Distributed Architecture for Managing Big Data in Smart Grid

A Thesis
Submitted to the Faculty of Graduate Studies and Research
In Partial Fulfillment of the Requirements
For the Degree of
Master of Applied Sciences
IN
Electronic Systems Engineering
University of Regina

By
Umar Ahsan
Regina, Saskatchewan
May, 2017

Copyright © 2017: U. Ahsan
Umar Ahsan, candidate for the degree of Master of Engineering in Electronic Systems Engineering, has presented a thesis titled, *Distributed Architecture for Managing Big Data in Smart Grid*, in an oral examination held on April 26, 2017. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner: Dr. Mohamed El-Darieby, Software Systems Engineering

Supervisor: Dr. Abdul Bais, Electronic Systems Engineering

Committee Member: Dr. Irfan Al-Anbagi, Electronic Systems Engineering

Chair of Defense: Dr. Daniel Kikulwe, Faculty of Social Work
Abstract

Smart grid is a technological advancement to the traditional power system that provides efficient and reliable utilization of energy resources. Large number of sensors are becoming part of the power network to improve its efficiency. These sensors enable communication between home appliances and power generators to enhance home appliance automation, monitoring and remote control capabilities. As smart power grid incorporates a large number of data-generating embedded sensors; key questions are where in the network to process and analyze the data, and how to perform the analysis. The data in smart grid can be processed either by one central processor or through multiple distributed processors. This thesis proposes a smart grid distributed architecture involving home sensors talking to a smart gateway in the home for local processing, which then passes processed data to a central processor for further analysis.

As part of our work, a test bed is designed to highlight advantages of distributed smart grid architecture by comparing central and local processing of data. An open data set is used to feed power sensor data into the test setup. This thesis discusses variety of operations that can be performed in the distributed architecture of smart grid. It is shown that the local processing of data can improve efficiency by effectively utilizing available network bandwidth. Furthermore, local processing is favorable for smart grid applications that are time critical as local processing has less delay and jitters for data communication round trip time as compared to central processing.
Moreover, we discuss that certain calculations like energy usage prediction for home appliances can effectively be done locally while central processor can be used for coordination between different local processors.
Acknowledgements

I would like to express my gratitude to my adviser Dr. Abdul Bais for his valuable guidance and encouragement throughout this research work. He regularly reviewed this research and suggested improvements. I am thankful to FGSR and SaskPower Clean Energy Research Grant (An Application Framework for Real-Time Streaming Big Data Analysis) for supporting this research.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>i</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>iii</td>
</tr>
<tr>
<td>Table of Contents</td>
<td>v</td>
</tr>
<tr>
<td>List of Tables</td>
<td>vi</td>
</tr>
<tr>
<td>List of Figures</td>
<td>x</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>3</td>
</tr>
<tr>
<td>1.2 Problem Statement</td>
<td>7</td>
</tr>
<tr>
<td>1.3 Centralized Computing Approach in Smart Grid</td>
<td>8</td>
</tr>
<tr>
<td>1.4 Thesis Contribution</td>
<td>9</td>
</tr>
<tr>
<td>1.5 Thesis Outline</td>
<td>14</td>
</tr>
<tr>
<td><strong>2 Related Work</strong></td>
<td>15</td>
</tr>
<tr>
<td>2.1 Cloud Services Usage for Smart Homes</td>
<td>15</td>
</tr>
<tr>
<td>2.2 Advanced Metering Infrastructure</td>
<td>19</td>
</tr>
<tr>
<td>2.3 Distributed Data Processing</td>
<td>20</td>
</tr>
<tr>
<td>2.4 Available Test Beds</td>
<td>23</td>
</tr>
<tr>
<td>2.5 Contribution to the State of the Art</td>
<td>24</td>
</tr>
</tbody>
</table>
3 Proposed Architecture

3.1 Software Aspect of Distributed Architecture
  3.1.1 Dataset for Proposed Architecture Testing
  3.1.2 Operating System for Local Processor
  3.1.3 Central Processor Selection
  3.1.4 Mode of Communication

3.2 Hardware Aspect of Distributed Architecture
  3.2.1 Hardware for Sensors
  3.2.2 Hardware for Local Processor

3.3 Overall Design

4 Test Bed and Experimental Results

4.1 Test Bed

4.2 Results
  4.2.1 Local vs Central Processing
  4.2.2 Bandwidth Saving
  4.2.3 Predictability of Energy Usage in Smart Grid

4.3 Discussion

5 Conclusion and Future Work

Bibliography
List of Tables

4.1 Bandwidth saving algorithm result ........................................ 57
List of Figures

1.1 Different regions in smart grid for sensors installation [1] ©2010 IEEE 4
1.2 Smart home sensors sending data to a central processor [2] ©2012 IEEE 8
1.3 Proposed approach for distributed data processing for smart homes . 11
1.4 Smart home appliances connected to local processor using heterogeneous communication technologies. The local processor is further connected to central processor . . . . . . . . . . . . . . . . . . . . . . . . 11

2.1 Distributed cloud architecture for smart grid [3] ©2015 IEEE . . . . 16
2.2 An overview of APIs installation at sensor level enabling them to connect to cloud for smart grid data management [4] ©2010 IEEE . . . 17
2.3 Smart home applications sending data to cloud service through arduino processors [5] ©2013 IEEE . . . . . . . . . . . . . . . . . . . . . . . . . . 18
2.4 Sensors within home are connected to local processor for load management [6] ©2011 IEEE . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
2.5 Working principle of data distribution between smart meters for parallel processing [7] ©2016 IEEE . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21
2.6 Processing time comparison for central and distributed architecture [7] ©2016 IEEE . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
2.7 Smart gateway based approach for data processing [8] ©2016 IEEE . 23

3.1 Sample data set of home A . . . . . . . . . . . . . . . . . . . . . . . . . . 29
3.2 Mode of communication in our architecture. Sensor is communicating with local processor i.e. smart gateway through Bluetooth while HTTP protocol is used for communication between smart gateway and central processor.

3.3 Dragon board 410c [9]

3.4 High level architecture of our proposed design

4.1 Dragon board receiving data from sensors

4.2 Data displaying on AWS as received from dragon board

4.3 Description of round trip time for Bluetooth communication

4.4 Round trip time from 11 pm - 12 am

4.5 Round trip time from 3 am - 4 am

4.6 Round trip time from 9 am - 10 am

4.7 Round trip time from 3 pm - 4 pm

4.8 Round trip time from 6 pm - 7 am

4.9 Round trip time from 10 pm - 11 pm

4.10 Average round trip time for central processor during different times of a day

4.11 Average round trip time for local processor during different times of a day

4.12 Standard deviation of round trip time for central processor during different times of a day

4.13 Standard deviation of round trip time for local processor during different times of a day

4.14 Load Profile of Furnace HRV from Home A data set showing significant difference in value of its power consumption when it is on or off

4.15 Average % value retained corresponding to different thresholds on June 29
4.35 Average % error in the predicted value of furnace HRV using external, internal temperature and humidity combined as input parameters for ANN model

4.36 Average % error in the predicted value of furnace HRV using external temperature change during 15 minutes duration as input parameter for ANN model

4.37 Average % error in the predicted value of furnace HRV using internal temperature change during 15 minutes duration as input parameter for ANN model

4.38 Average % error using different sensors as input parameters for ANN model after seven days

4.39 Percentage error in predicted energy usage of furnace HRV using external temperature as input parameter for regression model after each 15 minutes interval

4.40 Average % error in the predicted energy usage of furnace HRV using external temperature as input parameter for regression model after every day

4.41 Average % error in the predicted energy usage of furnace HRV using average external temperature, humidity and change in external temperature as input parameter for regression model after every day
Abbreviations

AMI  Advanced Metering Infrastructure
AMR  Automatic Meter Reading
ANN  Artificial Neural Network
API  Application Program Interface
HAN  Home Area Network
HRV  Heat Recovery Ventilator
NAN  Neighborhood Area Network
PLC  Power Line Communication
WAN  Wide Area Network
Chapter 1

Introduction

Globally the power industry is going through significant transformation as the generation, transmission, distribution, and control infrastructure are aging while energy consumption is increasing. This increasing energy usage motivates researchers to investigate options for upgrading traditional power system. According to an estimate, residential buildings collectively consume 73% of the total consumption of a grid [10]. Therefore, efficient utilization of energy within this sector can greatly reduce load on a grid. The energy consumption within homes can be monitored and controlled through demand-side energy management [6]. The power consumption of different loads can be monitored to maximize efficiency of the system by reducing peak power value and its duration. In addition to that, dependency on external power usage can be reduced through scheduling power usage according to available local renewable energy [11], [12].

The conventional power system is developed considering the centralized approach with a few high capacity power generators that supply power to consumers. The power system is based on unidirectional energy flow from generators to consumers with consumers having limited control on the power flow [13]. Although the power grid has been serving humanity for more than 100 years, the traditional vision
of energy production and consumption is changing. Consumers are now generating and using their own energy locally. Since, energy generation is a continuous process so they generate energy even if they are not always the only or final consumers of it. The surplus energy is exported to the grid, which means that electricity transfer has to be bidirectional. In this sense, a modern power system differs from a traditional power grid through its ability to predict, monitor and manage bidirectional energy flow for reliable, economical and sustainable power supply [14]. Thus, the evolution of renewable power generation at distribution level and advancement in technology that supports bidirectional energy flow pave way for an intelligent power system, smart grid [1]. Intelligent Electronic Devices (IED) are now installed throughout the power network that continuously exchange data with each other for efficient and economical operation of grid.

The importance of modernizing power grid can be judged by public and private investment in this area. For example, the U.S. government invested a sum of $ 3.4 billion in 2011 for modernizing their power system [15]. Similarly, many local power distributors within the U.S. are upgrading their system, through integrating sensors in their network, to receive real time update about power consumption and generation [16]. ENEL Telegestore project in Italy, Hydro One project in Canada, the Evora InovGrid project in Portugal, and the Modellstadt Mannheim project in Germany are few examples that utilize smart grid concept [3]. In the U.S. only 6% of households had smart meters, an integral part of smart grid, installed in 2008 which rose to 89% in 2012, and by 2019, 30 million more homes will have smart meters installed at their premises [7]. Thus, extensive modernization of traditional power system is evident in recent years. Since, power system is an extensive network involving generators, consumers and distributors therefore its modernization covers multiple areas that are discussed next.
1.1 Background

The smart grid includes sensor networks that use modern communication technology to transmit their generated data for effective working of the network. Since there are large number of devices that are part of the network therefore managing massive amount of data generated by these devices is a challenging task [17]. According to an estimate, the amount of smart meter data generated by two million customers is 22 gigabytes per day [18]. Austin Energy in Texas installed 50,000 smart meters in their system and these meters, while sending data after every fifteen minutes, require 200TB of storage space per day. If that data exchanging interval is changed to five minutes then Austin Energy data storage needs will increase [19]. Few other examples of smart meters include A3 Alpha [20] and Home Energy Management Systems (HEMS) [21]. Moreover simple data collection in smart grid, like current power consumption and generation, is useless without its real time analysis [4]. It is a challenging task to manage massive amount of data, generated by smart grid infrastructure, which involves selecting, monitoring and analyzing data in real time [22]. The demand side energy management requires continuous monitoring and controllig of electrical loads in homes. The monitoring and controlling can be performed throug deploying control algorithms either at central processor or at multiple distributed processors. Many researchers have worked for extensive spatial and temporal monitoring algorithms of electrical loads e.g. [23], [24], [25] and [26]. While on the control side, home automation protocols like X10 and Insteon are designed for load controlling [6]. Addition of these monitoring and control algorithms in the power system is important for its modernization.

The smart grid involves integration of communication system in the existing infrastructure to transfer sensors generated data for its processing [27]. Smart grid sensors are broadly categorized into generation, transmission and consumption units as shown in the Figure 1.1. Each category has specialized sensors with specific func-
Sensors are an integral part of the smart grid as they reliably sense the power network through measuring system parameters like transmission line temperature, power usage etc. Moreover, they can send signals to actuators, deployed throughout the system, in case of any anomaly to counter overall failure of the network [29], [30]. In order to achieve this goal, a smart grid uses wired and wireless sensors to create a mesh of low cost, low power and efficient sensors network. These sensors continuously exchange data with smart meters, connected within homes, and with power distribution centers to optimize resources through real time data processing [31]. The data processing is achieved through deploying processors throughout the
power system. Finding an optimal location for data processing in smart grid is a challenging task.

In the existing power system the generating units have to design their generating capacity while considering the peak demand of consumers. Due to wide variation in consumer needs the energy generating resources are not used to the fullest. The installation of huge generators to counter peak demand of consumers and less utilization of these resources to the fullest decrease efficiency of existing power system [32]. Models for predicting energy utilization like Markov models are introduced in the power system to reduce peak average load [33]. The predicted energy consumption is used to switch on or off intermittent loads at optimal moments for reducing peak energy usage. Currently, multiple energy forecasting models are developed like Multiple Linear Regression, Auto Regressive Moving Average, Auto Regressive Integrated Moving Average, Artificial Neural Network (ANN), Fuzzy Logic and Support Vector Machine [34] to predict energy usage. These models can be deployed either at the consumer or at generation side to predict energy consumption in the power system. Selecting the right location to deploy these algorithms is important for power system modernization. The selection involves multiple factors like capability at the consumer side to implement these algorithms and sufficient network resources to send all the consumers appliances data to generator side in case of one centralized model.

The effective working of smart grid can be achieved through categorizing sensors hierarchically. In the hierarchical structure, sensors that are installed within a home are categorized as Home Area Network (HAN). Data from large number of HANs are collected by one local processor forming Neighborhood Area Network (NAN). The local processor installed within NAN acts as a coordinator between different HANs for efficient energy distribution. Many NANs are integrated and collectively form Wide Area Network (WAN), providing real time electricity management of the entire system.
The transmission of sensors data within the network is a challenging task even for current state-of-the-art technology [35]. According to an estimate the bandwidth requirement for a power distribution system of moderate size is nearly 100 Mbps [1]. Multiple literature reviews for smart grids communication can be found in [36] and [15]. There are variety of available technologies for data communication between sensors such as Power Line Communication (PLC) [37], [38], [39], cellular data carrier technology [40], IP based network, fiber optical cables, Bluetooth, Wi-Fi, WiMAX and ZigBee [41]. The selection of technology is based on multiple factors like its availability, installation and maintenance cost, advancement of that technology for particular application and security requirements [40], [42]. Overall, selection of communication technology for sensors data transmission involves following important factors:

- **Latency**: Latency of a communication technology is critical while evaluating its usage in smart grid. Certain time sensitive applications like wide area situational awareness system require low latency while certain other applications like Advanced Metering Infrastructure (AMI) are not time critical and can tolerate high latency communication technology [43], [44].

- **Reliability**: The communication technology used in smart grid infrastructure needs to have required data transferring capability to avoid any lose of smart grid data.

- **Interoperability**: Smart grid contains heterogeneous sensors with every sensor having different data transfer rate, format, and protocol. Therefore interoperability needs to be considered while designing the communication infrastructure [45]. As an example, certain applications require high data rate like wide area situational awareness system while others require low data rate like AMI [46]. Similarly, real-time communications are required in the case of fault
detection, service restoration or quality monitoring sensors; periodic communication is required in Automatic Meter Reading systems (AMR); while bulk data transfer capability is required to read logs and energy quality information from sensors. Therefore, smart grid sensors have variable communication requirement.

- Security: Secure data communication and implementation of advance security measures in the communication system is important to protect critical assets of smart grid [42].

Consideration of all these factors while designing systems for smart grid data is critical for its implementation.

1.2 Problem Statement

The modernization of current grid involves research in the following areas:

- Addition of sensors in the existing power network. The question is where and how many sensors can be added in it.

- Development of effective communication system for data transferring between these sensors.

- Addition of intelligent and extensive data processing platforms in the system. The data processing platforms can be one central processor or multiple distributed processors.

- Discussion of how predictable is home energy usage and how this prediction can be used by local and central processor to improve efficiency of smart grid.

These areas need detail analysis and are point of discussion in this thesis. The addition of central processor that can process data generated by sensors centrally is discussed next.
1.3 Centralized Computing Approach in Smart Grid

In centralized technique, data from sensors present within multiple homes is collected and processed at control center as shown in Figure 1.2.

As seen in the Figure 1.2 that all sensors of HANs of smart grid are directly communicating with the central processor for effective working of the system. Although this approach is useful to upgrade traditional power system but there are certain issues related to this approach. Centralized infrastructure needs to have generic communication infrastructure to communicate with heterogeneous sensors that are continuously becoming part of smart grid network. Moreover, high cost is also associated with the improvement of sensors to enable them to connect to central processor [47]. Inefficiency, due to extensive usage of network bandwidth for data communication, is also another problem associated with this approach [48]. The centralized system has also single point of failure which makes it highly unreliable for critical applications. In addition to that, scaling is an issue as the number of sensors
and the generated data is increasing at an exponential rate [49]. Finally, central processor approach involves unpredictable round trip time, depending on network load, for data communication which is undesirable for time sensitive applications. Although, central processor based solutions are currently used for smart grid but issues, like big data, extensive bandwidth requirement, inconsistent communication time, vulnerable to security attacks etc., are linked to that solution. Due to these issues centralized processor approach for continuously evolving smart grid system is not a feasible solution. In order to counter these issues, distributed data processing infrastructure is an alternative solution which forms the basis of this thesis.

1.4 Thesis Contribution

We propose a hierarchical architecture for smart home (subset of smart grid) data management through distributed processing. This approach decreases service delays and saves network bandwidth by locally processing the data. In our proposed solution the data generated by home sensors is collected by smart gateway, local processor. The local data processing power enables our solution to be more suitable for real time data analysis [50]. The data which we need in real time for successful analysis, operation and control of the grid is its current state. In our solution, the state of the grid (voltage phasors) can be directly measured locally multiple times per second for getting accurate insight into the real time dynamics of the system. Currently, as these measurements are collected by Supervisory Control and Data Acquisition (SCADA) system by polling over 2 – 4 seconds, the measurements do not represent a snapshot of the actual system state at one particular time. With local processing, due to low latency, the current local energy usage can be measured with more accuracy through multiple readings per second. In addition to that, Bonomi et al. [51] discuss multiple advantages of local processing in smart grid like location awareness, mobility,
flexible geographical distribution, low latency, real time data analysis and interoperability. The gateway allows interoperability in smart grid by acting as a bridge for data communication between different communication protocols. The smart gateway is further connected to central processor (cloud resources in this case) for integration of data generated by multiple local processors. The local processors filter home sensors data and send the processed information to central processor for reporting the current status of the network. Wei et al. [52] discuss that local processing significantly reduces communication overhead as all users handle their data locally and only consult with the central processor for information like price query. In addition to that, smart gateway not only collects data from sensors but it also provides real time feedback to sensors after data processing. It is important to note that, the local processing is based on data provided by sensors as well as real time data received from central processor through bidirectional communication. Furthermore, the gateway also sends processed information to the central processor for storage, analysis and other calculations that require extensive data processing capability. Finally, multiple gateways are integrated with the central processor to provide overall prototype of actual smart grid. An overview of our proposed solution is shown in the Figure 1.3 and 1.4.

In Figure 1.3 all the sensors within home are connected wirelessly or through wire to smart gateway. The gateway needs to communicate with different type of sensors, each with different physical media for communication like Bluetooth, Wi-Fi, or ZigBee etc. The variation in physical media is because of variable communication requirement of sensors. A communication paradigm based on IP is proposed between smart gateway and central processor. There is an extensive bidirectional communication within the entire network. The purpose of bidirectional communication is to optimize smart grid working through processing generated data and sending control signals to actuators within the network. The processing is based on monitoring and
Figure 1.3: Proposed approach for distributed data processing for smart homes

Figure 1.4: Smart home appliances connected to local processor using heterogeneous communication technologies. The local processor is further connected to central processor
control algorithms that are installed in the processors.

In our proposed architecture the central processor is brought closer to end devices by introducing local processors. In this way, storage and processing capability is at the edge of the network. This edge technique is referred as fog computing. In the smart grid architecture, fog computing concept is introduced by Hong et al. [53] in which low-latency processing functions are performed at edge by local processors while delay tolerant functions are performed on powerful centralized processor. This technique reduces significant data communication load on the entire network as only processed information is forwarded to central processor. In this thesis we discuss our proposed solution that involves development of a test bed to evaluate distributed smart grid architecture. A considerable number of research groups are working toward establishing test beds to validate designs and implemented protocols related to smart grid. These test beds have various aims, scale, limitations, and features. Most of these test beds conduct experiments either in a lab environment (e.g., SmartGrid-Lab, VAST, Micro Grid Lab), or in a residential area (e.g., PowerMatching City) or commercial space (e.g., Smart Microgrid) [54]. Although our proposed architecture is well suited for data processing in all the sections of smart grid but our test bed focuses on minimizing big data generated by smart homes through local processing. We conduct experiments in lab environment using actual residential data set available online.

Following are the contributions of this thesis for distributed data processing of smart homes in power system:

- It highlights the possible options for distributed processing of data generated by heterogeneous sensors and usage of monitoring and control algorithms for local processing.

- It discusses the importance of central processor for collaboration between distributed processors to realize an effective smart grid network.
• Many researchers have proposed distributed smart grid architecture in their studies like [53], [8], and [7] but very limited work is done on implementing this concept to provide quantitative information. In this thesis, we demonstrate working principles of distributed architecture by designing actual test bed.

• It is discussed by many researchers that the local processing of data would provide low latency and using central processors for data processing causes delay but the quantitative data about delay is missing in their work. This thesis provides results that are useful in quantifying the low latency of distributed architecture.

• The local processing of data provide robustness to distributed smart grid architecture. Since the communication with central server is done through the Internet which works on best effort service paradigm. Therefore, Quality of Service (QoS) can not be guaranteed for data that is sent to central server. It is through local processing that the QoS of smart grid can be improved by delaying less time sensitive data and sending only time sensitive data to central server to improve QoS of the network [27].

• Accurate prediction of home energy usage is an important part of current grid advancement. In this thesis, we study the predictability of energy consumption at the scale of individual appliances.

• The results highlight possible algorithms and processes that can be processed locally.

This thesis contributes towards publication of two conference papers titled "Distributed Smart Grid Architecture for Managing Big Data" [55] and "A Review on Big Data Analysis and Internet of Things" [17]. In addition to that submission of one journal paper related to this work is also under consideration.
1.5 Thesis Outline

The rest of the thesis is written in chapter form in the following arrangement:

• Chapter 2: Closely related work for the advancement of smart grid network including different approaches for central and distributed processing of data within smart homes is presented in this chapter.

• Chapter 3: Proposed architecture is explained in this chapter. It provides an overview of our proposed solution and discusses in detail all software and hardware requirements for testing our architecture.

• Chapter 4: Results of distributed smart home architecture are presented in this chapter along with experimental setup. The advantages of distributed architecture are compared with centralized architecture quantitatively in it.

• Chapter 5: Conclusion of our research work is presented along with an outlook for future work.
Chapter 2

Related Work

Smart home infrastructure can be designed using cloud based centralized processor or through distributed processors using edge processing technique. Many researchers propose both approaches. In this chapter we highlight the advantages and limitations of both architectures and establish a link of our work with the existing work in this area.

2.1 Cloud Services Usage for Smart Homes

Baek et al. [3] propose a cloud computing layered architecture for information management in smart grid. The main idea behind the solution is to discuss hierarchical structure to compute and manage data generated within smart grid network. They divide the framework into three levels: top, regional and end level. The top two levels contain cloud based technology while the last layer contains sensors of smart grid. In their approach, smart grid is divided into multiple regions with all of them managed by central cloud. The detail structure of regional and central cloud architecture is shown in Figure 2.1.
The regional layer provides initial processing to data that is received from end devices. Our approach in this thesis is highly linked with Baek et al. methodology but their focus is on enhancement of security features in smart grid network. As a proof of their concept, they provide limited scenarios like authentication of data generated within a particular region. Our work is an extension of their methodology of distributed processing but through designing an actual test bed.

A conceptual view regarding cloud computing as a platform for management of smart grid data is discussed by Rusitschka et al. in their paper [4]. Their primary focus is on using the Internet as a backbone for data flow between sensors and central processor. The data management in their approach is performed at the central level. According to Rusitschka et al. the interoperability in smart grid can be achieved through installing simple web based Application Program Interface (API) at sensor level. These APIs enable data flow between sensors and central processor through the Internet. A detail overview of their concept can be seen in the Figure 2.2.
Rusitschka et al. in their paper [4] discuss the possibility of distributed data management and parallel processing of smart grid data. According to them, the data generated by smart grid sensors is time series e.g. current utilization and production of power etc. The time series data can be stored in distributed structure using key-value pairs. The key is an identifier associated to a particular sensor and the value is data generated by the sensors along with its generated time. This key-value based system enables efficient analysis of the data. As an example, a threshold value can be set for a specific sensor that can trigger outage management alarms in case of any anomaly in the value of that sensor. This key-value system enables distributed as well as parallel data management within smart grid network [4]. This parallel and distributed system can be used to upgrade the smart grid network. The discussion on connectivity issues for sensors that are unable to directly connect to central processor
is not done in their approach. In order to solve this issue, we propose an addition of smart gateway, that is installed locally and capable of not only directly connecting to central processor but also having multiple other connectivity options like Bluetooth, and Wi-Fi etc. Moreover, using cloud for data processing is not scalable as the number of sensors in smart grid network is increasing that consume extensive network bandwidth during data sharing. In order to solve this issue, we propose a smart gateway that can process data locally through collaborating with central processor and only sends processed information to the central level. This gateway not only enables time sensitive applications to be part of smart grid network, due to its low latency, but also saves extensive network bandwidth. We also highlight that not all the data from sensors is required at the central processor for effective working of smart grid.

Zhou et al. [5] present smart home applications that are connected to cloud. In their architecture the data from sensors is collected through Arduino processor that sends data to cloud for its processing as shown in the Figure 2.3.

![Figure 2.3: Smart home applications sending data to cloud service through arduino processors](image)

Figure 2.3: Smart home applications sending data to cloud service through arduino processors [5] ©2013 IEEE
The purpose of their approach is to enable users to visualize the sensors' state anytime via the Internet. They do not discuss any methods for processing that generated data. While in our proposed methodology, we not only locally process the data but are also providing feedback to sensors to counter any anomaly in the entire network. Moreover, their work is limited to those applications that can be connected to cloud services through the Internet. While our approach enables applications that are not capable of connecting to the Internet directly, to be part of smart grid infrastructure.

2.2 Advanced Metering Infrastructure

AMI is among the first advancements in the traditional power system. AMIs are included in the traditional power system to not only measure periodic energy consumption by users but also to investigate other parameters related to power like its voltage, frequency, etc. These measurements are provided by AMIs to central processor for calculations like real-time price, and critical peak hours for energy consumption.

Irwin et al. [6] present a smart gateway for monitoring and controlling home automation devices. All the sensors within home are connected to a gateway as shown in Figure 2.4.
The gateway is referred as AutoMeter in the Figure 2.4 that controls and monitors all the appliances connected to it. The communication between sensors and AutoMeter is performed through power line. The system is capable of monitoring individual loads through continuous data collection from individual loads after specific time interval. However, this system does not have external connectivity. Therefore the decisions for energy consumption can only be made based on predefined logic. Hence, this system cannot be used for real time decision making which requires extensive communication with other units of smart grid. Our approach is an extension of this concept with the addition of connectivity of gateway to central processor for real time decisions.

2.3 Distributed Data Processing

Yan et al. [7] in their paper discuss distributed data processing infrastructure within existing smart grid system to increase its reliability, cost and efficiency. According to their solution, smart meters, that are vital part of smart grid, can be
grouped to form clusters. The rationale behind grouping smart meters is to perform parallel processing of data. In their approach, there is a master meter within a cluster that manages and controls all other meters. After a fixed time interval, data generated within cluster is captured and distributed between smart meters for parallel processing. Master node plays an important role for data distribution. The Figure 2.5 shows the detail working of their approach.

All the smart grid components like distributed renewable energy generators and home sensors are directly connected to a cluster. As the smart meters are directly connected to sensors therefore data can be processed locally. The smart meters are interlinked through central node to decrease latency and improve quality of service. The proof of concept is demonstrated through 15 low-cost single board PCs that form the cluster. The data is pre-loaded in each PC and frequency anomaly is detected in

Figure 2.5: Working principle of data distribution between smart meters for parallel processing [7] ©2016 IEEE
the data set. The processing time for this infrastructure is shown in Figure 2.6.

![Processing time comparison for central and distributed architecture](image)

**Figure 2.6:** Processing time comparison for central and distributed architecture [7] ©2016 IEEE

From the Figure 2.6 it is evident that the processing time for large data set is significantly low in distributed architecture as compared to centralized one. The main drawback in the solution by Yan et al. is that their system is not communicating with the central processor for decisions. In smart grid architecture there are certain decisions that depend on real time information, like current electricity price, from central server. Although the results for distributed processing are quite encouraging but their approach is limited to only those decisions that require local data for decisions. Considering the lower processing time in distributed architecture, we also quantify the advantages of distributed data processing in this thesis.

Lee et al. [8] discuss gateway based distributed architecture for wireless sensors networks. In their architecture the interoperability in wireless sensors is achieved through a gateway, capable of connecting to heterogeneous sensors. The sensors send data to the gateway which further processes the data through collaboration with other local gateways as shown in Figure 2.7.
This gateway based approach increases local data processing efficiency as proposed and implemented in our approach but it lags central processor connectivity which is highly desirable in smart grid applications.

2.4 Available Test Beds

Radhika et al. [56] present a smart grid test bed based on GSM technology which is capable of load management, fault detection and self healing. The test bed system consists of automated power switches, smart meters, energy sources and load. The communication for the system is achieved through GSM modules. The test bed can operate in two modes either in a centralized control mode or in a distributed control mode, in their paper [56] they present a distributed control method so that each automated power switches module have intelligence of its own and decides the control measures at each stage through distributed processing. The test bed allows the implementation of protocols and methodologies that is used for load management in smart grid.

West Virginia Super Circuit (WVSC) [57] smart grid demonstration project implement a hardware-based multi-agent systems for distribution automation and
control. The facility can demonstrate various distributed network applications such as energy storage, and dynamic feeder reconfiguration. Agent Communication Language (ACL) by Foundation for Intelligent Physical Agents (FIPA) is adopted for interoperability as a universal communication protocol in their test bed.

Stanovich et al. [58] design and develop hardware-based test bed to evaluate distributed control algorithms in smart grid. They use fiber optical medium to communicate data between distributed processors. As part of their work, they conclude that the implementation of distributed control algorithms involves unanticipated latency in data communication. It is further discussed that advancement in smart grid area requires distributed data processing and the infrastructure of smart grid should be flexible to add new control algorithms. Chen et al. [59] design test bed enabling researchers to evaluate performance of network for smart grid applications. In addition to these, Cintuglu et al. [60] in their paper provide a detail overview of available test beds that are used for research related to smart grid architecture. According to [60] the interdisciplinary structure of smart grid requires heterogeneous test beds, each having specialized capability. Our proposed solution specifically focuses on managing big data in smart grid infrastructure through distributed data processing.

2.5 Contribution to the State of the Art

The existing methods for advancement in smart grid either focus on local or centralized processing and both have pros and cons as highlighted earlier. Our solution is a mixture of both approaches i.e. effective coordination between local and central processors for data processing, which is basically the fog computing model. Our approach is highly linked with Baek et al. [3] work but with the addition of distributed processing technique as discussed by Yan et al. [7]. The interoperability in our work is motivated by the concept of Lee et al. [8]. Irwin et al. [6] explain different
types of local processes in smart grid architecture and we implement similar kind of processes for smart home. Overall, our work is an extension of all these concepts including quantitative results of our proposed architecture that are achieved through a physical test bed.
Chapter 3

Proposed Architecture

The advancement in the traditional power system includes addition of heterogeneous sensors and installation of processors to optimize working of the system. The data generated by sensors in power system is useless without its processing. In order to process the generated data there are two basic architectures i.e. a central processor approach or multiple distributed processors approach. The amount of generated data in smart grid is escalating that creates a need for data processing closer to the source. Although number of sensors are continuously increasing in all sub-sections of smart grid i.e. consumers, generation and transmission sector but this thesis is limited to consumer side smart grid advancement. The designing of our proposed architecture for modernization of home side management of smart grid involves following requirements:

- The data processing should be closer to the source.
- The advantages of local data processing should be highlighted.
- The local processor should be capable of connecting to heterogeneous sensors and can process the data locally.
- Multiple local processors should be connected to one central processor to ex-
plain the dependency of localized decisions on information received from central processor.

- Real time data analysis should be possible both at local and central level.

In our proposed architecture the data generated by sensors within a home is collected by a processor installed locally. The local processor is further connected to a central processor that receives processed data from multiple local processors. The central processor acts as a coordinator to streamline working of the entire system. An important point to consider is that there is a bidirectional communication between central processor and local processors. In addition to that, local processors also communicate with the actuators within their respective home to react to any anomaly. A block diagram of our proposed methodology is shown in the Figure 3.4.

The proof of concept of our proposed architecture can be provided through simulated environment or through designing actual system. Multiple researchers have simulated variety of conditions to demonstrate the concept of distributed smart grid architecture like [61], [62], [63] and [64]. While on the other hand, very few researchers like [65] have developed actual test beds for testing the proposed architecture. One main reason for lack of actual test beds is the cost associated with it [66]. In our work we are using a mixed approach. We are using real time data generated by sensors as part of Smart* project [66] to reduce cost for sensors deployment, but we are using physical local and central processors to process the data. This mixed approach is useful as it lowers the cost for testing our methodology without affecting any results. The data used in our work is an open data set generated by actual sensors and it is discussed in detail in the relevant sub-section.

The test bed that is used for testing our proposed distributed architecture includes software and hardware aspects of the solution. The selection of software and hardware of the local processor involves selecting right platform that should allow connectivity of heterogeneous sensors and can process the generated data locally. In
addition to that, it should also be able to connect to a central processor for centralized data processing. Similarly, the software and hardware of central processor should allow connectivity of multiple local processors simultaneously along with the capability to process the information generated by local processors. Overall, the entire architecture should handle bidirectional communication between all the connected sensors and processors. We will further discuss in detail the software and hardware aspects of our proposed solution and explain overall working of the system.

3.1 Software Aspect of Distributed Architecture

Each section of our solution, central and local processor, has diverse requirements and therefore variety of software are used. In addition to that, selection of right communication mode between the processors for data sharing is also critical part of our solution. In the following sub-section we discuss in detail the data set that is used for our experiment.

3.1.1 Dataset for Proposed Architecture Testing

The open dataset that is used for testing our proposed architecture is part of Smart* project [66]. In the project they deployed multiple sensors in three different homes, Home A, B, and C, and continuously gather operational and environmental data generated by them. The main focus of their work was to measure, through sensors, maximum aspects of a home energy usage that could affect smart grid. The sensors measured environmental condition, overall electricity usage of home and energy consumption of each individual circuit and load.

Home A data set involved a 1700 square feet home having eight rooms and three occupants. The main appliances included 3 air conditioners, dryer, washing machine, dish washer, freezer, refrigerator and heat recovery ventilation. The electric panel of
Home A included 26 different individual circuits, with each individual appliance put on different circuit and measuring value of each circuit provided data related to that appliance usage. The available data set includes power usage of all these 26 circuits measured through sensors. The sensors reported their data at intervals depending on the energy consumption of the appliance i.e. the reporting frequency was higher when appliance used energy as compared to when it was in idle state. Contrary to Home A data set, Home B and Home C data set involves overall usage of power within that home after different time intervals. Overall, all three homes had sensors that generated data to report the actual power usage of respective home. A sample data set of Home A can be seen in the Figure 3.1.

<table>
<thead>
<tr>
<th>Sensor Name</th>
<th>ID</th>
<th>Time</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid</td>
<td>1</td>
<td>1341047862</td>
<td>1004.972</td>
</tr>
<tr>
<td>Dryer</td>
<td>2</td>
<td>1341047862</td>
<td>1.88</td>
</tr>
<tr>
<td>OfficeOutlets</td>
<td>3</td>
<td>1341047862</td>
<td>10.582</td>
</tr>
<tr>
<td>LivingRoomOutlets</td>
<td>4</td>
<td>1341047862</td>
<td>70.698</td>
</tr>
<tr>
<td>GuestBathOutlets</td>
<td>5</td>
<td>1341047862</td>
<td>0.594</td>
</tr>
<tr>
<td>MasterBathOutlets</td>
<td>6</td>
<td>1341047862</td>
<td>2.022</td>
</tr>
<tr>
<td>LivingRoomPatioLights</td>
<td>7</td>
<td>1341047862</td>
<td>4.562</td>
</tr>
<tr>
<td>GuestBathHallLights</td>
<td>8</td>
<td>1341047862</td>
<td>5.643</td>
</tr>
<tr>
<td>DiningRoomOutlets</td>
<td>9</td>
<td>1341047862</td>
<td>77.542</td>
</tr>
<tr>
<td>OutsideOutlets</td>
<td>10</td>
<td>1341047862</td>
<td>0.582</td>
</tr>
<tr>
<td>CellarLights</td>
<td>11</td>
<td>1341047862</td>
<td>-0.11</td>
</tr>
<tr>
<td>Grid</td>
<td>11</td>
<td>1341047865</td>
<td>1003.391</td>
</tr>
<tr>
<td>FurnaceHRV</td>
<td>12</td>
<td>1341047872</td>
<td>490.454</td>
</tr>
<tr>
<td>CellarOutlets</td>
<td>13</td>
<td>1341047872</td>
<td>118.062</td>
</tr>
<tr>
<td>WashingMachine</td>
<td>14</td>
<td>1341047872</td>
<td>1.262</td>
</tr>
<tr>
<td>FridgeRange</td>
<td>15</td>
<td>1341047872</td>
<td>5.196</td>
</tr>
<tr>
<td>DisposalDishwasher</td>
<td>16</td>
<td>1341047872</td>
<td>4.885</td>
</tr>
<tr>
<td>Microwave</td>
<td>17</td>
<td>1341047872</td>
<td>3.311</td>
</tr>
<tr>
<td>KitchenLights</td>
<td>18</td>
<td>1341047872</td>
<td>3.536</td>
</tr>
<tr>
<td>BedroomOutlets</td>
<td>19</td>
<td>1341047872</td>
<td>68.1</td>
</tr>
<tr>
<td>BedroomLights</td>
<td>20</td>
<td>1341047872</td>
<td>4.935</td>
</tr>
<tr>
<td>MasterOutlets</td>
<td>21</td>
<td>1341047872</td>
<td>110.829</td>
</tr>
<tr>
<td>Grid</td>
<td>1</td>
<td>1341047874</td>
<td>1010.471</td>
</tr>
</tbody>
</table>

Figure 3.1: Sample data set of home A
In the data set the first column represents the name of sensor installed at respective circuit while the second column represents unique ID of every sensor. The third column represents the time in Unix (UTC) Time Stamp at which the data is generated while the last column shows actual power in Watt that is consumed by the respective device. As it can be seen from the figure that the data set has extensive time series data for multiple sensors that are present in home therefore the usage of data set is equivalent to getting data from actual sensors. The data set that is available online on the Smart* project [66] website covers data from May 1, 2012 to July 31, 2012. We are using selective part of the data set based on our needs as discussed in the next chapter.

3.1.2 Operating System for Local Processor

The right operating system for local processor should have the ability to connect to multiple devices simultaneously and should be capable to process the generated data by the sensors. In addition to that, it should be capable to communicate with central processor for coordination and data exchange. There are multiple options available for choosing the Operating System (OS) for local processor e.g. Windows, Linux, Android etc. Both Linux and Android suits our requirements that are discussed previously. Android is selected for deployment at the local processor contrary to Linux due the availability of APIs of Android for Bluetooth communication. Other factors that count for its usage over other OSs are:

- It is an open source OS and therefore many researchers and developers are not only familiar with its working but are also contributing for its advancement. Being an open source there is an extensive free support available that is useful for its customization according to our requirement.

- There are many free security tools and options available that can be used to
improve its security.

- Android OS also supports multiple connectivity options like Wi-Fi, Bluetooth, and USB etc. which is highly desirable for local processor due to interoperability. As discussed before that with the advancement of smart grid concept many new sensors are becoming part of the system, therefore integration of these sensors with the local processor requires flexible OS like Android.

- Android also supports connectivity to central processor through the Internet. Since, the central processor in the modern smart grid is far away from the local processor therefore IP based communication is the only possible option. Android best suits this requirement.

- The local processors at homes should be economical and small in order to be used for smart grid network. The characteristic of portability and quick processing on even small processors makes Android suitable for our work.

Overall, interoperability, open source, multiple communication support and portability make Android an ideal choice for local processor OS.

### 3.1.3 Central Processor Selection

The central processor should not just provide support for data storage and its analysis but its selection involves multiple other factors. In our work we have used Amazon Web Service (AWS) as the central processor. There are various reasons for our selection of AWS over other popular cloud services like Google cloud and Microsoft Azure.

- The most important factor that is considered while selecting cloud service is the availability of Software Development Kit (SDK). SDK acts as a firmware in connected devices and supports their connection to the cloud service. AWS has
a flexible SDK and it is well adopted by many manufacturers of devices. Google Cloud does not have built-in support in its SDK for connected devices and needs to be configured according to the requirements. Microsoft Azure SDK can only be configured for Windows OS which is a serious drawback as we are interested in designing a prototype that can be easily replicated on all the devices. The SDK that is available for AWS SDK is installed on our local processors, running Android OS, and it provides an excellent support for connecting local processors with AWS.

- Selection of central processor includes its support for standard communication protocols like Hyper Text Transfer Protocol (HTTP), used for extensive bidirectional data communication, and Message Queue Telemetry Transport (MQTT), used for communicating small messages between server and devices. AWS provides too extensive support for all the common communication protocols. In our proposed solution we are using HTTP as a communication protocol between local and central processor because of wide-ranging communication requirements between local and central processor.

- The effective working of smart grid network is based on bidirectional communication within a network. The AWS SDK provides support for bidirectional communication between our local servers and AWS central server using HTTP.

- Data security is important while selection of ideal cloud service for central processor. AWS has a strong data security feature along with Microsoft Azure and Google Cloud.

- The effective working of smart grid depends on storage of data generated by heterogeneous sensors. AWS has the most matured data storage capability through its DynamoDB service as compared to Google Cloud and Microsoft Azure.
- Smart grid is constantly evolving and many researchers with different backgrounds are working for its development. The central processor should be capable for supporting multiple programming languages for its configuration. AWS is an excellent choice in this regard as it provides support for multiple programming languages like Python, Java etc.

In addition to all these factors, AWS has services like capability of real time data analysis through RedShift and implementation of machine learning algorithms through its machine learning service. All of these features collectively act as decisive factors in selecting AWS for our central processor.

### 3.1.4 Mode of Communication

The distributed smart home architecture contains two major domains for data exchange. One domain includes communication between sensors connected within home and other domain includes communication between local and central processor.

There are heterogeneous sensors having characteristic of communicating using multiple standard protocols like Wi-Fi, Bluetooth, RS 232, Zigbee etc in smart home. In our test bed, Bluetooth is used for communication between sensors and local processor within home as it is a secure protocol and provides effective short range wireless communication between sensors and processor. Typically, sensors that are present in a home are within 100 meter from local processor. The range of Bluetooth communication is also upto 100 meters [67] while modern advancement is underway that can increase its range to 250 meters. The effective range of Bluetooth makes it suitable for communication between sensors and local processor. In addition to that, the maximum data bit rate for Bluetooth is 1 Mbps (Megabits per second) which is also sufficient for data exchange between sensors and local processor. The number of sensors that can communicate with local processor via Bluetooth is limited to 7 sensors at a time in the initial version of Bluetooth. With the advancement in the
Bluetooth technology, the number of sensors that can concurrently communicate with local processor is very high i.e. more than 250 sensors [68]. Overall, considering all these characteristics of Bluetooth, we have used it for effective bidirectional data flow within home. The inclusion of other mode of communications like Wi-Fi, and RS232 can be possible advancement in our prototype. These technologies can easily be integrated with our test bed as the OS of local processor i.e. Android supports all these technologies by using their APIs. The hardware of our local processor, Dragon Board 410c, also supports these technologies. Although Bluetooth well suits the requirements for communication in distributed architecture of smart home but Zigbee also has all these features. In fact Zigbee enables 36000 sensors to send data to local processor simultaneously and it has better range for communication i.e. approximately 300 meters. The lack of support for Zigbee in Android OS limits its usage in our test bed.

The selection of technology for communication between local and central processor involves different characteristic. Since, central and local processors are far apart therefore Internet Protocol (IP) is natural choice for communication. Currently, we are using HTTP as a protocol for communication between local and central processor via the Internet. HTTP is suitable for this type of communication as it is reliable and it supports quick variable data transfer rate. MQTT is another possible option for data transferring between local and central processor but MQTT has low data rate and therefore can only be used if there is low data rate requirement.

The appropriate mode of communication within our proposed distributed smart grid network can be seen in the Figure 3.2.
3.2 Hardware Aspect of Distributed Architecture

Hardware forms an integral part of our proposed methodology and test bed. During the designing of our test bed, sensors and local processor hardware specifications are considered. While, we rely on AWS hardware for central processor. Since the basic idea behind using any cloud service is to consider only the software aspect of that service therefore we will not discuss the hardware of AWS in this thesis. We will further discuss in detail the hardware used for sensors and local processors in the following sub-sections.

3.2.1 Hardware for Sensors

In our proposed architecture, the local processor acts as a smart gateway that has the capability to connect to heterogeneous sensors having different communication technology. The rationale behind selection of Android OS for local processors is its characteristic to support multiple communication technologies like Bluetooth, Wi-Fi, and RS 232 etc. Therefore, according to our proposed methodology multiple types of sensors can be wireless or wired connected to the local processor. We further discuss the hardware of local processor in the following subsection.
3.2.2 Hardware for Local Processor

In our proposed methodology the local processor should have the following characteristics:

- Capable to communicate with heterogeneous sensors.
- Process the data generated by these sensors.
- Communicate with central processor through the Internet.

The capability of communicating with heterogeneous sensors can be achieved through having multiple communication hardware like Wi-Fi, Bluetooth, GPS, RS 232, analog, and digital connectors. In view of these characteristics we are using Dragon Board 410c, a product of Arrow electronics, in our test bed [9]. The Dragon Board 410c supports Wi-Fi 802.11, Bluetooth 4.1, GPS antenna, 40-pin low speed expansion connector, 60-pin high speed expansion connector, and additional 16-pin analog expansion connector [9]. In addition to that, Android OS can easily be installed on it. The Dragon Board 410c can be seen in the Figure 3.3.

![Dragon board 410c](image)

Figure 3.3: Dragon board 410c [9]
Interoperability in Dragon Board 410c is possible due to presence of multiple connectivity options in it. As an example, two different sensors, one with only Bluetooth hardware and other with only Wi-Fi hardware can connect with it simultaneously for data communication. Moreover, GPS antenna can provide location information for calculations that require regional information like weather forecasting data for predicting energy usage. Overall, Dragon board hardware is well designed to manage interoperability that is required in smart home sensors network. Once the data is received by the local processor it has to be processed locally. The processing involves analyzing and making decisions locally on the basis of available data. In our proposed methodology the local processor is not only connected to sensors but also connected to central processor. The decision made by local processor is based on the data received from sensors as well as information received from the central processor. A possible example of that kind of decision that includes data from sensors and information from central processor is to turn on a particular home appliance on the basis of current price. In this scenario, the real time price of electricity can only be received from central processor. Overall, Dragon Board 410c, owing to diverse communication hardware, has all the characteristics for being a local processor in our design.

3.3 Overall Design

There are three main units that are coordinating with each other for successful working of our design. Sensors and actuators, local processor, and central processor.

An important part of our design is bi-directional communication between these modules. The data is generated by sensors that are part of home and is sent to local processor. The local processor has the capability to connect to heterogeneous sensors. So, in order to collect data from all type of sensors the local processor acts as a smart
gateway and aggregate data from all types of sensors present within home. The local processor is connected to sensors as well as central processor and processes the data on the basis of data received from sensors and central processor. Furthermore, the local processor sends the processed data to central processor that aggregates and processes information received from multiple local processors. Overall, our design divides the smart grid into layers with each layer sending only processed data to upper layer. The overall design of our proposed architecture is shown in Figure 3.4.

Figure 3.4: High level architecture of our proposed design
Chapter 4

Test Bed and Experimental Results

The architecture discussed in the previous chapter is tested through an experimental setup which is discussed in this chapter.

4.1 Test Bed

The testing of our proposed architecture involves collective working of three important components:

- Sensors and actuators
- Local processor
- Central processor

We are using an open data set from Smart* project [66] and designed an android application installed on an android device. The data set is stored in the memory of that device. Currently, we are using three different homes data as provided by the Smart project organizers. For each home we are using a separate android device, acting as a source for sensors data. The data set is preloaded in Random Access Memory (RAM) of the android device. In the given data set for Home A there are 26
sensors, measuring internal energy usage of home, and 11 external sensors, measuring environmental factors within proximity of that home. While for Home B and Home C data set we have only 1 sensor value, measuring total power use within home, and 11 external sensors value, measuring external conditions. In the given data sets, all sensors generated data after irregular time intervals, stamped by UTC corresponding to the time it was generated. The sensor android device collectively sends only that portion of data set to Dragon Board 410c that has same UTC.

At first, the sensor application creates bidirectional communication path with local processor i.e. Dragon Board 410c via Bluetooth. It then starts reading the pre-loaded data set from start, referring the first UTC as time zero, and sends the chunk of data having same UTC to the local processor. The board has a separate android application to receive and process data. The wait time for resuming data transfer depends on the value of next unique UTC in the data set. For example, if the difference between initial and next UTC is 5 seconds then the sensor application will send the next data after 5 seconds. In this way an actual smart home, having multiple sensors, is simulated. The local processor also sends the processed data to central processor i.e. Amazon Web Service (AWS). We are currently using two important services of AWS in our test bed. These services are AWS Lambda and DynamoDB. The AWS Lambda service processes data received from Dragon Board 410c while DynamoDB stores that data for future calculations. In addition to that, AWS Lambda also allows bidirectional communication with the board through IP based communication. Figure 4.1 - 4.2 show actual test bed. In the next section we discuss results that are achieved using our test bed.
### Receiver Node

Connected to Galaxy Nexus

External Sensor items: 287

Internal Sensor items: 296

<table>
<thead>
<tr>
<th>No.</th>
<th>Name</th>
<th>Time(UTC)</th>
<th>RealPower</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Grid</td>
<td>1341048008</td>
<td>551.637</td>
</tr>
<tr>
<td>2</td>
<td>Dryer</td>
<td>1341047898</td>
<td>0.281</td>
</tr>
<tr>
<td>3</td>
<td>OfficeOutlets</td>
<td>1341048008</td>
<td>10.272</td>
</tr>
<tr>
<td>4</td>
<td>LivingRoomOutlet-1341048008</td>
<td>71.975</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>GuestBathOutlet-1341047862</td>
<td>0.594</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MasterBathOutlet-1341047862</td>
<td>2.022</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>LivingRoomPatioLights</td>
<td>1341047862</td>
<td>4.562</td>
</tr>
<tr>
<td>8</td>
<td>GuestBathHallLights</td>
<td>1341047998</td>
<td>6.053</td>
</tr>
<tr>
<td>9</td>
<td>DiningRoomOutlet-1341047903</td>
<td>77.948</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>OutsideOutlets</td>
<td>1341047862</td>
<td>0.582</td>
</tr>
</tbody>
</table>

Figure 4.1: Dragon board receiving data from sensors
4.2 Results

The main purpose of our test bed is to analyze advantages of local vs central processing, bandwidth saving in case of localized processing and possibility of including machine learning algorithms at local and central level for decision making. In the following sub-sections we discuss these advantages in detail.

4.2.1 Local vs Central Processing

Simple calculations are used to measure advantages of local processing. In this part we focus on comparison between round trip time for data from sensors to local and central processor. Although all the sensors continuously send data to local processor depending on UTC value but we specifically focus on 'total power usage' sensor value from home A data set for our calculations. 1000 Watt(W) is set as threshold for excessive power usage in home A. In our experimental setup, the sensor application continuously sends total power usage value from the data set to Dragon Board 410c on the basis of UTC. As the total power usage value gets higher than 1000
W, the sensor application generates signal for excessive energy usage, we refer it as point A in Figure 4.3. It sends that signal to local processor. The time Dragon Board 410c receives that signal is referred as point B in Figure 4.3. Meanwhile the sensor application continuously sends next total power usage values on the basis of UTC in the data set. After receiving excessive power usage signal, the local processor starts monitoring values of total power usage. Once, the value goes below our set threshold the local processor generates signal for low power usage and sends the signal to sensor application. We refer this time as point C in Figure 4.3. The time at which sensor application receives low power usage signal from Dragon Board 410c is referred as point D in the figure. The difference between B and C, referred as E, is the total time total power usage remains higher than the set threshold while the difference between A and D, referred as F, is the total time the power usage value remains higher than the threshold including round trip time for Bluetooth communication. Therefore, the round trip time for Bluetooth communication between sensor application and local processor is the difference between F and E, calculated by sensor application. The detail description of round trip time for Bluetooth communication is shown in Figure 4.3.

![Figure 4.3: Description of round trip time for Bluetooth communication](image)

We further extend this process to central processor i.e. AWS in which monitor-
ing the value of total power usage is done through AWS Lambda service. Similar to local processor the AWS Lambda once received signal for excessive energy usage keeps track of the total power usage values. Once, the value goes less than the set threshold i.e. 1000 W, it generates signal for low power usage and sends it to local processor which forwards it to the sensor application. The sensor application processes that signal as it does for local processor to calculate round trip time for communication between sensor application and AWS.

As discussed earlier that the purpose of using simple calculations is to understand the round trip time for signal from sensor to local processor and from sensor to central processor. In order to observe round trip time we used data set from July 1, 2012 to July 7, 2012 and stored it in the sensor application. To understand the effects of other Internet users on our central processor round trip time we monitored our test bed during different duration of a day. Specifically, we monitored it during six different duration of a day i.e. 11 pm – 12 am, 3 am – 4 am, 9 am – 10 am, 3 pm – 4 pm, 6 pm – 7 pm, and 10 pm – 11 pm. The reason for selection of these time duration is to cover the depth and breadth of our analysis while at the same time simplifying our analysis. Figures 4.4 - 4.9 contain the round trip time for Bluetooth communication between sensor application and local processor as well as central processor. The data points in the figures represent the number of times data is being transferred from sensor application to local and central processor.
Figure 4.4: Round trip time from 11 pm - 12 am

Figure 4.5: Round trip time from 3 am - 4 am

Figure 4.6: Round trip time from 9 am - 10 am
Figure 4.7: Round trip time from 3 pm - 4 pm

Figure 4.8: Round trip time from 6 pm - 7 am

Figure 4.9: Round trip time from 10 pm - 11 pm
The average for local and central processor over these durations can be seen in Figure 4.10 and Figure 4.11. The standard deviation for local and central processor round trip time over these durations is shown in Figure 4.12 and 4.13.

Figure 4.10: Average round trip time for central processor during different times of a day

Figure 4.11: Average round trip time for local processor during different times of a day
It can be observed from the results that the average round trip time between sensors and local processor, using Bluetooth technology, is approximately 65 milliseconds while for central processor that includes the IP based communication medium is approximately 485 milliseconds. The round trip time between sensors and local processor is comparable to round trip time, 77 milliseconds, of [69] that are also using Bluetooth medium for communication. Furthermore, the standard deviation for local processor communication is only 2.2 milliseconds while for central processor it is 36.8 milliseconds throughout a day. The small standard deviation for Bluetooth communication enables the local round trip time to be highly predictable as compared to central one.
4.2.2 Bandwidth Saving

Measuring and reporting local load information continuously at certain time intervals, depending on UTC value, generates massive amount of data within smart grid network. One way is to transmit each measurement to the central processor but that will generate excessive traffic. This might be affordable with small number of sensors in the network but become infeasible if the number of sensors is high. In order to demonstrate the potential bandwidth saving capacity of our test bed we implement a simple scenario. In this scenario our objective is to monitor home energy usage for real-time price calculations. Since real-time price calculation is not based on single home therefore energy usage from multiple homes has to be aggregated. As the energy usage of appliances within home does not change significantly every single millisecond, we do not have to send all the data to central processor. If sufficient amount of information is extracted locally and the amount of data to be transmitted centrally is reduced then network bandwidth can be saved. To highlight bandwidth saving by locally processing and storing sensor’s data, we transmit the value of sensor to central server only if it differs more than a predefined threshold from the last transmitted value. As discussed earlier that the basic underlying assumption for this approach is that the load profiles of most household appliances follow the on/off model. In other words, switching an appliance on or off will cause an abrupt change in the load profile while between the changes the power consumption remains approximately constant. An example load profile of Furnace Heat Recovery Ventilator (HRV) from Home A data set can be seen in Figure 4.14. It can be observed from Figure 4.14 that when Furnace HRV is in on state than its power consumption is approximately 500 Watt and during off state its power consumption is approximately 30 Watt, following on/off model as discussed above.
We formulated the problem of deciding when to transmit the sensor value as a general change detection problem. Let $x_1, x_2, \ldots$, be a sequence of independent random measurements with an unknown change point $m$, such that the measurements $x_1, \ldots, x_{m-1}$, have values within a predefined range. The goal is to detect the change point $m$ and report the new value of sensor while avoiding to make too many redundant transmissions if a change has not occurred. We used the Shewhart test that is one of the simplest change detection algorithms and suits our goal extremely well.
We propose to transmit the current load value if it differs from the previously reported value by more than a predefined threshold, a new measurement $x_t$ is transmitted if

$$|x_t - \lambda| > \delta$$

(4.1)

where $\lambda$ is the previously reported load, and $\delta$ is the set threshold. In this way, the maximum tracking error at the receiving end is $\delta$. That is, the error can never exceed $\delta$, otherwise it would trigger a transmission of a new value. The choice of the threshold depends on the desired error and transmission rate. Thus, the network bandwidth can be saved by sending sensor values from local to central processor only when the value shows change by a certain threshold.

For testing purpose, we focus on the “total energy usage” value present in home B data set and test our approach by varying the threshold value from 1 Watt to 100 Watts. To explain it further, if the threshold value is set to 10 Watt then the local processor, Dragon Board 410c, sends the total energy usage value to central processor, AWS, only when it shows change by more than 10 Watts. In this way, we are able to avoid excessive network usage while at the same time introducing error in central processor. Moreover, we quantify the error that is introduced at the central level due to our bandwidth saving approach. The error is calculated by taking absolute of difference between last value reported to central processor and the actual value and is shown in Equation 4.2.

$$\text{Error} = |\text{Last Value Reported to Central Server} - \text{Actual Current Value}|$$  

(4.2)

We test the algorithm on one week data set of Home B data set i.e. July 1 to July 7 and calculate the average error corresponding to each threshold value over the span of seven days. The selection of seven day data is used by multiple researchers
in their testing like [70] as it covers the users behavior over the entire week and it is likely that the same behavior is repeated again next week. Figures 4.15 - 4.28 show % value retained at the local server and average % error introduced at the central level corresponding to thresholds ranging from 1 Watt - 100 Watts. % value retained and % error is calculated using Equation 4.3 and Equation 4.4 correspondingly.

\[
\text{% Value Retained} = \left(\frac{\text{No. of values not sent} - \text{Total No. of values}}{\text{Total No. of values}}\right) \times 100 \tag{4.3}
\]

\[
\text{% Error} = \left(\frac{\text{Last Value Reported to Central Server} - \text{Acutal Current Value}}{\text{Acutal Current Value}}\right) \times 100 \tag{4.4}
\]

Figure 4.15: Average % value retained corresponding to different thresholds on June 29
Figure 4.16: Average % error corresponding to different thresholds on June 29

Figure 4.17: Average % value retained corresponding to different thresholds on June 30

Figure 4.18: Average % error corresponding to different thresholds on June 30
Figure 4.19: Average % value retained corresponding to different thresholds on July 1

Figure 4.20: Average % error corresponding to different thresholds on July 1

Figure 4.21: Average % value retained corresponding to different thresholds on July 2
Figure 4.22: Average % error corresponding to different thresholds on July 2

Figure 4.23: Average % value retained corresponding to different thresholds on July 3

Figure 4.24: Average % error corresponding to different thresholds on July 3
Figure 4.25: Average % value retained corresponding to different thresholds on July 4

Figure 4.26: Average % error retained corresponding to different thresholds on July 4

Figure 4.27: Average % value retained corresponding to different thresholds on July 5
Figure 4.28: Average % error corresponding to different thresholds on July 5

The average for % error introduced centrally and % value retained at local level during a week can be seen in the Table 4.1 and Figure 4.29 - 4.30.

<table>
<thead>
<tr>
<th>Threshold Value</th>
<th>Average % Error</th>
<th>% Bandwidth saving</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.039971429</td>
<td>49.37090798</td>
</tr>
<tr>
<td>2</td>
<td>0.092971429</td>
<td>67.2788503</td>
</tr>
<tr>
<td>5</td>
<td>0.248742857</td>
<td>85.6441061</td>
</tr>
<tr>
<td>10</td>
<td>0.519471429</td>
<td>92.42986603</td>
</tr>
<tr>
<td>100</td>
<td>3.946385714</td>
<td>99.11587388</td>
</tr>
</tbody>
</table>

Table 4.1: Bandwidth saving algorithm result

Figure 4.29: Weekly average % error introduced at central level corresponding to different thresholds
Our results show that the proposed reporting scheme reduces the communication load between local and central processor in exchange for a small tracking error at the central level. Specifically, if threshold is 1 Watt then the total average error centrally in “total energy usage” sensor value is 0.04 % while the error is 3.95 % with threshold of 100 Watts. It is evident through these results that with minimal error of 0.04 % at central processor we are able to save almost half of the network usage.

4.2.3 Predictability of Energy Usage in Smart Grid

The ability to predict future energy requirements is a critical component of variety of applications in smart grid that seeks to conserve or improve management of energy resources. Utilities, for example, use forecasting of future demand to determine how to manage energy generation. With the recent widespread development of smart grid, forecasting at the scale of individual homes becomes necessary for enabling effective demand response systems and user-side energy management. A system that is able to accurately estimate energy requirements at the scale of individual appliances in home can enable feedback and recommendations to support load shifting to reduce peak demand or help home owners better understand how they can modify usage to minimize electricity bills.

In this thesis, we undertake a study to answer the following question: how pre-
dictable is home energy usage? Our goal is not to design a new prediction algorithm, a topic that has been comprehensively studied in the literature, but to derive insights into the predictability of home energy usage for demand-response systems in particular and home energy management in general. To answer the above question, we study the predictability of energy consumption at the scale of individual appliances. For calculation purposes, we compare the prediction accuracy at the time horizon of fifteen minutes into the future, which can be expanded to an hour, day or week.

Our second contribution is to understand the performance of different machine learning algorithms that can be used for energy usage prediction. We show that simple algorithm that uses basic features has comparable performance to sophisticated machine learning algorithms and time series predictors. Since a simple algorithm has low computational and storage needs, it can considerably reduce the need for large processors and can easily be deployed on small distributed processors.

For calculations and to understand the importance of machine learning for predicting energy usage in distributed architecture of smart grid, we predict Furnace HRV energy usage in Home A using data from July 1 to July 7, 2012. The model that is designed to predict Furnace HRV energy consumption has input parameters like external temperature, internal home temperature, external humidity etc. As we have an extensive data set that contains values of sensors that are measuring these parameters therefore we can easily use the data set for our calculation. In our process we accumulate data generated by sensors into 15 minute intervals i.e sensor data is averaged over all readings within the interval and reported at the end of that interval. Each day has intervals 0,...,95, where interval 0 occurs at 12 midnight and reports on readings of respective sensor from 11:45 PM to midnight while interval 95 reports the readings from 11:30 PM to 11:45 PM. These readings are then used to predict Furnace HRV usage.

We generate forecasts for next 15 minutes Furnace HRV usage, based on present
readings of sensors and readings gathered in previous intervals. To simplify calculations on our local processor we initially test our model in MATLAB and use its ANN tool to determine which sensor value, within our data set, can best predict the Furnace HRV energy usage. The sensor that gives minimum error when it is used to predict Furnace HRV energy usage is then used in real test bed. The purpose of this strategy is to get maximum accuracy while keeping model simple. The % error is calculated according to Equation 4.5.

\[
\text{% Error} = \left( \frac{|\text{Predicted Value} - \text{Actual Value}|}{\text{Actual Value}} \right) \times 100
\] (4.5)

The ANN model that is used to implement this strategy is shown in Figure 4.31. The model consists of 1 input layer, 1 output layer and hidden layer with 15 neurons. The input layer in this case is different sensors values while the output layer is the Furnace HRV usage. 15 number of hidden layers are selected based on multiple iterations in hidden layer neurons number and selecting the number that gives minimum error. The training of data set is performed using Levenberg-Marquardt algorithm and the minima in error is achieved after 8 iterations. 70% of the data set is set as a training data while both testing and validation are 15% of the data set respectively.

![Figure 4.31: ANN model to predict energy usage of furnace HRV](image)

We vary different sensors values i.e. average external temperature during the interval, average internal temperature, change in both these temperatures during 15
minutes interval, average humidity etc. In addition to that, we also use combination of these sensors as input parameters for our predicting model. Specifically we collectively use internal temperature, external temperature and humidity as input parameters for predicting Furnace HRV usage as shown in Figure 4.35. After simulating results for a week from July 1 to July 7, we calculate that value of Furnace HRV sensor is highly dependent on change in external temperature value. The average % error achieved with different sensors is shown in Figures 4.32 - 4.37. It can be observed from these figures that:

- As data set increases from only 1 day to full week, the average % error decreases in predicting Furnace HRV usage. The decrease in the error is because of the fact that ANN model is using batch processing of data for prediction. Therefore as the data set increases the accuracy of prediction also increases.

- The Furnace HRV usage is highly dependent on few parameters like average external temperature and its change during the interval, while it is less dependent on other parameters like average internal temperature.

![Average % error in the predicted value of furnace HRV using external temperature as input parameter for ANN model](image)

Figure 4.32: Average % error in the predicted value of furnace HRV using external temperature as input parameter for ANN model
Figure 4.33: Average % error in the predicted value of furnace HRV using internal temperature as input parameter for ANN model

Figure 4.34: Average % error in the predicted value of furnace HRV using humidity as input parameter for ANN model
Figure 4.35: Average % error in the predicted value of furnace HRV using external, internal temperature and humidity combined as input parameters for ANN model

Figure 4.36: Average % error in the predicted value of furnace HRV using external temperature change during 15 minutes duration as input parameter for ANN model

Figure 4.37: Average % error in the predicted value of furnace HRV using internal temperature change during 15 minutes duration as input parameter for ANN model
The average % error in the predicted values after seven days for multiple sensors can be seen in Figure 4.38.

Figure 4.38: Average % error using different sensors as input parameters for ANN model after seven days

After successfully verifying our model through MATLAB and observing the dependency of Furnace HRV usage on change in external temperature value, we incorporate Apache regression model [71] in the local processor to forecast Furnace HRV energy usage. Regression analysis is a statistical process for estimating the relationships among variables. Its focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps to understand how the typical value of the dependent variable (or 'criterion variable') changes when the independent variables are varied. After developing a regression model, if an additional value of X is then given without its accompanying value of Y, the fitted model can be used to make a prediction of the value of Y. Regression model describes the relationship between a dependent variable, Y, in our case it is energy usage of Furnace HRV, and independent variable or variables, X, it is change in external temperature in our case. Mathematically, it can be represented as shown in Equation 4.6.
\[ Y = I + S \times X \quad (4.6) \]

Where I is ‘intercept’ and S is ‘slope’ and they are related to line that fits the model. The modeling accuracy is determined using least square estimation with one independent variable, X. The accuracy of a line through the sample points is measured by the sum of squared residuals (vertical distances between all the points of the data set and the fitted line), and the goal is to make this sum as small as possible i.e. to minimize the value of Q as shown in Equation 4.7.

\[
Find \quad \text{min} Q(I, S), \quad \text{for} \quad Q(I, S) = \sum_{i=1}^{n} E_{i}^2 = \sum_{i=1}^{n} (Y_{i} - I - S \times X_{i})^2 \quad (4.7)
\]

Where n represents the number of points in the data set. \(Y_{i}\) and \(X_{i}\) are all pairs of data set that are used to design model. In order to minimize the value of Q we compute the gradient of Equation 4.7 with respect to I and S separately as shown in Equation 4.8 and 4.9.

\[
\Delta \sum_{i=1}^{n} E_{i}^2(I) = -2 \sum_{i=1}^{n} (Y_{i} - I - S \times X_{i}) \quad (4.8)
\]

\[
\Delta \sum_{i=1}^{n} E_{i}^2(S) = -2 \sum_{i=1}^{n} (Y_{i} - I - S \times X_{i}) \times X_{i} \quad (4.9)
\]

The minimum value of Q is calculated by setting the gradients as calculated in Equation 4.8 and 4.9 equal to 0. The regression model can be extended to multiple input parameters. The multiple linear regression model as represented by Equation 4.10 is used for predicting the value of one dependent variable from the values of two or more independent variables.

\[
Y = I + S_{1} \times X_{1} + S_{2} \times X_{2} \quad (4.10)
\]
Where $I$ represents intercept and $S_1, S_2$ represents slope of independent variables $X_1, X_2$ respectively.

When we implement the regression model in our real time data set we are able to achieve average % error of 5.24 after a week using change in external temperature during 15 minutes interval as input parameter. Similarly, in addition to change in external temperature, when we use average humidity and average external temperature during 15 minutes as input parameters to our multiple linear regression model, we are able to achieve average % error of 4.91 after a week. Since, we are using different algorithms and models in simulation and actual test bed therefore we are getting different results. The main purpose of simulation results of MATLAB is to understand the input within data set that can most accurately predict Furnace HRV value. The predicted value of Furnace HRV is sent to AWS which after receiving energy usage from other local processors predicts the entire network energy usage for next 15 minutes and broadcast the energy usage price for next 15 minutes to all the local processors.

One main advantage of using regression model is that we can restart forecasting quickly after the external temperature sensor failure. Since only two previous time intervals readings are needed, forecasting can resume within half an hour of sensor malfunction or data miscommunication. Another benefit of our approach is that it requires no training data set as the Apache regression model allows for dynamic updating of the data and uses online machine learning method. In online learning method, as more data becomes available in sequential order it can be added to the regression model to update the model for future prediction. To explain it further, lets consider that currently we have 'n' actual values of Furnace HRV, represented by $y$, i.e. $y_1, \ldots, y_n$. These values are corresponding to 'n' change in external temperature sensor values, represented by $x$, i.e. $x_1, \ldots, x_n$. The linear regression model is designed using the data set as shown below:
This model can be represented according to Equation 4.11.

\[
Y = HS
\]  

(4.11)

Where in Equation 4.11 H represents design matrix while S represents parameters that need to be optimized. The regression model is fitted using sum of square residuals as discussed in Equation 4.7 and can represented as shown in Equation 4.12.

\[
RSS(S) = (\hat{Y} - HS)^T \times (\hat{Y} - HS)
\]  

(4.12)

Where in Equation 4.12 \(\hat{Y}\) represents actual values of data sample. Now to minimize error of Equation 4.12, we take gradient of Equation 4.12 with respect to S which can be represented as:

\[
\delta RSS(S) = -2H^T(\hat{Y} - HS)
\]  

(4.13)

The minima of Equation 4.13 is calculated by equating it to zero and the values of S for which error is minimized is given as:

\[
S = H^T H^{-1} H^T \hat{Y}
\]  

(4.14)

Now, using the fitted model the next value of Furnace HRV is predicted. Once, the actual value becomes available after 15 minutes interval, it is added to our training data set and new regression model with updated S parameters is computed using
Equation 4.14. The updated model will have new intercept \( I' \) and slope \( S' \) as seen in Equation 4.15.

\[
Y = I' + S' \times X
\]  

(4.15)

The multiple regression model is updated through online learning model using similar approach as discussed above. In case of multiple linear regression model there can be 'd' number of input parameters on which 'Y' (Furnace HRV usage in our case) is dependent. The overall model for multiple regression model is shown below:

\[
\begin{bmatrix}
    y_0 \\
    y_1 \\
    \vdots \\
    y_n
\end{bmatrix}
= \begin{bmatrix}
    1 & x_{10} \ldots x_{d0} \\
    1 & x_{11} \ldots x_{d1} \\
    \vdots & \vdots \\
    1 & x_{1n} \ldots x_{dn}
\end{bmatrix}
\begin{bmatrix}
    s_0 \\
    s_1 \\
    \vdots \\
    s_d
\end{bmatrix}
\]

It can be observed from Figure 4.39 and 4.40 that initially the accuracy of our model is quite unpredictable but as more data is available then the updated model is able to predict energy usage of furnace HRV with higher accuracy and smaller standard deviation. Figure 4.40 shows the average % error in prediction after every day using external temperature as input parameter while Figure 4.41 shows the average % error in prediction after every day using change in external temperature, average external temperature and humidity as input parameters. Figure 4.39 is the % error in prediction after every 15 minutes interval.
Figure 4.39: Percentage error in predicted energy usage of furnace HRV using external
temperature as input parameter for regression model after each 15 minutes interval

Figure 4.40: Average % error in the predicted energy usage of furnace HRV using
external temperature as input parameter for regression model after every day

Figure 4.41: Average % error in the predicted energy usage of furnace HRV using
average external temperature, humidity and change in external temperature as input
parameter for regression model after every day
So, overall we are able to predict energy usage of Furnace HRV locally with approximately 5\% error using online stream processing technique. Since, Furnace HRV sensor is generating value approximately after every five seconds therefore the main purpose of predicting energy usage locally is to decrease network usage and only use the network for sending final calculated energy prediction to the AWS.

4.3 Discussion

Overall, through the above stated results we are able to demonstrate advantages of local processing of sensors generated data over central processing. The summary of these results is:

- The local processing of data is favorable for applications that are time critical as it takes approximately 65 milliseconds to send and receive data back between local processor and sensors. While, for the data processing between local and central processor, the time varies from 400 milliseconds to 700 milliseconds depending on the network load. The average round trip time for central processor communication is 485 milliseconds.

- The demand response applications and critical peak pricing [72] can be well managed through our proposed architecture. Since the local data communication has less standard deviation as compared to central server therefore critical peak pricing and demand response can be possibly done locally through collaborating with central server. The local processing of the data also improves the security and privacy of users data as the data is not exposed to external world.

- It is discussed in [73] that future smart grid contains applications like Time-Critical Wide Area Measurement and Control system that require the data processing in micro seconds for effective working of system. Due to high standard deviation and latency in communication with central server these kind of
calculations will not be possible. It is through using the distributed architecture that time sensitive calculations can be performed.

- The network bandwidth can be saved by sending the value of sensors, like overall power usage within home, from local to central processor only when it changes by a certain threshold. The test results show that with a change of value of 1 unit to 100 units and we are able to save bandwidth from 50 % to 99 %. Although, the load on network decreases due to this approach but it adds error in the values of central server that varies from 0.04 % to 3.95 %. Therefore, the value of threshold is application dependent as there are certain applications that can not tolerate error of even 3.95 % that is achieved for 100 unit threshold. In that case, the amount of threshold needs to be decreased to improve accuracy in the central readings.

- Successful prediction of energy usage locally of one appliance i.e. Furnace HRV with error of approximately 5 % shows that in distributed architecture there are certain calculations that can be done locally to avoid excessive network usage. In addition to that, there are certain other calculations that can not be done locally like calculation of current electricity price. In order to calculate the price, central server must collaborate with all the local processors to receive their local energy usage.

- In addition to all these calculations, the overall power usage of each home is also sent to the central server according to the time stamp that is present in the data set. The central server collects and stores that information. The aggregated power usage is then used to calculate real time power price that is being sent back to the local processor after every 10 milliseconds for future local calculations. In this way, an overall real smart grid system can be realized.
Chapter 5

Conclusion and Future Work

Smart grid is considered as the revolutionary power generation and distribution system with advance features of real time monitoring of entire networks with the help of distributed sensors. We have successfully implemented distributed architecture of smart home, part of smart grid, and discuss in detail time delay, network bandwidth saving and possible local processing algorithms within smart grid using our test bed. The centralized data processing system that receives data from all sensors of smart grid creates a bottle-neck for data transmission resulting in network congestion issues. This challenging task is solved through a distributed architecture. The distributed architecture contains local processors that can collect and process data from sensors connected to them. Although Dragon board 410c is used as local processor for data processing but real time data analysis is also possible through AMI as discussed in [74].

In our test bed the round trip time for sensors to communicate with local processor is small i.e. approximately 65 milliseconds contrary to large round trip time for central processor i.e around 485 milliseconds. Furthermore, the network bandwidth saving is also an additional advantage due to the usage of distributed processing and successfully tested in our test bed. While, introduction of artificial intelligence in our system is in preliminary stage but we are able to produce encouraging results
to predict energy usage due to Furnace HRV present within home. The predicted usage value is sent to the centralized server that after receiving predictions from all the homes predicts the next duration energy usage. This system enables network saving, real-time processing and intelligent local data processing. Distributed smart home architecture requires extensive bidirectional communication between homes and central processor and decisions about power usage need to be made at the edge for efficient network usage and quick processing. There are multiple areas in which distributed smart grid architecture can be improved:

• Optimal placement of distributed processors: The smart grid requires optimal solutions to the placement of processors in the network to achieve smart grid goals of efficiency through loss minimization and high-quality power delivered to the ultimate user. One possible addition to our proposed architecture is inclusion of a middle processor between local and central processor that can act as a collaborator for multiple local processors within a certain boundary. In this way, central processor can be brought closer to the edge.

• Integrated forecasting models: Load and generation forecasts are intricately entwined to varying degrees of complexity. Formulations and solutions are required to allow for new realistic decision making.

• Communication medium: New or improvement in communication infrastructure is required for self-healing grids and to enhance reliability for data flow in the network.

• Advancement in control algorithms: Multiple control and prediction models can be added in smart grid to optimize its working. As an example, machine learning algorithms can also be added in the central processor that can aggregate all the predicted energy usage received from local processors to predict overall energy usage of the entire grid.
• Local processors: Possibility of implementing control algorithms for smart homes in already existing embedded devices like network routers can be an interesting area of research.

• Addition of renewable energy to smart grid: More distribution generation by renewable energy sources can be added to the smart grid. Furthermore, the related problems due to intermittency of such energies need to be evaluated. Number of applications of renewable energies are assumed necessary before and in parallel with the advancement of smart grid.

In nutshell, distributed smart home architecture is definitely the next development step in smart grid advancement but its advantages need to be quantified further to develop it in the right direction.
Bibliography


76


