ABSTRACT

POOSAMANI, NITHYANANTHAN. Enabling Accurate and Energy-Efficient Context-Aware Systems for Smart Objects using Cellular Signals. (Under the direction of Dr. Injong Rhee.)

The Internet of Things (IoT) paradigm aims to interconnect a variety of heterogeneous Smart Objects (e.g., sensors, smart devices, home automation equipment) using Machine-to-Machine communications. Smart devices have become one of the primary ways for people to access entertainment and other business applications, both inside and outside of their homes. This has led to two significant problems: substantial increase in monthly wireless data usage, and a rapid drain in smart phone battery life. Another recent trend with small form-factors in devices has lead to a bulk of the device components fused together using adhesives without being exposed to outside world (e.g., battery is glued to panel case or screen without exposing the circuit terminals). This prevents researchers from measuring energy consumption ratings for the different sub-systems in the phone using power monitoring devices.

Smart devices that provide health monitoring, smart home and workplace, enterprise device management, and many others need to constantly sense their context and communicate with the network to collaborate with others. Mobile applications that provide location-specific services require either the absolute or logical location of users in indoor settings. Identifying the context of a user (e.g., in front of the store, suits section, billing counter, home, office, conference room) in a timely and energy-efficient manner is important for the applications to disburse appropriate deals or activate a set of device-specific policies. In all these cases, though sub-meter level accuracy is not required or expected, a practical and an infrastructure-independent solution which can be easily deployed in real world is highly preferred.

In this research, we first analyze the detailed statistical properties of cellular signals in indoor environments and construct a reliable database of cellular signal signatures for different indoor locations. We show that it is feasible to accurately distinguish between neighbouring indoor locations in a reliable and energy-efficient manner. We then profile the energy usage of Wi-Fi
in mobile devices under different device screen activation scenarios and quantify the energy wastage due to unnecessary scan and association events under poor link conditions, which to the best of our knowledge has not been reported in previous literature. In our first work, iSha, we develop a fine-grained energy consumption analyzer system to estimate the energy consumption values of specific sub-components in smart devices which eliminates the need for specialized hardware power monitoring equipments.

In our second work, PRiSM, we develop a novel and light-weight signature matching system to automatically discover Wi-Fi hotspots without turning on the Wi-Fi interface in the smart device. It uses signal strengths received from cellular base stations to statistically predict the presence of Wi-Fi and connects directly to the hotspot without scanning. The system continuously learns based on user movement behaviours and auto-tunes its parameters accordingly. Hence, PRiSM, provides a practical and infrastructure-independent system to maximize Wi-Fi data offloading and simultaneously minimize Wi-Fi sensing costs.

In our final work, PILS, we develop a indoor localization system which logically maps the contextual information of the smart device with a specific indoor location using cellular multi-homing. We utilize a variety of back-channel parameters such as Received Signal Code Power (RSCP) from 3G radio cellular systems, Reference Signal Received Power (RSRP) and Reference Signal Received Quality (RSRQ) from 4G radio cellular systems in addition to Received Signal Strength (RSS) values from 2G radio cellular systems. We show the effects on location accuracy with using only connected base stations and with neighbouring base stations, self-sourced data and crowd-sourced data. We also show that by choosing a combination of signals from different cellular radio technologies specific to different locations provide better location accuracy than relying on one single radio technology for all indoor locations.

In short, we aim to address three important challenges in ubiquitous and pervasive mobile computing: maximal data offloading from cellular networks to Wi-Fi with minimal energy consumption, fine-grained energy consumption analysis for small form-factor devices, and cost-effective and infrastructure-independent indoor localization system for wide-area IoT networks.
We show the effectiveness of our solutions with working system prototypes and real world data analysis results. We also show that our solution methodologies are robust and applicable to all major mobile computing platforms.
© Copyright 2015 by Nithyananthan Poosamani

All Rights Reserved
Enabling Accurate and Energy-Efficient Context-Aware Systems for Smart Objects using Cellular Signals

by
Nithyananthan Poosamani

A dissertation submitted to the Graduate Faculty of North Carolina State University in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

Computer Science

Raleigh, North Carolina
2015

APPROVED BY:

__________________________  _________________________
Dr. Mladen Vouk               Dr. Rudra Dutta

__________________________  _________________________
Dr. Huaiyu Dai               Dr. Injong Rhee
Chair of Advisory Committee
DEDICATION

I bow to Dakshinamurthy, the God facing the south, who is the greatest of teachers, who is the personification of unalloyed wisdom, who sits under a banyan tree, who always observes silence in Turiya state, His disciples getting all doubts cleared without the young teacher uttering a single word, who is the doctor to those afflicted by the disease of birth and death, and who is the treasure hose of all knowledge. Salutations to Him who shines and exhibits, Himself by the beatific Chinmudhra of the hand, that He exists within the humans as self, forever and non changing, even during the changing states of childhood, youth and old age and even during the states of sleep, dream and wakefulness. The truth as represented by the concept of Brahman, exists always.
BIOGRAPHY

Nithyananthan Poosamani was born in Coimbatore, Tamilnadu, India. He received his B.E. in Electronics and Communication Engineering from P.S.G. College of Technology, (affiliated to) Anna University, Tamilnadu, India in 2007. He worked as a software engineer in the area of 3G Radio Network Controllers for two years before joining North Carolina State University for M.S. in Computer Engineering in Fall 2009. He enrolled in to the Ph.D. program in Computer Science in Spring 2012 and received his M.S. “en-route” in Computer Science in Spring 2014. His research interests include human mobility pattern analysis, mobile systems design for future Internet of Things, Access Network Discovery and Selection for Smart Objects, and energy-efficient indoor localization techniques.
ACKNOWLEDGEMENTS

I would like to first thank my advisor, Dr. Injong Rhee, without whose guidance, I would never have started on my doctoral program. He supported me through important stages of my research life. I would also like to thank the members of my dissertation committee, Dr. Mladen Vouk, Dr. Rudra Dutta and Dr. Huaiyu Dai for their kind help and patience. They had insightful questions and suggestions which helped me clarify my ideas and execute them. I am also extremely thankful for Dr. Khaled Harfoush and Dr. Mihail Sichitiu for suggestions and help during my doctoral examinations and for their feedback.

I would like to extend my sincere appreciation and thanks to all the staff members of Computer Science department for their help and support through these years. Special thanks to the Director of Graduate Program Dr. Douglas Reeves and Dr. George Rouskas. Special mention to administrative staff Ms. Kathy Luca, Ms. Linda Honeycutt, and Ms. Ann Hunt for their help and support through various day to day activities. I also like to thank Mr. Marhn Fullmer for assisting me with computer network related functions.

I would like to thank my colleagues with whom I had a wonderful opportunity to work with. Special thanks to Dr. Kyunghan Lee, Dr. Yaogong Wang, Dr. Sungro Yoon for their career and research related advice. I am also grateful to other lab members Jeongki Min, Arpit Gupta, Haiqing Jiang, Shuiqing Wang and SeungEun Chung for having shared precious moments in my doctoral study time. Special thanks to Shireesh Bhat for his help and support in crystallizing my thoughts and ideas. I also like to thank my mentors and advisors in both NetApp and Samsung Research for sharing their time and knowledge with me. Special mention to my mentors, Greg and Daniel from NetApp who helped me obtain a solid foundation for my industrial career.

My life at NC State will be incomplete without the mention of following six organizations a.k.a. “families”: The Office of International Services (OIS), The NCSU Student Government (SG), The University Graduate Student Association (UGSA), The NCSU STARS Student Leadership Corps (SLC), The Indian Graduate Student Association at NC State (Maitri), and
The Campus Recreation at NC State. My first and only on-campus job was at the Carmichael Recreation center as an official and later as a Supervisor for recreational sports. Some of my best friends in the United States are from my memorable days at Campus Recreation. Thanks for taking me in to your family and showering with your warm southern culture. I will never forget the fun times be it the tailgating, or simple house parties. Thank you for making me one among you crazy bunch! I am gratified and humbled by my experiences with Maitri both as an Executive and as a student. My association with all the president’s I worked with brings back all the sweet memories. This section will not be able to hold the names of all the wonderful friends I made during my association with Maitri. Thank you all.

I would like to thank all my project coordinators and advisor’s in STARS for their encouragement and support. Special thanks to Dr. Trisha Biswas, Veronica Catete and Robinson Udechukwu for putting up with me through thick and thin. Serving as the Treasurer in the executive body of UGSA for 2014-2015 opened my doors for financial accounting in huge organizations and to understand its nuances. It also helped me identify my abilities as an able administrator and engage for organizational success. Special thanks to my friends Bryan Hoynacke, Dr. David Fiala, Dr. Milena Bobea, Veronica Catete, Veronica Mbaneme, Chirag Gajjar, Katie Kennedy and Brooke Jordan for the awesome times in UGSA. Special mention to my advisors Dr. Barbi Honeycutt and Dr. Beth Overman and finance staff Ms. Ashley Chilton in helping me get through the job ever so easily.

Student government at NC State is definitely one of the main reasons for my successes in student involvement and engagement, and played a major role in honing my administrative abilities. I can easily attribute a variety of my acquired skills at NC State to student government. It gave me a glimpse of everything from serving as a committee chair to the Vice-president of Student Senate, interacting with Vice-Chancellor to Provost to The Chancellor himself, working with a variety of departments ranging from Alumni association to the department of Transportation. It taught me to work closely with your subordinates and your superiors on equal footing and achieve your goals for furthering the success of your organization. It made me
experience the pleasure of public service when the transportation department made changes to the Centennial Route 10 after working closely with them for three years and will ever be cherished in my mind. Though there are innumerable people to personally thank, I shall keep it to my advisors who I interacted with the most and my personal well wishers: “Godmother” Ms. Eileen Coombes, Ms. Laura Stott, Ms. Deborah Felder, Ms. Allyson Clagett and Mr. Mike Giancola. You all made my five years of association with Student Government a wonderful memory. Special shout out to all my best friends I made during my time at SG.

Last but not least, an integral part of my life at NC State will be my association with The Office of International Services both as an orientation team member and as a student. Some of my friends for life were gifted to me here in OIS. Most of them are Ph.D. holders but I am going to address them as I normally would. Forgive my brevity! Lakshmi a.k.a. “Lauren Ball”, Ranjitha a.k.a. “Ranjith Kumar”, PP a.k.a. “PingPong Ganokon Urkasensin”, jajaja a.k.a. “Jose”, Habibi a.k.a. “Ahmad”, Manjula a.k.a. “Manoj”, Joao a.k.a. “The Creepy Guy”, RT a.k.a. “Arthi Kannan”, and Saudi Prince a.k.a. “Tayseer”. The staff members here are more than just university employees and are a family to me. The best part is I get to talk with them more about my personal life than the regular student queries. My heartfelt appreciation and gratitude to all current and former members of the Office of International Services. Special shout out to Alexis, Kelia, Rebecca, Stacy, Mike, Thomas, Fernanda, Michael and Elizabeth!

Survival in a foreign country is not just knowing people but to feed yourself (literally!). I lost count of the number of times I ate food at my friends place. Vivake Ramasubramanian, Ramaswamy Adaikalavan, Balaji Soundararajan - I cannot thank enough for the wonderful support and love given by you and your family members. An important group of highly successful, talented and inspiring individuals who constantly motivate me are my school friends and their family: Adarsh, Ramu, Ganesen, Ashwin, Ajay. Thank you Supni, Kuppu, “stop-n-wait”, my undergraduate colleagues and numerous others for wishing me nothing but the best!

Finally, I will not be able to elucidate the sacrifices made by my mom Mrs. Ramathal Poosamani and dad Mr. Poosamani Nachimuthu, living separated from their only son during all
these years of study. I owe everything to both of you and I only wanted to say that I love you!

“Matha, Pitha, Guru, Deivam”. I bow to The Supreme Being with reverence. Tat Tvam Asi.
# TABLE OF CONTENTS

## LIST OF TABLES

### xi

## LIST OF FIGURES

### xii

## Chapter 1 Introduction

1.1 Dissertation Roadmap ........................................... 3

## Chapter 2 Background and Related Work

2.1 Energy Measurements ............................................ 4
  2.1.1 Screen Activation ........................................ 4
  2.1.2 Sub-component Power ...................................... 4

2.2 Wi-Fi Sensing .................................................. 5
  2.2.1 Wi-Fi Power Consumption and Reduction ................. 5
  2.2.2 Wi-Fi Network Sensing .................................. 5
  2.2.3 Wi-Fi Data Offloading .................................. 6
  2.2.4 Wi-Fi Fingerprinting ................................... 6
  2.2.5 Smart Phone Usage pattern ............................... 7

2.3 Indoor Positioning Techniques .............................. 7
  2.3.1 Infrastructure-based Approaches ......................... 7
  2.3.2 Fingerprinting-based Approaches ....................... 8
  2.3.3 Model-based Approaches ................................ 8
  2.3.4 Inertial Sensor-based Approaches ...................... 9
  2.3.5 FM-based Approaches ................................... 9
  2.3.6 GSM-based Approaches ................................ 9
  2.3.7 Localization Algorithms ................................. 10

## Chapter 3 iSha: A Fine-Grained Energy Analyzer System using Assembler and Disassembler

3.1 Motivation ..................................................... 14

3.2 Assembler and Disassembler ................................ 15

3.3 System Design ................................................ 16

3.4 Evaluation .................................................... 17
  3.4.1 Fine-grained Wi-Fi Energy Measurements ............... 17
  3.4.2 Procedure to Compile Platform Source Code ........... 18
  3.4.3 Procedure to Replace Supplicant Binary ............... 20

3.5 Discussion .................................................... 22

3.6 Concluding Remarks .......................................... 23

## Chapter 4 PRiSM: Wi-Fi Hotspot Auto-Discovery System for Smart Objects

4.1 Motivation ..................................................... 25

4.2 Design ........................................................ 26
  4.2.1 Wi-Fi Power Consumption ............................... 26
LIST OF TABLES

Table 3.1  Fine-grained energy measurements on Nexus One. .......................... 22
Table 4.1  PRiSM cellular signal sampling policy. ........................................... 38
Table 4.2  Dataset information. ................................................................. 39
Table 4.3  Fine-grained measurements for Wi-Fi sensing. ............................. 45
Table 4.4  Total Wi-Fi usage and battery savings for users in dataset ‘D2’ for Wi-Fi
           offloading with $\tau = -80\,dBm$. .............................................. 49
Table 4.5  Nexus One practical energy evaluation. ......................................... 51
Table 5.1  Energy consumption information per second between PILS and other ap-
           proaches used for continuous location sensing. .......................... 66
Table 5.2  Dataset information. ................................................................. 67
| Figure 3.1 | Future is here: Internet-of-Things (image courtesy of theconnectivist.com). | 13 |
| Figure 3.2 | Architecture | 15 |
| Figure 3.3 | Repeated scan/association events under poor AP signal when the device screen is (a) ON, (b) OFF. | 18 |
| Figure 4.1 | Working of default Wi-Fi when (a) an AP is available to connect with, and (b) an AP is not available. | 27 |
| Figure 4.2 | Repeated scan/association events under poor AP signal when the device screen is (a) ON, (b) OFF. | 28 |
| Figure 4.3 | Default Wi-Fi energy consumption for one minute under various screen activation conditions. | 28 |
| Figure 4.4 | The evolution of signal strength distributions from the most frequently connected base station for 3 different APs are depicted in (a), (b), and (c). For each AP, the data is aggregated over time whenever connected with the AP. | 29 |
| Figure 4.5 | The personalized signatures for three APs: (a) AP_X, (b) AP_Y, and (c) AP_Z. The distance between AP_X and AP_Y is about 7 km, AP_Y and AP_Z is about 30 meters. AP_Y and AP_Z are located in the same building. The observed base station IDs and their average signal strengths are given in the legend. | 31 |
| Figure 4.6 | PRiSM system architecture. | 35 |
| Figure 4.7 | PRiSM operation includes three tasks: bootstrapping, signature matching, and online training. | 36 |
| Figure 4.8 | (a) shows the launch screen of the service where user information is gathered, (b) shows that the system service runs constantly in the background and logs all required information. | 40 |
| Figure 4.9 | The number of hours logged by each individual user is shown during data collection phase. | 41 |
| Figure 4.10 | (a, b) ROC curves and (c) $\rho_{FP}$ Vs. $\rho_{FN}$ values for a randomly selected user for all algorithms in dataset ‘D1’. ATiS achieves very high true positive and true negative values and very low $\rho_{FP}$ and $\rho_{FN}$ values simultaneously. | 42 |
| Figure 4.11 | (a) Average $\rho_{FP}$ and $\rho_{FN}$ for users in dataset ‘D1’ and (b) $\rho_{FP}$ and $\rho_{FN}$ for 5 consecutive days for a user. | 43 |
| Figure 4.12 | Monsoon Power Monitor setup connected with the phone. | 44 |
| Figure 4.13 | Wi-Fi energy consumed every minute for (a) screen ON, (b) screen OFF, and (c) under poor Wi-Fi signals. For Footprint, $\Delta 1$ is estimated to be 0.673 mWh for screen on and $\Delta 2$ is estimated to be 0.719 mWh for screen off conditions. | 46 |
| Figure 4.14 | Mean battery savings for all users in the dataset with 95 % confidence interval. (a) vary $\delta$ given $\tau = -80$ dBm, (b) vary $\tau$ given $\delta = 1$ sec. | 48 |
| Figure 5.1 | PILS signature element. | 56 |
| Figure 5.2 | Cellular signal strength distribution of the mostly observed base stations at (a) location A and (b) location B. | 57 |
Figure 5.3 (a) The number of observed base stations over time at a location. It fluctuates from 0 to 7. (b) The observed base station IDs over time. (c) CDF of the number of observed base stations. .......................... 58

Figure 5.4 (a) Evolution of cellular signatures located at a location. .................... 59

Figure 5.5 QQPlot of sample data Vs. Standard Normal values ......................... 61

Figure 5.6 PILS system architecture. .................................................. 64

Figure 5.7 PILS contextual signature storage database. ............................. 65

Figure 5.8 Office location layout. Cross marks represent the trial locations. .......... 68

Figure 5.9 Home-SJ location. The different radio combinations are shown in different colored bars. The bars show average prediction probability and 95% confidence intervals error lines. ........................................ 68

Figure 5.10 Prediction probability for various environments (a) Self-Sourced Data (b) Crowd-Sourced Data. .......................................................... 69

Figure 5.11 Prediction probability for various environments with special sorting techniques for BSSET and MSE (a) Self-Sourced Data (b) Crowd-Sourced Data. . ... 71

Figure 5.12 Prediction error for various environments (a) Self-Sourced Data (b) Crowd-Sourced Data. .......................................................... 72

Figure 5.13 Prediction probability for change in number of observed neighbour BS (a) Home-SJ (b) Office. .................................................. 74

Figure 5.14 Prediction probability for change in number of observed neighbour BS (a) University Room 1231 (b) Library. ........................................ 75

Figure 5.15 Prediction probability for change in number of observed neighbour BS (a) University Room 1231 (b) Library. ........................................ 77

Figure 5.16 Prediction probability for change in distance between adjoining test locations during hallway walk. ........................................ 78

Figure 5.17 Prediction probability for room level accuracy (a) All connected and neighbour BS (b) Connected BS only. ................................. 78
Chapter 1

Introduction

The Internet of Things (IoT) paradigm aims to interconnect a variety of heterogeneous Smart Objects (e.g., sensors, smart devices, home automation equipment) using Machine-to-Machine communications. A majority of consumer devices sold today come equipped with a variety of wireless communication technology (e.g., Bluetooth Low Energy, Wireless LAN, Near Field Communication, Cellular) to connect with the world-wide network.

With billions of such devices predicted to connect with the future internet, bringing the physical data from these devices to the digital world requires energy-efficient methodologies and standard communication protocols. Devices that provide health monitoring, smart home and workplace, enterprise data security, and many others need to constantly sense their context and communicate with the network to collaborate with others. A variety of the smart devices mentioned above are self-powered or connected to a power source glued to their components and hence the circuit terminals are not exposed to measure the energy consumption.

Recent proliferation of smart devices with varying form factors and attractive pricing has met with increased customer adoption of these devices. Global shipments of smart phones have already surpassed those of conventional personal computers [1], and their numbers keep increasing each day. A plethora of applications specific to various smart phone platforms (e.g., Android [2], iOS [3], Windows [4]) are created to enhance the user experience in these devices. The
applications range from a simple calendar manager to complex business productivity software suites.

An ever-increasing majority of these applications expect *always-on* internet connectivity (e.g., iCloud [5], other cloud-based services) so as to provide unlimited storage and processing capabilities. Dedicated cameras nowadays come with Wi-Fi interfaces. Smart devices have become one of the primary ways for people to access entertainment [6] (e.g., social networking, communication, and browsing applications), both inside and outside of their homes. This has led to two significant problems: substantial increase in monthly data usage [7], and rapid drain in smart phone battery life.

Smart devices have become one of the primary ways for people to access various business applications. Most of these applications provide location-specific services and hence, require either the absolute or logical location of users in indoor settings. Big retail giants and shop vendors in indoor locations such as malls, public convention centers aim to provide specific deals and discounts to users who are within walking distance from their shops. Identifying the context of a user (e.g., in front of the store, suits section, billing counter) in a timely and practical manner is very important for the retail outlets to disburse appropriate deals.

Another fast developing trend is to selectively activate certain security features for smart devices in Enterprise Device Management (e.g., turn off camera inside office space, disable voice recorder in conference room). In the above applications, ‘front of the store’, ‘billing counter’, ‘conference room’ are few examples of logical locations or in the broad-sense referred to as the ‘context’ of the smart device. In all these cases, though sub-meter level accuracy is not required or expected, accuracy of the order of few feet ($\approx 5$ to $10$ ft) is highly preferred. However, to design a precise and an energy-efficient indoor localization system in an automated manner is (still) a very non-trivial task. The reasons include the need for: infrastructure-independent solutions, ease of practical deployment, and minimal battery consumption for users.

We aim to address three important challenges in ubiquitous and pervasive mobile computing: operating-system-independent fine-grained energy consumption analysis, maximal data offloading
with minimal energy consumption, and context-aware indoor localization with minimal sensor costs. We propose iSha, PRiSM and PILS for those challenges and to prove the effectiveness of our solutions with working system prototypes and real data traces.

1.1 Dissertation Roadmap

The rest of this organized as follows:

Chapter 2 provides the background information and also provides description about the previous works related to the different topics under consideration.

Chapter 3 provides information about the system and implementation of a fine-grained light-weight energy analyser system, iSha. The system uses a OS-independent solution using smali/backsmali, a assembler/disassembler module functionality described in Chapter 3.2. The system design is presented in Chapter 3.3. Discussion about possible future work is given in Chapter 3.5. Concluding remarks for this topic is presented in Chapter 3.6.

Chapter 4 describes PRiSM, a Wi-Fi hotspot auto-discovery system. It discusses the various components of system design in Chapter 4.2 and the real-world system implementation is Chapter 4.3. Simulation results based on real world traces and practical verification of energy savings are discussed in Chapter 4.4. Concluding remarks for this topic is presented in Chapter 4.6.

Chapter 5 provides information about a context-aware indoor localization system called PILS using the technique of cellular multi-homing. Details about cellular multi-homing and the suitability of cellular signals for such indoor location matching systems are discussed in Chapter 5.2. The prototype system implementation is described in Chapter 5.3. The performance measurements are discussed in Chapter 5.4 and possible drawbacks and future optimizations are provided in Chapter 5.5. Concluding remarks for this topic is presented in Chapter 5.6.

Chapter 6 summarizes the challenges solved by this dissertation and proposes new activities for future work.
Chapter 2

Background and Related Work

This chapter provides the background and existing research in the areas pertaining to our dissertation: energy measurements, Wi-Fi sensing related mechanisms, and indoor localization techniques.

2.1 Energy Measurements

2.1.1 Screen Activation

Huge energy savings are reported for power measurements for cellular radio/LTE traffic obtained during screen-off conditions [8], Wi-Fi [9–12]. However, we perform measurements under both screen on and off conditions, show that screen off energy is more compared to screen on due to use of CPU wakelocks, and use appropriate energy values for user logs. Hence, the final energy figures in our experiments more accurately match actual Wi-Fi power consumptions.

2.1.2 Sub-component Power

The variability and complexity of hardware in smart devices has resulted in highly dynamic power variations as opposed to some existing power models with high baseline-power level and small dynamic range [13, 14]. Though there is an array of works in the field of energy
measurements, there following are a few works which directly relate to measuring the energy consumption in mobile or smart devices: [13,14]. Since these works do not take in to account the recent OS changes and the power dynamics associated with latest integrated Circuit chips, they do not account for the dynamic power variations in smart devices. These works assumed a constant baseline-power level for mobile devices and calculated the excess power consumption to other operating components. They also only allow for small range of variations. iSha [15,16] differs from the previous works in that it dynamically modifies the executable code and inserts specific log-triggers to achieve fine-grained energy measurements.

2.2 Wi-Fi Sensing

2.2.1 Wi-Fi Power Consumption and Reduction

Wi-Fi power consumption has been studied in previous works: TailEnder [11], [12]. The measurements were done on Android G1 (0.175 mWh) and Nokia N95 (1.4 mWh [11], 0.328 mWh [12]) mobile phones. Our measurements on Android Nexus One show comparatively less values as shown in Table 4.3. Wi-Fi has high initial cost for scan/association [11]. Many techniques are proposed to mitigate the excessive power consumption by Wi-Fi radios. Wake-on-Wireless [17] and E-Mili [18] reduce the idle state power consumption of Wi-Fi by installing a secondary low-power transceiver for idle listening and by down-clocking the Wi-Fi chipset during idle periods respectively. Recently, NAPman [19], SleepWell [20] proposed intelligent idle period reduction schemes to enable Wi-Fi to stay in the PSM mode longer than usual. We try to reduce Wi-Fi sensing costs (radio power up/down, scan, association and DHCP) and do not focus on power consumption during idle periods or during data transfer.

2.2.2 Wi-Fi Network Sensing

Wi-Fi networks are resourceful but are scarcely available when compared to cellular networks [10]. Hence much research is focussed on developing optimal sensing intervals for Wi-Fi scans [9,21].
The algorithms consider information such as AP inter-arrival time, AP density and user velocity. These algorithms either increase/decrease Wi-Fi sensing intervals upon failure to meet APs and hence will not work well for all users because the Wi-Fi connectivity and movement patterns of users differ significantly.

Some other works determine the Wi-Fi sensing policy using on-the-board sensors in smart phones (e.g., Accelerometers [9], GPS [22–24], Bluetooth [25]) or off-the-board sensors (e.g., Zigbee [26]). However collecting information from those sensors pose additional energy overhead (e.g., Accelerometers consume close to 0.667 mWh every 30 sec [27]) and some resources may not be available always (e.g., GPS is not available indoors, availability of Bluetooth users). We utilize readily available GSM cellular signals at zero extra energy cost and predicts AP availability without the aid of alternative sensor information.

2.2.3 Wi-Fi Data Offloading

Prior research works quantify the efficiency of mobile data offloading through available Wi-Fi networks. [28] predicts future Wi-Fi throughput and waits to delay data transfer only if the 3G savings expected are within the application’s delay tolerance. [29] shows that over 70% of data can be offloaded if delayed by two hours. [12] selects 3G or Wi-Fi links to transfer data based on the Lyapunov optimization framework to minimize energy expenditure. PRiSM [30,31] on the other hand does not provide quantitative bounds on the amount of data that can be offloaded or decides between Wi-Fi and 3G, instead, we try to maximize such offloading opportunities with minimal energy consumption.

2.2.4 Wi-Fi Fingerprinting

Wi-Fi hotspots are scarcely distributed when compared to cellular networks [10]. Wi-Fi signal fingerprinting techniques [32,33] use extensive offline pre-processing stage to construct signal strength models and to calibrate the radio maps. Also, these techniques take more time to converge. PRiSM [30,31] does not assume anything about the underlying data model or
distribution and hence takes a non-parametric approach.

Few other works use multi-modal sensors (e.g., Accelerometers [9], GPS [22–24], Bluetooth [25], Zigbee [26]) in addition to identify context. Unlike PRiSM, the sensors may not be available always and they consume extra battery energy (e.g., Accelerometers consume close to 0.667 mWh every 30 sec [27]). Some require infrastructural changes and extensive war-driving efforts to obtain feature-rich data sets. Also, most commercial systems (e.g., WiFi Sense [34], Place Lab [35]) turn on the radio interfaces continuously to identify context which results in battery drain.

2.2.5 Smart Phone Usage pattern

The usage pattern of smart phones differs on an individual basis. [6] characterizes the user interaction with the device and the variety of applications used and the impact of those activities on network and energy usage. [36] provides the network performance experienced by the users solely when they are interacting with their devices. It also argues how poor network performance results in a bad user experience.

2.3 Indoor Positioning Techniques

2.3.1 Infrastructure-based Approaches

Infrastructure-based approaches utilize specialized hardware and RF signals or beacons to achieve accuracy in indoor locations: Infrared [37], Ultrasound [38], RFID [39], Bluetooth [40], MIMO [41]. Although these systems provide high accuracy, they depend on additional infrastructure and meticulous engineering for efficient working. Also, they pose high overhead for deployment in practical scenarios. [41] provides very fine-grained location tracking in indoors but it requires changes in the AP software and in client devices for fast calibration. Hence it might not be a practical solution in current market setup. Our work, PILS [30,42], however, does not require any additional hardware or software changes and can work from current off-the-shelf smart


2.3.2 Fingerprinting-based Approaches

The most widely used approach for mapping RF signals to location uses received signal strength (RSS). Here a signal map of RSS fingerprints is constructed for every location. This RSS fingerprint was introduced by RADAR [32] and it constructed Wi-Fi signal maps at locations from one or more access points, achieving accuracies of the order of a few meters. Later systems such as Horus [33] used probabilistic techniques to improve accuracy. However, the above systems had extensive calibration overhead.

A recent approach [43] tried to reduce calibration and increase accuracy further using statistical modeling of signal strengths. Thus in general, fingerprinting-based approaches consume considerable time and effort to generate the signals maps of the locations. Moreover, current state-of-art techniques utilize Wi-Fi signals for indoor localization which are prone to multi-path and fading effects from static objects and human movement. A recent work [44] proposed using OFDM PHY layer information for more robust performance but still requires extensive signal logging and hence depletes battery energy quickly. PILS [42] uses cellular signals already received by smart phones at no extra-cost and runs in the background collecting signals for the places visited by the user. Also, we try to reduce the dependency on RSS signals by utilizing other network-related signals (e.g., Received Signal Reference Power, Timing Advance).

2.3.3 Model-based Approaches

Model-based approaches [45–47] avoid costly profiling needed in fingerprinting approaches. They generate signal maps based on recorded RSS from few known locations and use radio propagation characteristics to model the entire area. Some of them use floor plan and location of APs to generate the signal map. Other than the RSS, few other techniques model based on time of arrival [48], time difference of arrival [49], and angle of arrival [41, 50]. However, all the techniques require potentially expensive hardware and complex pre- or online processing to
compute the required database. Our work [30,42], on the other hand, stores cellular signals in a novel way within the user’s phone and reduces computation complexity.

2.3.4 Inertial Sensor-based Approaches

Current smart devices in the market come with a variety of sensors such as accelerometer, gyroscope, barometer, and magnetometer. In these cases, if the starting location is known, the user can be tracked in indoor settings. Accelerometer signals have been used along with dead-reckoning to provide indoor pedestrian localization [51,52]. A mixture of sensors have been utilized to synthesize unique signatures for different locations [53]. All these multi-modal sensing techniques still require other radio signals (e.g., Wi-Fi, cellular) to increase the accuracy and continuous running of these sensors pose extra energy overhead for the battery.

2.3.5 FM-based Approaches

The operating frequency range of Wi-Fi signals makes it sensitive to static objects and human movement due to multi-path and fading. To overcome these limitations in indoor settings, few research works utilized FM signals for localization [54,55]. FM signals are shown to experience less temporal variations when compared to Wi-Fi signals in indoors and hence provide good location accuracy. However, the explicit need to know the location of FM stations and floor plans to generate FM signal maps and the need for large windows in the building [55] make these techniques not feasible for practical deployment.

2.3.6 GSM-based Approaches

GSM signals have also been used to generate RSS maps to predict indoor locations. One of the research works very similar to PILS utilizes extensive signal fingerprints from up to 29 different GSM channels for enhanced accuracy [56]. However, such values cannot be obtained from commercial GSM phones as the signal strengths are limited up to a maximum of 6 neighbors [57,58] and a connected cell. PlaceLab [58] provides very coarse location with
100 – 150 meter accuracy. Though the volume of research is huge in indoor localization, not many works have utilized GSM signals. Our work, PILS [42] is very different from above previous works in that it utilizes the statistical properties of the entire spectrum of signals received from both connected and neighbor cells. It also does cellular multi-homing where GSM, UMTS and LTE signals are used to increase the accuracy of location prediction.

2.3.7 Localization Algorithms

A simple yet lightweight algorithm adopted for indoor and outdoor localization systems uses a set of base station IDs for matching (BSSET). In order to evaluate the likelihood of matching a fingerprint in the database, the algorithm can simply count the number of common BSs or can sum up the weight values of common BSs, where the weight is assigned to each BS based on its frequency of observation. Another class of algorithms use mean squared error (MSE) for matching [59, 60].

Most Artificial Intelligence (AI) techniques typically identify the top \( k \) fingerprints showing the smallest MSE values and then calculate the center from the locations paired with \( k \) fingerprints. This extension is called \( kNN \) (k-nearest neighbor) but they have the following problems: minimal training phase but costly testing phase including both time and memory, and assumes that data is in feature/metric space which means it is associated with some distance. Our work requires entire signal distribution clusters of the training data for quicker prediction and hence, uses a specialized hybrid algorithm to including lazy learning techniques and statistical algorithms from belief networks. Both BSSET and MSE algorithms need their own hard-coded threshold value (\( C \)) but PILS auto-tunes its threshold parameters regularly.

Some others [33] use a model-based approach to build radio signal maps. They take more time to converge and require extensive war-driving to generate the data set. We do not assume anything about the underlying data model or distribution and hence takes a non-parametric approach. SLAM based approaches [61] have used Kalman filters and Markov chains for localization. Recent approaches have used particle filtering [62, 63] to good effect. Our
work [30, 42] uses a hybrid combination of cluster-based and statistical expectation maximization (EM) techniques.
Chapter 3

iSha: A Fine-Grained Energy Analyzer System using Assembler and Disassembler

The Internet of Things (IoT) paradigm aims to interconnect a variety of heterogeneous Smart Objects (e.g., sensors, smart devices, home automation equipment) using Machine-to-Machine communications. A simple illustration of how these smart objects have become intertwined with our lives is shown in Figure 3.1. A majority of consumer devices sold today come equipped with a variety of wireless communication technology (e.g., Bluetooth Low Energy, Wireless LAN, Near Field Communication, Cellular) to connect with the world-wide network. With billions of such devices predicted to connect with the future internet, bringing the physical data from these devices to the digital world requires energy-efficient methodologies and standard communication protocols. Devices that provide health monitoring, smart home and workplace, enterprise data security, and many others need to constantly sense their context and communicate with the network to collaborate with others.

A variety of the smart devices mentioned above are self-powered or connected to a power source glued to their components and hence the circuit terminals are not exposed to measure
the energy consumption. It is also to be noted that the main components in these devices (e.g., Bluetooth, Wi-Fi, NFC) are mostly based on open-source implementations. Hence, it is easily possible to obtain the executable code running inside those devices at run-time.

The overall idea behind this work is based on the fact that if the manufacturer provides the energy consumption measurements for the internal components at laboratory settings, then, by modifying the executable code at run-time and inserting log-triggers at specific places of the code will enable the developers to modify the working of the internal components and tune them to ensure minimal energy usage. This will also result in universal energy prediction values in these devices and reduce energy mismatch values between different researchers. We also intend to store the energy models in a database so that the results can be used to predict the energy consumption values at different real-life environment conditions. Here, we develop a novel and light-weight system, iSha [15], to measure the detailed power consumption patterns of various
system sub-components under different device screen states.

3.1 Motivation

The variability and complexity of hardware in smart devices has resulted in highly dynamic power variations as opposed to some existing power models with high baseline-power level and small dynamic range [13,14]. So, the overall system power increasingly depends on highly granular sub-component power measurements. A majority of customer applications utilize Wi-Fi and Bluetooth, both of which have become integral components in most low-power wearable computing and smart devices. While new standards (e.g., 802.11ad) continuously emerge, it provides new opportunities for application and protocol developers to modify existing functionality and improve the user experience in these devices.

In very recent times, the popular mobile operating systems (e.g., Android [2], Apple iOS [3], Windows Phone OS [4], Blackberry OS [64], BADA [65]) have come up with specific Application Programming Interfaces (API) to show detailed statistics about power consumption by individual system components. However, there is a need to develop a universal approach to accurately measure energy consumption of certain device system components through real world usage patterns and not just under laboratory settings.

To that end, we need to develop a new light-weight system to analyze and model the energy consumption patterns of these technologies at a very fine-grained level and be simultaneously applicable for multiple devices. As a result, the developers can directly evaluate and optimize the energy efficiency of their protocols/methods without the need for a power monitor and also bring about energy savings to all applications which use them rather than profiling individual applications. Thus the question we ask ourselves is, “Can we introduce a mobile OS-independent solution which can help researchers analyse energy consumption measurements in a variety of smart devices without using physical power monitoring devices?”.
3.2 Assembler and Disassembler

An assembler is a utility program used to convert the syntax for operations in a programming language and the mnemonics into the object code suitable for the hardware to run. A disassembler is a computer program to translate the machine language back into the assembly language. This operation is the inverse of what the assembler does. More information about assemblers and disassemblers can be found here [66,67].

There are a variety of assembler/disassembler programs for different operating system codes. In this work, we utilize smalli and backsmali as the assembler/disassembler module to work on Android .dex format files. smalli/baksmali is an assembler/disassembler for the dex format used by dalvik, Android’s Java VM implementation. The syntax is loosely based on Jasmin’s/dedexer’s syntax, and supports the full functionality of the dex format (annotations, debug info, line info, etc.). The source code for the programs used can be found in the Google open source code base here [68].
3.3 System Design

In this work, we develop a novel and light-weight system, iSha, to insert specific log triggers in the executable code using an assembler/disassembler module called smali/backsmali. The overall system architecture is shown in Figure 3.2. The system consists of three main processes: modifying the open source code of the internal phone component by the developer, modifying the runtime executable code inside the device dynamically using an assembler/disassembler module such as smali/backsmali, storing the energy measurement values and develop a energy model for future predictions. In these steps, it is assumed that the fine-grained energy measurements for individual sub-components in the phone or the device is available. The values can be published either by the manufacturer or a researcher who gets access to these measurement sheets or by way of manual measurement techniques.

Since most of the major modifiable components in these devices are open-source programs (e.g., wpa_supplicant for WiFi, bluetooth stack, NFC codebase), the developer alters the source code for a particular sub-routine to modify the functionality. During runtime, the assemble/disassembly takes places within the device and inserts log-triggers at specific places in the modified source code. By tracking the time information and the occurrence of log values, the system will automatically calculate the amount of energy consumed by a specific sub-module and displays it to the developer. In this way, it is possible for the developer to test the suitability of his code modifications in real world scenarios in addition to laboratory settings. Later, the measured detailed power consumption patterns of components under different device screen states are used to generate a model using stochastic approach. We also impart real-world data into the energy model for the developers to emulate “in-the-wild” variations from within their laboratory settings.
3.4 Evaluation

In this section, we will discuss about our specific case study where we implemented iSha to deduce the fine-grained energy measurements if the Wi-Fi sub system and the measurement results obtained after multiple runs.

3.4.1 Fine-grained Wi-Fi Energy Measurements

In a smart phone, a Wi-Fi scan is initiated in response to two actions: by turning on the screen or when an application specifically requests for a scan. When an AP is available to connect, the Wi-Fi driver scans the available channels and connects to the pre-configured AP as shown in Figure 4.1 (a). If no such AP is found in the pre-configured list, it periodically scans until the device is successfully connected to an AP or until a connection time-out occurs in the Wi-Fi driver after 15 mins.

The default time interval for consecutive scans vary between 5-30 sec in various \textit{wpa-supplicant} implementations. Upon screen off, the Wi-Fi radio chipset is turned off after a delay of 2 mins to avoid race conditions in the driver. CPU Wake locks are obtained for operations during screen off. While in connected state, if the link quality deteriorates, the Wi-Fi radio driver is kept in high power state constantly due to repeated scan and association requests. Also to avoid packet loss, the driver operates at lower modulation rates. Our measurements using a power monitor show the repeated scan/association operations in Figure 4.2. We start off by measuring the detailed power consumption patterns of Wi-Fi in mobile phones for different screen states (i.e., On, Off) under various Wi-Fi availability conditions (i.e., Good, Poor, Null) and data rates. The current power models do not consider such fine-grained variations, rather only consider the change in baseline power due to overall screen display brightness levels. Due to open source nature of Wi-Fi module (\textit{wpa_supplicant}) in Nexus One phones, we added logs in appropriate places to correlate the energy consumption with the specific system process.

The energy consumed for some important processes during the Wi-Fi start-up is shown in Table 3.1. These measurements account for the specific processes alone and do not include
the baseline system power. Hence, it captures all the dynamic power variations in the process including tail energy for the series of chipsets. Using iSha, there is no need for developers to use physical power monitoring devices and can dynamically deduce the change in power consumption measurements.

3.4.2 Procedure to Compile Platform Source Code

In order to modify the Wi-Fi sub-component and replace the binary, the modified source code should be compiled within the entire platform source code for Android. In the section, we provide details to compile the platform source code of Android as follows. Some of the details for new Android OS releases may be different than that provided below.

- Set up Build Environment. Get Python installed.
- Install JDK 6 if you want to use Gingerbread or newer
  
  - `sudo add-apt-repository "deb http://archive.canonical.com/ lucidpartner"`

– `sudo apt-get update`

– `sudo apt-get install sun-java6-jdk`

• Install all other required packages

• In this explanation, Linux version is Ubuntu 11.04. All following commands for ADB setup is for this version. For other lower versions, it may change. Test device is HTC Nexus One (Passion) running Linux kernel 2.6.32.x

• Get USB access to Linux System for using ADB (Android Debug Interface).

• Download Android source from git

  – Make sure you have a `bin/` directory in your home directory, and that it is included in your path.

    * `mkdir /bin`

    * `PATH = /bin:$PATH`

  – Download the Repo script and ensure it is executable.

    * `curl https://android.git.kernel.org/repo > /bin/repo`

    * `chmod a+x /bin/repo`

  – After installing Repo, set up your client to access the android source repository. Create an empty directory to hold your working files.

    * `mkdir WORKING_DIRECTORY`

    * `cd WORKING_DIRECTORY`

  – Run repo init to bring down the latest version of Repo with all its most recent bug fixes. You must specify a URL for the manifest, which specifies where the various repositories included in the Android source will be placed within your working directory.
* repo init -u git : //android.git.kernel.org/platform/manifest.git

− To pull down files to your working directory from the repositories as specified in the default manifest, run repo sync.

− To compile the code after setting up all the environments including adb in your virtual machine, run the following in order.
  
  * Run Nexus one script. Your phone should be connected with the virtual machine.
    
    · cd WORKING DIRECTORY
    · cd ./device/htc/passion/
    · ./extract – files.sh
  
  * build the setting
    
    · cd WORKING_DIRECTORY
    · repo sync –j16
    · . build/envsetup.sh
    · lunch full_passion – userdebug
  
  * To enable Debug Logs printed in wpa, change the log level to MSG_DEBUG from MSG_INFO at line 23 of the file wpa_debug.c

  * Now build it at WORKING_DIR. It takes around 2 hrs for initial build.
    
    · cd WORKING_DIRECTORY
    · make –j16
  
  * The wpa_supplicant binary can be found under /out/target/product/passion/system/bin/

### 3.4.3 Procedure to Replace Supplicant Binary

We made changes to sub-modular functions such as scanning, authentication, association etc and found that the final energy consumption values obtained using iSha remained within 2% of the actual power monitor. The prototype implementation is performed in HTC Nexus One phones
since the Wi-Fi functionality is open-sourced and is easily available without any additional 
OEM modification. The steps are given below:

- Copy the wpa_supplicant binary file to SDCARD
- In the HTC Nexus One phone, in “wireless and networks” menu, disable WiFi by uncheck-
ing the checkbox
- Make sure you have rooted the phone and have given it “super-user” access
- Open a command window from within the “platform-tools” directory in your device
- Type the following commands in order
  - adb shell
  - su
  - mount –o rw,remount –t yaffs2 /dev/block/mtdblock3 /system
  - chmod 777 /system/bin
  - cat /sdcard/wpa_supplicant_nithy > /system/bin/wpa_supplicant
- Collecting the logs on Android phone without USB cable normally is achieved via following 
  commands
  - adb wait-for-device shell
  - logcat –v threadtime > /data/logcat.log &
- The power values and timing measurements of logs from the wpa_supplicant can be 
obtained using advanced commands such as below
  - logcat –v threadtime power : I wpa_supplicant : I WifiStatetracker : D >
    /data/logcat_nithy.log
  - logcat –v time power : I wpa_supplicant : I WifiStatetracker : D > /data/logcat_nithy.log
Table 3.1: Fine-grained energy measurements on Nexus One.

<table>
<thead>
<tr>
<th>Item</th>
<th>Energy (µWh)</th>
<th>Screen On</th>
<th>Screen Off</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radio Up</td>
<td>79.90</td>
<td>100.10</td>
<td></td>
</tr>
<tr>
<td>Scan</td>
<td>83.40</td>
<td>118.50</td>
<td></td>
</tr>
<tr>
<td>Association</td>
<td>77.10</td>
<td>108.00</td>
<td></td>
</tr>
<tr>
<td>DHCP</td>
<td>28.90</td>
<td>53.90</td>
<td></td>
</tr>
<tr>
<td>Radio Down</td>
<td>39.70</td>
<td>59.40</td>
<td></td>
</tr>
</tbody>
</table>

Although [69] has previously utilized system call tracing for power modeling, it incurs high kernel-level logging overhead (both log memory and energy) as opposed to our approach which is light-weight. The Wi-Fi usage patterns from different devices are logged on to a centralized database and we build an effective energy consumption model suitable for most devices based on logged process events. Thus, we expect to reduce the error from complexity and hidden states. The developers can later obtain logs from a variety of users with various mobility and usage patterns and apply the energy model to evaluate the energy efficiency of their methodologies at various instances and can also emulate “in-the-wild” variations from within the laboratory settings.

### 3.5 Discussion

The main assumption used in this work is that the energy values provided to by the database is accurate. If the database gets corrupted due to inaccurate energy measurement values, the upcoming results are also compromised. However, we believe this work is an effort to provide an option for developers to understand the energy consumption characteristics of their programs who otherwise have none available to them.

Although, we developed a prototype system to accumulate energy consumption values for different sub-processes within a sub-system, it is complex and difficult to identify all the important sub-processes and model the transition between different energy states. Hence, much
work is needed in to developing a powerful database which has information about all sub-processes and hence can avoid hidden states.

Wi-Fi sub-component is an open source implementation however, smart phone vendors highly customize the open source code (e.g., wpa_supplicant) and introduce new functionality. This can disrupt the embedding of log-triggers in the assemble code and hence can result in faulty time logs. This can easily be solved if we get to access the vendor modified executable files. Another feature which needs optimization is to minimize the logging overhead. There is a possibility of high system overhead (both log memory and energy) in the low-memory wearable devices as present in system-call tracing techniques.

It is also imperative to identify the battery age of a device and suitably re-calibrate its energy consumption values. Hence, the database needs to be properly segregated based on the lifetime of the devices. We believe that in future work, with proper crowd-sourcing techniques, some of the above optimizations can be easily performed and iSha can play a bigger role in energy measurements with the multitude of wearable devices entering the market.

### 3.6 Concluding Remarks

In this work, we develop a novel and light-weight system, iSha, to insert specific log triggers in the executable code using an assembler/disassembler module. We measure the detailed power consumption patterns of components under different device screen states and generate a model using stochastic approach. We also impart real-world data into the energy model for the developers to emulate “in-the-wild” variations from within their laboratory settings.

With more and more chipsets having integrated or combo radios, we will extend our work to other components such as Bluetooth and NFC to obtain a comprehensive energy consumption model for these low power communicating technologies which are more suited for the wearable infotainment devices and the Internet of Things (IoT) architecture.
Chapter 4

PRiSM: Wi-Fi Hotspot
Auto-Discovery System for Smart Objects

Recent proliferation of smart devices with varying form factors and attractive pricing has met with increased customer adoption of these devices. Global shipments of smart phones have already surpassed those of conventional personal computers [1], and their numbers keep increasing each day. A plethora of applications specific to various smart phone platforms (e.g., Android [2], iOS [3], Windows [4]) are created to enhance the user experience in these devices. The applications range from a simple calendar manager to complex business productivity software suites. An ever increasing majority of these applications expect always-on internet connectivity (e.g., iCloud [5], other cloud-based services) so as to provide unlimited storage and processing capabilities. Dedicated cameras nowadays come with Wi-Fi interfaces.

Smart devices have become one of the primary ways for people to access entertainment [6] (e.g., social networking, communication and browsing applications), both inside and outside of their homes. This has led to two significant problems: substantial increase in monthly data usage [7], and a rapid drain in smart phone battery life.
The cellular network carriers are struggling to keep pace with the increased data generation and consumption from these devices by utilizing alternate ways of data transfer (e.g., Wi-Fi data offloading) so as to mitigate network congestion. Network operators see Wi-Fi as a cost-effective means to offload large amounts of cellular data due to the globally available spectrum capacity and widespread existing deployments of Wi-Fi. Furthermore, new deployments can be easily made in locations (e.g., transportation hubs, shopping malls) which generate substantial user traffic. To reduce the cellular data costs and to enjoy increased data rates, customers too are willing to connect to public Wi-Fi hotspots. Wi-Fi has also been proven to consume less energy than cellular technologies (e.g., 3G, 4G) for data downloads.

But to connect to a hotspot, there is a need for constant scanning of Wi-Fi access points (APs) in these devices and results in undesired battery drain. To design an accurate and an energy-efficient Wi-Fi sensing system is (still) a very non-trivial task. The reasons include: almost 60% of battery drain in smart devices result from Wi-Fi [70], not all public Wi-Fi hotspots offer good connectivity and leads to poor user experience [36], frequent disconnection and re-association events with APs incur high energy costs than normal. Here, we develop a new Wi-Fi detection system, PRiSM [30] (Practical and Resource-aware Information Sensing Methodology), which utilizes the freely available cellular signal information of GSM signals to statistically map the Wi-Fi APs with a logical location information.

### 4.1 Motivation

Prior works with Wi-Fi detection mechanisms used optimal scanning intervals for Wi-Fi to identify hotspots [9, 21]. The scanning intervals are increased or decreased based on parameters like AP inter-arrival time, AP density and user velocity. Since the Wi-Fi connectivity times and movement patterns vary among users, these methods do not adapt well for all users. Wi-Fi signal fingerprinting techniques [32, 33] use extensive offline pre-processing stage to construct signal strength models and to calibrate the radio maps. Also, the techniques take more time to converge.
Multi-modal sensing techniques (e.g., Accelerometers [9], GPS [22–24], Bluetooth [25], Zigbee [26]) are also developed to identify context. Few others use average received signal strengths from connected cellular base stations to predict user location [71–73]. However, averaging the signal strength values results in loss of granularity and use of additional sensors consume significant extra battery energy (e.g., Accelerometers consume close to \(0.667\, \text{mWh every 30 sec}\) [27]). Some require infrastructural changes and extensive war-driving efforts to obtain feature-rich data sets. Also, most commercial systems (e.g., WiFi Sense [34], Place Lab [35]) turn on the radio interfaces continuously to identify context which results in excessive battery drain where Wi-Fi scan/association is observed to have high initial costs [11,12].

Upon observing the existing solutions, we sense a need for a new system which is light-weight and does not require extensive data pre-processing. It should consume minimal battery energy, provide ways to continuously accommodate the signal fluctuations, and be easily deployable in real world. Thus the question we ask ourselves is, “How can we maximally discover Wi-Fi APs in a practical and energy-efficient way with zero extra sensing costs?” Given that the Wi-Fi scanning and transmission incur the same energy [71], this question draws more attention.

4.2 Design

4.2.1 Wi-Fi Power Consumption

In a smart phone, a Wi-Fi scan is initiated in response to two actions: by turning on the screen or when an application specifically requests for a scan. When an AP is available to connect, the Wi-Fi driver scans the available channels and connects to the pre-configured AP as shown in Figure 4.1 (a). If no such AP is found in the pre-configured list, it periodically scans until the device is successfully connected to an AP or until a connection time-out occurs in the Wi-Fi driver after 15 mins.

The default time interval for consecutive scans vary between 5-30 sec in various \(wpa_{-}\text{supplicant}\) implementations. Upon screen off, the Wi-Fi radio chipset is turned off after a delay
of 2 mins to avoid race conditions in the driver. CPU Wake locks are obtained for operations during screen off. While in connected state, if the link quality deteriorates, the Wi-Fi radio driver is kept in high power state constantly due to repeated scan and association requests. Also to avoid packet loss, the driver operates at lower modulation rates. Our measurements using a power monitor show the repeated scan/association operations in Figure 4.2.

When there is no AP available to connect, the Wi-Fi radio driver scans continuously and results in energy wastage (Figure 4.1 (b)). The energy consumed by the Wi-Fi radio under various screen conditions and AP availability conditions is shown in Figure 4.3. Thus, PRiSM can save substantial energy by intelligently avoiding poor and no Wi-Fi conditions in an accurate manner.

### 4.2.2 Cellular Signatures

We investigate the feasibility of constructing a database using the statistical information of cellular signals for each Wi-Fi AP and the ability to distinctly identify the APs in the database based on their signatures. Cellular signals are ubiquitous in nature and are received continuously
Figure 4.2: Repeated scan/association events under poor AP signal when the device screen is (a) ON, (b) OFF.

Figure 4.3: Default Wi-Fi energy consumption for one minute under various screen activation conditions.
Figure 4.4: The evolution of signal strength distributions from the most frequently connected base station for 3 different APs are depicted in (a), (b), and (c). For each AP, the data is aggregated over time whenever connected with the AP.
by the phones. A smart phone can receive signals from more than ten base stations (BSs) in dense urban areas [74]. GSM based Android phones can overhear signals from up to seven (six neighbouring and one connected) BSs in ASU (Active Set Updates) units at any time instant. The linear equation between dBm and ASU values for GSM networks is $dBm = 2ASU - 113$. ASU values range from 0 to 31 and 99, which indicates unknown signal strength. The total time interval of observation of every base station within the signature differs and depends both on the total time spent by the user while connected to the particular Wi-Fi and also on the occurrence pattern of the base station.

To capture the entire signal characteristics that a user uniquely experiences for an AP, we propose to build cellular signal signatures using “probability distributions” of signal strengths from all observable connected and neighbor base stations rather than using abstracted information (e.g., “average signal strengths”). A Wi-Fi signature is defined as the set of probability density functions (PDFs) of signal strengths from all connected and neighbor Base Stations (BS) when the smart phone is associated with that unique Wi-Fi AP. We performed the statistical measurements for all users in our dataset, but for explanation purposes, we will take random users to show the following results. Figure 5.4 shows the evolution of signatures recorded by a user over time for three Wi-Fi APs to which the user has connected most frequently. For better readability, we plotted only the signal strength distribution from the most frequently connected BS per Wi-Fi AP. The figure shows the PDF of signal strengths received from the connected BS at different intervals of time. Simply put, the distribution shown after 10 hrs includes the data used for the distribution shown at 5 hrs plus five more hours. Note that the signal strength distributions do not converge to a Gaussian distribution even after 25 hrs of signal accumulation. Hence, we develop a non-parametric algorithm which does not assume anything about the underlying data distribution. The correlation coefficient ($\rho_{X_1,X_2}$) between probability distributions accumulating signals for different amounts of time clarifies the existence of characteristic patterns in the signatures. High value of correlation coefficients for signatures after 25 hrs of signal accumulation and low cross-correlation values indicate that our statistical
Figure 4.5: The personalized signatures for three APs: (a) $AP_X$, (b) $AP_Y$, and (c) $AP_Z$. The distance between $AP_X$ and $AP_Y$ is about 7 km, $AP_Y$ and $AP_Z$ is about 30 meters. $AP_Y$ and $AP_Z$ are located in the same building. The observed base station IDs and their average signal strengths are given in the legend.
technique is likely to provide good performance in matching accuracy.

Figure 4.5 further shows that the signatures recorded by a user for different APs located far from or near to each other have significant dissimilarities. We again choose three Wi-Fi APs: \(AP_X\), \(AP_Y\), and \(AP_Z\) from a user’s database, where distances between \(AP_X\) and \(AP_Y\) is about 7 km and between \(AP_Y\) and \(AP_Z\) is about 30 meters (\(AP_Y\) and \(AP_Z\) are in the same building). In the figures, base station IDs and their average signal strengths are given in the legend. As expected, the signatures for \(AP_X\) and \(AP_Y\) contain completely different sets of BSs and different patterns of signal distributions. On the other hand, the signatures for \(AP_Y\) and \(AP_Z\) show similar sets of BSs. However, they are still distinguishable because the signal distributions show unique patterns. Considering the possible differences in the environment and the behaviour of a user, observing dissimilar signal distributions even for nearby APs is not surprising and actually helps to identify the APs more reliably.

4.2.3 Existing Localization Algorithms

A class of algorithms (referred as \(BSSET\)) uses the set of cellular BS ID’s to evaluate the likelihood of matching a fingerprint in the database. It can simply count the number of common BSs or can sum up the weight values of common BSs, where the weight is assigned to each BS based on its frequency of observation. Another set of algorithms (referred as \(MSE\)) use mean squared error for matching [59], [60]. An error is defined as the difference between the signal strength in current observation and the average signal strength recorded in the fingerprint for the same BS.

Most Artificial Intelligence (AI) techniques typically identify the top \(k\) fingerprints showing the smallest MSE values and then calculate the center from the locations paired with \(k\) fingerprints. This extension is called \(kNN\) (k-nearest neighbor) but they have the following problems: minimal training phase but costly testing phase including both time and memory, and assumes that data is in feature/metric space which means it is associated with some distance. PRiSM requires entire signal distribution clusters of the training data for quicker prediction.
and hence, uses a specialized hybrid algorithm which includes lazy learning techniques and statistical likelihood estimation. Both BSSET and MSE algorithms need their own hard-coded threshold value \( C \) but PRiSM auto-tunes its threshold parameters regularly. Some others [33] use a model-based approach to build radio signal maps. They take more time to converge and require extensive war-driving to generate the data set. PRiSM does not assume anything about the underlying data model or distribution and hence takes a non-parametric approach.

### 4.2.4 Proposed Algorithm

We design an algorithm, ATiS, that can utilize detailed statistical properties of cellular signals instead of the averaged signal strength values. A simplified version of ATiS (Automatically Tuned Location Sensing) is explained in Algorithm 2. Since the entire signal distribution is available, ATiS predicts the location in near real-time. A higher level intuition of the algorithm is that if the probability of seeing a particular signal strength within the PDF of a base station (BS) is high and the probability of the BS observed when connected to an AP is high, the total joint distribution is maximized and we get a more accurate signature match.

ATiS utilizes a set of signatures \( P \) each consisting of a set of base stations \( R_j \) and corresponding signal strength distributions \( f_{k,j}(S) \), where \( k \in R_j \) and \( j \in P \). Note that \( j \) and \( k \) are signature ID’s (e.g., Wi-Fi AP) and cellular base station ID’s respectively. Each signature \( P \) has information pertaining to the number of occurrences made by its individual base stations in \( n(k,j) \) and the total occurrences of all its base stations collectively in \( N_j \). At any time interval \( t \in [t_1, t_2] \) during the testing phase, the signal observed from a particular base station \( k \) is measured to be \( s_k(t) \). For any signature \( j \) which has observed this particular unique base station \( k \) over the course of its training time period, the likelihood of occurrence of the currently observed signals from the base station \( k \) is calculated as \( v(k,j) = (\prod_{i=t_1}^{t_2} f_{k,j}(S = s_i(k))) \). Similarly, the likelihood is calculated for every base station \( k_1, k_2, \ldots k_n \) which is observed during the time frame of measurement \( t \) and which matches within the signature database. The overall maximum likelihood score of simultaneous occurrence of all such base stations within a particular signature
Algorithm 1: **ATiS Signature Score Generation**

1: **INPUT:** Signature database for all Wi-Fi APs connected by the user
2: **INPUT:** Set of currently observed BSs and their corresponding signal strengths at time $t$
3: **INPUT:** Hashmap of unique Wi-Fi APs and reverse Hashmap of observed BS IDs to APs for fast lookup
4: **OUTPUT:** List of Wi-Fi APs in descending order of likelihood

5: **Step 1.** For given input BS, look-up the reverse Hashmap to identify the signature cluster subset to reduce computation

6: **Step 2.** Calculate the score for the individual signatures
7: for all signatures in cluster subset do
8: for all Base Station ID’s within signature do
9: if Base Station ID exists in input at time($t$) then
10: if Requested signal strength bin is Empty then
11: Normalize ‘$x$’ adjacent bins
12: end if
13: Evaluate likelihood of occurrence using expectation maximization from Bayesian-based approach
14: end if
15: end for
16: Accumulate final likelihood scores for all signatures
17: end for
18: **Step 3.** Apply the lower and upper bound thresholds ($[C_L, C_U]$) on generated scores
19: **Step 4.** Return Wi-Fi APs which satisfy the thresholds
20: **Step 5.** Check with the ground truth and update the signature thresholds if needed

$j$ is then calculated as $s(j) = \left( \prod_{k \in P(j)} v(k, j) \right)$. For any input BS, ATiS does a local normalization of signal strength values surrounding the target signal strength in the database and hence, performs well even under signal fluctuations. For example, assume in the signature database, a Wi-Fi location has recorded signal distribution for a BS having signal strength values only for $asw$ values 17, 18, 20, and 21 out of the possible 0 to 31 values. During testing phase, if the input signal strength for the same BS is 19, ATiS does not mark the probability of finding the signal strength 19 as zero, instead, it normalizes the values of signal strength bins 18, 19 and 20. If all the requisite bins (here 18, 19 and 20) are empty, ATiS normalizes the expected value to be $1/|n(k, j)|$. ATiS pre-emptively calculates the value ahead of database update because after the current estimation time period $t$, the observed signals for this particular base station will be updated in the database. Hence, ATiS can perform well even under slight signal variation conditions. The closer the match of input base
stations within a signature, the better is the score for the Wi-Fi. All signatures whose likelihood scores \( s(j) \) satisfy the lower bound \( (C_L) \) and upper bound \( (C_U) \) thresholds are returned as output in descending order of their scores. Note that the values of \([C_L, C_U]\) are initialized with \([1, 0]\) initially and are decreased or increased over time to achieve a tight threshold range. The novel part of ATiS is that it auto-tunes thresholds within \( 0 - 1 \) based on likelihood scores by checking the ground-truth (i.e. Wi-Fi AP (un)available) after each connection attempt and hence, does not overfit the data for any particular scenario. Also, by design, PRiSM utilizes a *cluster-reduction approach* to only compare the currently received signals with a small subset of the signatures in the database irrespective of the total database size and saves on computation time to compare from all the signatures otherwise.

### 4.3 Implementation

#### 4.3.1 Architecture

The primary modules of PRiSM as shown in Figure 4.6 include: *PRiSM Manager* at the application layer and *PRiSM Controller* at the platform layer of the Android stack. The manager runs in the system background and builds a list of unique signatures (inside the phone for privacy) for all connected Wi-Fi APs through the trainer service. The sensing service overhears the cellular signals at programmed time intervals to predict AP availability. The
Figure 4.7: PRiSM operation includes three tasks: bootstrapping, signature matching, and online training.

decision engine ranks the scores from the Bayesian network based algorithm and outputs the result. The controller implements a novel selective-channel Wi-Fi scanning framework to connect to APs directly without scanning or association via `wpa_supplicant` module in the phone system. It uses appropriate frequency channel information of APs stored in the database. The existing configuration file `wpa_supplicant.conf` is intelligently modified at runtime to provide access to the manager and the controller simultaneously. Hence, PRiSM can serve as a middleware for all Location Based Service (LBS) applications in the smart phone. PRiSM suppresses Wi-Fi connection to an AP in poor signal strength regions and when the user moves closer to the same AP, it automatically matches the good signature of the AP and connects to it.

4.3.2 Operation

The three important tasks performed by PRiSM is shown in Figure 4.7 and they include: bootstrapping, signature matching, and online training. **Bootstrapping** is the first process when a signature database is created for every user for the first time. Here, an *event* represents the process of connecting to a Wi-Fi AP. Since most people show regular movement patterns on a weekly basis [75], the signatures are continuously updated as time evolves but most signatures get stabilized quickly within a week. The process of computing the likelihood score for an AP from all matching signatures and threshold parameters is called as **Signature Matching**. The decision engine notifies the Wi-Fi on/off decision along with the AP channel information to the Wi-Fi controller within a sub-second time period. ‘LBS’ (Location Based Service) applications,
though not a main part of PRiSM operation, is shown here (shaded in Figure 4.7) since PRiSM also can serve as a middleware for all such applications in the smart phone. Only upon successful connection to an AP, we enter Online Training through which the signature database is kept up-to-date. It is done to capture environmental changes such as configuration updates in an AP, changes in indoor signal propagation paths and behavioural changes in the user. PRiSM suppresses Wi-Fi connection to an AP in poor signal strength regions and when the user moves closer to the same AP, it automatically matches the good signature of the AP and connects to it.

When PRiSM predicts an AP, it tries to connect to the AP even without scanning. If the ground truth (checked by connecting to the AP after every prediction) has an AP (i.e., true positive), the connection attempt becomes successful and hence reduces the time to connect to an AP by 33.7%. If the ground truth has no AP (i.e., false positive), the connection attempt will be unsuccessful and it auto-tunes the threshold parameters. PRiSM predicts no AP under two conditions: Zero Match (i.e., overheard BS ID’s do not match with any stored Wi-Fi signature) and Threshold Mismatch (i.e., overheard BS ID’s matched with some Wi-Fi signature but failed to satisfy the threshold parameters). In the case of zero match, PRiSM assumes the user is in a new place and scans all channels once to provide the results to the user. Here, it simultaneously aids for user experience and reduces energy on repeated scans until the user decides to connect to any AP. In the case of threshold mismatch, it first scans only those channels associated with its known list of APs in the database. If the scan results match with an AP in the database (i.e., false negative), it connects with the AP and simultaneously tunes its threshold parameters and hence saves energy instead of scanning all channels. If no match is found (i.e., true negative), PRiSM stops further scans and turns off the Wi-Fi interface to save energy from excessive unnecessary scans.
4.3.3 Cost Analysis

Cellular signals are received and processed all the time by the phone MODEM at no extra cost. PRiSM activates the CPU only to read cellular signal values from the MODEM and to compute using ATiS. At all other times, CPU is not activated by PRiSM and consumes negligible energy (0.6 – 1.1 µWh) on top of CPU base energy. The sampling policy is shown in Table 4.1. The overall energy costs for continuous Wi-Fi sensing using PRiSM is minimal when compared to normal Wi-Fi scan. Using the reverse hashmap, the signatures are computed only for the MACs with current observed BS IDs. Hence, PRiSM only compares the currently received signals with a small subset of signatures in the database irrespective of the total database size and saves on computation time. Thus the space and time complexity needed for computation is a function of the density of APs in the nearby environment and is almost constant. In our traces, the signature comparisons never exceeded 35 even though some users had up to 337 unique signatures stored in their database. Hence, PRiSM is more robust to handle database explosion.

4.4 Evaluation

In this section, we will provide information about the datasets used in the experiments and the results obtained. The simulation and practical verification results are separately discussed.

Table 4.1: PRiSM cellular signal sampling policy.

<table>
<thead>
<tr>
<th>Screen</th>
<th>Wi-Fi State</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disconnected</td>
</tr>
<tr>
<td>ON</td>
<td>1 sample every 20 sec</td>
</tr>
<tr>
<td>OFF</td>
<td>1 sample every 20 sec</td>
</tr>
</tbody>
</table>
Table 4.2: Dataset information.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># of Volunteers</th>
<th>Total hours</th>
<th>Avg. Wi-Fi %</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>24</td>
<td>2592</td>
<td>89.6</td>
</tr>
<tr>
<td>D2</td>
<td>16</td>
<td>1440</td>
<td>81.3</td>
</tr>
</tbody>
</table>

4.4.1 Datasets

We obtained Institutional Review Board (IRB) approval from North Carolina State University to gather datasets (Table 4.2) from Android based devices running our customized monitoring application. Data was collected for over two weeks from graduate students (29), undergraduate students (6), and employees (5). Undergraduate students predominantly covered locations within the campus. Graduate students had both on-campus and off-campus locations. Each employee data is from a different urban city in the US. The Android application using which the data was collected is shown in Figure 4.8. Since, the users were paid according to the number of hours they logged, the logging service also informs the number of hours the user has logged since the start. This is shown in Figure 4.9.

Dataset ‘D1’ is obtained from our lab Nexus One phones used by volunteers as their primary device. It includes timestamp, Wi-Fi signal statistics for connected and neighbor APs, screen unlock info, and cellular signal statistics for connected and neighbor BSs. Dataset ‘D2’ is obtained from personal phones of volunteers due to non availability of test phones in large numbers. It includes screen on/off information in addition to screen unlock information present in ‘D1’ but lacks neighbor BS information due to the closed nature of GSM API found in those phones. In both datasets, cellular signal and screen information are recorded at each second and Wi-Fi information at each minute. Since fine-grained screen activity information is required to accurately predict energy savings, we use ‘D1’ to analyze the algorithm accuracy and apply those parameters (false positives, false negatives, etc) to ‘D2’ to predict energy savings. The devices recorded up to 35 APs in some campus locations. Also, the students recorded higher number of signatures for unique APs (up to 337) than the employees due to their movement.
patterns and the number of unique locations visited throughout the data collection period.

Figure 4.8: (a) shows the launch screen of the service where user information is gathered, (b) shows that the system service runs constantly in the background and logs all required information.

4.4.2 Accuracy Measurements

A trace-driven simulator builds the signatures and evaluates the accuracy of the algorithms by checking with the ground truth values in the dataset. The robustness of an algorithm depends on the proportion of true positives and true negatives correctly identified. The Receiver Operating
Figure 4.9: The number of hours logged by each individual user is shown during data collection phase.

Characteristic (ROC) curves shown in Figures 4.10 (a) and (b), the diagonal line represents the random prediction of an algorithm using a large random sample dataset, points above and below the diagonal represent good and bad prediction accuracy. ATiS obtains higher percentage of true positives and true negatives compared to other algorithms because ATiS uses the entire signal distribution from BSs and auto-tunes its threshold parameters as time evolves by adjusting itself to signal variations. However, other class of algorithms (BSSET and MSE are discussed in § 4.2.3) use average signal strength values from BS and persistent threshold values which either overfit or underfit the data. Though the results are shown from a single user for clarity,
Figure 4.10: (a, b) ROC curves and (c) $\rho_{FP}$ Vs. $\rho_{FN}$ values for a randomly selected user for all algorithms in dataset ‘D1’. ATiS achieves very high true positive and true negative values and very low $\rho_{FP}$ and $\rho_{FN}$ values simultaneously.
we observed a similar pattern across all users in the dataset. 

False positive rate (FPR) is defined as the ratio of number of false positives over the sum of false positives and true negatives. True positive rate (TPR) is defined as the ratio of true positives over the sum of true positives and false negatives. Similarly True negative rate (TNR) is defined as the ratio of number of true negatives over the sum of true negatives and false positives. False negative rate (FNR) is defined as the ratio of number of false negatives over the sum of false negatives and true positives. 

False positive ratio ($\rho_{FP}$) is defined as the number of cases that an algorithm detects an AP when there is no such AP in the ground truth divided by the total number of cases. Similarly, false negative ratio ($\rho_{FN}$) is defined as the number of cases that an algorithm detects no AP when there is an AP in the ground truth divided by the total number of cases. Higher $\rho_{FP}$ indicates losing more chances for energy saving and higher $\rho_{FN}$ indicates losing more connection opportunities. Figure 4.10 (c) shows that BSSET and MSE class of algorithms require very high

![Figure 4.10](image.png)

Figure 4.11: (a) Average $\rho_{FP}$ and $\rho_{FN}$ for users in dataset ‘D1’ and (b) $\rho_{FP}$ and $\rho_{FN}$ for 5 consecutive days for a user.

threshold values to achieve lower $\rho_{FP}$ values, which results in undesired higher $\rho_{FN}$ values. ATiS achieves lower $\rho_{FP}$ and $\rho_{FN}$ values simultaneously and hence results in minimum lost
opportunities for connection with maximum energy saving. Figure 4.11 (a) shows the variation in mean $\rho_{FP}$ and $\rho_{FN}$ values for individual users over their entire evaluation period suggesting the difference in their mobility patterns and the places they visit. The overall $\rho_{FP}$ and $\rho_{FN}$ values for all the users in the dataset ‘D1’ averaged to 1.10% and 0.19%, which is very close to zero (ideal value). Since PRiSM starts predictions from day-1, we show the variation in the values after each day in Figure 4.11 (b). Hence, even with small number of samples in acquired data during the initial few days, ATiS keeps the false positives and false negatives low and it further improves as time evolves.

4.4.3 Energy Measurement Setup and Calculations

Energy measurements are obtained from Monsoon power monitor [76] with values recorded every 200 $\mu$s. For practical purposes, we avoid using mobile power monitors as in [72]. The setup of the Monsoon Power Monitor connected to the phone is shown in Figure 4.12. Since PRiSM modifies default Wi-Fi connection framework, we obtain fine-grained energy information for important Wi-Fi processes as shown in Table 4.3. The Wi-Fi measurements are obtained by subtracting the background energy (which includes CPU, LCD, and backlight) from total consumed energy. Extensive trials are performed using an automated program to avoid finger
touch events on the LCD screen and to avoid sensitive fluctuations in power consumption. We also remove all background processes and turn off other sensors not associated with the Wi-Fi to avoid energy variations. For trace-based simulation, we first extract Wi-Fi event information (e.g., radio-enable, scan, authentication) for various screen activity conditions recorded in the dataset and combine with practical usage values in Table 4.3 to accurately calculate the total energy consumption by Wi-Fi usage specific to each particular user for each day.

Table 4.3: Fine-grained measurements for Wi-Fi sensing.

<table>
<thead>
<tr>
<th>Item</th>
<th>Energy Consumption (mWh)</th>
<th>HTC Nexus One</th>
<th>Samsung Galaxy S5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>Screen</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wi-Fi Radio Up</td>
<td>0.0943</td>
<td>0.1181</td>
<td>0.2528</td>
</tr>
<tr>
<td>Wi-Fi Radio Down</td>
<td>0.0405</td>
<td>0.0606</td>
<td>0.0510</td>
</tr>
<tr>
<td>Scan</td>
<td>0.1376</td>
<td>0.1955</td>
<td>0.5333</td>
</tr>
<tr>
<td>Auth/Assoc</td>
<td>0.1588</td>
<td>0.2711</td>
<td>0.2570</td>
</tr>
<tr>
<td>PRiSM Active</td>
<td>0.0019</td>
<td>0.0173</td>
<td>0.0015</td>
</tr>
<tr>
<td>Wakelock</td>
<td>NA</td>
<td>0.0241</td>
<td>NA</td>
</tr>
<tr>
<td>CPU Normal</td>
<td>0.2706</td>
<td>0.0059</td>
<td>0.0871</td>
</tr>
</tbody>
</table>

4.4.4 Default Wi-Fi Vs. Footprint Vs. PRiSM

In Figure 4.13, we compare the energy consumed by PRiSM with three other Wi-Fi sensing systems: Default Wi-Fi refers to Wi-Fi in off-the-shelf phones, Footprint refers to the Wi-Fi sensing system in [71], and Ideal refers to the imaginary oracle sensing system introduced for user clarity. We define the characteristics of an ideal system as: uses zero system/CPU energy to identify Wi-Fi APs, connects automatically to the APs without scanning, and shuts down Wi-Fi radio immediately in places where Wi-Fi is absent. PRiSM implements a full version of a sub-optimal algorithm, PRiSM-SubOpt, which scans for APs before connection and a prototype version of an optimal algorithm, PRiSM-Opt, which knows AP channel information and connects directly without scanning.
Figure 4.13: Wi-Fi energy consumed every minute for (a) screen ON, (b) screen OFF, and (c) under poor Wi-Fi signals. For Footprint, $\Delta 1$ is estimated to be $0.673 \text{ mWh}$ for screen on and $\Delta 2$ is estimated to be $0.719 \text{ mWh}$ for screen off conditions.
Footprint triggers scan based on distance moved by the user (more than 10 m indoors or 20 m outdoors). In no Wi-Fi areas, Footprint scans for Wi-Fi first and later records all places which do not have Wi-Fi bloating its history list. Even in areas with Wi-Fi, unless the user moves, it does not scan even if the Wi-Fi radio is turned off after screen off delay. PRiSM, however, checks for Wi-Fi availability every sampling period and connects to Wi-Fi if needed, else, it maintains the radio in off state. Also, it connects directly to the AP without scanning and avoids turning on Wi-Fi in poor signal strength areas and hence saves energy intelligently. Since energy measurements for Footprint is not available and implementing the entire system is out of scope in our experiments, we combine the accelerometer energy value (0.667 mWh) obtained in [27] and our own test measurements to sample cellular signals thrice (0.006 mWh for screen-on and 0.052 mWh for screen-off) to calculate the additional cost incurred by Footprint in both screen on (Δ1) and off (Δ2) conditions to be 0.673 mWh and 0.719 mWh per minute. For a stationary user, Footprint effectively suppresses Wi-Fi scans in no Wi-Fi areas, but still incurs the overhead energy from accelerometer usage, which is significantly high compared to PRiSM. For a moving user, Footprint consumes more energy than default Wi-Fi and PRiSM. When Wi-Fi is available, PRiSM-SubOpt consumes slightly higher energy than default Wi-Fi since it uses extra energy for cellular overhearing. However, PRiSM-Opt always consumes less energy than default Wi-Fi by design. The Ideal system always consumes the lowest possible energy and provides a baseline to compare for the maximum amount of energy that can be saved by any Wi-Fi sensing system.

4.4.5 Effect of Sensing Intervals (δ) and Wi-Fi Thresholds (τ)

The energy consumed by PRiSM and an Ideal system for various sensing intervals (δ) and Wi-Fi thresholds (τ) is discussed here. We do not compare Footprint because of the non-availability of accelerometer values in our dataset. Also, PRiSM does not measure distance from APs or discriminate between indoor and outdoor locations. The energy savings vary between users and depends on their individual mobility patterns and Wi-Fi availability (e.g., Users who often
experience poor and no Wi-Fi situations save more energy than users who experience good Wi-Fi. The reason is in good Wi-Fi areas, the only avenue to save energy is to avoid scan costs). We define battery capacity as the maximum amount of energy that can be extracted from a smart phone battery and is assumed to be 5000 mWh in our energy calculations.

Sensing interval of $\delta = 1 \text{ sec}$ is equivalent to keeping the Wi-Fi interface continuously ON. When $\delta$ increases, the average battery saving for all users combined decreases steadily as shown in Figure 4.14 (a). The decrease in energy saving from that of 1 sec scanning is because of following reasons: scan is not performed continuously and during the time slots (e.g., $\delta = 30 \text{ sec}$, 45 sec.. 5 min), only 20 sec of time slot is utilized for sensing operations and the Wi-Fi radio is turned OFF for remaining time. Email sync applications are shown to take close to 18.54 sec [77]. Hence, we assume a constant time of 20 sec for the purpose of evaluation and can be varied if necessary. We see that PRiSM-Opt achieves close to 96% in average battery savings to that of an ideal system. The variation in average battery savings for all users for different thresholding values ($\tau$) is shown in Figure 4.14 (b). The decrease in savings with smaller thresholds ($\tau = -90$) is due to increased energy usage to connect to Wi-Fi in poor signal areas. Even under

Figure 4.14: Mean battery savings for all users in the dataset with 95% confidence interval. (a) vary $\delta$ given $\tau = -80 \text{ dBm}$, (b) vary $\tau$ given $\delta = 1 \text{ sec}$. 
no thresholding ($\tau = \text{None}$), ATiS achieves close to 90% of that achieved by an ideal system. This shows that the huge energy savings of PRiSM are mainly due to the better performance of ATiS algorithm and not just the RSSI thresholding parameter. However, to provide better user experience and also to save on battery energy, PRiSM as a system, uses a default value of $\tau = -80 \text{dBm}$.

### 4.4.6 Overall Energy Impact

Wi-Fi sensing measurements in Table 4.3 show that latest Samsung Galaxy S5 phones consume more energy compared to older Google Nexus One phones because of powerful Wi-Fi chipsets. We infer that in spite of all the commercial advancements made in recent times to reduce the power utilization in Wi-Fi radio’s (e.g., better sleep cycles, reduced idle times), scanning for Wi-Fi APs still requires substantial energy. Hence, PRiSM in general can save substantial battery energy in all phones without discrimination. From Table 4.4, users spend about 30% of battery energy on average for Wi-Fi sensing operations. We observed that about 11.24% of
that energy is wasted for Wi-Fi sensing in regions with poor/no Wi-Fi combined, which is very significant. On average, PRiSM saves about 16.51% of total battery energy, which is equivalent to saving almost 825.5 mWh worth of energy spent on Wi-Fi if we assume the battery capacity to be 5000 mWh. [78] estimates the average battery lifetime\(^1\) of a smart phone to be 40 hrs and 27 hrs for casual and regular usage respectively. Using this result, we observe that PRiSM on average can extend the battery lifetime by 6.6 hrs and 4.5 hrs for casual and regular phone usage respectively. Given that about 70% of users in our dataset travelled in ‘Good Wi-Fi’ areas, energy savings for PRiSM will be much higher if users had high mobility patterns in ‘No Wi-Fi’ and ‘Poor Wi-Fi’ areas, which happens more often in practice.

\textbf{4.4.7 Practical Verification of Energy Savings}

We identified test phones with similar battery aging by comparing the amount of time it took them for a full battery discharge with bare-essential Android system processes. Practical verification of energy savings shown in Table 4.5 involved two phones: one normal phone and other running PRiSM-SubOpt. A mock application was installed on both phones to check for Wi-Fi at different sensing intervals. The application just connects to the AP and no data transmission is done since energy consumption may change with different data transfer rates even to same AP’s at a particular time instant. For an RSSI threshold setting of \(\tau = \text{None}\), the sensing intervals for 30 sec, 60 sec, and 120 sec saw average Wi-Fi availability of 55.20 %, 77.32 %, and 0 % respectively. This Wi-Fi availability is calculated by comparing the Wi-Fi connectivity information recorded from user logs and the signature database file given to the users for test. For \(\delta = 30\ sec\), PRiSM obtained huge energy savings in no-WiFi areas and incurred minimal energy overhead in areas with Wi-Fi. For \(\delta = 60\ sec\), PRiSM should have had more battery left at the end since the sensing intervals are less frequent but recall that PRiSM only saves energy on scan costs in areas with good Wi-Fi. Given that users saw an average of 77.32 % Wi-Fi, only about 9 % of battery remained. For \(\delta = 120\ sec\), we specifically tested

\(^1\)In this paper, ‘battery lifetime’ refers to the operating time of the battery from one full charge to full discharge.
Table 4.5: Nexus One practical energy evaluation.

<table>
<thead>
<tr>
<th>System</th>
<th>Wi-Fi Sensing Interval (δ)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30 sec</td>
</tr>
<tr>
<td>Normal</td>
<td>lasted 14.5 hrs</td>
</tr>
<tr>
<td>PRiSM</td>
<td>had 54% left</td>
</tr>
</tbody>
</table>

the scenario where user visits totally new places (i.e., with zero stored information about the location in database and hence 0% Wi-Fi availability to connect). Hence, the energy savings by PRiSM is more in this scenario than previous sensing intervals even though the Wi-Fi sensing operations are less frequent at every 120 sec. The results show that in all cases, PRiSM-SubOpt had substantial amount of battery left when the normal phone is completely discharged. We believe that PRiSM-Opt will save even more energy in these situations.

4.5 Discussion

PRiSM places no restrictions to the users in the way they hold their phones or the places they visit. Each user accumulated dynamic cellular signal variations from distinct antenna gains and antenna placement in their devices. PRiSM constructs unique signatures in every device and hence, operates on the assumption that a user connected to the logical Wi-Fi location previously before evaluation phase. As discussed in § 4.3.2, for Zero-match case, PRiSM creates signatures for the new location only after manual user input. Hence, it only creates signatures for user specified logical locations and reduces meaningless signatures but requires manual supervision. To reduce this limitation, a centralized signature database can be implemented and shared using crowd sourced data. Initially, we only used Nexus One phones to collect data since we get neighbour cell information in addition to connected cell information from the Android software stack. In recent months, many popular devices from Motorola, HTC, Sony and LG started providing the neighbor values. Hence, the scope and impact of PRiSM is wide.

Load balancing or Cell breathing techniques are used in CDMA systems where the Base
Station (BS) output power is split among active users. Hence, coverage range of cellular towers is shrunk based on user load. In GSM and current LTE systems, each BS usually transmits with full transmission power in the downlink. Based on the location of the mobile within the coverage area, it receives a percentage of the transmitted power which PRiSM utilizes to logically localize locations. PRiSM is not heavily impacted by this because it uses both connected and neighboring cell towers for estimation and hence, absence of one or two BSs does not affect its working. Also note that PRiSM periodically updates the signal strengths for the signatures whenever the user visits the logical locations and hence, provides a robust method to update the database.

An example rule (not given in XML format for the sake of clarity) input that can be implemented in the decision engine is “do not connect to a particular AP on weekdays but do connect on weekends”. Hence, PRiSM filters its decision based on the day, even though the AP is available at all days. We see many potential applications for the decision engine in PRiSM (e.g., Cellular network carriers can dynamically modify data-offloading rules for their customers based on their real-time network congestion levels at a location. This can facilitate fast handover between Wi-Fi and cellular data usage and readily complement commercial ISP solutions like Hotspot 2.0 [79]. Note that Hotspot 2.0 promises seamless Wi-Fi authentication and handoff, but still needs to identify places with good Wi-Fi).

Some APs do not allow data transmission even after successful connection with the AP (i.e., closed). PRiSM currently does not handle this specific case, but works well for both open and password protected APs if the device has connected to those APs previously. This limitation can be rectified by first performing additional data connectivity check on each newly connected AP. Closed APs can then be added to a separate list to avoid future connections. Recently, Apple blocked Wi-Fi scans initiated from any user application and made its Application Programming Interface (API) private. However, PRiSM does not initiate scans, instead, it records information such as AP name, MAC ID’s, and the signal strengths after system initiated connection. Though PRiSM is implemented in Android, its functioning holds good for most mobile operating systems in general.
4.6 Concluding Remarks

We developed a new Wi-Fi detection system, PRiSM, which utilizes the freely available cellular signal information of GSM signals to statistically map the Wi-Fi APs with a logical location information. The signal strengths from these base stations are recorded, however, the geographical coordinate location of these base stations is not required. PRiSM runs in the background and reads cellular signals based on a scheduling policy and hence consumes minimal energy overhead. We use a novel technique to dynamically build and update the signature clusters in near real-time and thus avoid the need for an extensive training phase.

We develop a specialized statistical matching algorithm which uses a likelihood estimation technique to automatically tune the decision thresholds for every signature. The threshold values are tuned by connecting to APs and comparing against the ground truth values (i.e., AP available, unavailable). We also implement a novel selective-channel Wi-Fi scanning framework to automatically connect to the APs without scanning or association by utilizing their stored frequency channel information. The empirically constructed signal distributions and decision thresholds for a Wi-Fi location can be adapted or learned as time evolves. We implement PRiSM on Android devices and perform both trace-based simulation and practical evaluation. PRiSM obtains up to 96% of energy savings in Wi-Fi sensing operations equivalent to saving up to 16% of total battery capacity, together with an average prediction accuracy of up to 98%. 
Chapter 5

PILS: A Context-Aware Indoor Positioning System via Cellular Multi-Homing

Smart devices have become one of the primary ways for people to access entertainment and other business applications. Most of these applications provide location-specific services and hence, require either the absolute or logical location of users in indoor settings. Big retail giants and shop vendors in indoor locations such as malls, public convention center’s aim to provide specific deals and discounts to users who are within walking distance from their shops. Identifying the context of a user (e.g., in front of the store, suits section, billing counter) in a timely and practical manner is very important for the retail outlets to disburse appropriate deals.

Another fast developing trend is to selectively activate certain security features for smart devices in Enterprise Device Management (e.g., turn off camera inside office space, disable voice recorder in conference room). In the above applications, ‘front of the store’, ‘billing counter’, ‘conference room’ are few examples of logical locations or in the broad-sense referred to as the ‘context’ of the smart device. In all these cases, though sub-meter level accuracy is not required or expected, accuracy of the order of few feet ($\approx 5$ to $10ft$) is highly preferred. However, to
design a precise and an energy-efficient indoor localization system in an automated manner is (still) a very non-trivial task. The reasons include the need for: infrastructure-independent solutions, ease of practical deployment, and minimal battery consumption for users. With the advent of small cells and millimeter wave technologies, we aim to focus the research community to look to utilize 3G and 4G cellular network characteristics for energy-efficient and practical indoor localization. Here, we develop PILS [42], a new context-aware indoor location detection system, to statistically map a context to an indoor location in a logical manner.

5.1 Motivation

Predicting the location of a user in indoor settings in a practical and energy-efficient manner is (still) a very non-trivial task. The latest challenge in indoor localization is not to design specialized sensors but to design and implement practical data fusion methods using the already available technologies. Current state-of-the-art indoor localization techniques utilize Wi-Fi and a variety of sensors inside smart phones to predict user location. Some also require site-specific input such as indoor floor plans or the location of Wi-Fi access points.

Context-aware services are being used extensively from providing coupons and deals to setting site-specific security features in smart devices using Enterprise Device Management (EDM). In the above applications, ‘produce section’, ‘conference room’ are few examples of logical locations. In these cases, though sub-meter level accuracy is not required or expected, accuracy of the order of few feet is highly preferred. However, to design a fast and an energy-efficient indoor localization system in an automated manner is (still) a very non-trivial task. Existing works in indoor localization use specialized hardware and RF signals or beacons to achieve accuracy [37–41] and demand additional infrastructure and meticulous engineering for efficient working. Some others use Wi-Fi signal fingerprinting as in RADAR [32], Horus [33]. However, the above systems have extensive calibration and practical deployment overhead. Moreover, Wi-Fi signals are more prone to multi-path and fading effects from static objects and human movement.
Sensors inside smart phones such as accelerometers [51, 52] are also used to predict user location but they suffer from high battery energy consumption. Some systems use 2G cellular signals for indoor localization [56–58] but they only use averaged Received Signal Strengths (RSS) which is less granular since it contains power belonging to serving cells, co-channel cells, thermal noise etc. With the mobile carrier’s planning to phase out outdated 2G systems by the end-of-year 2016 [80] and the current advent of small cells and millimeter wave technologies, we aim to focus the research community to utilize multitude of 3G and 4G cellular network characteristics for energy-efficient and practical indoor localization.

Based on the existing solutions, we need a system that consumes minimal battery energy and provides ways to continuously accommodate the cellular signal fluctuations, and be easily deployable in real world. Thus the question we ask ourselves is, “How can we reliably identify the context of a user in indoor environments using a multitude of cellular network signals such as 2G, 3G or 4G?”.

5.2 Design

5.2.1 Cellular Multi-Homing

Cellular Multi-homing is the technique used by any user equipment (UE) or smart device to camp on the cellular tower signals from multiple generations (e.g., 2G, 3G, 4G-LTE). Using
Figure 5.2: Cellular signal strength distribution of the mostly observed base stations at (a) location A and (b) location B.

this technology, the smart device can receive much better information about its context and surroundings because it can correlate the signal strength information from multiple cellular base station towers. We attempt to measure a variety of statistical information pertaining to cellular signals received by smart phones and study how we can construct a database of reliable cellular signal signatures per location. We then investigate the feasibility of distinction among locations recorded in the database based on their signatures. Here, we provide our findings for general GSM-based cellular signals. We store the signature for a location as a XML element as in Figure 5.1. The structure of the specialized data structure designed for PILS is shown in § 5.3.2.

Cellular signals are ubiquitous in nature and are received continuously by the phones. A smart phone can receive signals from more than ten base stations (BSs) in dense urban areas [74]. GSM based Android phones can overhear signals from up to seven (six neighbouring and one connected) BSs in ASU (Active Set Updates) units. The linear equation between dBm and ASU values for GSM networks is $dBm = 2ASU - 113$. ASU values range from 0 to 31 and 99, which indicates unknown signal strength. We analyzed the statistics for signals collected at different locations but for explanation purposes, we take random locations to show the following results. Figures 5.2 (a) and 5.2 (b) show PDFs (Probability Density Functions) of signal strengths for the same BS from two adjacent rooms. They see very different signal strength patterns and
Figure 5.3: (a) The number of observed base stations over time at a location. It fluctuates from 0 to 7. (b) The observed base station IDs over time. (c) CDF of the number of observed base stations.
Figure 5.4: (a) Evolution of cellular signatures located at a location.

hence can be effectively used to distinguish the locations if we capture the entire signal statistics for that locations. To do so, we propose to build cellular signal signatures using “probability distributions” of signal strengths from observable base stations rather than using abstracted information (e.g., “average signal strengths”).

Figure 5.3 (a) shows the variation in the number of observed BSs between 1 and 7. It is due to the changes in user movement pattern and environment conditions. Different locations observe different BSs or different signal strength patterns from the same common BS. Figure 5.3 (b) shows the variation in the BS observation pattern over time. While many BSs are observed intermittently, some reliable BSs are observed continuously. The connected BSs change over time even at a given location due to channel fading. Figure 5.3 (c) shows the CDF (Cumulative Density Function) of the number of observable BSs at a location. The number is more biased toward seven for urban settings where the density of BSs is more.

Figure 5.4 shows the evolution of signatures recorded at a location for a considerable number of hours. For better readability, we plotted only the signal strength distribution from the BS which has been most frequently observed in the corresponding signature. Simply put, the cumulative distribution after 100 hrs includes the distribution after 50 hrs plus fifty more hours. Note that the signal strength distributions do not converge to a Gaussian distribution even after
a good amount of time. Hence, we need a non-parametric algorithm which does not assume anything about the underlying data distribution. Multiple peaks shown in each distribution confirm that different locations experiences characteristic signal patterns. The correlation coefficient ($\rho_{X_1,X_2}$) between probability distributions accumulating signals for different amounts of time clarifies the existence of characteristic patterns in the signatures. From our experiments, high value of correlation coefficients and low cross-correlation values indicate that our statistical technique is likely to provide good performance in matching accuracy.

We also find that the signatures recorded at locations far from or near to each other have significant dissimilarities. We choose three locations: $X$, $Y$, and $Z$, where the radial distances between $X$ and $Y$ is about 4 miles and between $Y$ and $Z$ is about 15 – 20 meters ($Y$ and $Z$ are in the same building). In the figures, base station IDs and their average signal strengths are given in the legend. As expected, the signatures for $X$ and $Y$ contain completely different sets of BSs and different patterns of signal distributions. On the other hand, the signatures for $Y$ and $Z$ show similar sets of BSs. However they are still distinguishable because the signal distributions show unique patterns. Considering the possible differences in the environment factors, observing dissimilar signal distributions even for nearby APs is not surprising and actually helps to identify the APs more reliably.

It is also to be noted that, in addition to visually verifying that the data does not fall under a Guassian distribution, we also statistically verify it by drawing QQPlot. In a QQPlot, the null hypothesis is usually that the data samples are distributed normally with an unspecified mean and variance. is a plot of the sorted values from the data set against the expected values of the corresponding quantiles from the standard normal distribution. If the proposed null hypothesis holds true for the dataset, then the plotted points should approximately lie on a straight line. This is shown for a location is Figure 5.5.
5.2.2 RSRP and RSRQ parameters

We performed similar experiments on 3G and 4G cellular radio signals and found similar distribution characteristics for signal strengths as discussed previously for 2G radios in § 5.2.1. In 4G LTE radio system, in addition to observing the signal strength values, we also recorded the RSRP and RSRQ values. These parameters are described in the 3GPP standard specification [81]. RSRP (Reference Signal Received Power) is a type of RSSI-based measurement. It measures average received power over cell-specific reference signals and hence, can provide better power information related to different locations unlike regular RSSI (Receive Strength Signal Indicator). RSSI values contain power belonging to serving cells, co-channel cells, thermal noise etc. RSRQ (Reference Signal Received Quality) is a parameter which provides the quality of received signals in the user device. These parameters are used in cell handover process and hence, we believe using these back-channel parameters in addition to the regular signal strength values can substantially increase the localization accuracy.

5.2.3 Practical Algorithm

We design an algorithm that can utilize detailed statistical properties of cellular signals instead of the averaged signal strength values. A simplified version of ATiS (Automatically
Algorithm 2: **ATiS Signature Score Generation**

1: **INPUT**: Signature database for locations visited by the user
2: **INPUT**: Set of currently observed BSs and their corresponding signal strengths at time $t$
3: **OUTPUT**: Estimated user location

4: **Step 1.** For given input BS, identify the location cluster subset to reduce computation
5: **Step 2.** Calculate the score for the individual signatures
6: for radiotype do
7: for all signatures in cluster subset do
8: for all Base Station ID’s within signature do
9: if Base Station ID exists in input at time($t$) then
10: Estimate the BS Impact Factor
11: if Requested signal strength bin is Empty then
12: Normalize ‘x’ adjacent bins
13: end if
14: Evaluate likelihood of occurrence using expectation maximization from Bayesian-based approach
15: end if
16: end for
17: Accumulate final likelihood scores for all signatures
18: end for
19: end for
20: **Step 3.** Apply the lower and upper bound thresholds to generated scores (i.e., $C_L \leq \text{score}(j) \leq C_U$)
21: **Step 4.** Apply either BS Match or BS Impact Factor sorting technique dynamically based on data storage pattern
22: **Step 5.** Now order the locations based on score values
23: **Step 6.** Return estimated user location
24: **Step 7.** Check with the ground truth and update the signature thresholds if needed
Tuned Location Sensing) is explained in Algorithm 2. Since the entire signal distribution is available, ATiS predicts the location in near real-time. A higher level intuition of the algorithm is that if the probability of seeing a particular signal strength within the PDF of a base station (BS) is high and the probability of the BS observed at a particular location is high, the total joint distribution is maximized and we get a more accurate signature match. ATiS utilizes a set of signatures ($P$) each consisting of a set of base stations $R_j$ and corresponding signal strength distributions $f_{k,j}(S)$, where $k \in R_j$ and $j \in P$. Note that $j$ and $k$ are signature ID’s (e.g., Location IDs) and cellular base station ID’s respectively. Each signature $P$ has information pertaining to the number of occurrences made by its individual base stations in $n(k,j)$ and the total occurrences of all its base stations collectively in $N_j$. The maximum likelihood of the currently observed signals, $s_k(t)$ for $t \in [t_1, t_2]$, from the base station $k$ is calculated as $v(k, j)$ for the signature $j$.

ATiS computes the likelihood scores using expectation maximization techniques based on the Bayesian Networks. For any input BS, it does a local normalization of signal strength values surrounding the target signal strength in the database and hence, performs well even under signal fluctuations. The closer the match of input BS within a signature, the better is the score for the location. All signatures whose likelihood scores satisfy the lower bound ($C_L$) and upper bound ($C_U$) thresholds are returned as output in descending order of their scores. The values of $[C_L, C_U]$ are initialized with $[1, 0]$ initially. The upper bound gradually decreases and the lower bound gradually increases based on ground truth to achieve a tight threshold range. ATiS auto-tunes thresholds within $0 - 1$ based on likelihood scores and hence, does not overfit the data for any particular scenario. PILS also uses a hashmap of unique location labels to store the signatures in a cluster and a reverse hashmap of observed BS IDs to labels. The signatures are thus computed only for the locations with current observed BS IDs. Hence, by design, PILS utilizes a cluster-reduction approach to only compare the currently received signals with a small subset of the signatures in the database irrespective of the total database size and saves on computation time to compare from all the signatures otherwise.
5.3 Implementation

5.3.1 Architecture

As shown in Figure 5.6, the primary modules of PILS include: Trainer Service, System Service, Signature Database and the Decision Engine. The trainer service and the system service runs at the application layer and the decision engine is at the platform layer of the Android stack. The trainer runs in the system background and constructs a list of unique signatures (inside the phone for privacy) for all locations visited by the user. The system service overhears the cellular signals at requisite time intervals to predict the location. The decision engine is designed to include a XML rule-set to decide which Location Based Services (LBS) get access to this location information from PILS. Hence, PILS can be attributed to a middleware component providing location information to all system services running in the device. If the location is predicted correctly, the signatures can be kept up-to-date by triggering a run-time training service to update the signature database. We implemented a prototype version of PILS in Android based devices for practical verification.

5.3.2 Data Storage and Retrieval

PILS has a novel implementation for the signal statistics distribution storage. An overview of the storage database implementation for storing the contextual signatures is shown in Figure 5.7.
The locations are organized in a hashmap based on the location ID’s and every location ID has a cluster of observed BS ID’s from different radio types. For every BS ID, a list of values are stored corresponding to every unique signal strength. Whenever, a new database update is triggered, the individual count values in the list are just incremented and hence, PILS achieves near-real-time database updates. This is very significant because, PILS totally avoids the training phase required by most AI algorithms. Also, there is no need to stop the localization service during database update. Every such location cluster is mapped as a XML element as shown in Figure 5.1. When a user needs to change the phone, the entire dataset distribution can retrieved as an XML document and transferred to the new device. Hence, PILS can facilitate signature data distribution easily between multiple devices and can help provide a truly ubiquitous experience.
Table 5.1: Energy consumption information per second between PILS and other approaches used for continuous location sensing.

<table>
<thead>
<tr>
<th>Item</th>
<th>Energy Consumed (mWh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi-Fi Scan</td>
<td>0.1185</td>
</tr>
<tr>
<td>Accelerometer</td>
<td>0.6670</td>
</tr>
<tr>
<td>GPS</td>
<td>1.5800</td>
</tr>
<tr>
<td>PILS</td>
<td>0.0173</td>
</tr>
</tbody>
</table>

5.3.3 Cost Analysis

Cellular signals are received and processed all the time by the phone MODEM at no extra cost. PILS activates the CPU only to read cellular signal values from the MODEM and to compute using ATiS. At all other times, CPU is not activated by PILS and consumes negligible energy (0.6 – 1.1 µWh) on top of CPU base energy. The sampling policy is usually set at 20 contiguous samples every 60 seconds. However, the sampling rate can be increased or decreased dynamically based on the situation needs (e.g., While user is walking, the sampling rate can be higher to provide quick location updates). The energy consumption for running PILS is compared to other approaches used for location sensing in Table 5.1. The energy given in the table includes the energy for signal sampling, ATiS functioning and base CPU energy. Using the reverse hashmap, the signatures are computed only for the location clusters with current observed BS IDs. Hence, by design, PRiSM only compares the currently received signals with a small subset of signatures in the database irrespective of the total database size and saves on computation time to compare from all the signatures otherwise. Thus the space and time complexity needed for computation is a function of the location clusters identified by the current observed BS ID and is almost constant.
5.4 Evaluation

5.4.1 Datasets

We developed a customized monitoring application for Android based devices to obtain data from locations inside buildings. Dataset information is provided in Table 5.2. We obtained data from several different environments: home, office, shopping mall, library, university classroom and hallways. The layout of the office is shown in Figure 5.8. The actual dimensions of the library is not available and different classrooms had different dimensions ranging from 7.25mx6.75m to 13.25mx13.25m. The figure shows that trial locations include both rooms and open locations. Three Samsung Galaxy S5 phones were used by lab personnel to gather data from the home and office locations. We obtained IRB\(^1\) approval to gather data from Nexus One devices run by volunteers. The radial distance between the the adjoining spots in the office and home estimate between 5 – 20 meters with each other. More information regarding the dataset is given in table 5.2.

5.4.2 Accuracy Measurements

A trace-driven simulator builds the signatures and evaluates the accuracy of the algorithms by checking with the ground truth values in the dataset. The training and test data are collected separately at different time intervals. The signature database is trained using the training set and the test data is evaluated against the database. In the following figures, self-sourced data infers that the data is collected from the users personal phone and crowd-sourced data infers

---

\(^1\)In this paper, ‘IRB’ refers to ‘Institutional Review Board’.
that the data collected from the devices of all users are combined and the results are validated against the entire database.

The overall prediction probability for various environments is shown in Figure 5.10 (a) and (b) for self-sourced and crowd-sourced data. Self-sourced data is the data which is stored in the users phone via his movement patterns. Crowd-sourced data is the aggregated data collected from different users data. We note that even though, the accuracy varies from location to location based on signal strengths and other BS availability conditions, we see that PILS provides better accuracy when compared to other matching algorithms such as base station
Figure 5.10: Prediction probability for various environments (a) Self-Sourced Data (b) Crowd-Sourced Data.
set or cell ID matching (BSSET) and mean square error matching (MSE). MSE gets very low values because, the average signal strength values for various spots within an environment is almost the same and hence it is difficult to correctly identify them. Note that when BSSET and MSE techniques include the specialized sorting techniques used in PILS, they do not match to the performance shown in PILS mainly due to the statistical matching performed in PILS as opposed to average values used in BSSET and MSE. This is shown in Figure 5.11 (a) and(b).

We also see that though the accuracy decreases slightly, it holds good for crowd-sourced data also. We also plot the localization error that occur while predicting the location spots. We define the localization error as the amount of distance by which the correct location and the predicted location differ. This is shown in Figure 5.12 (a) and Figure 5.12 (b).
Figure 5.12: Prediction error for various environments (a) Self-Sourced Data (b) Crowd-Sourced Data.
To evaluate the advantage of using the detailed statistical distribution of multiple radio signals, we plot the average prediction accuracy values of ATiS algorithm for different 2G, 3G and 4G radio combination signal strengths as shown in Figure 5.9. For 4G radio, we include measurements for RSRP and RSRQ in addition to the regular RSSI measurements. We find that the individual radio types have varying prediction accuracies in the different locations. Combining two radio types increases the prediction accuracy but the combination of all three radio technologies always provide the highest accuracy for any location type. Thus, we justify our premise for using the multi-homing technique to localize in indoor settings. Also, this measurement study was performed at all the locations and is verified.

5.4.3 Accuracy varies with number of base stations

The prediction accuracy should increase with increase in number of observed base stations. The change in prediction probability with change in neighbour BS occurrence for different environment locations is shown in Figure 5.13 and Figure 5.15. We should also take care of the fact sometimes locations with more BS matches obtained less likelihood scores (e.g., 0.1 * 0.1 is less than 0.1). Since ATiS tunes the thresholds for a location based on the collective score for all observed BS’s and not on per-BS score, it will result in these mis-classifications. But, we avoided these mis-classifications by sorting the predicted scores based on the number of BS matched in the signature. Hence, we were able to achieve higher prediction probability values.

But interestingly, we also found that for Mall, the prediction probability is very low (around 50%). This is due to the fact that mall location had very large open indoor space and it was very difficult to distinguish the adjoining location spots. So we tried an advanced technique to do path matching based on WiFi beacon value. To our expectation, we then received close to 80% prediction probability accuracy in Mall. However, more detailed measurements are needed in places such as mall where there are large open spaces within an indoor architecture for the shops.
Figure 5.13: Prediction probability for change in number of observed neighbour BS (a) Home-SJ (b) Office.
Figure 5.14: Prediction probability for change in number of observed neighbour BS (a) University Room 1231 (b) Library.
5.4.4 RSRP values provide better accuracy than RSS values

RSS values contain average signal power including power belonging to serving cells, co-channel cells, and thermal noise and are more prone to environmental degradation. PHY layer RSRP values are measured over cell-specific reference signals and are less prone to local perturbations in the environment. Also, the RSS values from 4G networks yield better prediction than RSS signals from 2G networks in office location due to presence of small cells in the buildings. In home location, the 4G towers were located far away than the 2G towers and hence, the RSS values were less helpful in differentiating the rooms. We observed the received signal strength values and the geographical coordinate of the base stations.

5.4.5 RSRQ values provide less accuracy when used alone

As explained in previous sections, RSRQ value is derived from RSRP and RSSI values. It is affected by adjacent channel interference and thermal noise and hence, when used alone, provides less accuracy than RSRP values. However, when RSRQ values are used alongside RSRP values, the prediction accuracy of the system is found to increase more than just using RSRP signals.

5.4.6 Radio network combinations increase accuracy

The prediction accuracy of the system increases with increased radio combination as opposed to one single radio type since the radio signal from particular signal technology may not hold good to distinguish locations at all places. The radio signal distributions for BSs in different radio networks vary for different locations. Hence, a radio combination can help avoid mis-classification rather than relying on a single radio network where two adjacent indoor locations might observe similar signal distributions over time. This further encourages our claim to use for multiple radio combination values. The graph is shown in Figure 5.9.
5.4.7 Effect of accuracy on walking

In this section, we provide information about the variation in accuracy measurements with the different distance between adjacent spots during the hallway walk in our university campus building. The spots are located every 4m apart up to 30m apart. We can clearly see that the prediction probability increases with increased distance between the adjacent spots. Note that in these experiments, we do not use any path matching or accelerometer to track user movement. Hence receiving close to 60% prediction probability for hallway locations of 15m is considered to be very positive result for practical applications. The graph is provided in figure 5.16.

5.4.8 Effect of accuracy on room level detection

In order to observe whether our solution will hold practical value in multiple environment scenarios, we made a set of measurements to check the accuracy of PILS which deciding on following things: inside or outside the home, inside or outside the classroom, etc. We received average prediction probability measurements above 70% for all locations tested. The results are showed for both all neighbours and zero neighbour (i.e., connected BS only) in figure 5.17 (a) and (b) respectively. Hence, PILS can be a viable practical solution in terms of energy cost and deployment cost when considered with other approaches such as WiFi.

5.5 Discussion

PILS makes no restrictions to the users in the way they hold their phones or the places they visit. Each user accumulated dynamic cellular signal variations from distinct antenna gains and antenna placement in their phones. Load balancing or Cell breathing techniques are used in CDMA systems where the BS output power is split among active users. Hence, coverage range of cellular towers is shrunked based on user load. In GSM and current LTE systems, each BS usually transmits with full transmission power in the downlink. Based on the location of the mobile within the coverage area, it receives a percentage of the transmitted power which PILS
Figure 5.15: Prediction probability for change in number of observed neighbour BS (a) University Room 1231 (b) Library.
Figure 5.16: Prediction probability for change in distance between adjoining test locations during hallway walk.

Figure 5.17: Prediction probability for room level accuracy (a) All connected and neighbour BS (b) Connected BS only.
utilizes to logically localize locations. Hence, PILS is not directly affected by fluctuations due to cell balancing.

PILS constructs unique signatures in every device. With deeper knowledge on signal reception characteristics in different devices, a centralized database can be easily implemented and shared between users so that PILS can provide location estimations even in new places where the user has not previously visited. In recent months, many popular devices from Motorola, HTC, Sony and LG started providing the cellular signal values for all radio types due to the changes made in the Android operating system platform source code. Hence, the scope and impact of PILS is wide and relevant to current times.

An example rule (not given in XML format for the sake of clarity) input into decision engine is “do not connect to Wi-Fi hotspots at particular location”. Hence, PILS filters its decision based on the rule-set in the decision engine. We see many potential applications for the decision engine in PILS (e.g., Cellular network carriers can dynamically modify data-offloading rules for their customers based on their real-time network congestion levels at a location. This can facilitate fast handover between Wi-Fi hotspot at a location and cellular data usage and readily complement commercial ISP solutions like Hotspot 2.0 [79]. Note that Hotspot 2.0 promises seamless Wi-Fi authentication and handoff, but still needs to identify locations with Wi-Fi.).

5.6 Concluding Remarks

We proposed to advance the state-of-art in indoor localization in following ways: utilize detailed statistical properties of cellular signals including both connected and neighbor base stations for GSM, UMTS, and LTE networks through cellular multi-homing. We developed a new context-aware indoor location detection system, PILS, which utilizes the freely available cellular signal information from multiple cellular generation networks to statistically map a context to the indoor location in a logical manner. A context or location signature is defined as the set of probability density functions (PDFs) of signal strengths from all observable Base Stations (BS) when the smart phone observes a context or when present at a location. PILS runs in
the background and reads cellular signals based on a scheduling policy and hence consumes minimal energy overhead. We introduced a novel technique to dynamically build and update the signature clusters in near real-time and thus avoid the need for a specialized training phase. We developed a specialized non-parametric statistical matching algorithm which uses expectation maximization techniques based on Bayesian networks and automatically tunes the decision thresholds for every signature. We also implemented a XML rule-based decision engine to help facilitate sharing location information with any LBS applications.

Through this work, we provide information about the feasibility of using such back-channel parameter (RSCP, RSRP, RSRQ) measurements in indoor localization efforts. To the best of our knowledge, we are the first to provide a working prototype implementation using these back-channel parameters. As a result, we hope that these network related parameters shall be opened up by the network operators to the research community and more detailed research on the factors will follow.

Through extensive experiments, we showed that detailed statistical properties of cellular signals alone is sufficient enough to logically distinguish a location with sufficient accuracy. By not relying on any extra sensors, PILS conserves as much battery energy as possible. The empirically constructed signal distributions and decision thresholds can be adapted/learned as time evolves. Therefore, PILS does not require any extensive war-driving or crowd-sourcing to gather data and is more robust to changes in cellular signal environments. We evaluated PILS on Android smart phones in different contextual/location scenarios and demonstrated its effectiveness.
Chapter 6

Conclusion

6.1 Summary

The dissertation addresses three important challenges in ubiquitous and pervasive mobile computing: operating-system-independent fine-grained energy consumption analysis, maximal data offloading with minimal energy consumption, and context-aware indoor localization with minimal sensor costs. We have proposed iSha, PRiSM and PILS for the above challenges and prove the effectiveness of our solutions with working system prototypes.

iSha is a mobile operating-system-independent solution to analyse and predict fine-grained energy consumption values of system sub-components. It inserts specific log triggers in the executable code using an assembler/disassembler module. We measure the detailed power consumption patterns of components under different device screen states and generate a model using stochastic approach. We also impart real-world data into the energy model for the developers to emulate “in-the-wild” variations from within their laboratory settings.

We developed a new Wi-Fi detection system, PRiSM (Practical and Resource-aware Information Sensing Methodology), which utilizes the freely available cellular signal information of GSM signals to statistically map the Wi-Fi APs with a logical location information. We developed a specialized statistical matching algorithm which uses a likelihood estimation technique
to automatically tune the decision thresholds for every signature. The threshold values are
tuned by connecting to APs and comparing against the ground truth values (i.e., AP available,
unavailable). We also implemented a novel selective-channel Wi-Fi scanning framework to
automatically connect to the APs without scanning or association by utilizing their stored
frequency channel information. The empirically constructed signal distributions and decision
thresholds for a Wi-Fi location are learned as time evolves.

We proposed to advance the state-of-art in indoor localization to utilize detailed statistical
properties of cellular signals including both connected and neighbor base stations for GSM,
UMTS, and LTE networks through cellular multi-homing. We developed a new context-aware
indoor location detection system, PILS, which maps a context to a specific indoor location
in a logical manner. We introduced a novel technique to dynamically build and update the
signature clusters in near real-time and thus avoid the need for a specialized training phase. We
developed a specialized non-parametric statistical matching algorithm which uses expectation
maximization techniques based on Bayesian networks and automatically tunes the decision
thresholds for every signature. We also implemented a XML rule-based decision engine to help
facilitate sharing location information with any LBS applications.

Through this work, we provide information about the feasibility of using such back-channel
parameter (RSCP, RSRP, RSRQ) measurements in indoor localization efforts. To the best of
our knowledge, we are the first to provide a working prototype implementation using these
back-channel parameters. As a result, we hope that these network related parameters shall be
opened up by the network operators to the research community and more detailed research on
the factors will follow.

6.2 Future Work

With more and more chipsets having integrated or combo radios, we will extend our work to
other components such as Bluetooth and NFC to obtain a comprehensive energy consumption
model for these low power communicating technologies which are more suited for the wearable
infotainment devices and the Internet of Things (IoT) architecture. There is still a lot of scope to extend the work pertaining to the implementation of a crowd-sourced database for storing the signatures. Some of the practical issues are related to dealing with the highly fragmented mobile ecosystem and identifying the quality of the data traces collected for storage. It also requires a deeper knowledge on signal reception characteristics in different devices. One another area where our work can be extended is to remove the user input involved in getting the location-specific context information. By automating the location naming module, we can easily map the indoor locations using unsupervised learning algorithms, which are really a whole different area of research.
REFERENCES


[58] Anthony LaMarca, Yatin Chawathe, Sunny Consolvo, Jeffrey Hightower, Ian Smith, James Scott, Timothy Sohn, James Howard, Jeff Hughes, Fred Potter, Jason Tabert, Pauline Powledge, Gaetano Borriello, and Bill Schilit. Place lab: Device positioning using radio beacons in the wild. In PERVASIVE, 2005.


89


