mobile sensing: A survey and future directions. *Communications Surveys Tutorials, IEEE*, PP(99):1–1, 2014.

[116] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In *5th European Conference on Computer Systems*, pages 265–278, 2010.

[117] M. Zhou, R. Zhang, D. Zeng, and W. Qian. Services in the cloud computing era: A survey. In *2010 4th International Universal Communication Symposium*, pages 40–46. IEEE, 2010.

[118] Z. Zhuang et al. Improving energy efficiency of location sensing on smartphones. In *MobiSys*. ACM, 2010.