Virtualizing The Internet of Things

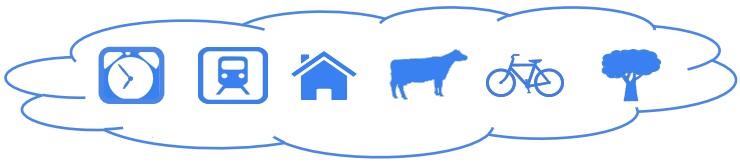
Sarkar, Chayan

DOI
10.4233/uuid:9020a4c1-f81a-4a8c-86f4-024452585505

Publication date
2016

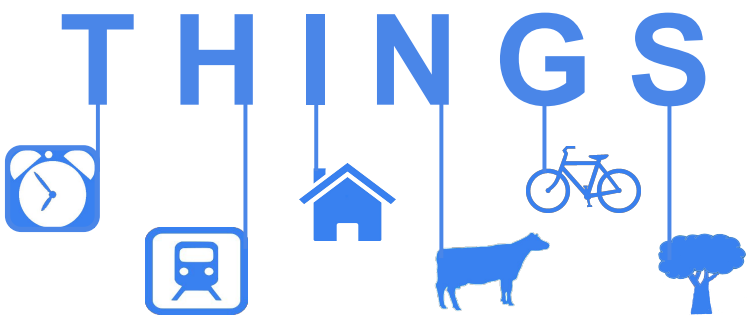
Document Version
Final published version

Citation (APA)
Sarkar, C. (2016). *Virtualizing The Internet of Things*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:9020a4c1-f81a-4a8c-86f4-024452585505



VIRTUALIZING

The Internet of



Chayan Sarkar

Virtualizing The Internet of Things

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op
woensdag 23 november 2016 om 10:00 uur

door

Chayan SARKAR

Master of Technology, Computer Science and Engineering,
IIT Bombay, Mumbai, India
geboren te Mathabhanga, India.

This dissertation has been approved by the

promotor: Prof. dr. K.G. Langendoen
copromotor: Dr. R. Venkatesha Prasad

Composition of the doctoral committee:

Rector Magnificus	chairman
Prof. dr. K.G. Langendoen	Delft University of Technology
Dr. R. Venkatesha Prasad	Delft University of Technology

Independent Members:

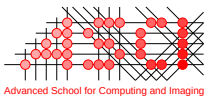
Prof. dr. ing. P.J.M. Havinga	University of Twente
Prof. S. Sitharama Iyengar	Florida International University
Prof. dr. ir. P. Van Mieghem	Delft University of Technology
Prof. L. Muñoz	University of Cantabria
Prof. dr. ir. I.G.M.M. Niemegeers	Delft University of Technology

Author email: chayan@ieee.org

ISBN 978-94-6186-748-3

An electronic version is available at `http://repository.tudelft.nl/`.

This work was carried out in the TU Delft graduate school.



This work was carried out in the ASCI graduate school. ASCI dissertation series number 366.

This work was funded by iCore, a project sponsored by the EU FP7 program.

The true laboratory is the mind,
where behind illusions we uncover the laws of truth.

Jagadish Chandra Bose

Contents

Summary

Computers were invented to automate the labour-intensive computing process. The advancement of semiconductor technology has reduced the form-factor and cost of computers, and increased their usability. This has gradually introduced computers in various control and automation systems. The further rise of miniatuarized computing devices paves the way for autonomous monitoring using embedded devices. In the last two decades, we observed a huge surge of such monitoring and control systems. These systems are generally termed as the wireless sensor and actuator networks (WSAN). In a WSAN, a number of sensor nodes monitors a deployment area where data collection by humans is either difficult or costly. These devices collaboratively report their sensor readings to a centralized node called the sink. The sink is connected with the Internet, thus delivers the data to the outside world. This way the deployment region can be monitored remotely. Similarly, some actuators can also be controlled remotely through the sink.

In the last decade, the concept of the Internet of Things (IoT) has evolved where any device can be reached by any other device/system/human being from anywhere and anytime. Thus, WSANs can be seen as a precursor of IoT. However, the vision of IoT is not limited to mere remote connectivity. Unlike traditional WSAN, where devices are deployed in remote/critical locations for specific purposes, IoT devices would be integrated into our daily surroundings assisting us in every aspect of life. As the embedded devices are resource constrained, energy and computational efficiency is a major challenge for both WSAN and IoT devices. However, the problem escalates as the IoT devices are expected to perform a number of tasks as opposed to a specific task as performed by classical WSANs. Moreover, the goal of IoT is to take humans out of the control loop or reduce the human intervention as much as possible. This requires devices to exchange data and cooperate among themselves. Thus, IoT devices need to act smartly fulfilling various requirements within its resource constraints.

Every existing and upcoming device and network would be part of the IoT ecosystem. As the number of devices is expected to grow multifold, managing these devices will be a challenge. Especially since these devices are under the control of various entities/organizations. Not to mention that the manufacturers of various devices and their specifications would also vary significantly. To accomplish the vision of IoT these devices need to be able to cooperate and collaborate among themselves even if they are managed differently. This thesis brings forward the concept of virtualization in IoT to tackle the challenges of a global IoT ecosystem.

The first challenge that we tackle is how to virtualize the IoT. We propose a reference architectural model for IoT called DIAT. The reference architecture follows a layered design principle where each layer groups a number of similar functionalities together. This enables easy development of existing and new functionalities of each layer independently. To validate the feasibility and usability of such an architectural model, we developed a system based on a practical IoT-application scenario. To this extent, we developed a controller (iLTC) that operates the heating and lighting systems in an office environment such that these devices operate energy efficiently. At the same time the system ensures a comfortable surroundings for the occupants while eliminating any direct involvement from the occupants.

As WSANs are an integral part of the IoT ecosystem, next, we revisited some of the classic problems of WSANs in the wake of virtualizing the IoT. As energy efficiency is one of the biggest issues in WSANs, we propose a solution to reduce the overall traffic in a network without affecting the quality of data/monitoring. We achieved this by virtualizing the WSAN, which leads to higher cooperation among the devices and a higher operational optimization. We developed the virtual sensing framework (VSF) that exploits the inherent correlation among the sensor nodes to predict sensor readings (virtual sensing). The basic idea is that if a number of nodes are highly correlated, sensor readings from only one of them is sufficient to predict the readings for rest of them. Due to virtualization, such a cooperation among the nodes is possible. This reduces the amount of data transfer within the network, which leads to energy-efficient network operation.

Further, we developed an efficient data collection protocol, called Sleeping Beauty that complements the virtualized sensor network. Based on a centralized schedule, nodes deliver their sensor readings to the sink reliably and efficiently. The accomplishment of a centralized schedule depends on network-wide time synchronization. As the hardware clock of an embedded device drifts significantly within a short time span, we developed a simple self-rectification mechanism such that the overhead of synchronizing the network periodically can be reduced significantly. This technique can be used by any protocol that requires time synchronization other than Sleeping Beauty.

Timely data collection is another desired aspect of IoT as opposed to classical WSANs where latency is generally compromised in order to achieve a higher energy efficiency. We developed a communication mechanism, called Rapid that not only delivers the sensor readings in a fixed time bound, it also reduces the energy consumption. Rapid forms a number of clusters on-the-fly, where the cluster-heads collect data from the cluster-members and send an aggregated packet to the sink. By exploiting the capture effect, Rapid achieves parallelization for intra-cluster communications. Further, it exploits the constructive interference based fast flooding to deliver the aggregated data, which eliminates hop-by-hop flow scheduling. These two factors reduces the overall end-to-end delay of all the flows.

The proposition of this thesis is that by means of virtualization, traditional WSANs can be easily integrated into the grand vision of IoT. We proposed a reference architecture, validated by means of a case study, and developed several amendments to classical WSAN data collection, making it consume less energy and achieve lower latency. We are convinced that virtualization can be applied effectively to other (WSAN) functionality as well. The future of IoT is looking bright.

Samenvatting

Computers zijn ooit uitgevonden om arbeidsintensieve berekeningen te automatiseren. Echter, door de voortschrijdende ontwikkelingen op het gebied van halfgeleiders zijn ze steeds kleiner en goedkoper geworden, wat ervoor gezorgd heeft dat het scala aan toepassingen navenant gegroeid is. Zo worden computers tegenwoordig ook gebruikt om productieprocessen en machines te controleren en te automatiseren. De verdergaande miniaturisatie maakt het binnenkort mogelijk om dit soort toepassingen geheel door 'embedded devices' te laten verrichten. Dit kan men afleiden uit de opkomst de afgelopen twee decennia van zogeheten 'wireless sensor en actuator netwerken' (WSAN). In deze netwerken wordt er automatisch data verzameld op moeilijk toegankelijke plekken of om kosten te besparen. De embedded devices sturen de gemeten data naar een zogenaamde 'sink node' die met het Internet verbonden is, waardoor het mogelijk wordt om zaken op afstand te monitoren. Als het netwerk ook actuatoren bevat kunnen die, in omgekeerde richting, aangestuurd worden.

In het afgelopen decennium is het 'Internet of Things' (IoT) concept geëvolueerd tot een systeem waarbij ieder apparaat willekeurig waar en wanneer geraadpleegd kan worden. Echter de IoT visie strekt verder dan dat. Apparaten moeten opgaan in hun omgeving en de dagelijkse bezigheden op elk denkbare manier ondersteunen. Dat is een formidabele uitdaging omdat de meeste IoT apparaten slechts weinig middelen tot hun beschikking zullen hebben. Verder is de idee dat ze hun werk doen zonder, of met minimale inspanning van mensen. Dat maakt dat apparaten zullen moeten samenwerken, immers ieder device op zich is bij lange na niet krachtig genoeg. Het is zaak slim met de beperkte middelen om te gaan om een maximaal resultaat te bereiken.

Als alle huidige en toekomstige apparaten en netwerken inderdaad deel zullen uit maken van het IoT-ecosysteem, zal het aantal apparaten zeer sterk groeien. Het beheer ervan zal dus een heikel punt worden. Vooral omdat deze apparaten gedeeld zullen worden door meerdere partijen. Maar ook omdat ze gemaakt worden door verschillende producenten volgens verschillende specificaties. Het realiseren van de IoT-visie wordt hierdoor bemoeilijkt omdat deze heterogene verzameling apparaten moet samen werken terwijl ze verschillend worden beheerd. Dit proefschrift zoekt de oplossing in het virtualiseren van het Internet of Things.

De eerste bijdrage van ons onderzoek is de definitie van een referentiearchitectuur (genaamd DIAT) voor de IoT virtualisatie. DIAT volgt het gelaagdheidsprincipe waardoor het integreren van bestaande en ontwikkelen van nieuwe functionaliteit

onafhankelijk van andere software lagen kan plaats vinden. Om de werking van het referentie model aan te tonen hebben we het toegepast op een typisch IoT scenario. Ihb. hebben we de besturingssoftware ontwikkeld voor het regelen van de temperatuur en lichtniveau in een kantooromgeving op een energiezuinige wijze. Tegelijkertijd houdt het systeem automatisch rekening met het gebruikers comfort zonder directe tussenkomst van de mens.

Omdat WSANs een integraal onderdeel zijn van het IoT-ecosysteem hebben we bestudeerd wat virtualisatie betekent voor klassieke WSAN bouwstenen zoals bijv. periodic sensing. Omdat energiezuinigheid één van de top prioriteiten is hebben we een methode ontwikkeld die de hoeveelheid data verkeer reduceert zonder de nauwkeurigheid van de verzamelde data te beïnvloeden. Deze methode maakt, uiteraard, gebruik van virtualisatie hetgeen een betere samenwerking mogelijk maakt en er zodoende beter geoptimaliseerd kan worden. De crux is het ontwikkelde Virtual Sensing Framework dat de correlatie tussen de gemeten waarden van verschillende sensoren benut om waarden met grote nauwkeurigheid te kunnen voorspellen ipv. te meten. De toepassing merkt hier niets van omdat het virtualisatie raamwerk automatisch de sensoren clustert en voorspellingen doet op basis van de metingen van één representatieve sensor (en de rest afschakelt om energie te besparen). Deze optimalisatie reduceert het netwerkverkeer aanzienlijk en leidt in het algemeen tot een energiezuinige werking van het geheel.

Omdat niet al het netwerkverkeer vermeden kan worden, hebben we ook een nieuw data collectie protocol ontwikkeld, genaamd Sleeping Beauty (Doornroosje). De acties van de sensoren wordt centraal gecoördineerd om de metingen zo efficiënt mogelijk bij de sink te verzamelen. Een belangrijk element van het protocol is een nauwkeurige tijdssynchronisatie van de nodes onderling waardoor de radio hardware alleen op die momenten ingeschakeld wordt dat er ook daadwerkelijk berichten verstuurd worden. Omdat de (goedkope) interne klokken van de meeste IoT apparaten vrij onnauwkeurig zijn bevat ons protocol een simpel, maar effectief zelfregulerend mechanisme waardoor met minimale hoeveelheid communicatie sensor nodes toch in de pas blijven lopen. Dit zelfregulerend tijdssynchronisatiemechanisme kan ook voor andere toepassingen ingezet worden.

Tenslotte hebben we onze pijlen gericht op het tijdig afleveren van metingen verzameld in het IoT netwerk, dit in tegenstelling tot klassieke WSAN oplossingen die vaak data opsparen om energie te winnen. Het Rapid protocol garandeert een maximale aflever-tijd (latency) in een multi-hop netwerk, en doet dat op een energiezuinige manier. Rapid doet dit door het netwerk op te splitsen in clusters, waarvan de leden hun data bij het cluster-hoofd afleveren, die vervolgens alle verzamelde data in 1 bericht doorstuurt naar de sink. Door het 'capture' effect te benutten kan de dataverzameling in de verschillende clusters in parallel plaats vinden. Voor de overdracht van de cluster-hoofden naar de sink wordt gebruik gemaakt van het 'constructive interference' principe, waardoor een bericht met 1 enkele flood verstuurd

kan worden en kostbare multi-hop communicatie vermeden wordt. Tezamen zorgen deze twee mechanismen voor een forse reductie in latency.

Tot slot, de stelling van dit proefschrift is dat virtualisatie het gemakkelijk maakt om traditionele WSANs te integreren in de brede IoT visie. Daartoe hebben we een referentiearchitectuur voorgesteld, gevalideerd middels een concrete toepassing, en diverse verbeteringen aangebracht aan klassieke WSAN data collectie protocollen die zowel energie zuinigheid als de snelheid (latency) ten goede komen. We zijn er van overtuigd dat virtualisatie ook effectief op andere (WSAN) protocollen toegepast kan worden. De toekomst van het Internet of Things ziet er rooskleurig uit.

1

Introduction

You can't cross the sea merely by standing and staring at the water.

Rabindranath Tagore

In many countries like India, every year a huge number of farmers face a significant amount of monetary loss either due to over-dependence on unpredictable weather or due to over-cultivation of a particular crop in a region. Imagine a future where farmers get suggestion about what crop to grow based on rain prediction and/or ground water level, and precise information about water usage whenever required. Additionally, the cultivation landscape within various regions is monitored continuously such that farmers can cooperatively decide what crop to grow in how much acres of land. Such a cooperation among the farmers can not only avoid over cultivation of the same crop, it can also ensure that raw materials from various farmers do not reach the market all at the same time. To achieve such a goal, a huge amount of data needs to be collected continuously, and smarter decisions can be made by processing these data. Such an enormous goal can only be feasible if data collection, sharing, and processing can be done automatically using low-cost technologies. The advancement of inexpensive sensing and communication platforms brings us to the doorstep of such an era. Not only farming, but every aspect of daily life and business can be enhanced using such technologies.

In today's connected world, there are several means of ephemeral communication amongst devices, e.g., Bluetooth, GSM, NFC, WiFi, ZigBee, etc. However, the idea now is not only to connect with other communicating devices, but also to be aware of our surroundings or an area of interest. This paradigm shift opens up

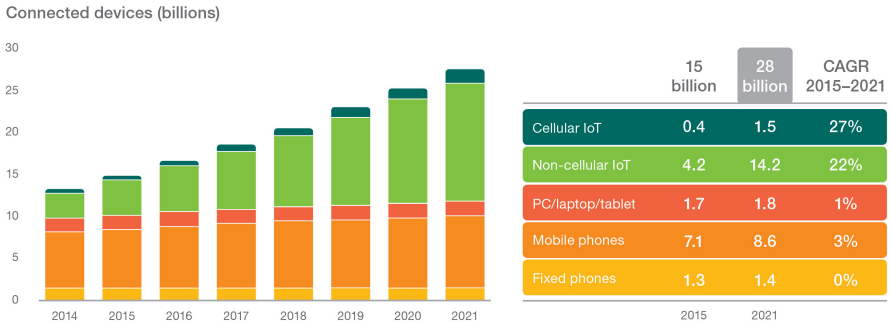


Figure 1.1: Expected growth of IoT devices as compared to traditional connected devices.[1]

the possibility of a number of smart services, e.g., ambient temperature monitoring and thermal control in buildings, traffic monitoring and instant route suggestion, efficient usage of lighting systems, cooperative farming and precision agriculture, industrial automation, etc. An important aspect of these services can be captured by the words of Mark Weiser. In his seminal paper [121], he provided a vision – "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it". Today's advancement in miniaturized technologies and communication substrates makes it possible to easily incorporate a device into our surroundings. Thus, we are about to witness a future where there will be thousands of devices that will seamlessly communicate with each other to support everyday life in a smart way. In general, this vision is referred to as the "Internet of Things" (IoT). The idea is to form an intelligent network of these colossal number of devices, systems and equipment.

In its earlier incarnation, IoTs were represented by Wireless Sensor Networks (WSN) within a limited scope. A WSN is a distributed system composed of many miniaturized sensor devices. These networked devices are used to monitor an area of interest autonomously. They collaboratively collect and report sensed data to a central location/gateway, called the sink, through a wireless communication mechanism. If the sink is connected with the Internet, the reported data can be accessed from any location on earth, a.k.a., the area of interest can be monitored remotely. Some of these revamped networks include actuator devices that can be triggered automatically/manually from a remote location too. They are referred to as Wireless Sensor Actuator Networks (WSAN).

The ease of deployment and management has extended the scope of WSAN application areas from early data collection scenarios like, habitat monitoring [73], structural health monitoring [55], volcano monitoring [123], etc., to ambient assisted living, precision agriculture, industrial automation, etc. The improvement

[1]Image source: Ericsson mobility report, June 2016.

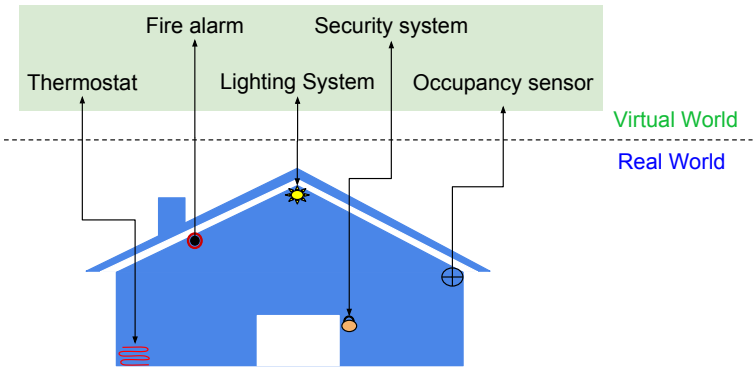


Figure 1.2: Physical IoT devices with heterogeneous features and resources form the real world. Their virtual representation breaks the barrier of heterogeneity, which fosters cooperation and operational optimization.

of embedded technologies has also boosted the growth in the number of connected devices around us towards the goal of IoT. A recent mobility report by Ericsson has projected a huge surge of IoT/embedded devices in the near future as shown in Fig. 1.1 [3]. As IoT is seen as a global communication substrate, it can help devices to connect with each other by exploiting any possible communication capability.

Over a decade, a significant amount of research work from both academia and industry has solved the connectivity issues of WSANs. However, the purview of IoT is not limited to remote connectivity. Thus, the technological advancements in the field of WSANs are not sufficient to fulfill the grand vision of IoT. Here we enlist some of the biggest differences between traditional WSANs and modern IoT systems.

- Traditional WSANs were deployed to collect data unattended where manual data collection would otherwise be labor intensive or difficult to achieve. In some cases, actuators are also able to be controlled manually. The overarching IoT vision is to keep humans out of the control loop, where miniaturized devices continuously sense our surroundings and perform a number of automated actions smartly without any human intervention. Even if some human intervention is required, it should be kept minimal.

- IoT may deploy heterogeneous devices to achieve a diverse range of goals as opposed to homogeneous types of devices employed by traditional WSANs towards a single goal. Thus the IoT systems have to facilitate a mechanism for cooperation among these heterogeneous devices.

- As most IoT devices are resource constrained (especially battery operated), their operational optimization is one of the strongest requirements. In the past, a number of efforts were made to optimize the activities of individual or a group

of homogeneous devices. There is a larger scope of optimization when all the devices belonging to a system are considered as a whole.

These differences lead to a number of barriers that need to be solved to attain the IoT vision. In this thesis, we propose the virtualization of IoT and show how it can overcome these barriers.

In computing, virtualization refers to the process of creating a virtual version of something, rather than real, which includes computer hardware platforms, operating systems, storage devices, and computer network resources [2]. Usage/deployment of multiple applications simultaneously is costly if each requires dedicated infrastructure/resources. The concept of virtualization is developed to resolve such scenarios. Virtualization can create multiple virtual copies of the same resource, and each copy can be used by a separate process. In contrast with partitioning resources and assigning a dedicated chunk to each processes, virtualization provides the abstract view of owning the whole set of resources by each processes, and also using the whole (or most of them) resources in a time-shared manner. This provides a higher utilization of resources, and saves money.

We employ the same philosophy to tackle some of the fundamental challenges of IoT. Fig. 1.2 shows an example of various IoT devices available in a smart house. Due to IoT, the thermostat can already be controlled remotely, but smarter control based on occupancy provides a scope of energy saving. Similarly the lighting system can also be controlled energy efficiently based on occupancy. A trivial approach would deploy two separate occupancy sensors for the two applications. Instead of using two different sensors, a single occupancy sensor can be shared between them. Using virtualization, there is an opportunity for better optimization in resource sharing.

Even though virtualization can reduce infrastructure cost, still there will be a large number of devices around us. These devices will be manufactured and managed by different owners/organizations. In order to share resources among multiple applications, these devices should be able to interact with each other seamlessly. Using virtualization, an IoT device and its functionalities/resources can be represented in a homogeneous way. This facilitates easy cooperation among the devices, which fosters operational optimization of the whole system. Moreover, due to the abstraction achieved through virtualization, a high level of intelligence can be applied to the system to reduce human intervention. The system-level intelligence can then be split and translated to individual device-level smart operations. This thesis advocates to virtualize each IoT device in order to virtualize the whole IoT ecosystem.

[2] https://en.wikipedia.org/wiki/Virtualization

Problem Statement

The goal of virtualizing the IoT in order to achieve higher optimization and cooperation among the existing and new IoT devices comes with a set of challenges. In the following we explore some of the conceptual and system level challenges of the topic. The discussion takes the plunge towards the main contribution of this thesis – a virtualization framework for IoT and enhancement of WSANs as an enabler of IoT through virtualization.

How to virtualize IoT

Virtualizing IoT is the first challenge. The problem lies with the fact that in the coming years, the number of IoT devices are expected to grow multifold. All existing and evolving networks are presumed to be part of this future IoT infrastructure. IoT will offer unique identification of the objects and their virtual representation as the basis for independent development of services and applications. These will be characterized by voluminous and self-governing data capture, event transfer, network connectivity and interoperability.

Many IoT application domains have been identified, e.g., smart home, smart logistics, smart transportation, smart health care, smart agriculture, etc. [12]. A common factor in all such applications is the inherent *smartness*. Being part of a "smart" application, various devices within an application domain can automatically collect data, share information, and initiate and execute services with minimal human intervention. The main challenge in IoT is to manage and maintain a large number of devices and react smartly according to the data generated by them. In this light, we enlist some of the key factors that dictate the challenges in IoT related research.

- Heterogeneity: IoT Devices are deployed by different persons/authorities/entities. These devices have different operating conditions, functionalities, resolutions, etc. Thus, enabling a seamless integration of these devices is a huge challenge.

- Scalability: The rapid growth of embedded technologies is leading to enormous deployment of miniaturized devices (sensors, actuators, etc.). As the number of devices grows, the data produced by these devices grow multifold. Thus, handling the growth in number of devices and information they produce is a massive challenge for IoT.

- Interoperability: In an IoT application, there are heterogeneous entities (devices and cloud services) with different functionalities such as, data provider, data consumer, service facilitator, etc. Seamless interaction amongst the various entities is crucial to realize the vision of IoT. To ensure interoperability, capabilities/features of the IoT devices need to be projected/exposed in a uni-

fied manner, especially when devices can be managed differently. Similarly, there is a need for standardizing the APIs for coherent synergy among different entities.

- Security & Privacy: Due to the large number and the heterogeneity of the actors involved in IoT, ensuring data authentication, usage control, consistency and protection are a few core issues. To evolve a holistic system design, information security, privacy and data protection need to be addressed properly.

Existing architectural models for IoT usually target a specific application scenario. For example, an architecture for a smart office application has been proposed by Castellani *et al.* [20], and an architecture focusing on the RFID network and smart logistics system has been proposed by Beier *et al.* [14]. Being too closely coupled with their particular use-case, these models cannot be applied for cross domain applications without significant adaptation and improvements. Thus, a new reference architecture for IoT has to be developed so that it can be used as a placeholder for any application domain. It should accelerate the growth of smart applications while efficiently handling the major hurdles of a global IoT ecosystem. The reference architecture should act as a proponent of IoT virtualization such that inter-domain and cross-domain cooperation among IoT devices becomes possible.

System-level challenges

Virtualization not only enables interoperability among heterogeneous devices, it also exposes capabilities of a device in various ways such that it can be part of multiple IoT applications simultaneously. As most of the IoT devices are resource constrained in terms of energy, memory, and processing capability, efforts need to be put forward such that the IoT devices are not overburdened by the virtualization. For example, in order to host/support two different applications at the virtual level, the associated device may have to compute and communicate more frequently as compared to serving a single application. Thus, it is required to perform operational optimization of an individual device while fulfilling the requirements of multiple applications. However, a bigger goal of virtualization is to enable cooperation among the devices to achieve high operational optimization even if they are part of a single application domain. In both cases, optimization at the virtual level has to be reflected and complemented at the physical level. Next, we go through some of the key problems in WSAN domains in the light of revised requirements.

Overall data traffic reduction: Energy is a precious resource in a battery-powered device. Thus, a vast number of existing works have focused on energy efficiency in the WSAN domain [33, 78, 127]. Since the radio transceiver is a major energy-consuming component in a wireless embedded device and data transmission happens infrequently as compared to wireless computer networks, efforts are made to keep the radio transceiver in low-power mode as much as possible. The idea is to

reduce energy waste due to idle-listening and packet overhearing without affecting the essential data transmissions. Duty cycling is the standard approach to reduce energy consumption of individual nodes [3] by periodically putting the nodes to sleep (or low-power mode) [15, 30]. Similarly, many efforts were also made to reduce the overall data traffic within the deployment [74, 107].

When a number of sensor nodes is deployed at a site, the nodes should be deployed in such a way that every part of the monitored area is sensed by at least one node. However, optimal node deployment is a difficult task and requires some *a priori* knowledge about the field that is to be monitored. Moreover, the lack of reliability of the embedded devices can cause node failures. Thus, a typical deployment includes more nodes than needed to accurately and reliably sense the phenomena of interest. This over-provisioning of nodes can be utilized to reduce data traffic within a network. Often nodes show high correlation among themselves based on their sensed data. If two nodes show high data correlation, it is sufficient that only one of them reports its data.

Many existing data gathering techniques that exploits correlations, make some strong assumptions [44, 52]. For example, nodes that are located closeby are highly correlated, whereas far away nodes are loosely correlated. However, in many real-life deployments, two distant nodes can show a very strong correlation. Moreover, correlation among the nodes is not a static phenomena, and it can change significantly over time. To support our claim, let us assume there is a WSN deployment in a forest, comprised of temperature sensors. Now, based on their exposure to the sunlight, we can broadly categorize them into two groups where nodes are correlated at any instant of time – the ones directly exposed to sun light and those in the shade. As the sun moves across the sky, the set of nodes that comes directly under the sun changes. As a result, the nodes belonging to each of the groups also change over time. Moreover, two far away nodes, if exposed to sun at the same time, are likely to show high correlation. Thus, we claim that any correlation-based data prediction should be dynamic and adaptive, and node correlation should be calculated based on data only (irrespective of their geographical collocation).

Routing that complements application layer optimization: As data from all the nodes are available at the sink, the number of correlated node groups in a network and their member nodes can be determined easily. As the nodes belonging to the same group are highly correlated, it can be exploited to predict data for the remaining of the nodes when data from them is not available. Deliberately sending data from only one node per group reduces total amount of traffic within the network and a higher level of energy efficiency can be achieved. However, the question is how to reproduce the data for the nodes from whom data is not available. Importantly, correlated nodes need not produce the same data, which makes data reproduction a non-trivial task. Another important aspect is how to cope with the

[3] In WSAN terminology, an IoT device (sensor and/or actuator) is referred as a node.

dynamics of correlation changing among the nodes over time. Moreover, the role of a representative node within a group needs to rotate in order to balance the burden. Thus, there is a need for *node-scheduling* that can tackle all these issues.

Based on the application scenario, a different node scheduling strategy has to be devised, e.g., k-coverage [11], point-coverage [17], spatial correlation [44], etc. Node-scheduling algorithms, which select the subset of active nodes, follow two general strategies. Either they select the set of active nodes at the application layer and leave the routing to the underlying protocols [11, 17], or they jointly perform node selection and routing [21]. In the first case, an advanced routing protocol is required that should use a minimal number of relay nodes besides the chosen active nodes in order not to diminish the overall efficiency. In the second case, the cross-layer solution is tightly coupled to the particular scheduling scheme making it non-reusable for other scheduling strategies. Thus there is a need for a generic data-collection algorithm that ensures highly-efficient network operation irrespective of the node-scheduling policy.

Neither the traditional data collection protocols nor the routing protocols for node-scheduling are suitable for applications where sensed data or actuation command need to be delivered within a fixed time bound, e.g., industrial automation, automatic lighting control, etc. Since these protocols trade off performance for energy-efficiency, real-time applications demand a specialized communication protocol that strikes a balance between latency and energy consumption.

Routing that supports timely data delivery: Though duty cycling is a very efficient approach to reduce energy consumption of a node, it seriously affects end-to-end latency as packets may, or rather will, be delayed at each hop through the network to wait for the next node to wake up [94, 108]. Protocols that want to ensure real-time data delivery therefor use slot-based synchronous communication instead, where a slot is defined as a small time period to forward a data packet from one node to another. Energy is saved by having nodes become active only during those slots in which they need to transmit or receive data by means of a global schedule that determines the sender/receiver pair for each slot. That in itself introduces two problems: (i) the schedule must be computed, which is non-trivial, and (ii) resources are wasted when nodes have no data to report/forward in their slots.

Scheduling multiple flows is known to be an NP-hard problem [94], where a flow is defined as a communication stream between a source and destination node pair. Flow scheduling becomes even more difficult when each flow needs to be completed within a time bound [93]. The question is then how a real-time communication protocol solves an NP-hard scheduling problem along with providing real-time guarantees at minimal energy expense.

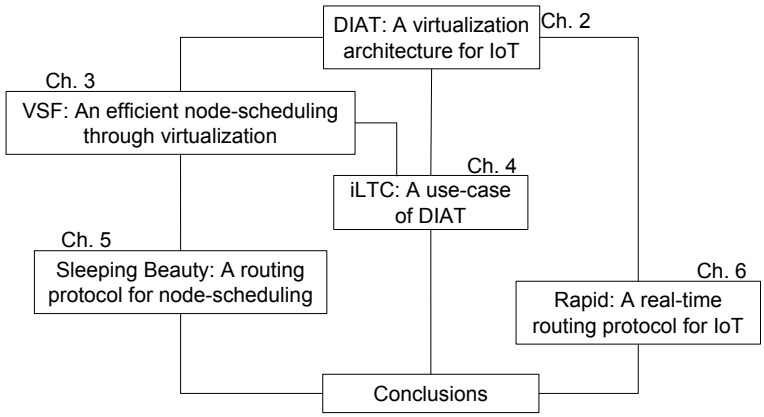


Figure 1.3: Organization of the thesis.

Contributions and Thesis Outline

This thesis uses virtualization for three major benefits – (i) tackling heterogeneity through a unified view of the device, (ii) reduce infrastructure cost by reusing and sharing devices, and (iii) achieve higher operational optimization (specially energy-efficiency) of the deployed devices. We focuses on the design and development of a framework, and enhancing functionalities that help traditional WSANs make a leap towards the IoT vision. Fig. 1.3 shows a pictorial overview of the thesis outline. The contributions of individual chapters are as follows.

A virtualization architecture – Chapter 2. We discuss an analysis of the desired characteristics of the envisioned IoT ecosystem, which helps to outline the requirements and associated challenges. We advocate virtualization as a technique that can underpin these characteristics. However, there are several challenges that need to be overcome to reach the goal. Lack of a generic architecture prompts us to develop a reference architectural model that can subdue these challenges. We propose a layered and distributed architecture for IoT, called *Distributed Internet-like Architecture for Things (DIAT)*. It accommodates heterogeneous objects and provides support for interoperability. Features such as automation, intelligence, zero-configuration, etc., are integral parts of DIAT. Several cognitive functions such as dynamic service creation, modeling and execution are incorporated in the proposed architecture. We will show that DIAT satisfies the key characteristics and goals of an IoT architecture through the study of a realistic use-case. The basic concepts of this architectural model were shaped in an EU project, called iCore [5].

Virtualizing wireless sensor networks – Chapter 3. IoT virtualization can only be beneficial if the concept can be realized at the system level. Before integrating the layered architectural concept of DIAT on a WSAN, first, we focus only on

achieving higher operational optimization through cooperation. One of the major challenges for WSANs is to increase the operational lifetime of the battery-powered nodes without requiring to change the battery. There exist many schemes and protocols that are focused on increasing the lifetime of a WSAN. In this thesis, we take a holistic view towards increasing the energy-efficiency of a WSAN deployment.

We introduce virtual sensing as the building block towards virtualizing a WSAN. A virtual sensor represents a real sensor device and it can accurately predict the sensed data without requiring the real sensor device to provide the measured data. To this extend, virtual sensors exploit the spatio-temporal correlation among the sensor devices. We describe a virtual sensing framework (VSF) as the virtualization scheme for WSAN. VSF requires actual sensed data only from a few nodes and data for the remaining nodes can be predicted using the virtual sensors. Thus VSF selects only a handful of nodes as active and keeps a large number of nodes in low-power sleeping mode at a time instant. This leads to a significant traffic reduction and higher energy-efficiency in a WSAN deployment.

Though VSF tries to keep as many nodes as possible in sleep mode, the number of active nodes depends on the correlation dynamics of the deployment. Unlike previous works, where *a priori* knowledge about correlation is assumed, VSF not only learns correlation on-the-fly, it also copes with changes in correlation patterns over time. We show that the selection of active nodes at a time instant is an NP-hard problem. Thus, we provide a heuristic algorithm to find a minimum number of active nodes. We report around 98% and 79% of data traffic reduction when VSF's activity reduction scheme is used on the IntelLab and GreenOrb datasets, respectively.

A use-case of DIAT – Chapter 4. To demonstrate the benefit of the layered architecture as outlined by DIAT, we explore its features in an IoT application scenario. This chapter describes **iLTC** an indoor environment controlling system that offers automated HVAC and lighting control. The goal of the system is to maintain a comfortable indoor environment for all occupants in a shared space (like an office), while reducing the energy consumption by the HVAC and artificial lighting systems. The obvious benefit is the reduction of the energy bill, which otherwise stands a significantly high proportion of the overall energy consumption of a building. Thus, iLTC achieves the grand vision of IoT, i.e., keeping the human out of the control loop while not only enhancing the human life but also achieving a bigger socio-economic cause.

By virtualizing all the IoT entities, iLTC showcases how operational optimization can be achieved. The distributed and layered design shows how the system scales with a large number of rooms and occupants, and how heterogeneous types of devices can cooperate towards a common goal. Existing solutions for automated lighting control use a dense deployment of light sensors in order to get fine-grained measurements. However, iLTC utilizes a limited sensor deployment and the virtual

sensors as discussed in Chapter 3, to achieve the fine-grained sensor measurements as required by the application. This showcases how virtualization can tackle device heterogeneity, reduce infrastructure cost, and achieve higher operational optimization. Results show that iLTC set-point selection can reduce energy consumption up to 39% and 60% by the HVAC and lighting systems, respectively, compared to the fixed set-point mechanism. We evaluate iLTC with 21 participants housed in multiple rooms and qualitative user evaluation shows over 78% of the participants felt comfortable with the deployed iLTC system.

Routing for virtualized IoT – Chapter 5. Although VSF (Chapter 3) can reduce the overall data traffic significantly, without a complementary routing mechanism, the energy efficiency can be diminished significantly. As mentioned earlier, node scheduling exploits this feature by limiting the number of active nodes to achieve energy-efficient network operation without violating the coverage requirements of the application. However, it is not sufficient to just limit the number of active nodes; active nodes should form a *connected* (sub)network. There exists a large body of work on selecting a connected subset of nodes covering the complete deployment [21, 133]. These optimized node-scheduling techniques, however, are generally not applicable to real-world deployments as they require complete topological information, which is difficult and expensive to obtain accurately.

We present Sleeping Beauty, an energy-efficient communication protocol that operates with partial topological information, yet outperforms state-of-the-art protocols (LWB [36] and FS-LWB [18]) in node-scheduling scenarios. Sleeping Beauty accomplishes this by including (i) an efficient neighbor-discovery mechanism that enables the selection of a minimal, but connected set of active nodes, and (ii) a simple, but elegant clock-offset estimation technique that allows nodes to sleep for a long time without the need for explicit resynchronization. The latter is important for any application that requires time synchronization. We compare the performance of Sleeping Beauty with state-of-the-art protocols on two public testbeds (Indriya and FlockLab), and show that the same performance in terms of packet reception ration (PRR) can be achieved at a fraction of the energy consumption.

Real-time communication – Chapter 6. Apart from energy-efficient data communication, timely data delivery is another desired feature by many IoT applications. Though a number of energy-efficient routing protocols exist in the WSANs domain, timely data delivery is often not the prime focus. For example, Sleeping Beauty (Chapter 5) aimed at energy-efficient routing, without any focus on latency. On the other hand, in the control system domain, a number of routing technique exist primarily focusing on latency of the data delivery with energy efficiency as a secondary concern. However, in resource-constrained IoT devices for real-time applications, energy-efficiency should be considered as another primary concern along with latency.

1

We present Rapid, a highly energy-efficient communication protocol that can provide real-time guarantees. It uses a cross-layer approach – slot-based, synchronous communication at the link layer, and flooding and clustering at the network layer. Using on-the-fly clustering, we split the data collection in two levels. This allows us to achieve parallel operations leading to higher energy-efficiency and low latency together. We compare Rapid with source and graph routing to study how it performs. We also compare performance of Rapid with LWB [36] and FS-LWB [18] on two public testbeds. Results show that a similar packet reception ratio (PRR) can be achieved by Rapid while significantly reduce latency and energy consumption. These benefits are primarily achieved because of the use of fast-flooding based multi-hop communication, which eliminates explicit routing requirements and hop-by-hop scheduling for each flow. A secondary gain follows from tackling the inefficiency of flooding the whole network for reaching a single destination by the use of a selective-flooding mechanism.

Chapters 2, 3, 4, 5, and 6 are based on the following publications:

- Chayan Sarkar, Akshay Uttama Nambi S. N., R. Venkatesha Prasad, Abdur Rahim, Ricardo Neisse, and Gianmarco Baldini, "DIAT: A Scalable Distributed Architecture for IoT", in *Internet of Things Journal*, IoTJ. IEEE, 2015.
- Chayan Sarkar, Vijay S. Rao, R. Venkatesha Prasad, Sankar Narayan Das, Sudip Misra, and Athanasios Vasilakos, "VSF: An Energy-Efficient Sensing Framework using Virtual Sensors", in *Sensors Journal*. IEEE, 2016.
- Chayan Sarkar, Akshay Uttama Nambi S.N., and R. Venkatesha Prasad, "iLTC: Achieving Individual Comfort in Shared Spaces", in *International Conference on Embedded Wireless Systems and Networks*, EWSN. ACM, 2016.
- Chayan Sarkar, R. Venkatesha Prasad, R. T. Rajan, and Koen Langendoen, "Sleeping Beauty: Efficient Communication for Node Scheduling", in *International Conference on Mobile Ad hoc and Sensor Systems*, MASS. IEEE, 2016.
- Chayan Sarkar, R. Venkatesha Prasad, Koen Langendoen, and Thiemo Voigt, "Rapid: Real-time and Energy-efficient Communication for IoT". [submitted]

2

A Virtualization Architecture

The need for a new reference architecture – comprising of smart control and actuation – has been identified by many researchers. This chapter describes the *Distributed Internet-like Architecture for Things (DIAT)*, which has been designed to overcome most of the obstacles in the process of large scale expansion of IoT. We propose a layered architecture that provides various levels of abstraction to tackle issues such as, scalability, heterogeneity, security and interoperability. The proposed architecture is coupled with cognitive capabilities that helps in intelligent decision making and enables automated service creation. Using a comprehensive use-case, comprising elements from multiple application domains, we illustrate the usability of the proposed architecture.

Introduction

IoT can be viewed as a global infrastructure substrate that helps to liaise with any object[1] by exploiting the captured data and its communication capabilities. All existing and evolving devices, networks, and systems are presumed to be part of this IoT infrastructure. The grand vision of IoT is to provide a number of automated services to support every aspect of life. Thus, the billions of IoT devices will play the role of enabler as they will seamlessly integrate into our surroundings, collect data continuously and autonomously, and perform actions without any human intervention.

Based on a common goal, a number of IoT services can be grouped together, which forms a number of IoT application domains, e.g., smart home, smart logistics,

Parts of this chapter have been published as – Sarkar *et al.*, "DIAT: A Scalable Distributed Architecture for IoT", IEEE Internet of Things Journal, 2015 [97].

[1]We use the terms object and device interchangeably.

smart transportation, smart health care, smart agriculture, etc. [12]. Not only the devices that belong to the same application domain, but also across domains devices will share information to cooperate flawlessly. Some of the desired characteristics of cooperative IoT objects as well as IoT applications are listed below [62].

- Automation: Automation is a key feature of any IoT device and application. Autonomous data collection, processing, contextual inference, collaboration with other IoT objects and decision making should be supported by any IoT infrastructure.
- Intelligence: Objects in IoT should act intelligently. Building intelligence into these objects and empowering them to operate adaptively based on different situations is an important aspect. Situation and context awareness are the key entities for an intelligent system that can operate with minimal human intervention.
- Dynamism: An object in a IoT ecosystem can move from one place to another. The IoT ecosystem should be able to dynamically recognize and adapt these objects based on the environment. Thus, dynamic management and integration of these objects across different environments and applications is crucial for a unified IoT architecture.
- Zero-configuration: To support easy integration of devices in the IoT ecosystem, plug-and-play functionality should be available. Zero-configuration support encourages an easy and decentralized growth of IoT systems [120].

The main challenge in IoT is to manage and maintain large numbers of devices and react smartly according to the data generated by them. Some answers addressing this challenge can be seen under the umbrella of the Future Internet and in projects such as BUTLER [1], COMPOSE [2], FIRE [4], IoT-A [6], etc. They deal with large-scale networking, cognitive networking, network of networks [47], as well as service-oriented architecture development for a converged communication and service infrastructure, to mention a few. In the light of growing IoT infrastructure, more sensors and actuators make the system more intelligent and highly responsive. Higher amounts of sensed information and precise control could help in achieving sophistication. Furthermore, there must be many devices to make services fault tolerant and dependable. However, *paripassu*, they also impart difficulties of maintaining them, controlling them and filtering the enormous amount of data generated by them.

As mentioned earlier, there are a number of challenges on the way to realize the vision of IoT. As the devices and systems are manufactured, deployed, owned, and managed by different entities, their behaviors and actions differ significantly. This heterogeneity of devices leads to the problem of interoperability, i.e., two devices can not communicate or cooperate seamlessly, which is otherwise a very important

aspect of IoT. Similarly, the large number of devices raises the scalability issue from various aspects, e.g., network management, data management, data processing, etc. Lastly, communication with the IoT devices should maintain secrecy and privacy, especially for sensitive data and critical actuation commands. Thus, security and privacy is one of the major obstacles for expansion of IoT. As most IoT devices are resource constrained, a significant adaptation and modification is required of traditional security measures while adopting them in IoT devices.

Contributions: In this chapter, we describe a layered and distributed architecture for IoT, called the *Distributed Internet-like Architecture for Things (DIAT)*. We believe that it has the potential to tackle many of the technical challenges described earlier. It also supports the desired characteristics of IoT objects and applications. Our key contributions are listed below:

- We define a layered IoT architecture. Layering binds similar functionalities together and provides a hierarchical structure for these functionalities. It also provides decoupling of orthogonal features (e.g. security, privacy) and encourages their independent development.

- We ensure integration of the desired characteristics of IoT systems with the help of various cognitive functionalities defined at various layers of the architecture. Specifically, we show how dynamic service creation and modeling can be achieved without having any human intervention.

- With the help of a composite (cross application domain) use-case, we showcase that the proposed architecture (DIAT) has *distributiveness*. It is also capable of tackling various technical challenges like, heterogeneity, scalability, and interoperability.

The rest of this chapter is organized as follows. First, we briefly discuss some of the existing IoT architectural design principals in Section 2.2. In Section 2.3, we describe our proposal of a layered architecture for IoT in detail. Then, we describe some of the key cognitive features of the architecture that provide the desired characteristics of IoT in Section 2.4. In Section 2.5, we provide a use-case example to validate our proposal. Finally, we conclude in Section 2.6.

Related Work

Many proposals attempt to define an architectural model for IoT that are usually applicable to a specific application domain. For example, Castellani *et al.,* have proposed an architecture for a smart office application [20]. Their main focus is only to interconnect wireless sensor networks and actuator networks to the Internet as a web service. Similarly, the EPC global IoT architecture has mainly focused on the RFID network and smart logistics system [14]. There is no suitable unified

Table 2.1: Comparison among Architectural Proposals.

IoT Characteristics	DIAT	Butler [1]	Compose [2]	IoT-A [6]
Heterogeneity	yes	partially	yes	yes
Scalability	yes	no	no	–
Interoperability	yes	no	partially	yes
Security & privacy	yes	yes	partially	yes
Automation	partially	yes	partially	yes
Distributiveness	yes	partially	no	–
Layered design	yes	no	yes	no

architecture till date that is appropriate for a global IoT infrastructure. The existing architectural proposals are defined for a particular type of application.

Many researchers have suggested layered architectures for IoT. For example, Gronbaek *et al.*, [41] and Dai *et al.*, [26] have proposed architectural model similar to the OSI architecture. Tan *et al.*, have also suggested a layered architecture for the IoT [112]. However, their design failed to emphasize how to tackle the key challenges in IoT such as, interoperability, scalability, security, etc. There are other works that provide directions for the development of IoT architectures. Ning *et al.*, have provided a comparative study between man-like neural system and social organization framework for IoT architecture development [81]. Kovatsch *et al.*, have proposed a centralized web-resource based architecture for decoupling the application development from the domain of heterogeneous devices [57]. However, these approaches neither guarantee scalability nor tackle interoperability of objects belonging to various application domains.

From the IoT perspective, every object can be seen as a potential service provider. However, these tiny services might not provide a complete application/solution. IoT applications need to holistically combine such multiple tiny services to provide a complete functionality. They add up in right measures and sequence to provide a complete solution to users. In Service-Oriented Architecture (SOA), computers run an arbitrary number of services, and services can exchange information among themselves without human intervention and without the need to make changes to the underlying program itself. Thus, SOA approaches are considered to be applicable for IoT by many researchers [43, 111]. However, the architecture needs to be adopted in such a way that the technical challenges (e.g. interoperability, security & privacy etc.) related to IoT can be tackled.

Overall, there are many pieces of solutions to enable IoT for smart applications; but a comprehensive and holistic design is required. In this chapter, we attempt to provide such a comprehensive design. We compare our proposal (DIAT) with some of the existing efforts based on potentiality of tackling technical challenges

and availability of the key characteristics of IoT (Table 2.1). From the table of comparison, it is evident that our proposal shows a great promise for a global IoT architecture.

IoT Distributed Architecture

Layering is a structuring technique that helps to group similar functionalities and encourages independent development of each layer. Thus, we follow a layered architecture for the IoT infrastructure. In IoT applications, similar to a service-oriented architecture, multiple services from individual service provider (or even individual objects) need to be merged with minimal human intervention. To fetch services from the objects without human interaction and to ensure easy collaboration among services, the objects need to expose their services in a homogeneous way [80]. Hence an abstract representation of the objects along with their features and capabilities is needed. On top of that, smart collaboration and coordination are required among the set of services that can serve towards a common goal. To create and manage the services, various service requests need to be properly analyzed before the execution. Thus, the functionalities of IoT infrastructure are grouped into three layers – (i) Virtual Object Layer (VOL), (ii) Composite Virtual Object Layer (CVOL), and (iii) Service Layer (SL). The three layers are responsible for object virtualization, service composition and execution, and service creation and management respectively.

To support some of the key features of IoT, like automation, intelligence, etc., a set of cognitive functionalities is integrated at each layer of the architecture. The three layers of the IoT architecture and their main functionalities are put together as a stack, called *IoT Daemon* (Figure 2.1). The *IoT Daemon* forms the basis for the distributed architecture, which we call "**D**istributed **I**nternet-like **A**rchitecture for **T**hings (**DIAT**)". Please note that the lowest layer (VOL) is mainly responsible for virtual representation of objects and acts as an interpreter between the physical and the cyber worlds. Features such as management, coordination, automation, etc., are implemented in the upper layers. Therefore, the cognitive functions as well as the resource requirement (e.g. processing power, memory, etc.) increases from lower to higher layers. Here we do not consider building physical communication amongst the devices since it has been dealt in depth in the literature.

In the IoT Daemon, the management of security and privacy policies is performed by the Security Management (SM) cross-layer module. The SM is responsible for the evaluation of policies that should be enforced by the SL, CVOL, and VOL. The integration of SM with DIAT is described in [97]. However, security and privacy of IoT devices is out of the scope of this thesis.

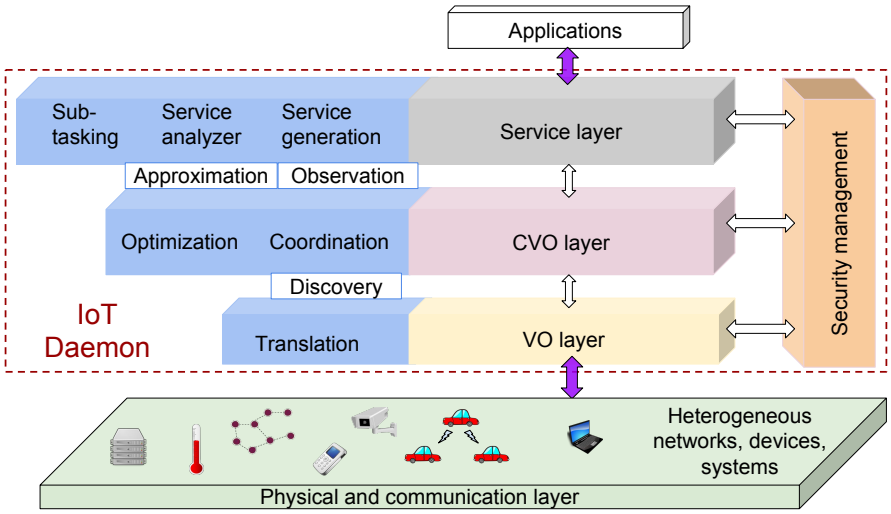


Figure 2.1: A layered architecture for the Internet of Things.

Virtual Object Layer

The *Virtual Object Layer (VOL)* is responsible for virtualization of physical objects or entities. It hosts the virtual representations of the real-world (physical) objects, called *Virtual Object (VO)*. A VO provides a semantic description of the capabilities and features of the associated real-world objects. Semantic modeling of VO enables efficient information exchange by expressing the information and its relationship among other VOs. Through semantic technologies, the VO layer provides universal methodologies to access all physical objects. Furthermore, the VOL plays the role of bridging the gap between the physical and the cyber world. As VOs are the digital representation of physical objects, the objects can be accessible in the cyber world through its corresponding VOs. Indeed, the VOs act as a translator between the digital and physical worlds. This abstraction helps to tackle the heterogeneity in various devices, systems and networks, and ensures interoperability and reusability.

Composite Virtual Object Layer

An individual VO represents its corresponding physical object and it is constrained by the capability of that object. In reality, multiple physical objects need to be engaged in order to accomplish a particular task. They need to communicate and coordinate among themselves to fulfill the goals. A *Composite Virtual Objects (CVO)* does this job. At runtime, a CVO is created by forming a mash-up of one/multiple VOs (and/or other existing CVOs) based on the necessity of a task. During the execution, a CVO dictates how the individual entities (VOs and/or CVOs) in its mash-up should work. A CVO plays the role of a coordinator. Additionally, it tries

to optimize the operations among its entities by doing smart scheduling. The CVO layer (CVOL) hosts the CVOs along with managing their creation and execution.

Before forming a CVO, the CVOL has to identify a set of suitable VOs (and other existing CVOs) that can collaboratively accomplish a service request. Thus, a *discovery* mechanism is identified as one of the key features of the CVOL. Semantic description of a VO (CVO) can easily describe its features and capabilities, and helps in the discovery mechanism. Discovery mechanism is out of the scope of this work; therefore, it is not discussed in detailed manner.

Service Layer

The *Service Layer (SL)* is responsible for creation and management of services. On one hand, it handles various service requests from users. On the other hand, it initiates service requests on its own in order to enable automatic service creation. Additionally, whenever a service request is received, the service layer analyzes and splits it into smaller subtasks. The service layer also decides how these subtasks are assembled to reach the final goal. This (abstract) description of a decomposed service request is provided to the CVOL to execute the task. Using the discovery mechanism, the CVOL identifies the mash-up of VOs and CVOs. If an exact match for VO (CVO) is not found (according to the requirement), an alternative VO can be selected if it can fulfill the requirement partially. Similarly, an exact service match (as requested) might not be available always and a close alternative can serve the purpose. This paradigm is termed as *approximate service* [86, 99].

IoT Daemon

All the proposed layers and their functionalities together is referred as *IoT Daemon*. As all the layers are associated with certain cognitive capabilities, the IoT daemon also consists of cognitive functionalities. It acts as the basis of the distributed architecture in DIAT. The abstracted view of an IoT daemon is shown in Figure 2.1. Every object with some processing power and memory runs its own IoT daemon. However, the footprint of the daemon varies with the capability of the devices.

A full-fledged IoT daemon contains three layers with all the functionalities. But some tiny embedded devices might lack processing power and memory to run a full version of the daemon, e.g., wireless sensors, actuators, etc. As a result, a depreciated IoT daemon may run on those devices with limited set of functionalities. For example, a basic VOL with the capability of hosting at least one VO is executed in a wireless sensor node.

Using IoT Daemon, our proposed architecture tackles all the challenges in IoT mentioned previously. The virtual objects (VOs) hosted by the VO layer are the abstract representation of the corresponding physical devices in the cyber/digital domain. Communication with the devices is performed through the VO layer which hides heterogeneity of devices in terms of capabilities and features.

The VO layer of an IoT daemon hosts the VOs that are directly associated with it. As a result, centralized VO hosting is not required. The same applies to the upper layers as well. A VO can connect with any other VO or CVO and they may not be hosted on the same device. This helps to overcome the memory and other resource requirements of the hosting entities. Thus DIAT is scalable.

The presence of IoT daemon in every object can relieve the burden of configuring each device manually, i.e., ensuring the scalability and zero-configuration requirement [120]. The APIs and interconnections among the three layers ensure interoperability.

To address a service request, often, data from multiple VOs are combined to draw a meaningful conclusion. These operational logics are applied at the higher layers (CVO and Service). Due to the layered design, lower layers of an IoT daemon residing in Device A for example, is able to communicate with upper layer of any other IoT daemon residing in Device B. As a result, all the functionalities need not be hosted in a centralized location. This depicts that the proposed architecture supports distributed operation amongst various IoT entities.

Cognitive Management in IoT Daemon

Smart features like, automation, intelligence, dynamism, etc., are the inherent characteristics of the IoT ecosystem. To support these features, we envisage a set of cognitive functions at the each layer of the DIAT architecture. These functions can be developed independently and integrated into the IoT daemon. Next, we describe some of the key cognitive functionalities of the IoT daemon.

Dynamic Service Creation

One of the major tasks of the service layer (SL) is automatic creation of services according to the context and situation. We define a cognitive entity, called the *observer* that plays a key role in automated machine-to-machine (M2M) communication in order to provide a service intelligently. The functionalities and the work-flow of an observer are spread across the CVOL and SL (Figure 2.2). The observer continuously monitors objects and assesses their situation. Based on available knowledge and current situation, it may decide to initiate a set of service requests without any human trigger.

Ubiquitous and pervasive computing is an inherent feature of IoT. Thus, context-awareness is an essential requirement for IoT. To initiate and execute an automated service intelligently, situations of all objects (humans, devices) need to be assessed carefully. Hence, context-awareness is an integrated part of the observer, which continuously monitors objects and derives the contextual information. The observer stores the contextual information about each object in its associated vector. Objects can be broadly categorized into two groups - human and non-human objects.

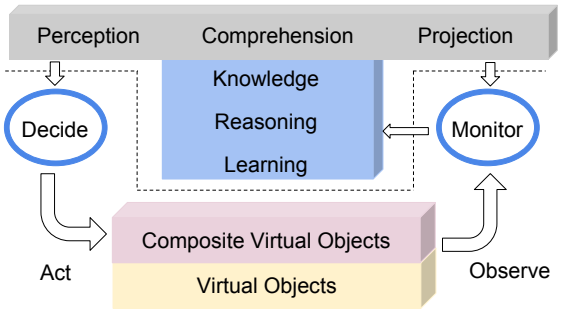


Figure 2.2: Functionalities and work-flow of an observer.

Two different types of vectors are used for these two sets of objects. A contextual information vector for a human object contains the following fields (refer to Fig. 2.3).

Current location
This field contains the current location of the human being. This is not raw data (like GPS data), but high-level information derived using low level data. For example, the current location can indicate *atHome, atOffice, atSupermarket*, etc. Additionally, this field has a sub-field, called *expected location*, which is derived based on the pre-scheduled job list and previous observations. If expected location information is available and if it differs from the current location, the user will be notified or a new service request will be initiated. For example, if the current location is *atHospital* and expected location is *atOffice* due to a scheduled meeting, a service request is generated to inform all the interested party for cancellation and rescheduling of the meeting.

Operating state
The operating state indicates what a person is currently involved in. Examples of operating state can be *inMeeting, isWorking, watchingTV, isSleeping, inHealthEmergency*, etc. This helps to initiate new service requests. For example, if a person is in a meeting, any incoming call is automatically diverted to auto-response mechanism or another person and the actual callee is notified in due time. Similar to the location field, this field also has a sub-field, called *expected operating state*. Any conflict between the sub-field and the main field initiates a new service request.

Next job queue
This field describes upcoming jobs that are derived from the to-do list or calendar events of a person. This field contains two sub-fields - *notification time* and *complementary service*. The notification time is the approximate time when an event should happen. The SL initiates a new service request at that moment. Additionally, the execution of a job might initiate some other jobs. The *observer* tries to

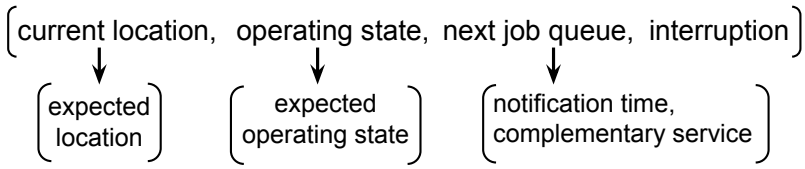


Figure 2.3: Contextual information vector for a human object.

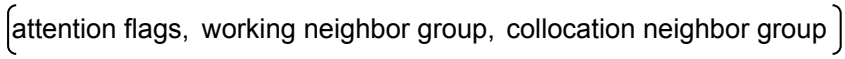


Figure 2.4: Contextual information vector for a non-human object.

determine those jobs and queue them in the list. The complementary service field points those jobs in the queue.

Interruption
Any service request (human-initiated or auto-generated) might require some human intervention at various points during the execution. The *observer* sets an interruption flag, if some service requires human attention. Based on the urgency level, a person is either notified immediately or it is postponed. The operating state of the person is important to decide the urgency level. For example, if a person is in a meeting and the central heating system of his/her house is broken, the person can be informed later. But if there is a burglary, the person should be informed immediately.

Similarly, the contextual information vector for a non-human object contains the fields as shown in Fig. 2.4.

Attention flags
This field contains various flags that indicate something is unusual with the object and attention is needed. The attention may not be necessarily from human beings. It may be in consultation (communication) with other objects. The level of urgency decides how quickly the attention needs to be provided. For example, an *observer* identifies a fire from the data produced by the fire detector VO and sets the attention flag to a value so that an immediate fire alarm service request can be initiated. On the contrary, if a room heater stops working, the attention flag is set to a relatively low priority value such that a service request is scheduled at an appropriate time.

Working neighbor group
This field creates a group amongst similar objects within a small area. The definition of an area can vary in different scenarios. In case of climate controller in an office

domain, the area is defined by the office premises (building). On the other hand, the area for a traffic sensor can spread across couple of blocks. The group members can communicate with each other for coordination and optimize their performance. An Observer can detect an anomaly in one object by consulting with other group members and set the attention flag.

Collocation neighbor group

Similar to the working neighbor group, this field creates and monitors a group among the neighboring objects. These objects need not be of the same type, but they are grouped based on their geographical collocation.

Dynamic Service Modeling

In the previous subsection, we described observer - a key entity to monitor objects and initiate a service request based on the context and situation. Execution of these services in real-time is a challenge in dynamic environments. As the CVOs are the service executors, to support dynamic service execution, we propose "Policy Based Models" for dynamic CVO creation in near real-time. Policies are an effective way to adapt to the dynamic environment and changing user requirements. Policies handle macro level specifications of service request and are dynamically created based on the inputs from observer, historical information and user requirements (Fig. 2.5).

Policies are modeled using semantic technologies namely, Ontology to describe in a machine-interpretable and interoperable manner. Policies represent a structured way to determine the CVOs required to accomplish a service request. A generic policy tuple consists of policy-id, modality, trigger, subject, target, behavior, constraint, role, desires, intentions, and assignment. *Modality* defines the authorization and obligation of the VO's. *Trigger* defines the state of the VO's or the time events. *Subject and target* define the roles of VO's in the system. *Behavior* defines the overall goal of the policy that is depicted by the CVO. *Constraint* specifies condition under which the policy is applicable. *Role* indicates the set of roles required to realize the goals. *Desire* is the set of sub-goals obtained upon on service decomposition. *Intention* specifies the execution manner of sub-goals; plans are updated dynamically based on availability of VO's and knowledge gained over time. *Assignment* is a dynamic mapping mechanism that assigns a role to a particular VO whose behavior matches to the goal of the policy. Further details on policy ontology is described in [80].

Policies are categorized into three groups in accordance with the three levels of the DIAT architecture.

- High level policy (SL level): Service level policy determines the high level abstract from the service request. This policy handles the macro level specification described by the service request.

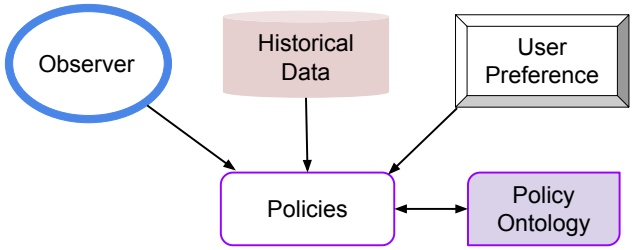


Figure 2.5: Creation of policies in DIAT.

- Concrete policy (CVO level): The CVO level policy determines the sub-goals and roles of the VOs using inputs from observers and policy ontology. Concrete policies defines how to create a CVO to accomplish sub goals in order to complete the service request.

- Low level policy (VO level): VO level policy contains the implementation details in terms of execution plan. VO policies determine what is the role of each VOs in the system.

We extend the Belief-Desire-Intention model in [91] to determine the policies for dynamic service execution. BDI model is extensively used in intelligent agents to model complex and dynamic systems. To enable dynamic service execution and CVO creation we develop Belief-Desire-Intention-Policy (BDIP) model. The major entities in BDIP model are, (i) **B**elief - What the IoT daemon believes about a particular service request. This refers to the initial belief of what the CVO is supposed to do to accomplish the service request. (ii) **D**esire - What are the goals that need to be achieved by the CVO. This describes what the CVO wants to achieve based on the initial beliefs derived. (iii) **I**ntention - What is the plan or sequence of action that need to be done to achieve the goal. (iv) **P**olicy - It guides the intentions, i.e., how to execute the plan of action and the roles of each CVO and VO.

Fig. 2.6, shows the interactions among the entities in BDIP model. To accomplish a service in static environment, we first determine the initial belief, based on the current service request and historical requests. Based on the initial belief, the BDIP model updates the desires of the CVO and determines the sequence of plan to complete the service. Observer monitors the VOs and creates necessary service requests in order to adapt to the dynamic environment. The proposed BDIP model can support dynamic service execution and CVO creation process with the help of policies. Upon the creation of new service requests, the policies influence the initial belief of the CVO to accomplish the new service requests. Policies also change the desires of the CVO and intention of the service requests based on the new service requirements and the dynamic environment. Fig. 2.6, shows sequence of interactions

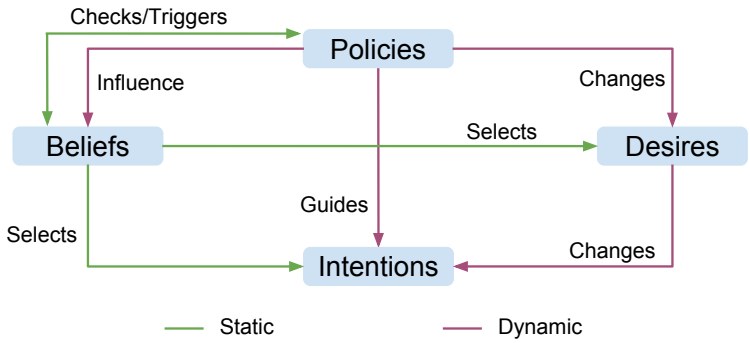


Figure 2.6: Belief-Desire-Intention-Policy (BDIP) model.

in the BDIP model based on static and dynamic service requests. Thus, our proposed BDIP model can adapt to the dynamic environment and user requirements making our architecture more scalable and robust.

A multi-application use-case

To validate our architecture, we describe a composite use-case in this section. The example consists of various IoT applications like smart home, smart transportation, smart health-care, etc., as shown in the Figure 2.7. The proposed architecture overcomes the barrier of (so called) separate IoT applications and achieves a global *Internet of Things* via interoperability. We assume an IoT daemon runs on every object, belonging to various application domains, and they communicate amongst themselves to carry out a task.

In the example shown in Figure 2.7, a smart fridge hosts a simple IoT daemon, which has only the VOL. The IoT daemon hosts a *fridge* VO, and it can inform a legitimate user (e.g. the owner) about its content. An *observer* of the fridge is hosted in a more capable pairing device, e.g., a PC, a smart phone, etc. The *observer* can identify essential items that are not available in the fridge. Let us assume that the *observer* is in the smart-phone of the fridge owner and it may add a task in the "next job queue" if some (essential) items are missing from the fridge. When the owner leaves his/her office in the evening, the *observer* identifies this by analyzing the contextual information vector associated with the person (current location and operating state). It also fetches the scheduled tasks from the job queue and creates a service request through the service generator. The service request forms a CVO, called *shopper*, using the BDIP model by engaging two VOs – the *fridge* VO and the *diet chart* VO. The *diet chart* VO is maintained at the "smart health care" domain. It is created based on a doctor's and/or a dietitian's advice. The *shopper* CVO also finds a convenient place (supermarket) to buy the items.

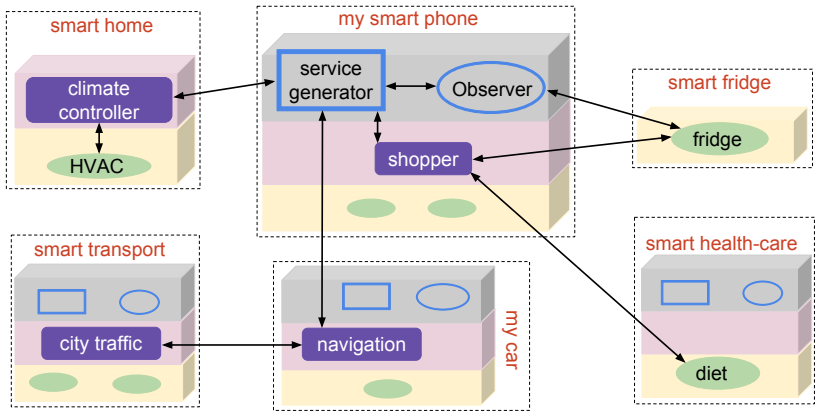


Figure 2.7: A use-case showing the unification of various applications based on the distributed IoT architecture.

From the contextual information vector, the *observer* also identifies that the person is driving currently. So the *service generator* informs the *navigation* CVO, residing in the car's IoT daemon, to fetch the route information to the nearest (or on the way to home) store. Then the *navigation CVO* gathers the information about the shortest route and an empty parking place from the *city traffic* CVO of the "smart transportation system". The *city traffic* CVO collects these information after consulting with the appropriate traffic sensors deployed across the city (through their corresponding VOs). Finally, the store is selected based on traffic condition, parking information and distance. Additionally, the *service generator* estimates the time to reach home. It conveys this information to the "smart home" climate control application. The *climate controller* CVO then engages the *heater* and the *ventilation* VOs to make the indoor climate as comfortable as possible according to the preference of the owner. The BDIP model guides the whole process starting from the service request initiation. It makes decision about the roles of each VOs and models the CVOs accordingly if certain changes occur during the execution of the service.

From the use-case as shown in Fig. 2.7, it is clear that the VOs, CVOs and Service logics are not hosted centrally. Rather, they are hosted locally by the local IoT daemon based on direct association. This makes the architecture scalable. Moreover, use-case shows a local CVO can combine data from the local VOs ("climate controller" CVO in smart home) or a remote CVO can also combine the data collected from VOs hosted by different IoT daemon ("shopper" CVO in my smart phone). This depicts that the proposed architecture supports distributed operation amongst various IoT entities.

Conclusions

The foundations for a generic IoT architecture have been discussed in this chapter. Based on these foundations, we have proposed a distributed layered architecture for IoT, called DIAT, which is a simple, yet scalable reference architecture. It accommodates heterogeneous objects and provides support for seamless interoperability among the objects. Various desired features such as automation, intelligence, dynamism, and zero-configuration are integral part of the architectural model. Several cognitive functions such as dynamic service creation, modeling and execution are incorporated in the proposed architecture. Using a realistic use-case, we have shown that DIAT meets the goal of IoT as a global infrastructure substrate. The basic concepts of the architectural model were materialized in an EU project, called iCore. In Chapter 3, we utilizes the concepts of DIAT by using the virtual object (VO) layer only. In Chapter 4, we exhibit a use-case of DIAT through an IoT application scenario.

3

Virtualizing Wireless Sensor Networks

This chapter describes the *Virtual Sensing Framework* (VSF), which virtualizes a WSN deployment (small/large scale) as part of IoT virtualization. Though DIAT provides a multi-layered reference architecture for IoT, VSF constitutes only the lower-most layer of DIAT, i.e., the virtual object (VO) layer. It focuses more on operational optimization of a WSN through virtualization.

The goal is to reduce sensing and data transmission activities of the nodes in a WSN without compromising on either the sensing interval or data quality. VSF creates virtual sensors at the sink to exploit the temporal and spatial correlations amongst the nodes based on sensed data. Using an adaptive model at every sensing iteration, the virtual sensors can predict *multiple consecutive* sensor readings for all the nodes with the help of sensed data from a few active nodes. We show that even when the sensed data represents different physical parameters (e.g., temperature and humidity), our proposed technique still works making it independent of the physical parameter sensed. Our technique can substantially reduce data communication among the nodes leading to reduced energy consumption per node yet maintaining high accuracy of the sensed data. In particular, using VSF on the temperature data from the IntelLab and the GreenOrb datasets, we have reduced the total data traffic within the network up to 98% and 79%, respectively. The corresponding average root mean squared error of the predicted data per node is as low as 0.36°C and 0.71°C, respectively.

Parts of this chapter have been published as – Sarkar *et al.*,"VSF: An Energy-Efficient Sensing Framework using Virtual Sensors", IEEE Sensors Journal, 2016 [101].

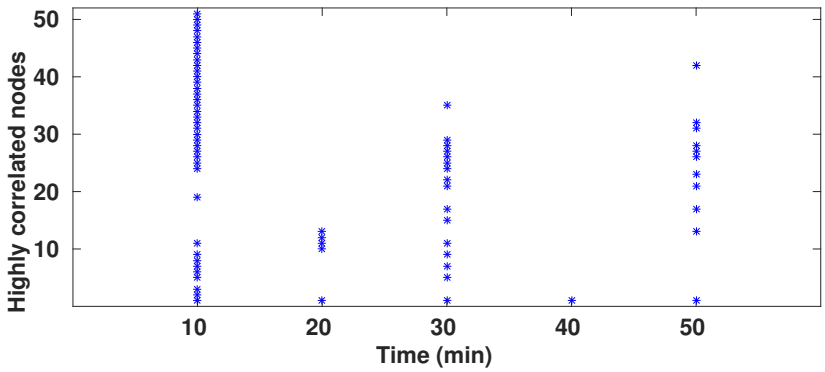


Figure 3.1: Nodes that are highly correlated with Node 1 at different period are marked in the vertical lines.

Introduction

Energy is one of the most precious resource a wireless node. Since the radio transceiver is the most energy hungry component in a node, researchers have either tried to keep the radio transceiver in low-power state or reduce its usage as much as possible. Thus, a number of MAC protocols [30, 78] have been developed to enforce duty cycling on the nodes. On the other hand, a number of efforts have also tried to reduce the overall data traffic within the network, e.g., clustering algorithms and in-network data aggregation [74, 107], coverage problems [114, 133], data compression [67, 125], etc. This work falls in the latter category, where energy efficiency is achieved by reducing the activity of the nodes.

Motivation

A popular technique to reduce overall data traffic is to utilize the over-provisioning of nodes, i.e., keep only a subset of nodes active at any point of time [119]. In such cases, data from the inactive nodes are either assumed to be the same as that of one of the active node or can be reproduced using the data from the active nodes. This reproduction is possible due to the fact that there is inherent spatial and temporal correlation amongst the sensors, and if two sensor nodes show very high correlation, the data from one node can be predicted accurately with the help of the other. In a wireless sensor networks (WSN), usually the data from a node is highly correlated with data from many other nodes. To maximize the energy savings in the network, as many nodes as possible need to be kept in low-power sleep mode (dormant) while their data can be predicted accurately with the help of a few active nodes. We term this as "maximum sleeping node policy".

Many existing correlation-based data gathering techniques assume *a priori* correlation among the sensor nodes [44, 46]. However, data correlation among the sensor

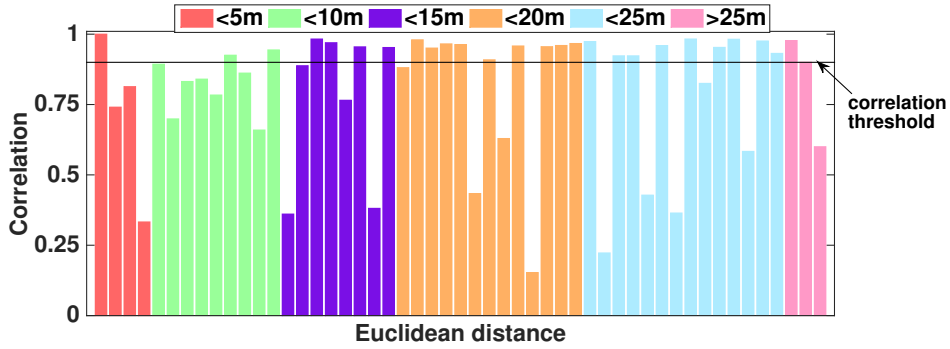


Figure 3.2: Correlation among the nodes cannot be described using Euclidean distance.

nodes often shows a dynamic behavior, i.e., the correlation among the nodes changes over time. Based on the temperature data of the IntelLab deployment [71], we have calculated the correlation among the nodes. Fig. 3.1 shows the nodes (asterisks on the vertical line) that are highly correlated with Node 1 at different time. From this figure it is clear that the correlation among the nodes varies significantly with time.

Another common assumption is that if two sensor nodes are geographically collocated, they produce highly correlated data [52, 129]. However, in many real-life deployments, two sensor nodes that are far apart can also show high data correlation; whereas, two close-by sensors may show poor data correlation. To demonstrate our claim, we show the correlation and geographical distance between Node 1 with all other nodes in Fig. 3.2. According to general assumption, the correlation among nodes decline with increase in distance. However, the figure shows that this claim is not always valid. Based on these observations, we claim that any correlation-based data prediction framework should be dynamic and adaptive.

Contribution

We propose *Virtual Sensing Framework* (VSF), an efficient data collection framework for WSN that reduces activities of the nodes to reduce overall energy consumption of the network, and maintains sensing requirements of the deployment. It considers changes in underlying correlation structure while exploiting the data correlation among the nodes. The following are our main contributions.

Virtualization of sensed data

To reduce energy consumption, VSF adopts the policy of keeping as many nodes as possible in low-power sleep mode and only a few nodes as active. It also helps avoiding data transmission for the active nodes whenever possible. The fewer measurement data is complemented by accurate prediction of the sensed data exploiting

correlations. The data prediction is done by the virtual sensors in successive sensing intervals with minimal involvement of the physical nodes.

Adaptive correlation exploitation
VSF provides an adaptive prediction scheme without considering any *a priori* correlation structure among the nodes. We consider a generic situation where geographical collocation of nodes may or may not imply higher correlation among the nodes. To the best of our knowledge, this is the first piece of work that considers this generalist view. Using an information theoretic approach, we also show that VSF can retain high data accuracy whenever there is high correlation among the nodes. Based on real-world datasets, we further show that VSF can help nodes to expend less energy while keeping high accuracy of predicted data.

Active node selection
We formulate a sensor selection problem for collection of sensor data that are correlated where maximal sleeping and minimum energy consumption policy is adopted. We propose a heuristic algorithm for the selection of active nodes. The algorithm selects a new set of active nodes after a few sensing intervals to attain uniformity in energy expenditure amongst the nodes over a longer period. Using VSF on real-world datasets, we show significant amount of energy savings while maintaining the required data accuracy. This eventually leads to lifetime improvement of the deployed WSN.

The rest of the chapter is organized as follows. First, we discuss some the existing energy-efficient data collection techniques for WSN in Section 3.2. Then, we discuss the virtual sensing framework and activity reduction technique in Section 3.3. In Section 3.4, we discuss the problem of active node selection and propose a heuristic algorithm. In Section 3.5, we provide a thorough evaluation of our system. In the first part of our evaluation, we show how the estimation mechanism performs and how to set various parameters of the system. In the latter part we compare our system with some of the existing approaches. Finally, we conclude this chapter in Section 3.7.

Related work

As data transmission is the major cause of energy consumption in a WSN, a large number of prevalent work exists in the literature on energy efficient data gathering techniques. Reducing the overall of traffic in the network is a common approach towards that direction. Clustering is an effective approach that reduces energy consumption by reducing the number of data transmissions within a network [22, 130]. A group of collocated nodes select one amongst them as a cluster-head (CH). Instead of directly reporting the sensed data to the sink node, the nodes deliver the data to the CH. The CH then takes the responsibility of delivering the data to the

Table 3.1: Comparison of VSF with other data collection techniques.

Techniques	Traffic reduction	Node scheduling	Correlation-based
Clustering	Yes	No	Yes/No
In-network data aggregation	Yes	No	Yes/No
Sensor data prediction	Yes	No	Yes
Coverage problem	Yes	Yes	No
Compressed sensing	Yes	No	No

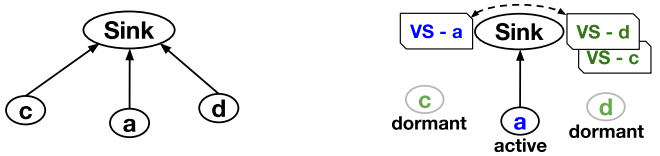
sink. Usually, in a large WSN, multiple cluster heads exist, and they collaborate among themselves to deliver the data to the sink. Thus, the amount of traffic is reduced as well as the energy expenditure for data transmission. However, every node needs to be active to sense and send their data.

Another important class of energy-efficient data collection technique is *in-network* data aggregation to restrict the amount of data to be delivered to the sink [25, 50, 107]. Data aggregation can be performed either in every node or in special nodes like the CH. The idea is to combine data from multiple sources and reduce data traffic within the network. Though in-network aggregation combined with clustering can significantly reduce the overall energy consumption of the network, nodes are still required to sense and send their data..

Apart from aggregating multiple data points in a single data packet, another common technique for data traffic reduction is leveraging correlation based estimation techniques. A data estimation model can be formulated if the estimated data do not deviate much from the sensed data and it is sufficiently accurate. Thus unnecessary transmissions can be avoided. Data is transmitted only when a large deviation is detected in the sensed data [42, 49, 95, 116]. Even though energy is saved significantly by avoiding data transmission, whenever possible, all nodes need to keep sensing continuously.

Many of the existing works do not find the correlation among nodes explicitly. Rather they assume a static correlation structure among the nodes [23, 24, 44, 88], and exploit the correlation to reduce energy consumption significantly. However, correlation among the nodes do not remain the constant. Thus, a mechanism is required to capture node correlation dynamically, and the data traffic reduction schemes should accommodate this dynamics.

Periodic on/off scheduling among the nodes in a WSN is an effective tool to save energy consumption, and some important contributions from the literature are briefly described here. The idea behind on/off scheduling is when more than one sensor node monitor a common sub-area or a target, data from only one node is sufficient. As multiple nodes produce same/similar data, only one node from this group actively sense/monitor the field and rest of the nodes are kept in sleep



(a) 3 sensor star network (b) virtual sensor

Figure 3.3: Data collection scenario in a WSN, and selective transmission using virtual sensing framework.

mode [119, 133]. The general assumption is that nodes within close proximity produce correlated data and the correlation pattern among all the nodes in the network is assumed to be known (or can be modelled easily).

Recently, compressed sensing based data collection mechanisms have received a lot of attention from the research community [67, 70, 125, 126]. The basic idea behind compressed sensing is that if a large dataset has high sparsity, it can be compressed. As a result, data transmission becomes less expensive. More sparsity means smaller data size. Based on this principle, a few samples of sensed data (from few sensor nodes) are collected at the sink. Later, whole data set for the entire network is reconstructed from these samples. Though this technique does not assume any correlation among the nodes, the basic assumption is that the data is sparse in some domain, e.g., frequency domain.

The above mentioned works are summarized in Table 3.1. In contrast, VSF exploits the inherent data correlation among the sensor nodes without assuming any *a priori* correlation amongst the nodes including the physical parameters these sensors measure. VSF can predict data for a large number of consecutive sensing instances with sufficiently high accuracy. Furthermore, VSF also tracks changes in correlation among the sensor nodes over time. Hence VSF can adapt to the changes in the environment and can work in a wide range of deployment.

Virtual sensing framework (VSF)

A sensor node consumes energy for each of its activity, e.g., sensing, processing, data transmission. VSF aims to reduce energy consumption by reducing frequency of sensing, processing and data transmissions. In a data collection WSN, each sensor node sense data in a predefined way and report the sensed value to the sink. If the sensing activity of the sensor nodes is reduced to lower the energy consumption, the purpose of the deployment cannot be fulfilled. VSF supplements the loss of sensing at the nodes by predicting their values. As a result, the energy consumption of the nodes is reduced while the application requirement is met.

VSF: Activity reduction scheme

Virtual sensors (VS) are the basic building blocks of VSF. A VS contains a prediction model for the associated physical sensor (PS). As virtual sensors are at the sink, the reconstruction of the sensed data does not affect the deployment goal (see Fig. 3.3). That is any application that uses the sensed data (for monitoring and/or decision making) will always receive the sensed data – actual measurement or predicted – from all the virtual sensors. The concept of virtual sensor was first presented in our earlier work [98].

As sensor values are usually correlated with their immediate past values, they can be predicted by exploiting its temporally correlated data. In order to increase the energy savings, a VS should predict successive values while its corresponding PS remains dormant. Therefore, changes in the physical parameter during long dormant periods might not be captured by the temporal correlation based method. In this case, prediction accuracy can be improved by exploiting cross correlations among the nodes. If two nodes have had very high correlation in the recent past, it is safe to assume that both the nodes will behave in a similar fashion for some time in the future too. Hence in VSF, we choose to exploit autocorrelation as well as cross correlations to fine-tune the prediction. The value of r ranges between [-1,+1], where +1 signifies exactly the same pattern (or highest correlation), 0 signifies absolutely no correlation, and -1 signifies exactly the opposite pattern. To define high correlation between two nodes, we define **correlated node pair** as

$$corr(s^i, s^j) = \begin{cases} 1, & \text{if} \quad |r(\underline{u}^i, \underline{u}^j)| \geq th_{corr}; \\ 0, & \text{otherwise}, \end{cases} \tag{3.1}$$

where node s^i and s^j form a highly correlated node pair (i, j) (or highly correlated nodes), if the above condition is satisfied; $\underline{u}^i$ and $\underline{u}^j$ are the sensed data vectors for nodes i and j, respectively. th_{corr} is a predefined correlation threshold that is used to decide sufficiently high correlation as any two vectors will always have some correlation – high or low.

In a correlated node pair, one node can remain dormant for the duration of prediction and the other one can remain active to help predict the sensing data of dormant node. This active node is referred to as *active companion*. A node can be highly correlated with more than one node, e.g., node s^i is highly correlated with m different nodes. This implies that node s^i is part of m different *correlated node pairs*. This subset (C^i) of m nodes are termed as **correlated companions**, and it can be defined as

$$s^k \in C^i \iff corr(s^i, s^k) = 1. \tag{3.2}$$

Every node in the network has a set of correlated companions. If a node becomes active, it can act as active companion for all of its correlated companions. Thus only a few nodes need to be active within a WSN, and most of the (remaining) nodes can be kept dormant. The correlated companions of a node change with time

as the data correlation change. Therefore, the companion of a dormant node is neither predefined nor fixated i.e., it can also change over time based on changing correlation between the sensors.

As mentioned earlier, VS uses a mix of autocorrelation and cross correlation based predictor, i.e., it utilizes the past values of itself and the currently sensed data from its active companion. This may drain energy of active nodes significantly. To circumvent this problem, VSF also conserves energy in the active nodes, whenever possible. Here, the node continues to sense the physical parameters, and also predicts the value using an autocorrelation based predictor. If the prediction error lies within a tolerable error bound, then the node does not transmit the sensed data. The data is transmitted only when the predicted data differs largely with that of sensed data. As energy spent by a node's CPU for making the selective transmission decision is less than the energy cost of a data transmission, by withholding data transmissions, significant energy is saved even in the active sensors. When an active node does not transmit the sensed data, then the associated virtual sensor predicts the data (using autocorrelation only). It is clear that the functionality of the virtual sensors associated with a dormant and an active node are different. Thus, two different types of VSs are used. We use the terms *Dormant Virtual Sensor* (DVS) when the corresponding PS is in dormant state, and *Active Virtual Sensor* (AVS) when the corresponding PS is active.

VSF: Adaptive node correlation

It is clear that a dormant node can conserve more energy than an active node. To ensure equal drain of batteries, state of nodes switch between dormant and active modes after a certain number of time-slots. As we do not assume *a priori* knowledge about the sensor data statistics, VSF needs to capture the correlation among sensors. It should also monitor the change in the correlation and adapt dynamically. To accomplish this, the whole data collection period is divided into three phases – training period, operational period and revalidation period as shown in Fig 3.4.

The sensing activity starts with training period. During this period, all the PSs collect data and transmit their data to the sink. At this point all the PSs are associated with their respective AVS. Using these training data, correlation among the sensor nodes is computed. At the end of this period, VSF marks the nodes to be either active or dormant based on their correlation. It also assigns an active companion for each of the dormant nodes. Then it creates prediction models for the dormant nodes (DVS). As mentioned earlier, a DVS uses a hybrid model of autocorrelation and cross correlation for its prediction. On the other hand, an AVS uses only an autocorrelation function. We have explained the prediction mechanism in Section 3.3.3.

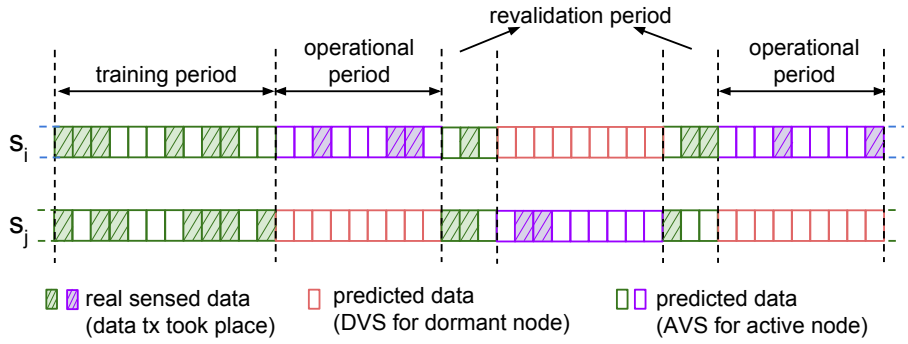


Figure 3.4: Data collection phases in VSF using activity reduction scheme.

The training period is followed by an operational period, where only active nodes sense. A dormant node remains in sleep mode during this period and its values are predicted by a DVS using the values from an active companion. On the other hand, an active node performs only selective data transmissions. It discards the sensed data if it identifies that the predicted data by the AVS is within a tolerable error bound. The active PS is able to calculate the prediction error, because at the beginning of an operational period, its corresponding AVS shares the prediction model with it. When the prediction error crosses certain error threshold, the active node transmits the data and updates the model parameter. When an AVS receives the sensed data, it also updates the model parameter. In this way, the model parameters remain synchronized at both the places. Method of model update is discussed in Section 3.3.4. The selective transmission process of an active node runs in every active node. On the other hand, the activity reduction scheme of VSF runs at the sink node (Section 3.4). Please note, we assume that the sink node has sufficient amount of memory and energy.

The revalidation period resumes all PSs to active mode and all DVSs are switched to AVSs. A revalidation period is shorter compared to the training period. The prediction models for the dormant nodes (in the last operational period) are validated. At the end of this period, the correlation pattern among the sensor nodes is updated, a new set of active and dormant nodes are selected and an active companion is assigned to each of the dormant node. Then, the operational period resumes. If a significant change in the correlation pattern is identified, instead of resuming an operational period, another training period may be started.

Prediction models for virtual sensors

As mentioned earlier, a dormant virtual sensor exploits autocorrelation as well as cross-correlation to predict the data. Fig. 3.5 shows the hybrid filter used for a DVS, which is a combination of an autoregression function (transversal filter) and

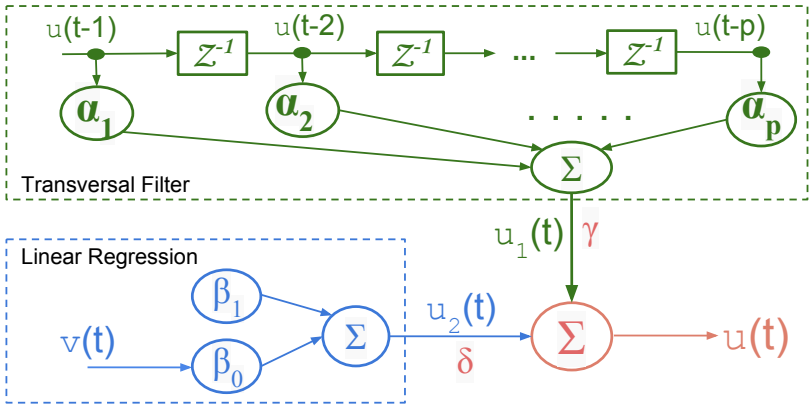


Figure 3.5: The filter used for a dormant virtual sensor is a hybrid of a transversal filter (based on autocorrelation) and linear regression (based on cross correlation).

linear regression function. On the other hand, an active virtual sensor exploits only autocorrelation of the sensed data; thus, only the autoregression function is used for it. An autocorrelation based transversal filter can be described as follows:

$$u(t) = \sum_{i=1}^{p} u(t-i)\alpha(i), \tag{3.3}$$

where $u(t)$ is the predicted time-series value at time instance t; $[u(t-1), u(t-2), ..., u(t-p)]$ are the previous p values of time-series; $[\alpha(1), \alpha(2), ..., \alpha(p)]$ are the filter coefficients; and p is the order of the autocorrelation model. Once the filter coefficients are determined, the current value of the series can be estimated using the past values. To find the filter coefficients, a set of training data is required from the time series. Suppose, there are T_p training data $(u(1), u(2), ..., u(T_p))$ available from the series. Then, the filter coefficients can be found by solving the following linear equations,

$$U\underline{\alpha} = \underline{u}, \tag{3.4}$$

where,

$$U = \begin{bmatrix} u(p) & u(p-1) & \cdots & u(1) \\ u(p+1) & u(p) & \cdots & u(2) \\ \vdots & \vdots & \ddots & \vdots \\ u(T_p-1) & u(T_p-2) & \cdots & u(T_p-p) \end{bmatrix},$$

$$\underline{\alpha} = [\alpha(1), \alpha(2), ..., \alpha(p)]^T,$$
$$\underline{u} = [u(p+1), u(p+2), ..., u(T_p)]^T.$$

This is a overdetermined system of linear equations, as the number of equations $(T_p - p)$ are more than the number of variables (p). As U is not a square matrix, an unique solution is not possible, and an approximate solution can be found using least-square method. The least-square solution of the Eq. 3.4 can be found using the following equation,

$$\underline{\alpha} = (U^T U)^{-1} U^T \underline{u}. \tag{3.5}$$

The linear regression model can be represented using the following equation,

$$u(t) = \beta(0) + v(t)\beta(1), \tag{3.6}$$

where $u(t)$ is the predicted time-series value at time instance t; $v(t)$ is the another time-series (correlated) value at time t; and $[\beta(0), \beta(1)]$ are the coefficients of the linear regression model. Similar to the autoregression model, the coefficients of the linear regression can also be found by solving the following linear equations,

$$V\underline{\beta} = \underline{u},$$

where,

$$V = \begin{bmatrix} 1 & v(p+1) \\ 1 & v(p+2) \\ \vdots & \vdots \\ 1 & v(T_p) \end{bmatrix}, \qquad \text{and} \qquad \underline{\beta} = [\beta(0), \beta(1)]^T.$$

The final prediction for a DVS is achieved by combining the outputs of both the regressors. Based on the accuracy of the models, the final value is calculated as a weighted sum of the predicted values,

$$\hat{u}(t) = \frac{(\gamma \cdot u_1(t) + \delta \cdot u_2(t))}{(\gamma + \delta)}. \tag{3.7}$$

The accuracy of the models are calculated using *Chi-squared statistics*, which is a well-known method to test goodness of fit [113]. To this end, the error values of the estimated signal need to be known. Using the model parameters and the training data set, first, the sensor value is estimated using both the models ($u_1(t)$ and $u_2(t)$ in Fig. 3.5). Then, the Chi-squared statistics can be obtained by taking normalized sum of the squared-errors. Chi-squared statistic is calculated as,

$$\chi_1^2 = \sum_{t=p+1}^{T_p} \frac{(u(t) - u_1(t))^2}{\sigma^2}, \tag{3.8}$$

where σ^2 is the variance of the observed signal. To get an inference from the statistics, a reduced Chi-squared statistic can be calculated by dividing it by the number

of degrees of freedom. The accuracy of the autoregression model (the goodness of fit), represented as γ, is given by,

$$\gamma = 1 - \frac{\chi_1^2}{\nu}, \tag{3.9}$$

where, ν, the degrees of freedom is equivalent to the number of samples $(T_p - p - 1)$. γ lies between $(0, 1)$, where 0 implies complete failure of capturing the system behaviour and 1 implies complete resemblance of the system behaviour by the model parameters. Similarly, the accuracy of the linear regression model (δ) can also be calculated.

Model parameter update

If the correlation structure among the nodes is known *a priori* and remain constant, a *Wiener filter* can be developed, which is said to be the *optimum in the mean-square error sense*. As the correlation is unknown and changes significantly with time, the filter coefficients can become outdated and may result in erroneous prediction. Thus, VSF uses an adaptive filtering technique to update the filter coefficients.

As mentioned earlier, a sensor node transmits the sensed data only when the error in prediction crosses a certain threshold value. To reduce the prediction error in future instances, the model parameters need to be updated. To update the model parameters, we have used least-mean-square based adaptive filtering technique. First, the prediction error is calculated using,

$$e_1(t) = u(t) - u_1(t),$$

where $u(t)$ and $u_1(t)$ are the actual and predicted sensor values, respectively. Then the filter coefficients are updated using,

$$\underline{\alpha} = \underline{\alpha} + \mu \cdot \underline{u} \cdot e_1(t),$$

where $\underline{u} = [u(t-1), u(t-2), ..., u(t-p)]^T$ is the input vector of the filter, and μ is the learning rate of the adaptive algorithm. Procedure to set μ can be found in [95]. Similarly, cross-correlation among the nodes is updated after each training and revalidation period.

Heterogeneous virtual sensor (HVS)

As discussed previously, VSF can accurately predict the sensed data for a dormant (or semi-dormant) sensor with the help of an active sensor. Here, the inherent assumption is that both the sensor nodes sense the same physical parameter. We extend this work to heterogeneous sensing, i.e., predicting the sensed data even if the two sensor nodes sense two different physical parameters, e.g., a temperature sensor data can be used to predict humidity sensor data, and vice-versa, a light

sensor data can be used to predict temperature sensor data, and vice-versa. The mechanism for a heterogeneous virtual sensor (HVS) is just like a DVS. HVS works because VSF considers only data correlation among two sensor nodes. However, for a HVS, some contextual information is required. For example, the error thresholds might be different for two different physical parameters. Similarly, heterogeneous virtual sensing can be applied for a particular time frame, e.g., light and temperature data show correlation only during the daytime.

Active node selection

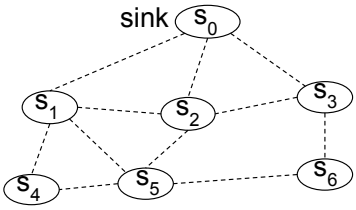
As described in Section 3.3, if a node has multiple correlated companions, it can become an active companion for all these nodes, i.e., sensed data only from this active node is sufficient to predict data for all other nodes that are highly correlated with it. However this raises a few questions such as, (i) whether a node should be active or dormant? (ii) how many nodes should be active? To answer these questions, in this section, we describe how nodes are assigned to either of these sets - active and dormant. The process is divided into two steps: (i) finding correlation structure of nodes; and (ii) assigning roles for nodes.

Finding correlation structure of nodes

Based on the sensed data, the correlation amongst the nodes can be calculated at the end of the training period. A threshold is set to define highly correlated nodes. As mentioned earlier (Section 3.3.1), the value of correlation coefficient is 1 when two time series shows identical trend over time. Thus, correlation coefficient close to 1 signifies a similar trend between two time series if not exactly identical. As a result, sufficiently high correlation based on high correlation threshold (close to 1) between nodes means their sensed data follow a similar pattern and the prediction is expected to be highly accurate. As mentioned earlier, every node in a WSN has its *correlated companions*. As an illustration, Fig. 3.6 shows a small WSN with six sensing nodes. The idea is to find the correlation between all pairs of nodes and listing those which are above a certain threshold as shown in Fig. 3.6. From this table, we can find the corresponding correlated companions.

Assigning roles to nodes

If a node is selected as an active node, all its correlated companions can be kept in dormant state. The goal is to select a minimal number of active nodes (maximal sleeping node policy) such that the union of the correlated companions of all the active nodes contains all the nodes in a WSN. For simplicity, we consider that all the nodes consists homogeneous hardware so that for any node, energy consumption in active mode is equivalent. As a result, a minimum number of active nodes can ensure minimum overall energy consumption by the WSN.



Node	Correlated companions
s_1	s_1, s_2, s_3, s_6
s_2	s_1, s_2, s_3
s_3	s_1, s_2, s_3, s_5
s_4	s_4, s_6
s_5	s_3, s_5, s_6
s_6	s_1, s_2, s_3, s_6

Figure 3.6: A small WSN with six sensing nodes and a sink node; a table showing the correlated companions of each node.

The problem is that if a random node is selected as active, it may not be able to communicate directly or through multihop path with the sink. Therefore, it is required to select a node as active, only if, it can send its sensed data directly or through another active node. The best selection is to find a set of minimum number of active nodes while satisfying the following conditions: (a) they form a connected graph, specifically, all of them can reach the sink node; (b) every dormant node has at least one active companion so that its sensed data can be predicted with a tolerable error bound. Once a node is selected as active, it consumes more energy than a dormant node. Naturally, it should not operate as an active node for a long time; otherwise it will drain out all of its energy quickly. To ensure fairness in energy consumption, a new set of active (and dormant) nodes are selected before every operational period.

Active node selection: combinatorial optimization

We represent the network as a node- and link-weighted, undirected graph $G = (S, L)$, where S is the set of all nodes in the network and L is the set of links in the network. A link between a nodes i and j exists if they are within the transmission range of each other. The cost of this link, denoted by $e_l(i, j)$ or $e_l(j, i)$, is equal to the energy consumption for transmitting a packet from one node to the other. Each node is also associated with a non-zero cost (node-weight), which is equal to the energy consumption based on its CPU and sensing activity. For a node i, let $e_a(i)$ be the amount of energy consumed in active state and $e_d(i)$ be the amount of energy consumed in dormant state. We now formally define the problem as follows.

Definition 1. *Active node selection: Given a node- and link-weighted undirected graph $G = (S, L)$ with n nodes and a collection of n subsets $\{S_1, ..., S_n\}$ such that S_i is the correlated companions of node i and $\bigcup_{i=1:n} S_i = S$, find the minimum cost subgraph $G' = (A, L')$, where $A \subseteq S$ and $L' \subseteq L$. Here cost of G' is defined as, $\sum_{i \in A} e_a(i) + \sum_{(i,j) \in L'} e_l(i, j)$. The minimum cost subgraph, G', has to satisfy the following conditions – (i) the union of the correlated companions of the nodes in A is S, i.e., $\bigcup_{k \in A} S_k = S$, and (ii) G' is a connected graph.*

Theorem 1. *Active node selection (decision) problem is NP-complete. The optimization version of the problem is NP-hard.*

Proof. It is easy to show that active node selection $\in$ NP, since a nondeterministic algorithm only needs to find a subgraph A and then verify in polynomial time that (i) the union of the correlated companions of the nodes in A is equal to S (complexity is O(n)), and (ii) the nodes in A are connected (complexity of depth first search (DFS) is O(n)). Therefore, active node selection (decision) problem is NP. The problem can be reduced to the set-cover problem. Without the additional constraint that the minimal subset of nodes should form a connected subgraph, the problem is same as the set-cover problem. This means the problem is as hard as the set cover problem, if not harder. As the set-cover problem is known to be NP-hard [53], the Active node selection problem is also NP-hard. Because of this direct equivalence with the Set cover problem, we have omitted detailed proof. □

A heuristic algorithm for active node selection

As explained in Section 3.4.1, the Problem 1 on hand is NP-hard. An optimal solution is not guaranteed to be found in polynomial time. To evolve a polynomial time solution for the active node selection problem, we propose a heuristic, called *active node selection heuristic* (ANSH). Apart from the connectivity constraint, an incautious active node selection may render some of the nodes to be active more often than the rest. Consequently this leads to run down of energy of these nodes faster than the rest. This might cause a disconnected network and reduces the lifetime of the network. Therefore, a careful active node selection should not only consider minimal energy consumption for a WSN, but also balance the energy expenditure across the nodes.

We separate the active node selection from the route finding process. To find the shortest path from each node to the sink node, we depend on the underlying routing protocol. The shortest path tree helps to derive the parent-child relationships for the nodes in the network. Now, if we ensure that the parent node of every active node is also selected as active, the connectivity can be ensured. Note that the sink node is always selected to be active.

The active node selection process, described in Algorithm 1, marks each node for either of the roles - active ($state = 1$) or dormant ($state = -1$). Initially, all the nodes are marked as undecided ($state = 0$). Then, the algorithm selects a node as active that has maximum residual energy among all the nodes. This selected node is then marked as active. To ensure connectivity, its parent node (also the parent's parent) needs to be marked as active. A recursive function is called to mark a node and its parent node as active (*markAsActive*). When a node is marked active, all its correlated companions can be marked dormant. However, some of these correlated companions might be marked active previously. Thus, only the undecided correlated companions of an active node are marked as dormant. After

Algorithm 1: *ANSH:* Algorithm for active node selection.

Input: Residual energy $(e^1_{res}, ..., e^n_{res})$, parent node $(p^1, ..., p^n)$, and the correlated companions $(C^1, ..., C^n)$ of every node.

Output: The nodes are split into two subsets: Active ($state = \alpha$) and Dormant ($state = \delta$).

Algorithm `ansh()`

```
state[1..n] ← φ; cover ← 0;
while (cover < n) do
    e_max ← 0;  i_max ← 0;
    for (i = 1 : n) do
        if (e_max < e^i_res AND state[i] == φ) then
            e_max ← e^i_res;  i_max ← i;
    cover ← markAsActive(i_max);
```

Procedure `markAsActive(i)`

```
if (state[i] == φ) then
    cover ← cover + 1;
state[i] ← α;
if (p^i ≠ sink AND state[p^i] ≠ α) then
    cover ← markAsActive(p^i);
for (k = 1 : n) do
    if (state[k] == φ AND s^k ∈ C^i) then
        state[k] ← δ;
        cover ← cover + 1;
return cover;
```

this, next active node is selected based on the maximum residual energy among nodes that are still undecided. The node selection process continues, until all the nodes are marked either active or dormant. Note that if a node is marked dormant, one of its highly correlated nodes is guaranteed to be marked as active.

To equalize the energy expenditure among the nodes, we mark nodes active based on maximum residual energy. If a node is marked active, its residual energy will be lesser in the next round of selection as compared to the dormant nodes. As a result, a new set of nodes will be marked as active. This ensures fair energy expenditure among the nodes in the network. If multiple nodes have the same residual energy, one random node among them is marked active.

Table 3.2: Deployment summary.

deployment name	nodes	data points/node	sensing interval	deployment type
IntelLab	51	5000	31 s	Office (indoor)
GreenOrb	220	432	10 min	Woods (outdoor)

Table 3.3: Current and energy consumption (with 31 s sensing interval) by a Tmote sky node for its various activity.

(a) Current consumption

Activity	Current
CPU active	1800 μA
CPU sleep	5.1 μA
Radio idle	20 μA
Radio sleep	1 μA
Radio transmit	17.4 mA
Radio receive	19.7 mA

(b) Energy consumption

Activity	Energy
CPU active	9.33 mJ
CPU sleep	0.57 mJ
Packet TX	4.74 mJ
Packet RX	5.23 mJ
Temp. sensing	0.33 mJ

Evaluation

We have evaluated virtual sensing framework based on the temperature and humidity data sets collected from the IntelLab deployment [71] and the GreenOrb deployment [68] along with some computer generated data sets. In this section, we present some of the results based on the temperature dataset obtained from these two real-world deployments. The summary of the deployments are shown in Table 3.2. Note that we consider only those nodes from these deployments that have sufficient data points.

We divide the evaluation into two parts. In the first part, we compare performance of the VSF, and how it is affected by changing the various parameters. Since choice of a proper set of values for the parameters is highly dependent on the data set, we provide an indication on how to set those parameters. In the second part, we compare performance of VSF with respect to some of the existing data collection techniques. Though the results described in this chapter are produced using a Matlab implementation, we have developed a working prototype of VSF using Java.

We used three metrics to evaluate performance of VSF – (i) the number of data packet transmission, (ii) accuracy of prediction, and (iii) energy expenditure by the nodes. We use root mean squared error (RMSE) to measure accuracy of

the prediction. To compute energy expenditure by the nodes during the simulation period, we have considered the Tmote-sky [79] for energy consumption measurement. Various current and energy consumption values are summarized in Table 3.3.

Data preprocessing

The dataset collected by the WSNs are often comprised of a number of fallacies such as erroneous sensed data, missing data points, disordered data stream, etc. As VSF applies estimation technique on a time series data, these fallacies need to be rectified before applying virtual sensing technique. Thus, we performed the following preprocessing on the dataset. First, we removed outliers from the datasets. We studied the datasets to identify the lower and upper bounds of the data points. For example, the temperature data in IntelLab deployment cannot go below 10° C and beyond 50° C. Thus any temperature value outside this range is considered as an outlier. When a data point is detected as outlier, it is replaced with the average of its immediate past and future data. If a data point is an outlier or not available then the current data point is removed from the dataset.

Second, we aligned the data points based on absolute time stamp across the nodes. Every node had a fixed sensing interval with a fixed offset amongst them. We identified the relative offset of sensing time for each node from a global time scale. A data point at a particular time instant is included in the processed data set if data points from all the nodes are available at that instant (after adjusting the offset). If data point for a particular node is missing, we tried to restore it by taking an average of its immediate past and future data points. If the current data point cannot be restored, data points from all the remaining nodes at that time instant are not included in the processed data. This way we ensure that when we calculate correlation of two time series, all the data points are properly aligned.

Performance of VSF

In VSF, nodes either remain active or dormant based on their assigned roles. As mentioned earlier, a node's role (active or dormant) changes over time. Fig. 3.7 shows the predicted data using VSF along with the sensed data for a total 5000 data points for Node 1 of the IntelLab deployment (upper plot). During this period, the node was assigned both the roles for some time. From this figure, it is clear that the predicted data closely follows the actual sensed data. The lower part of the plot shows the prediction error (absolute error values). Most of the time, the prediction error is within the limit of 1°C. There exists some occasional outliers, but the overall VSF error bound is suitable for many WSN applications. Similar results have also been observed for all the remaining nodes of both the deployments.

Heterogeneous virtual sensing

As mentioned earlier, VSF framework is not only applicable when all nodes have homogeneous sensor; it can be applied to any type of sensor data that show high correlation irrespective of data type. But it cannot be applied to any such sensed data that can change abruptly without showing any trend over time, e.g., indoor light (artificial light) can be completely different with a small time interval.

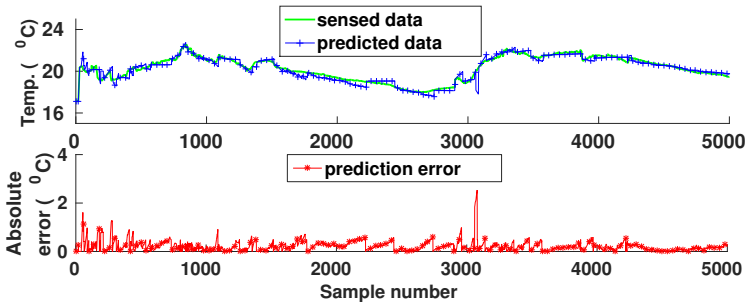


Figure 3.7: Sensed and predicted data of Node 1 over a period, where it was assigned both active and dormant role some time. The length of training, operational, and revalidation periods are set to 40, 20, and 10 data points respectively.

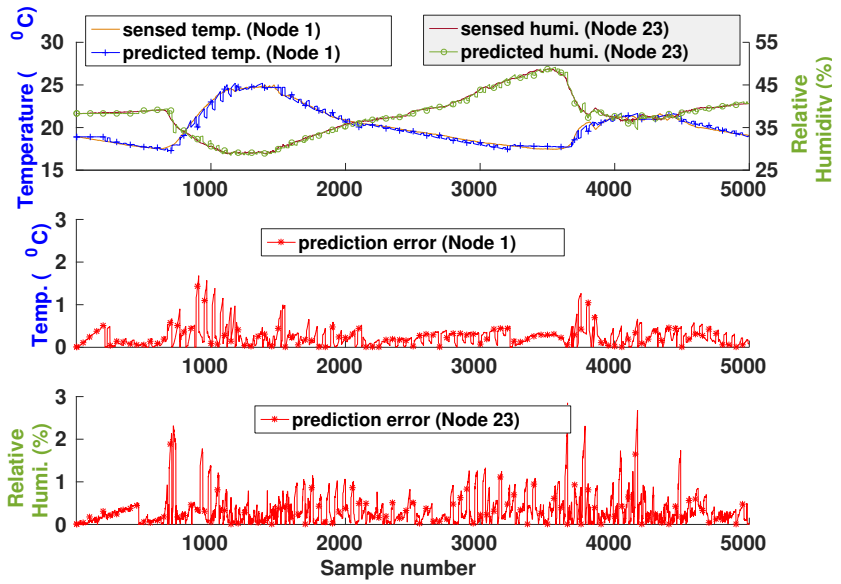


Figure 3.8: Temperature data from Node 1 is used to predict humidity data from Node 23 and vice-versa.

We have applied VSF on two heterogeneous data types. As an illustration, we have used temperature data from Node 1 and humidity data from Node 23 (see Fig. 3.8). We have preselected them, as we know that they have a very high data correlation. During the process, either of the node is selected as active. That means, temperature data is predicted based on the humidity data and vice-versa.

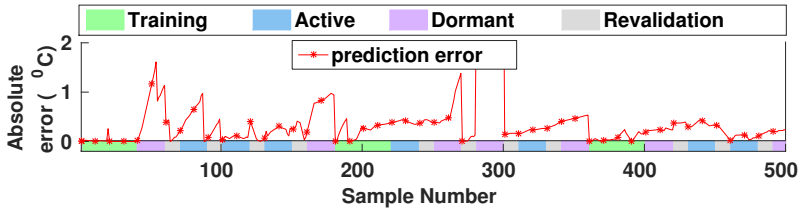
Adaptation to data correlation change

In VSF, all nodes go through the three operating modes, i.e., training, operational (active/dormant) and revalidation. As mentioned earlier, the operating mode of the nodes are decided based on the data correlation among the nodes. A small snapshot (500 data points) of the operating modes for Node 1 is shown in Fig. 3.9b. At the beginning, Node 1 enters the training period like all other nodes in the deployment and they are associated with a low threshold active virtual sensor (AVS). The length of the training period is set to 40 data points. At the end of this period, the operational period starts (20 data points). During this operational period, Node 1 is assigned dormant. Thus, it is associated with a dormant virtual sensor (DVS).

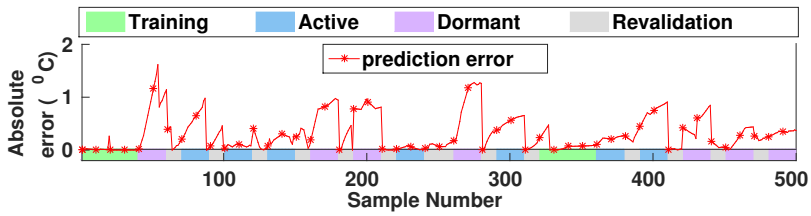
After the operational period, Node 1 enters the revalidation period as all other nodes in the deployment. Similar to the training period, nodes are also associated with an AVS during this period. After the revalidation period, usually another operational period starts with new role for the nodes (a different set of active and dormant nodes). In this case, Node 1 is assigned as active and is associated with an AVS. Presuming an operational period depends on the prevalence of correlation pattern among the nodes as it were during the last training or revalidation period. If a smaller fluctuation is noticed, a revalidation period can proceed to another revalidation period (after data point 250). In case of a huge data correlation deflection, a new training period is started (after data point 320).

The dynamic adaptation of the operating modes lead to a better prediction accuracy and lesser energy consumption by the nodes, as compared to a fixed schedule for the operating modes (Fig. 3.9b). The lengths of the operating modes are same in both the cases, i.e., 40, 20 and 10 data points for the training, operational, and revalidation period, respectively. But, for the fixed schedule case, a revalidation period is always followed by a operational period, and after 5 operational period (and 4 revalidation period in-between) the training period is initiated. In this case the revalidation period is used only for selecting a set of active node for the next operational period.

The dynamic adaptation of the operating modes is a better choice than the fixed schedule as it is clear from Fig. 3.10. Using 5000 sensed data from the 51 nodes of the IntelLab deployment, we found that the prediction error (RMSE) for most of the nodes is higher for fixed schedule case than the dynamic adaptation case (Fig. 3.10a). At the same time, energy consumption by most of the nodes for the fixed schedule case is also higher than the dynamic case (Fig. 3.10b). The energy

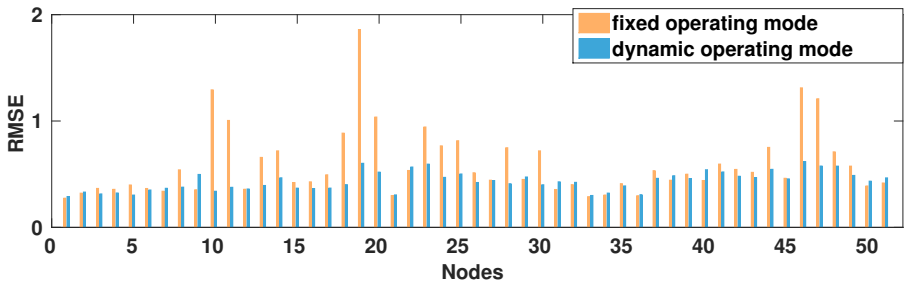


(a) Re-training at a fixed interval (after 5 revalidation period).

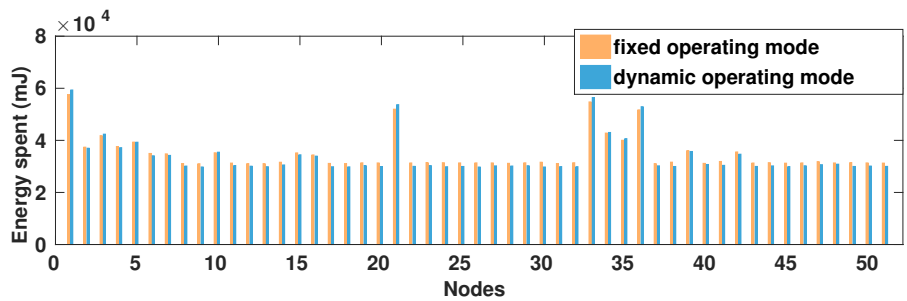


(b) Decision about re-training is taken dynamically based on data correlation.

Figure 3.9: Various operating modes of Node 1 in VSF for the first 500 data points. During the operational period, a node can be either active or dormant.



(a) Prediction error for every node



(b) Energy spent by each node

Figure 3.10: A comparison between fixed and dynamic re-training schedule. Average prediction error for a node is 0.59 °C and 0.43 °C for the fixed and dynamic schedules, respectively. Similarly, average energy consumption is 34.64 J and 33.90 J, respectively.

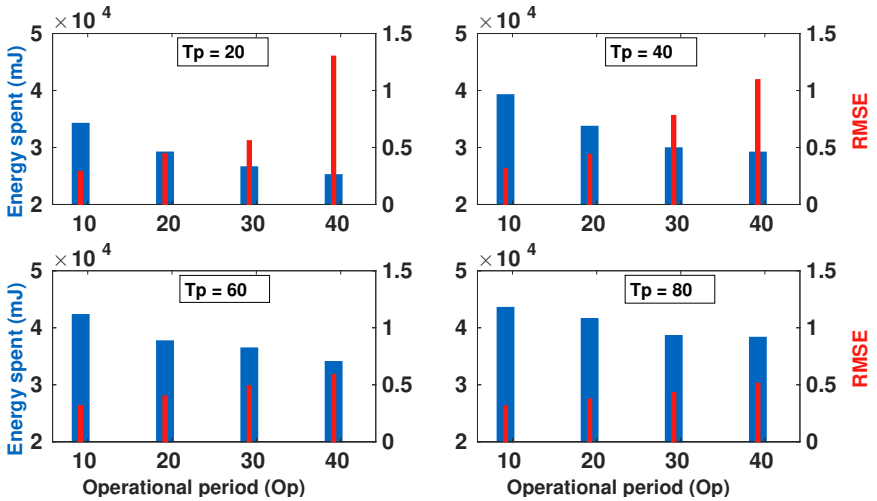


Figure 3.11: Average prediction error and energy consumption per node for various lengths of the operating periods based on the IntelLab deployment data.

consumption for the fixed schedule case can be reduced by increasing the interval between two training periods. But, this would only increase the prediction error. As a result, the dynamic scheduling of the operating modes can tackle the dynamic data correlation pattern among the nodes (over time).

Deciding the lengths of the operating periods

The lengths of the operating periods play a role in prediction error, and number of data transmissions (or energy consumption by the nodes) within the network. To reduce energy consumption, as many nodes as possible are to be kept in dormant mode for the maximal amount of time. Thus, a longer operational period can lead to an overall reduction in energy consumption as most of the nodes are kept dormant. However, longer operational period can lead to higher prediction error for the dormant nodes. These can be complemented by longer revalidation period. But, again, longer revalidation period leads to higher energy consumption by the nodes as all the nodes are accompanied by active virtual sensor (AVS) during this period.

Fig. 3.11 shows the average prediction error (RMSE) and average energy spent by a node for 5000 sensing intervals (data points). Four different lengths (20, 40, 60 and 80 data points) of training periods (T_p) are used. For each length of the training period, four different operational periods (O_p) of 10, 20, 30 and 40 data points are used. The revalidation periods (R_p) is kept fixed to 10 data points. From the figure, it is clear that energy expenditure is lower for the smaller training period and higher operational period, but it lacks prediction accuracy of the sensed values. On the

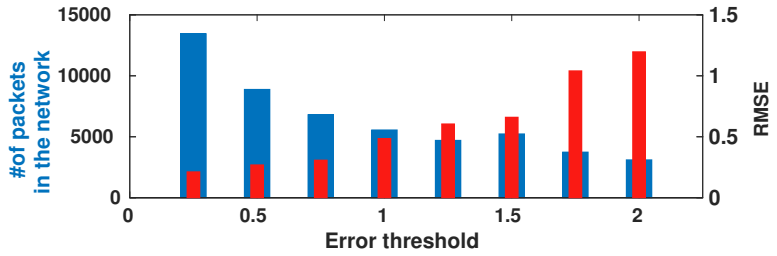


Figure 3.12: Tolerable prediction error has an immediate impact on the accuracy of VSF mechanism, as well the on the energy consumption of the nodes. Setting a tolerance level (threshold) depends on the application requirement.

other hand, prediction accuracy increases with the cost of more energy expenditure, i.e., longer training period and smaller operational period (more data transmissions within the network).

The length of these periods can be dependent on the deployment and their sensing intervals. Though the tendency should be larger operational period and smaller revalidation period, a suitable length of these periods can be selected in such a way that the operational periods are small enough to contain any possible drift of the data correlation among the nodes. To maintain minimal energy expenditure and high data accuracy, for the IntelLab deployment, a suitable $< T_p, O_p, R_p >$ combination is 40, 20, and 10 data points, respectively (Fig. 3.11).

Effect of the error threshold

In an active node, not all the sensed data is transmitted. If the prediction error is going to be within a predefined error threshold (tolerable error), the node avoids transmitting the data. If this threshold is set to a higher value, the significant number of data transmission can be avoided. But, this can decrease the accuracy of the overall data set. Fig. 3.12 shows how the prediction error increases when the error threshold increases. On the other hand, when the error threshold is lower, the data accuracy increases; but this also increases the number of data packets within the network, as well as energy consumption of the nodes. So, setting up this threshold is highly dependent on the application requirement.

Effectiveness of correlation based node grouping rather than collocation

We have argued for a system that node correlation is calculated based on their sensed data only, irrespective of their geographical collocation. This strategy provides a better accuracy for data estimation.

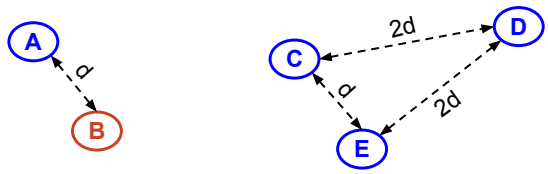


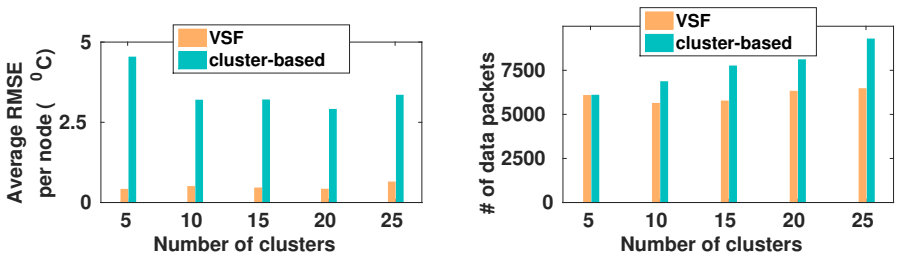
Figure 3.13: Node correlation based on data and geographical collocation.

Proposition 1. *Correlation based on sensed data is superior than the geographical collocation based, when data estimation is done exploiting the correlation.*

Proof. Let us suppose that the data correlation amongst the nodes is fixed and known a priori. To maintain high accuracy of the predicted data using VSF, member nodes in a particular group need to be highly correlated, i.e., their pairwise correlation is above certain correlation threshold (described as correlated companions in Section 3.3.1). Now, the nodes grouped together based on their geographical collocation. As shown in Fig. 3.2, nodes in close proximity need not always show high data correlation. If two nodes are assumed to be highly correlated just because they are geographically collocated, but not in reality, the prediction accuracy will be less.

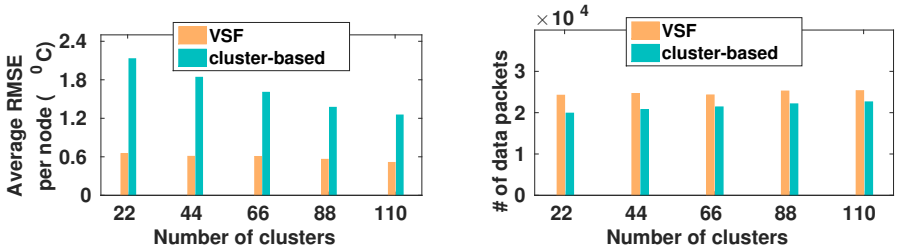
Let us take an example. In a WSN, consider nodes A and B are located very closed to each other, but have very poor data correlation, and nodes C and D have very high data correlation even though they are positioned apart from each other (Fig. 3.13). If geographical collocation based correlation is assumed, node A can act as an active companion of node B. But, the resulting data prediction would have lower accuracy. On the other hand, nodes C and D cannot be each other's active companion. Now, VSF would not assume A as an active companion of B, thus lower prediction accuracy can be avoided. Also, VSF can assign C as an active companion of D, thus, significant energy savings can be achieved by keeping node D in dormant state, while its data can be predicted with high accuracy. In case, two close by nodes are highly correlated (nodes C and E), both the methods will achieve higher prediction accuracy as well as save some energy. Thus, we can conclude that correlation based on sensed data is superior than the geographical collocation based correlation when data estimation is done based on the correlation. □

To further prove our claim, we applied experimental comparison based on the IntelLab and GreenOrb datasets. Node locations for both the deployments are known. We create cluster of nodes based on their geographical collocation, where nodes within a cluster are assumed to be highly correlated. The clusters are formed based on their Euclidean distance with each other. Instead of considering a fixed number of clusters within the network, we use different number of clusters. Thus, the number of nodes per cluster (cluster size) is varied. We use five different cases



(a) average RMSE per node (b) number of packet transmitted

Figure 3.14: IntelLab dataset: Comparison of VSF with geographical collocation-based clustering.

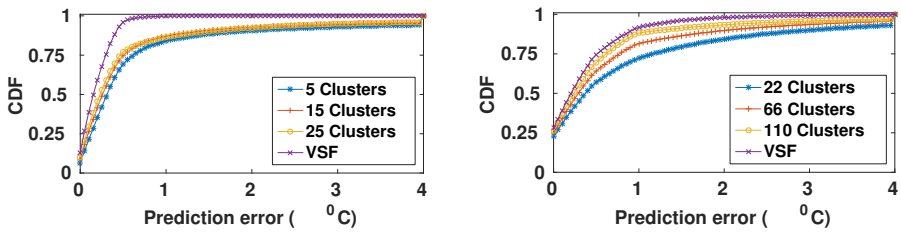


(a) average RMSE per node (b) number of packet transmitted

Figure 3.15: GreenOrb dataset: Comparison of VSF with geographical collocation-based clustering.

where the cluster sizes are equal to 10%, 20%, 30%, 40%, and 50% of the total nodes in the network. Thus, the number of clusters in IntelLab deployment are 5, 10, 15, 20, and 25, respectively. Similarly, in GreenOrb deployment, the number of clusters are 22, 44, 66, 88, and 110, respectively.

From Fig. 3.14a and Fig. 3.15a, we can conclude that the prediction accuracy for VSF is much higher than the geographical collocation based clustering method, as the average RMSE per node is much lower in VSF than the cluster-based method. The cluster-based method seems to be equivalent (or superior) as compared to VSF in terms of the number of data packet within the network (Fig. 3.14b and Fig. 3.15b). The reason behind this is that cluster-based method blindly selects an active node from a cluster and keeps the remaining node in dormant state. So, the number of active nodes are fixed and there are only n active nodes if there are n clusters. On the other hand, VSF assigns a node dormant if any of its correlated companions is assigned active. As the node correlation changes over time, the number of active nodes can also change over time. As a result, there can be more active nodes during a certain operational period, and thus, more number of data packets within the network. Naturally, this ensures high accuracy of predicted data for the dormant nodes. Please note that VSF also uses Autocorrelation to reduce the number of data transmission. Thus, over a long period the total number of data packets within a



(a) IntelLab dataset (b) GreenOrb dataset

Figure 3.16: Prediction error comparison between VSF and three geographical collocation-based clustering.

network using VSF is equivalent to the geographical collocation-based clustering (if not significantly less).

To further illustrate the data accuracy of VSF prediction mechanism, we show cumulative distribution function of the prediction error in Fig. 3.16a and Fig. 3.16b. From these figures, it is clear that VSF not only achieves lower prediction error on average, but it has a very fewer instances of high prediction error. The prediction error is bound by a lower value, where there are a few outliers, i.e., 95% cases prediction error is within 0.5°C. On the other hand, the geographical collocation based clustering and prediction mechanism has a significantly high number of instances with higher prediction error (error within 0.5°C for less than 75% of cases).

Comparison with LMS-based method

In this section, we compare VSF with the LMS-based sensor data estimation method by Santini *et al.* [95]. The proposed technique utilizes the inherent autocorrelation property of the sensed data by a node. Each node possesses an autocorrelation-based predictor, and makes a decision about transmitting the sensed data if the predictor is not able to predict the sensed value within a tolerable error. Though this method reduces the data transmission significantly while ensuring a high accuracy of the estimated data, it does not utilize the cross-correlation among the nodes. On the other hand, by utilizing cross-correlation among the nodes along with the autocorrelation, VSF achieves a higher reduction of data transmission within the network. The summary of comparison is shown in Table 3.4.

The results establish the fact that VSF can outperform LMS-based method in terms of data traffic reduction, and thus, energy consumption by the nodes. Though the prediction accuracy for the GreenOrb deployment using VSF is a bit lower than the LMS-based method, the prediction error is within a tolerable error bound (1°C). A further inspection of RMSE at the individual node level establishes the fact that the prediction accuracy achieved by VSF is comparable with the LMS-based method.

The improvement of VSF is in terms of energy consumption by the nodes as it was

Table 3.4: Performance comparison between VFS and LMS-based method [95].

	IntelLab			**GreenOrb**		
Method	data packets	transmission reduction	error (°C)	data packets	transmission reduction	error (°C)
LMS-based	75408	71%	0.53	29673	69%	0.47
VSF	4947	98%	0.36	19547	80%	0.71

already evident from the data transmission reduction (Table 3.4). VSF conserves energy by not only reducing the data transmission, but it achieves higher energy savings by keeping the nodes in deep sleep for higher amount of time.

Comparison with compressed sensing techniques

Compressed sensing based data collection techniques utilize the sparsity in the sensed data to reduce the amount of data transmission within the network. As opposed to our method, they do not consider correlation among the nodes. However, compressed sensing based methods also adopt a selective data transmission by the node, thus, they resemblance similarity with VSF.

Liu *et al.* proposed a compressed sensing based sensor data collection method, called CDC [67]. In this method, each node transmits its sensed data with a predefined probability p. As a result, the sink node would receive np packets (on an average) in each sensing rounds. Finally, the whole data set for all the nodes is reconstructed based on the partial data set using compressed sensing. The method works based on the assumption that the sensed data have a sparse representation when the data collected from all the nodes are converted to some other domain (e.g. frequency domain). If the data sparsity is very high, highly accurate reconstruction can be achieved with fewer samples. In that case the transmission probability can be set to a lower value. On the other hand, if the data sparsity is less, data reconstruction from a few samples can lead to larger inaccuracy. Naturally, a larger transmission probability is required so that sufficiently large samples can be collected and highly accurate reconstruction is achieved.

We compared the performance of VSF with CDC for various values of transmission probability (p). We tried 4 different values of p such that the average number of samples (or active nodes in case of VSF) are 5%, 10%, 15%, and 20% of the total number of nodes in the deployment. From Fig. 3.17 and Fig. 3.18, it is clear that the prediction error for CDC is much higher when there is only a few sample are used for the reconstruction. As the number of samples increases, the prediction accuracy for CDC also improves and goes closer to that of VSF. But, this increases the number of data packet transmission within the network (Fig. 3.17 and Fig. 3.18), which will also increase the energy consumption of the network.

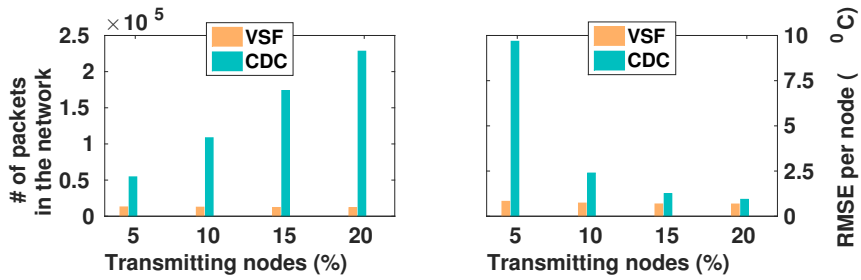


Figure 3.17: IntelLab dataset: Total number of data packet within the network for VSF as compared to CDC.

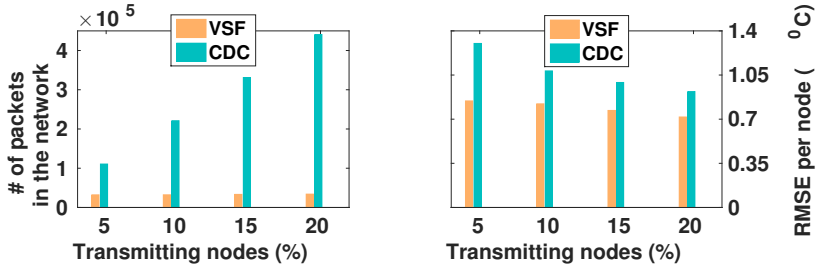


Figure 3.18: GreenOrb dataset: Total number of data packet within the network for VSF as compared to CDC.

Discussion

We briefly discuss some of the trade offs while using VSF. Since we are focusing on sensed data and predicting the data from sensor nodes using correlations, VSF is applicable to predict sensed data of any two nodes that are correlated irrespective of their geographical location. However, this assumption should be restricted for the nodes that are part of the same system/deployment. On the other hand, if two sensor nodes from two different cities show a high correlation for some period, VSF's activity reduction scheme should not be applied as their correlation can show a completely different pattern over a long period.

Even though VSF tries to capture the dynamics of data correlation, the prediction accuracy can drastically be affected when the data correlation is highly dynamic. In that case, a smaller operational period may be more suitable. Note that we have defined the operational period in terms of number of sensing intervals (time difference between two successive readings) rather than absolute time. If the monitored region in a WSN is highly dynamic, the nodes are more likely to sense more frequently. As a result, within a small time period many sensing events will occur and various operating states (training, operational, and revalidation) would also change quickly. Thus, a quick correlation change would have a lower effect on the prediction accuracy.

The heuristic algorithm for selecting the active nodes may result in choosing more than one active companion for a particular dormant node. In such a case selection of a particular active companion over the other may improve the prediction accuracy; but this needs an exhaustive search. Furthermore, utilizing the sensed data from multiple active companions can also improve the prediction accuracy. However, this will increase the complexity of the algorithm. Also, it is not always guaranteed to yield higher prediction accuracy. Thus, we adopted a simplistic approach, where only one active companion is chosen for each dormant node at the beginning of an operational period.

Though the framework is used for a relatively small number of data sources, e.g., 220 nodes in GreenOrb, it can also be used for large data sets. As every node needs to perform a simple estimation task and the estimation model is created based on its own sensed data, the number of nodes does not impose any burden on the nodes. Most of the computational intensive tasks are performed centrally at the sink, which is assumed to be connected with powerful computing devices. Even if the system is composed of multiple smaller networks each having separate sinks, the framework can manage such situations without any scalability issues.

Conclusions

There are many schemes and protocols to increase the lifetime of a WSN. In this chapter, we introduced the virtual sensing framework (VSF), which predicts multiple consecutive sensor data while some of the sensors remain dormant. We have utilized the inherent correlation amongst the sensor data without having: (i) any *a priori* knowledge of the statistics of the data; (ii) location of the sensor nodes and, (iii) type of the physical parameters observed. A case in point is predicting temperature with a light sensor within a tolerable error bound.

The prediction technique of the virtual sensors adapts to the changes in the sensor data. Using the VSF activity reduction technique, we have achieved a significant improvement in energy savings compared to other similar techniques while maintaining sufficiently high accuracy of the sensor data. Our maximal sleeping node policy can reduce the overall energy consumption of a WSN. However, the formulated minimum active node selection problem was shown to be an NP-hard problem. Thus, we provided a heuristic algorithm to find minimum number of active nodes at any instance. We have reported around 98% and 79% of data traffic reduction when the VSF activity reduction scheme is used on the IntelLab and GreenOrb datasets, respectively.

4

A Use-case of DIAT

The virtual sensing framework (VSF) virtualizes a WSN as part of virtualizing IoT. However, VSF does not reflect a virtualized IoT in its entirety. In this chapter we describe **iLTC**, which showcases the benefits of the layered architectural model as summarized by DIAT. Moreover, we demonstrate the feasibility and benefits of employing virtual sensing (building blocks of VSF) in a real scenario.

iLTC is an **i**ndoor **L**ighting and **T**emperature **C**ontroller system. Automatic control of HVAC and artificial lights has been one of the popular methods for achieving energy efficiency in buildings. The current systems use fixed set-point controls, which are decided based on a conservative approach. Additionally, the lighting systems require additional sensor deployment to cope with the continuous intensity fluctuation of natural light. iLTC eliminates the fixed set-points and requirement of additional light sensors. It decides operating set-points more aggressively, which is energy optimal and tries to provide maximal user comfort to all the co-occupants in a shared space. The flexibility of choosing energy-optimal set-points stems from the knowledge of individual temperature and lighting comfort functions. iLTC tries to ensure substantially *minimal* human intervention and only during a brief training phase.

To track the fluctuations in natural light, we employ a virtual-sensing based smart estimation technique that requires light measurements only once during the training phase. As iLTC follows the DIAT design principle, it is scalable to any number of rooms and users. Using the proposed system, we show the energy consumption of HVACs can be reduced up to 39%. Similarly, compared to traditional on/off based and multilevel lighting systems, energy consumption with iLTC can

Parts of this chapter have been published as – Sarkar *et al.*, " iLTC: Achieving Individual Comfort in Shared Spaces", ACM EWSN'16, February 2016 [102].

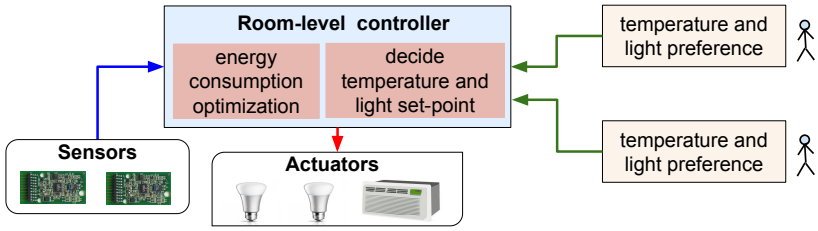


Figure 4.1: In iLTC a room-level controller sets an energy optimal operating point for the lighting and HVAC systems considering the comfort preferences of all the occupants.

be reduced up to 33% and 60%, respectively. We also provide a qualitative user-experience evaluation.

Introduction

Suitable lighting and thermal comfort play a significant role in people's physical and physiological well-being and productivity [118]. A significant amount of energy is spent on illuminating work places properly and also on maintaining thermal comfort. According to the United States Energy Information Administration (EIA), HVAC (heating, ventilation, and air conditioning) and artificial lighting systems account for about 32% and 25% of electricity usage in residential and commercial buildings [10]. Thus efficient usage of the HVAC and lighting is one major step towards reducing the energy consumption in indoor spaces.

Typically, building energy management systems (BEMS) are installed to lower the energy cost by automatically controlling the actuators[1] based on the ambient conditions and occupancy [16, 34]. The trick to achieve higher energy efficiency is to exploit natural conditions as much as possible and use the actuators to complement only the inadequacy, if any, in the natural conditions. However, most of these systems operate within a conservative range or set-point that is amenable to a large number of people providing only an average comfortable environment. For example, Monash set-point trials [9] suggested that ideally the temperature range of 21° C - 23° C, can provide thermal comfort to most of the occupants. However, the HVAC operating set-point can also be chosen from a broader temperature range (19° C – 28° C) based on external conditions and thermal preferences of the users in the building. Similarly, the suggested 500 lux for lighting an office environment may not be sufficient for some elderly people [124] and for young people fewer lumens may be sufficient. Thus developing a user-centric automation method per room can reduce the energy consumption and, *pari passu* increase comfort level.

With many occupants in a shared space like an office, there is a tradeoff in achieving the preferred comfort levels of the users and yet achieving energy ef-

[1] In the rest of this chapter the term *actuators* is used for both HVAC and artificial lighting systems.

ficiency. In this chapter, we build and test the **i**ndoor **L**ight and **T**emperature **C**ontroller (**iLTC**), which is a smart system that achieves higher energy efficiency while maintains the highest user comfort. A brief overview of the system is shown in Fig. 4.1. Unlike traditional BEMS, iLTC employs a room-level controller[2] to decide on an energy-optimal operating set-point for the actuators while trying to make all the co-occupants feel comfortable with respect to their comfort preferences. That is, the feeling of thermal and lighting comfort is not a single temperature or light intensity value for a person, but a range of values within which user can feel "equally" comfortable. Thus, iLTC needs to learn about the thermal and visual comfort preferences of each individual.

Implementing iLTC is highly challenging for the following reasons: *First*, the operating set-point for an actuator is a mere number. It is not easy to correlate comfort levels of humans with a certain light intensity and temperature value. A tangible scale of comfort levels needs to be mapped onto the set-points. *Second*, thermal comfort varies significantly from person to person. On the other hand, HVAC energy consumption is highly dependent on the temperature difference between the indoor and outdoor environments. Complete information about the comfortable temperature range of each co-occupant is required to decide on a common temperature set-point while keeping HVAC consuming the lowest possible energy. *Third*, lighting comfort also varies significantly from person to person [38]. Further, unlike temperature, light intensity also varies significantly at various locations inside a room. Thus, we need a mechanism to identify natural light intensity at specific locations (e.g., work-desk of a user).

This chapter addresses the above challenges and provides a complete working solution. Specifically we make the following contributions and some of them are, hitherto, unexplored.

1. We develop a layered design for iLTC that can offer room-level control for the lighting and HVAC systems. The system supports distributed implementation and thus it is scalable in terms of number of rooms and occupants (Section 4.3).

2. Our system employs a non-intrusive mechanism to derive user comfort preferences with minimal user intervention and training. We provide comprehensive mapping functions for a person, which can indicate comfort levels of the person for any given light and temperature value. To the best of our knowledge, we are the first to provide such a comprehensive mapping function (Section 4.4).

3. We estimate the natural light intensity at the work-desk of users by utilizing the light sensor available in their smartphones. We derive a relationship between the light intensity measured by the smartphone sensor and the outdoor light intensity with a single sensor. This eliminates the huge cost of deployment of additional sensors and their management (Section 4.4).

[2]This could be a simple addendum to the main controller of the HVAC in the building.

4. We propose an algorithm to determine the most energy-optimal operating set-point for HVAC and lighting systems while making all the co-occupants comfortable. The option of choosing a set-point from a range of comfort values offers a wide scope for energy savings (Section 4.5).

The core of iLTC is implemented using Java, while an Android application was developed to collect preferences of users. Matlab scripts were used for various model developments. We collected user preference data from 21 participants housed in different rooms to create individual temperature and lighting comfort functions. A detailed evaluation of iLTC is performed based on these comfort functions to measure energy savings. Furthermore, the proposed iLTC system was tested and evaluated by many users.

Related Work

A significant amount of energy is wasted by the HVAC and artificial lighting systems in a building due to their inefficient usage. Thus, a large body of current works focuses on the efficient usage of these systems from various aspects. Simple solutions proposed to save energy elicit turning off the actuators automatically when there are no occupants [38, 56]. The occupancy is detected using some sensor-based mechanisms. ThermoCoach [84] provides a personalized thermostat recommendation exploiting occupancy pattern. However, the technique used cannot be applied for shared spaces where there are multiple occupants.

Another set of work focuses on the reduction in energy consumption and stable operation of the HVAC systems. While some of the research efforts have been to optimize operational efficiency of the HVAC for a given set-point temperature [54, 58], a significant number of work also emphasize on selecting a suitable set-point temperature for the HVAC in order to fulfill thermal comfort of the occupants. There are two main methods for determining thermal comfort, heat-balanced way and the adaptive approach. The predicted mean value (PMV) is the heat-balanced way and depends on six parameters: metabolic rate, clothing insulation, air temperature, radiant temperature, air speed and humidity. Most of these parameters are difficult to obtain from typical sensors in BEMSs and hence the PMV proves to be impractical. The adaptive approach focuses on the adaptation of human psychological and physiological behavior [27]. Rather than using the traditional PMV model, which assigns a static comfort level to a user, recent studies have shown that participatory-based approaches can be used to optimize user thermal comfort and consequently reducing energy costs [39, 59, 82].

Participatory approaches allow occupants to give feedback based on their comfort level. From the feedback a consensus about common comfort value can be derived. One major challenge for such a system is that the set-point is resilient to outliers and thus a mechanism is required to cope up with the outliers. Zhang *et*

al. [132] provided a strategy that overcomes outliers. Another challenge is to find the balance between intrusiveness and user involvement in order for the occupants to maintain their incentive to participate. Most of these existing works require additional sensor deployment or detailed user information (preferences, demographics, etc.) and generate a fixed optimal set-point for each occupant given a room, which provides limited flexibility to decide set-point temperature. Erickson *et al.* [35] have used a participatory sensing approach to customize the HVAC conditioning. However, their approach does not consider shared spaces with multiple co-occupants. Moreover, it requires a significant amount of user participation to learn about one's comfort preferences.

With respect to reducing energy consumption by the lighting systems, the basic idea is to use artificial lights only when it is necessary. To do this light intensity inside the room need to be known. As the natural light level can change across days without any fixed pattern and also the light intensity varies at different locations within a room, measurements need to be done continuously at every location of interest. Thus many researchers have deployed a number of sensor nodes to monitor the fluctuation in natural light level [13, 83, 122]. However these approaches are intrusive and cumbersome.

In contrast to the above, we focus on: (a) selecting an energy optimal set-point as opposed to the fixed set-point; (b) employing only one reference sensor to provide the outdoor light intensity and using it to estimate the light intensity at desired locations across rooms; and c) use minimal data from occupants to learn their temperature and lighting comfort preferences.

iLTC: System Design

The system design for iLTC follows a layered design principal as described by DIAT (Fig. 4.2). Thus, the system is divided into three layers - (i) virtual object (VO), (ii) composite virtual object (CVO), and (iii) service layers. All the three layers are wrapped within a placeholder called 'daemon'.

There are two major building blocks for iLTC – 'user daemon' and 'room daemon'. A user daemon is associated with each user and it is hosted on her smartphone. The proposed system employed a smart-phone based App to learn individual temperature and lighting comfort levels for various temperature and light intensity values. The data collected through the App is utilized to create comprehensive comfort function. On the other hand, any activity associated with a room is handled by the room daemon. For each room in a building, a separate instance of room daemon is created for their independent operations. These daemons can be hosted on a centralized server at the building or at a room-level embedded device. The virtual representations of the actuators (VO) are hosted by the associated room daemon. Additionally, a temperature and humidity sensor is required for every room to get thermal feedback. As the modern HVAC systems can maintain a near homogeneous

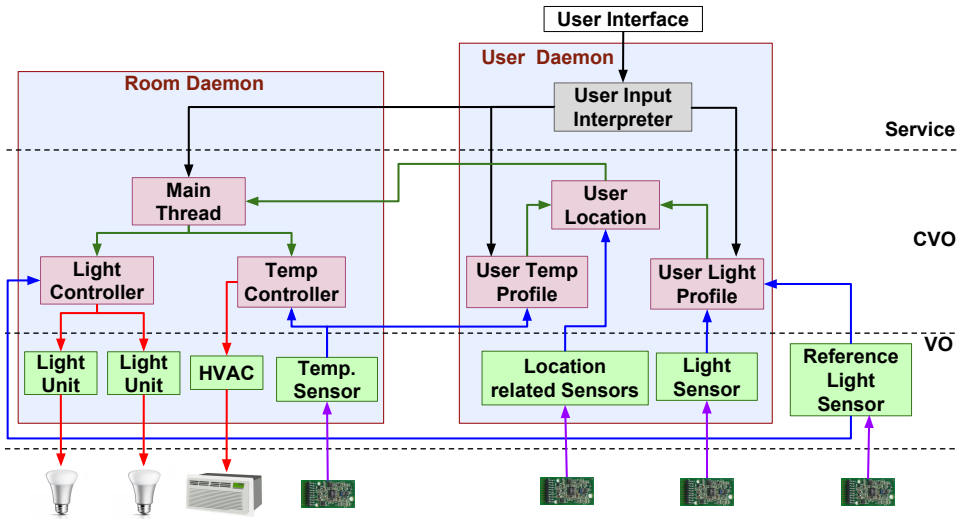


Figure 4.2: Layered design of iLTC and interactions amongst various components of its two main building blocks – the room daemon and the user daemon.

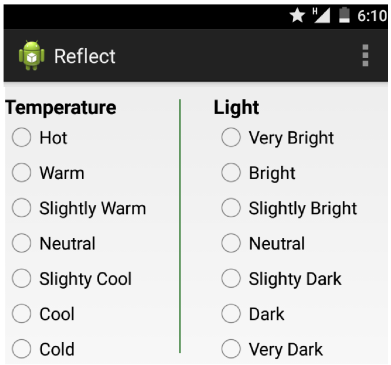
temperature within a room, it eliminates various thermal zones within a room. Thus, a single temperature sensor for a room is sufficient. The corresponding VO is hosted by the room daemon.

The proposed system uses artificial lights only when the received natural light at the work-desk fall below the comfort level. The natural light intensity at various positions inside a room varies significantly, and it keeps on changing. Thus, a trivial solution is to deploy light sensor at every work-desks. In contrast, we use a smart technique to avoid deployment of multiple light sensors inside a room. Rather we use a single light sensor for the whole building, and the effective natural light intensity at each user desk is estimated using this solitary sensor data. This reference light sensor is used by all users and room daemons, and is not part of either of the user and room daemons. Thus, the associated VO is hosted at a centralized location, which can be accessed by all stakeholders.

User Daemon

The user daemon obtains the preferences through the user interface and builds a comprehensive comfort function for thermal and visual preferences. In this section, we describe how individual user preferences are obtained and modeled. Furthermore, we also discuss how to estimate natural light intensity at a user's work-desk using only a single reference light sensor.

A user daemon hosts three CVOs – 'light profile', 'temperature profile', and 'location'. While the profile CVOs learn preferences of users during the training period,



(a) Application screenshot

comfort indicator	ASHRAE value	iLTC value
hot/very bright	+3	1
warm/bright	+2	2
slightly warm/bright	+1	3
neutral	0	4
slightly cool/dark	-1	3
cool/dark	-2	2
cold/very dark	-3	1

(b) Comfort levels

Figure 4.3: Screenshot of the App for collecting user preferences, and mapping them into a numeric scale.

the location CVO identifies whether the user is inside a room or not. This binary classification of user presence is performed using WiFi based indoor localization. We utilized existing WiFi access points deployed in the building along with smartphones of occupants. When a movement is detected (i.e., change in accelerometer data or step detector), the data collection for localization is initiated. The localization has training and evaluation phases. During the training phase, WiFi scans are performed periodically at the user location (in her office room). This phase is also called the fingerprinting stage, where data from WiFi scanning is used to determine the list of visible WiFi Access Points (APs) and their Received Signal Strength (RSS) along with the timestamp. Since, only room-level occupancy is required the duration of training phase can be varied. The feature vectors for different periods at each location are used as training set for classification. A Bayesian classifier model is built on the feature vectors to determine the presence of users in the room [72]. In the evaluation phase upon detection of movement, new feature vector is evaluated with the classifier model to determine if the user has left the room or not.

Individual User Profiling

The core of iLTC is to build individual comfort profiles for temperature and lighting. For a person, maximum thermal comfort is not a single temperature value rather a range of temperature values. Similarly, visual comfort also spans over a range of light intensities. Most users cannot easily correlate their comfort levels with temperature or light intensity values. Even if they do, there is a chance of significant deviations. To this end, iLTC utilizes a smartphone App to learn the preferences, coupled with precise measurements from corresponding sensors. When new users enter the system, we consider a conservative approach with respect to her prefer-

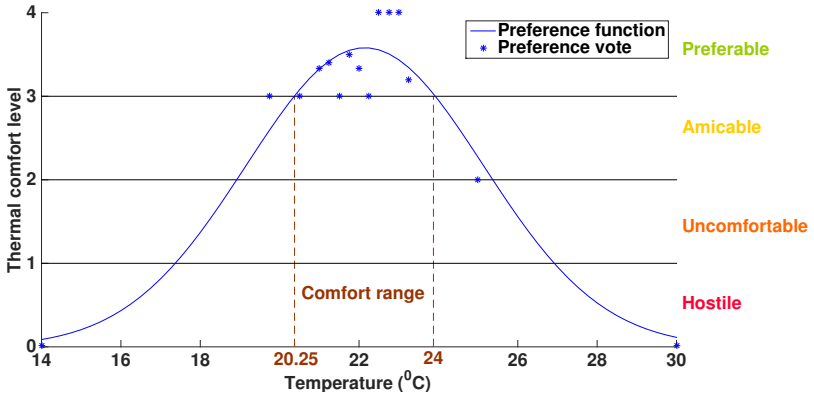
ences. Initially, we assign preference values derived from survey/ASHRAE, which are then personalized based on the preferences data collected over time.

Collection of user preferences
Thermal comfort of a person for a particular temperature cannot be determined without the feedback from the person. We use both explicit and implicit way of learning the preferences (feedback on comfort levels) – explicitly by asking the users to indicate their comfort levels and implicitly (later while iLTC is in operation) when the user overrides the controller settings. This allows dynamic adaptation of comfort preferences. Furthermore, if the new setting is significantly different, then preferences are adapted again by collecting additional voting data from the user. Thus, by capturing the changes in user preferences overtime, iLTC eliminates any outliers.

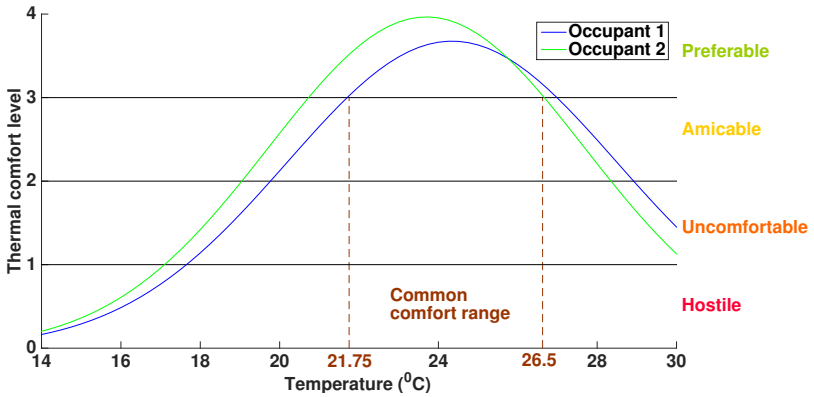
A screenshot of the App, which is used to collect feedback, is shown in Fig. 4.3a. To indicate comfort a seven-point scale is used as suggested by the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE). However, we convert the traditional numeric scale of [-3,+3] to [1,4] (see Fig. 4.3b), because indicator level 'neutral' is given the highest preference, whereas indicator level 'cold' and 'hot' are given the lowest preference. To learn the visual comfort a similar scaling is also adapted as indicated in Fig. 4.3. The data collected from the App is used to model the comfort preferences for temperature and light. Note that the explicit data collection is done only in the beginning when a user becomes part of iLTC.

Our goal is to create comprehensive comfort functions that can indicate the comfort level of the person for any given temperature or light intensity. To build such a function, ideally, we should have comfort level indicator for each possible value, which is not a feasible option. Thus we collected a few comfort indicators and then we try to model them. The data collection was conducted from 21 users with 5 different ethnic background, age varying from 24 to 51 years. The data collection is done over several weeks with a dedicated sensor node deployed in each of the rooms of the occupants (participants). Next, we explain the mapping from these measurements to comfort levels.

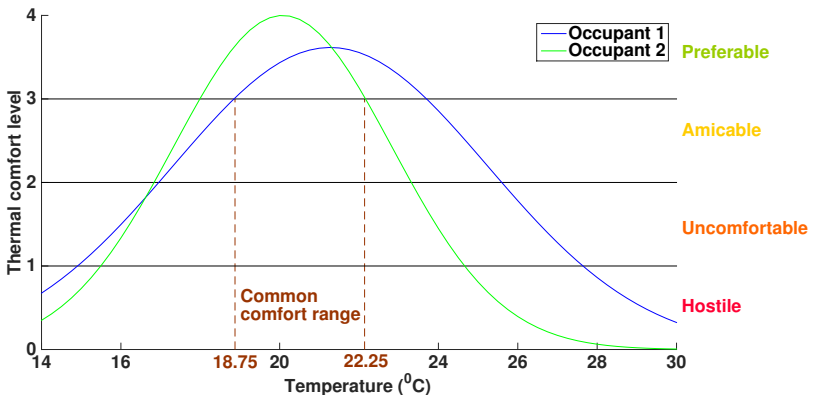
Modeling user profile: mapping from room temperature and luminance to comfort levels
Whenever a person indicates her comfort level, it is registered with its corresponding sensor values. During function creation, we cluster these comfort indicators in multiple equal sized bins. The bin size indicates a small range of sensor values for which user comfort remains unchanged. Different bin sizes were empirically evaluated across all the participants to determine the optimal bin size. More details on selecting appropriate bin size is discussed in Section 4.6. After analyzing user preference data collected from multiple users, we notice that thermal comfort function



(a) Comfort function of a person



(b) Common comfort range: room 1



(c) Common comfort range: room 2

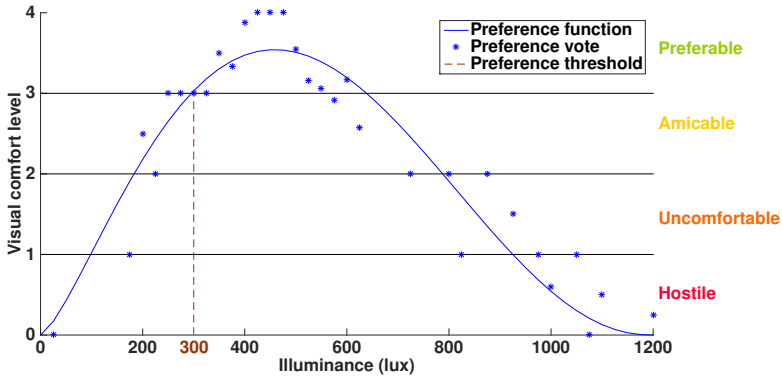
Figure 4.4: Comfort function of an individual based on preference voting, and common comfort range of a room based on individual comfort functions of all the co-occupants.

can be represented using a Gaussian function (eq. 4.1), whereas the light preference function can be represented using a Beta function (eq. 4.2).

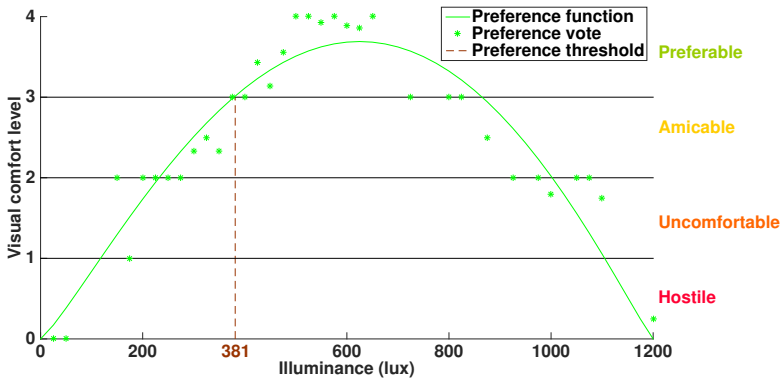
$$F_T(\underline{\alpha}, t) = \alpha_1 \exp^{-(\frac{t-\alpha_2}{\alpha_3})^2} \quad (4.1)$$

$$F_L(\underline{\beta}, l) = \beta_1 l^{(\beta_2-1)}(1-l)^{\beta_3-1} \quad (4.2)$$

For every user, the parameters ($\underline{\alpha}$ and $\underline{\beta}$) of these functions differ. Based on the comfort indicators, we derive the individual function parameters using the least square curve fitting. To derive a reflective function from the limited samples, we assume that any temperature beyond 14° C and 30° C will be uncomfortable for any person. Thus, with the existing comfort indicator data set, we add two additional data points of these two extreme temperatures with comfort value 0 using the least square method. Similarly, for light these two extreme values are 0 and 1200 lux.



(a) User 1 preference function



(b) User 2 preference function

Figure 4.5: Comfort functions of individuals based on preference vote and lower bound of lighting comfort.

Fig. 4.4a shows the clustered preference data and the thermal comfort function derived for an individual. From this function, we can conclude that this particular user feels maximum comfort within the range of 20.25° C and 24° C . When multiple users occupy a room, a common comfort range needs to be determined. Fig. 4.4b and 4.4c, shows the common comfort range in two different rooms occupied by different set of people (each room with two occupants). It is clear that common comfort range of two different rooms can be quite different based on the individual comfort ranges of the occupants. Thus, a common temperature set-point for all the room can cause discomfort for some of the occupants or it will expend more energy by the HVAC or lighting systems or both. iLTC exploits the comfort ranges of all the occupants in a room to determine the most energy optimal HVAC set-point for the room. Moreover, the comfort range for a user is location independent and it can be carried over when user changes her location.

Similarly, the clustered lighting comfort indicators and the fitted comfort function for two users is shown in Fig. 4.5. From these figures, we can conclude that the minimum desirable luminescence varies significantly from person to person. Unlike temperature, a common comfort range for lighting need not be derived for following reasons. First, when the light intensity becomes uncomfortable due to superfluous light, the artificial lighting system cannot be used to reduce the intensity (like a HVAC through cooling). Rather, the lighting systems remain completely off and window blinds can be used to block additional sunlight. Second, the particular light intensity from the lighting systems can illuminate differently at various parts of the room. Thus, a work-desk close to window might get sufficient sunlight, while a work-desk far away from the window might experience light deficiency. As the received light intensity from a light source (natural or artificial) differs from desk-to-desk, only individual visual comfort threshold need to be considered.

Modeling of received light at work-desks

As the light intensity varies within a room, it is important to measure the amount of natural light reaching each work-desk for a particular outdoor light intensity. Thus a single light sensor is not sufficient to measure the amount of natural light at different locations (in case there are more occupants in a room). Moreover, setting a particular set-point (brightness level) for a light unit does not mean a uniform light intensity in all parts of the room. This necessitates measuring received amount of light at the work-desks of the users from different sources of lights – artificial and natural.

Modeling of received natural light

iLTC employs the smartphone light-sensor of a user to measure the received light intensity at his/her work-desk. However, for *natural light*, one time measurement using the smartphone is not sufficient as the natural light can vary over the days. To

resolve this, we use one reference light sensor. Based on the measurement tuple of the reference light sensor and the smartphone light sensor, a relationship is established. This can be used to derive received natural light at the work-desk when there is a particular outdoor light intensity. Our goal is not to measure light intensity at every part of the room, but only at the work-desk of the occupant. Once this measurement is done, the relationship remains the same until the user changes his/her work-desk.

Though the gradient of light intensity can be expressed as, $l_r = \frac{l_s}{4\pi d^2}$, where l_s and l_r are the light intensity at the source and at a distance d from the source respectively, the gradient of light intensity inside a room cannot be described using the same relation. However, a similar form of relationship can be seen between outdoor and indoor light intensity as, $l_u = a_1 * l_o + a_2$, where l_u and l_o are the light intensity at the user work-desk and outdoor respectively. To determine the unknown parameters (a_1 and a_2) for a work-desk, we collected light values for a day after the user becomes part of the iLTC system (using her smartphone). Using the training data set, the parameters are estimated using least square method. Once these parameters are known for the work-desk, the received natural light can be estimated based on the reference light sensor values.

Intuitively, it is clear that once the parameters are found, the same parameters can be used to estimate light values for that location irrespective of the room and window dimension. However, the estimation accuracy suffers significantly if the same set of parameters is used irrespective of time of the day and weather condition. To improve the estimation accuracy, our approach splits the data set collected into multiple segments based on the light intensity values of the reference sensor. Then for each of these segments, we find the set of parameters.

Another important factor that influences the accuracy of natural light estimation is the visibility of the sun from the window (time of the day). Thus, we divide the data into various segments based on the location of the Sun and orientation of the window in the measurement room, and then determine the estimation parameters. Our detailed evaluation shows that dividing the data set into two parts – when the Sun is visible from the window, and when it is not visible from the window – improves estimation accuracy. By visibility of Sun, we mean the Sun's location is within the visible area of the sky through the window. Thus, to estimate the natural light at a work-desk, the right set of estimation parameters are chosen based on Sun's visibility from the window and light intensity segment of the reference light sensor.

Though the visibility of Sun from a window changes drastically throughout the year, it can be easily determined. If the direction of the window and geographical location of the room (latitude and longitude) are known, visibility of Sun can easily be derived from the Sun's azimuth. To derive Sun's azimuth, we have used the algorithm provided by the measurement and instrumentation data center (MIDC) of the national renewable energy laboratory (NREL) [7]. Note the window orientation and geographical location of the room is a one-time information.

Modeling received artificial light

Similar to the natural light, the amount of received artificial light also varies within a room. Thus, it is also necessary to measure the amount of received light from each of the light units at the work-desk of a user. During the training period, we turn on the light units in appropriate steps to measure the received light intensity from each of the light units at the work-desks. The light units were set to full brightness. Using the collected data, the gradient of brightness can be determined. For every light unit, there is a different gradient at different work-desks.

In building iLTC, we made the following two assumptions about modeling of preference: (a) since the temperature preferences is not location dependent, it can be utilized across various rooms, including a common meeting room and home environments; (b) the same is applicable for visual comfort. However, in a different room the gradient of light intensity from Sun and artificial sources varies differently. This necessitates a different set of parameters to model received light intensity for different work-desks. Since a user spends most of his/her time at a particular work-desk, one training campaign is sufficient.

Room Daemon

Most of the real-time activities are handled at the room daemon. Whenever a user enters or leaves the room the set-points for the actuators need to be adjusted. Moreover, if the natural conditions change, that also influences the set-point values of the actuators. Each room daemon hosts three CVOs – 'main thread', 'light controller', and 'temperature controller'.

Main thread CVO

The main thread CVO manages overall execution of the room daemon. When a person enters a room, it sends an 'arrival message' to the room daemon. This message contains the identity of the users and their preferences. Upon detecting arrival of an occupant, the daemon communicates preference values to the controller CVOs. It also makes a temporary local copy of the user preferences along with marking the presence of the user. It periodically monitors presence of occupants inside the room, and instructs the controller CVOs to adjust set-points of the actuators if required. On the other hand, the user daemon sends periodic 'hello messages' to indicate the presence. If no hello message is received from a user for significant amount of time, CVO assumes that the user has left the room, and removes her comfort preferences. If the user daemon itself identifies that the user left the room, it explicitly sends a 'departure message' to the room daemon. Whenever a departure is detected, the controller CVOs are invoked to adjust the set- points if required.

Light controller CVO

The goal of the light controller CVO is to ensure usage of artificial light only when the natural light is insufficient for lighting comfort. Given that it knows the amount of received natural light at a work-desk (l_n) and the corresponding user preference regarding the lighting comfort (l_p) (see Section 4.4.2), the amount of light deficiency can easily be calculated ($l_d = l_p - l_n$). Using the artificial light modeling described earlier, received amount of light at the work-desk can also be calculated if a particular light unit illuminates at certain brightness. Based on these, the CVO can decide the minimum brightness for each light unit so that the light deficiency of the user can be supplemented. A minimum amount of energy consumption is ensured since lower brightness means lower energy consumption.

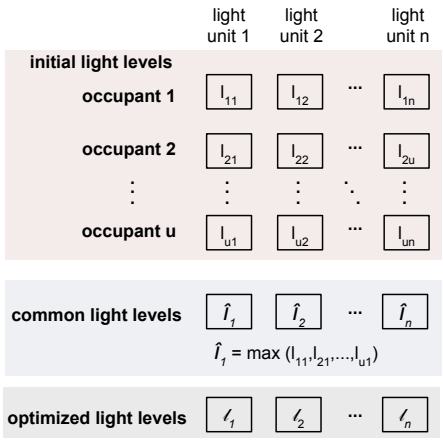
When there are multiple occupants inside a room, a certain brightness level for the light units may not satisfy everyone's lighting preference. Thus, to fulfill light deficiency of all the occupants while maintaining lower energy consumption, a suitable combination of set-points (brightness level) for each light unit needs to be decided. We formulate this as an optimization problem and it is given below.

$$\min \sum_{i=1}^{n} l(i) \tag{4.3}$$

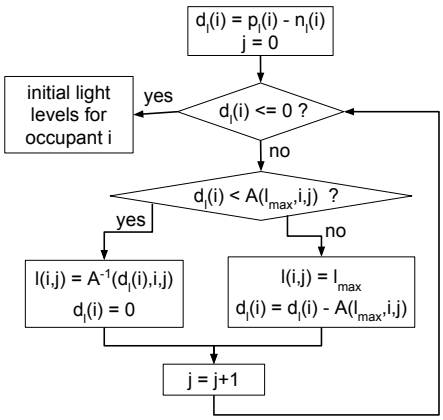
$$\texttt{subject to}: \quad \sum_{i=1}^{n} A(l(i), i, u) \geq d_l(u), \quad \forall u, \tag{4.4}$$

$$\texttt{where}, \quad A(l, i, j) = a_1(i, j) \times l + a_2(i, j). \tag{4.5}$$

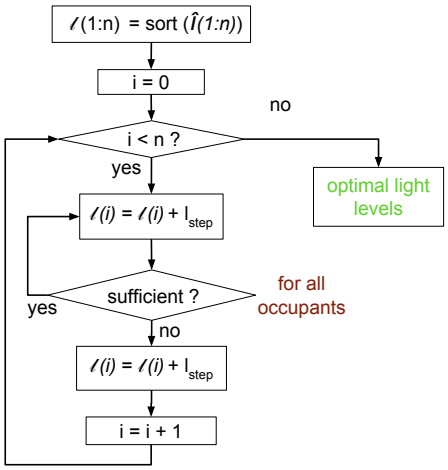
The objective of the optimization problem is to select a combination of set-points for each light unit in a room such that the light deficiency is fulfilled while having minimum energy consumption. If a light unit i sets its brightness to $l(i)$, then the received light amount for user u can be calculated using 4.5. Eventually, the combined received light amount should be at least equal to the light deficiency $l_d(u)$. If there are 4 light units and each light unit have 16 brightness levels, there are a total of 65536 combinations to choose an optimal brightness level. Light intensity can change quickly within a short time period. Thus, the light controller needs to adjust the set-points every now and then. Selecting an optimal set-point out of all possible combinations can incur significant computational cost. Thus, we propose a heuristic algorithm to find the set-point that maximizes user comfort and minimize energy consumed.



(a) overview of the algorithm



(b) initial light level for user i



(c) globally optimized light level

Figure 4.6: A three step algorithm to decide set-points of the light units inside a room when there is one or more occupant(s).

A brief overview of the algorithm is shown in Fig. 4.6. First, a combination of optimal set-points for all the light units are derived for each of the occupants based on her light deficiency. The common set-point for a particular light unit is decided by taking the maximum among individual brightness levels required for all the occupants affected by that light unit. This ensures that everyone would receive sufficient amount of light. In the final step of the algorithm, the brightness levels are decreased one step at a time to see whether this new combination can fulfill the deficiency of all the occupants. This iterative process stops, when no further decrease in brightness level is possible. Here the algorithm assumes that the brightness level

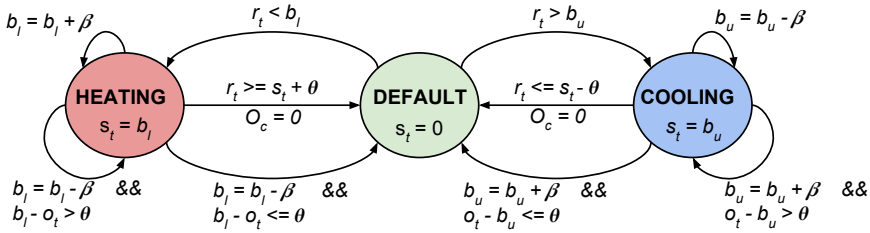


Figure 4.7: State diagram of the HVAC operation and associated temperature set-point (s_t) assignment. Transitions among the states depend on the room temperature (r_t), outside temperature (o_t), number of occupants (O_c), and the common thermal comfort range ($< b_l, b_u >$) of all the occupants.

of the light units can be varied. However, for traditional lighting system with only two states (on/off), we use a similar but simpler technique to decide whether a light unit should be On or Off at any time instant. When there is a quick drop in natural light levels, the brightness levels are not increased immediately, rather a similar iterative approach is considered but with faster rate of change in brightness level. This ensures occupants do not notice any immediate fluctuation in the light units.

Temperature controller CVO

The temperature controller CVO decides temperature set-point for the HVAC, which maximizes the comfort of all occupants and minimizes HVAC energy consumption. Current HVAC systems are quite efficient in terms of maintaining a room temperature based on the given set point. Furthermore, with the introduction of zone heating, HVAC systems can now eliminate hot or cold spots in a room and maintain a set temperature value across the room. Moreover, several research efforts are conducted to determine the optimal pre-conditioned temperature of a room before an occupant arrives or after she leaves. In this regard, this chapter focuses on how to obtain an optimal temperature set-point, which maximizes the comfort of all occupants and minimizes the energy consumed by the HVAC system. Determining a set-point is not trivial when there are multiple occupants present in a room. The temperature controller CVO finds the common comfort range of occupants based on the preferences collected previously.

A HVAC can be at three different operating states – default, heating or cooling. Default is a state when the HVAC consumes minimal amount of energy – without loss of generality, it can be the Off state. In heating state, the HVAC blows warm air inside the room such that the room temperature reaches a set-point value. The warmth of the air is decided based on difference between the desired set-point temperature and current temperature, whereas the energy consumption depends on the difference between the desired set-point temperature and outdoor temperature. In the cooling state, HVAC operation is similar.

We now describe how HVAC states are switched and how the set-point temperature is decided (Fig. 4.7) so that all the occupants feel comfortable. In the beginning, the HVAC is in the default state and its set-point is set to zero ($s_t = 0$). When an occupant enters the room, the room daemon receives her thermal comfort function. Then, the temperature controller CVO finds a comfort range for the person, which includes a lower (b_l) and upper (b_u) bound. The CVO also gets the current room temperature (r_t) from the temperature sensor VO. If the current room temperature is within this bound, then HVAC continues to remain in the default state. The CVO periodically checks if the room temperature falls out of this bound, and changes its operating state. This can happen for two reasons: (i) the room temperature changes due to occupants and/or the outdoor temperature; and (ii) the bounds of the comfort range are modified (narrowed) because of a new occupant entering the room. Hence our system periodically monitors the user presence in the room and the current temperature to maximize user comfort.

In case, the room temperature falls below the lower bound of the common comfort range ($r_t < b_l$), the HVAC enters the heating state, and the set- point is set to this lower bound ($s_t = b_l$). This ensures that all the occupants comfort preferences are met. At the heating state, if a new occupant leaves/arrives, the bounds for the common comfort range are modified. If the lower bound increases compared to the previous one ($b_l = b_l + \beta$), that means the heating need to be continued and the set-point is adjusted to the new lower bound. In case the lower bound gets reduced, there is a possibility of decreasing the heating intensity. If the new lower bound is significantly higher than the outside temperature ($b_l - o_t > \theta$), then the set-point is adjusted to the new lower bound and the HVAC continues in this state. Otherwise the HVAC is switched to the default state. At the heating state, if all the occupants leaves the room (O_c) or the room temperature reaches sufficiently higher than the set- point temperature ($r_t >= s_t + \theta$), the HVAC enters the default state with set-point being zero. The switching between heating to default state is guarded with a threshold θ to ensure that the state change does not happen frequently. A similar switching happens on the right side of the state diagram when room temperature goes beyond the upper bound of the common comfort range and the HVAC enters the cooling state. As mentioned earlier, this work focus on deciding the optimal set-point and assumes that the current HVAC system is capable of maintaining the set temperature value based on the physical conditions of the room. Thus by constantly monitoring the room condition and occupancy, iLTC adapts the set-points to maximize user comfort and minimize energy consumption of the HVAC system.

Evaluation

In this section, we describe our experimental setup and provide details of all the sensors used during the setup. We present results regarding modeling of received light at the work-desks and energy savings incurred with the deployment of iLTC.

Experimental Setup

Our iLTC system was deployed in an office environment with multiple rooms. The number of lights, window size and room size can vary depending upon the building considered. However, the functioning of the iLTC system is independent of these parameters. The set of devices used for our measurements, actuation and data collection includes: (i) Moteiv tmote-sky sensor nodes measuring temperature and light intensity in indoor and outdoor locations, (ii) Smartphones from different manufacturers for localization and user comfort indicator, (iii) Philips hue light bulbs for indoor lighting, and (iv) Plugwise circles were used to measure energy consumption of the hue bulbs.

Our experimentation considered 21 participants in an office environment from 5 different ethnic background with age varying from 25 to 51. The participant list includes both male and female users and all the users had their smartphone.

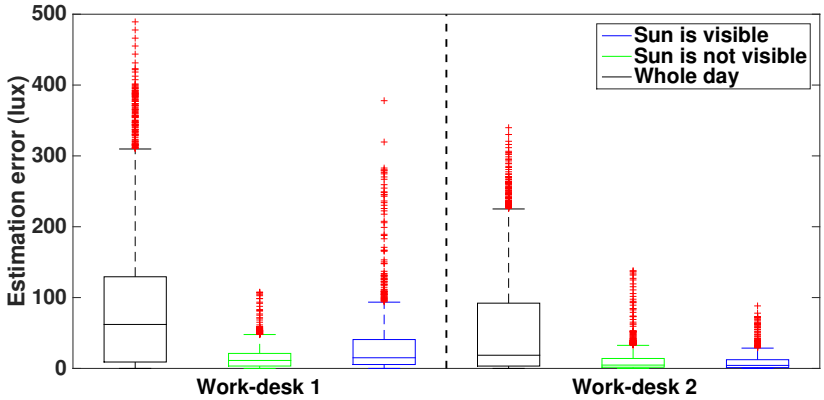


Figure 4.8: Estimation error of the received natural light at two work-desks.

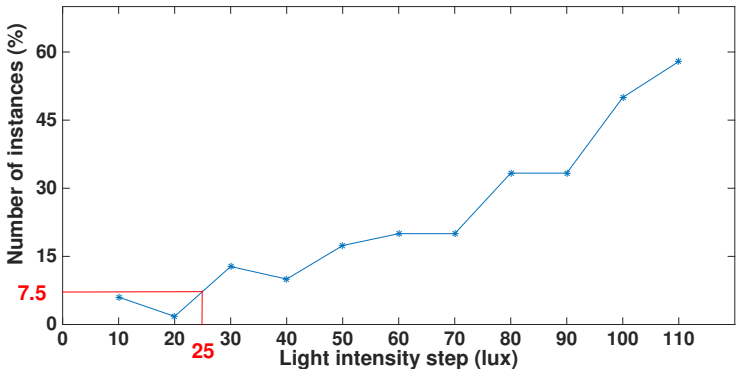


Figure 4.9: Various light intensity change in one step versus the number of instances when people felt annoyed with that sudden change.

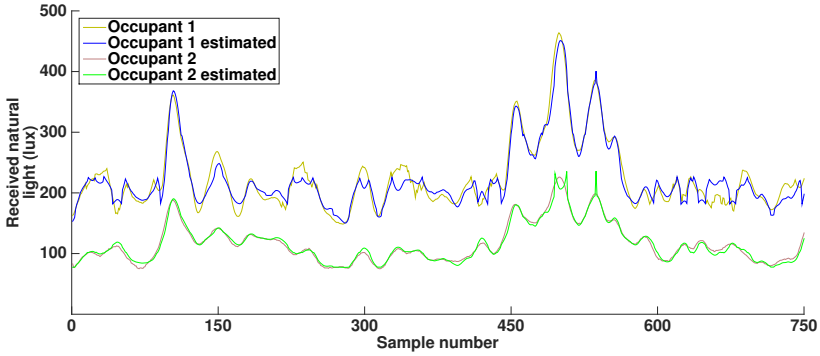
Results

The goal of iLTC is to decide set-points for the actuators such that (i) all the co-occupants in a room can be provided maximal comfort; and (ii) the energy consumption by the actuators is kept minimal. Deciding light unit set-point requires information about deficiency in natural light conditions. Thus, we first show the natural light estimation at the user work-desk using the estimation technique proposed in Section 4.4.2. Fig. 4.8 shows the estimation error of the received natural light at two work-desks. It can be seen that, when the whole day data is considered a high error is associated with the estimation. On the other hand, by splitting the data set based on visibility of sun significantly reduces the estimation error. This is mainly due to the consideration of rate of change of natural light with respect to sun visibility at the user work-desk.

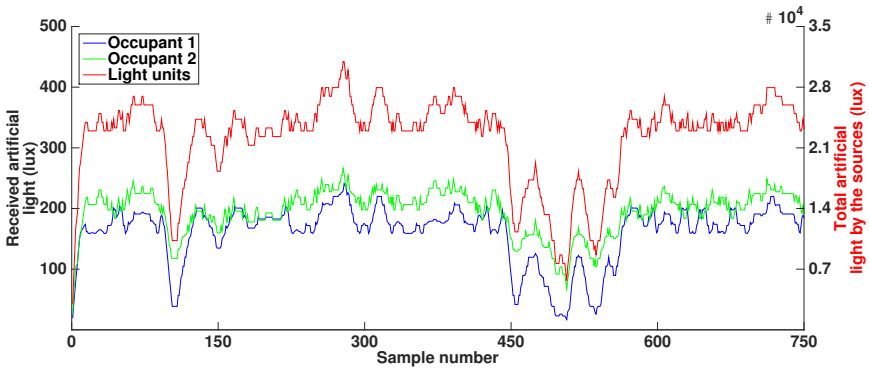
Fig. 4.9 shows the light intensity steps (in lux) and the corresponding number of instances when the occupant noticed the change in light intensity. We experimented this across participants and the average results are shown in Fig. 4.9. It can be seen that, when the combined received light intensity at the work-desk changes in a bigger steps more number of users feel annoyed. From our experiments, we determined 25 lux to be the step change, applied when increasing or decreasing the light intensity to prevent user inconvenience. In Section 4.4.1, we discussed that a range of temperature and light values are clustered into bin before deriving the comfort functions. For the light values, a bin size of 25 lux is used for the range of 0 to 1200 lux (This is also clear from Fig. 4.9). For temperature, it starts from 14° C until 30° C with a bin size of 0.25° C , that means any comfort label indicator for the range 18° C to 18.25° C is mapped to 18° C .

The light intensity for the hue bulbs considered in our experimentation are within the range 600 to 16000 lux near the source, and the energy consumption ranges from 0.58 W/s to 5.4 W/s. Unlike the temperature set-point, setting a particular brightness of light units does not guarantee the required level of lighting comfort at the user's work-desk as the light intensity degrades significantly while moving away from the source. Consider two users inside a room with minimum light comfort preference of 300 and 381 lux (Fig. 4.5). Fig. 4.10a shows the amount of natural light received at the work-desks over a time period. The light samples are measured every 10 s. During the real deployment of the system, the measured value of received light intensity will not be available. Thus, we use an estimate of received amount of light using the reference sensor as described earlier. It can be seen that, the estimated light intensity closely follows the actual received light intensity. This estimated light is used as input for the light controller CVO.

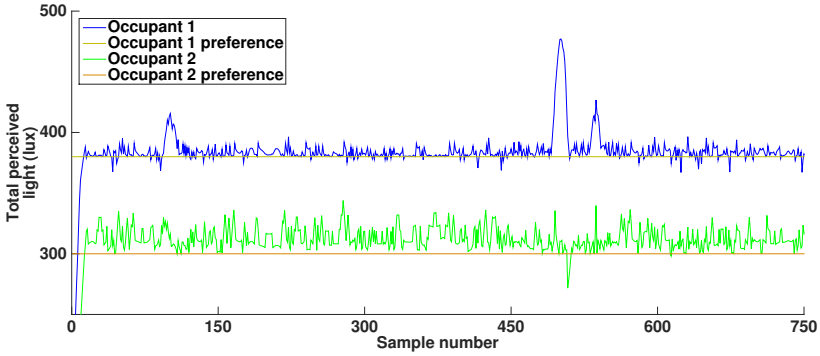
As mentioned in Section 4.4.2, we measured the gradient of light intensity from each source at each work-desks. Fig. 4.10b shows the total amount of light seen at the source (with all six lights) and the received light intensity at the user work-desks. The decrease of received light at the work-desks is indeed due to distance



(a) Measured and estimated natural light



(b) Received artificial light



(c) Total perceived light

Figure 4.10: Based on the estimation of the received natural light at two work-desks, and their corresponding light preferences set-points are decided for the light units.

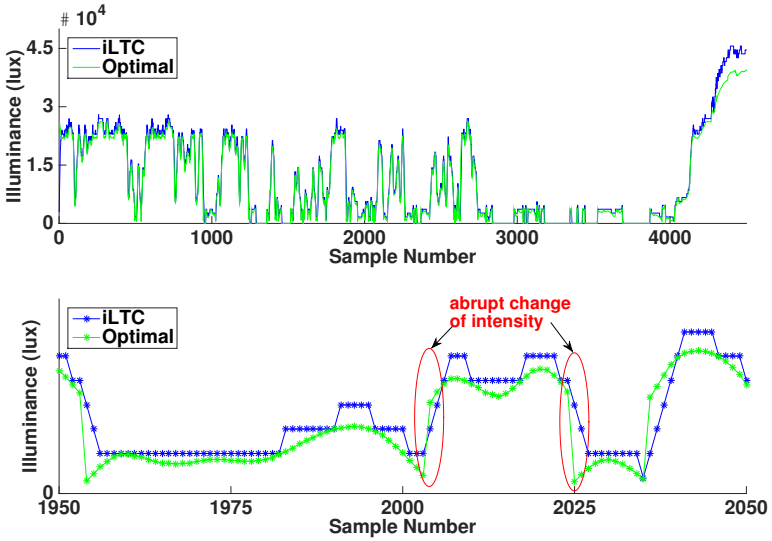


Figure 4.11: Combined luminance of all light units: a comparison between iLTC and the optimal light level selection.

from the artificial lights. Furthermore, Fig. 4.10c shows the total perceived light at the work-desk by considering both natural and artificial light. It can be seen that, lighting preferences of both the occupants are always met by adjusting the brightness level of the lighting system when necessary.

As mentioned in Section 4.5.2, the light controller CVO uses a computationally inexpensive algorithm to decide the set-points. From Fig. 4.10c, it is clear that the algorithm serves the purpose of providing lighting comfort to all the occupants. To evaluate efficiency of the algorithm, we derived the set-points using an optimal solution and compared it with iLTC solution. The optimal solution is found by testing all the possible combinations of brightness levels.

Fig. 4.11 shows the combined brightness of all the light units using the optimal and iLTC set-point solutions. As the energy consumption is directly proportional to the brightness level, this also reflects the level of energy consumption. From the figure, we can conclude that the iLTC solution is very close to the optimal solution. To be precise, iLTC uses only 6.92% higher light than the optimal solution. However, the number of iterations to find a suitable brightness levels using iLTC is mere 0.01% of the optimal solution. There is another significant drawback with the optimal solution. In order to find the least energy consuming brightness levels, the optimal solution can change the brightness levels of the light units too frequently with certain change in natural light level. This may cause annoying experience to the users. On the other hand, iLTC ensures that whenever the brightness level gets

decreased, it decrease by only one level of brightness (a closer look at Fig. 4.11 for samples between 1950 and 2050). From our empirical evaluation, we found that a lux difference of 25 is unnoticeable by the users at their work-desk and we used that as the minimum brightness step (see Fig. 4.9). This ensures that the user will hardly notice any change in the brightness when decreasing the light intensity.

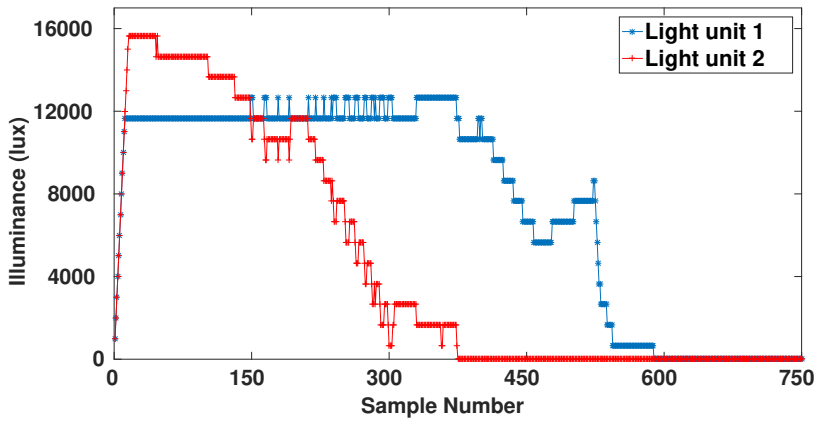


Figure 4.12: Brightness (set-point) decreases over time for two light units.

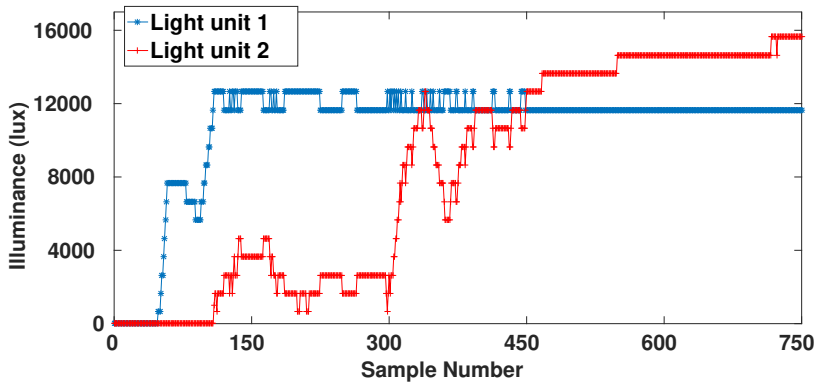


Figure 4.13: Brightness increases over time for two light units.

Fig. 4.12 shows the reduction in light intensity to maintain user comfort and minimize energy consumption across two lamp units. It can be seen that the light intensity is reduced iteratively such that not more than 25 lux difference is perceived at the user's work-desk. This approach ensures sudden fluctuations in natural light do not affect the user comfort. Moreover, when the natural light is not sufficient, artificial lights are turned ON to maintain the user comfort levels. For example, in evening the natural light perceived at the user work-desk might be lower than the

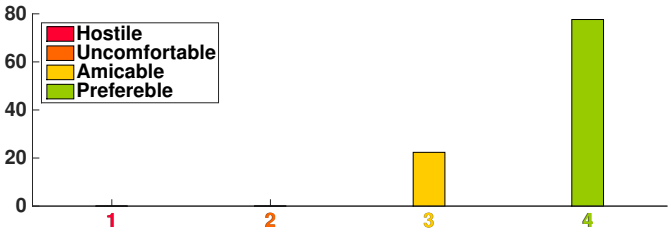


Figure 4.14: Visual comfort feeling of people when the artificial light units are controlled automatically using iLTC.

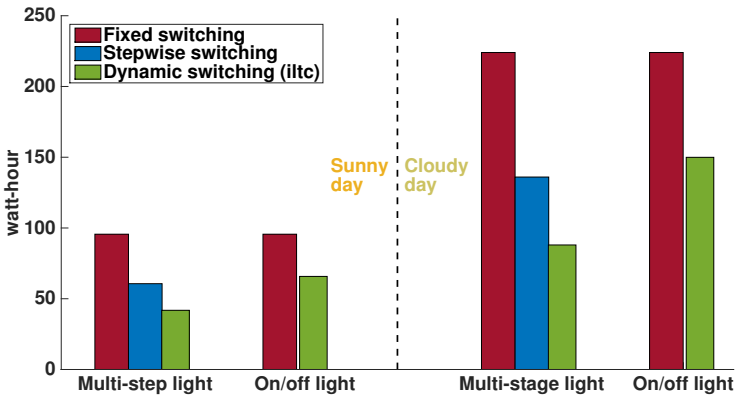


Figure 4.15: Combined energy consumption by all light units in a room for different switching strategies.

comfort preference. During this stage, we follow an iterative approach to increase the lux value, but at a faster rate rather than abrupt change in lux causing inconvenience to the user. Fig. 4.13 shows the rise in lux value to maintain user comfort across two light units. Thus iLTC avoids both abrupt increase and decrease in light intensity and follows a iterative approach to achieve the same without causing too much inconvenience to the users.

We also conducted a post-deployment user evaluation to determine the efficacy of the system (Fig. 4.14). Based on the user preferences collected, the set-points for the lighting system and the brightness level of the light units were constantly adapted. The feedback was collected using a questionnaire available on the smartphone App. The questionnaire comprised of questions related to visual comfort feeling. Each user selects one of the comfort levels *viz.,* (i) hostile, (ii) uncomfortable, (iii) amicable, and (iv) preferable based on the current set-points decided by the iLTC. The post-deployment evaluation was conducted on several days to generalize the outcome. On an average 78% of the feedback from 21 participants indicated preferable comfort

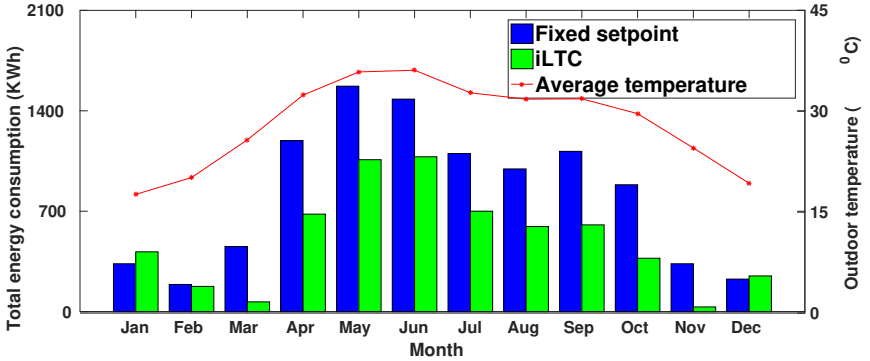
Table 4.1: Reduction of energy consumption by iLTC as compared to fixed and stepwise switching of the light units.

	Sunny day		**Cloudy day**	
Switching method	Multi-step	On/off	Multi-step	On/off
Fixed	56.25%	31.20%	60.71%	33.06%
Stepwise	31.03%	–	35.29%	–

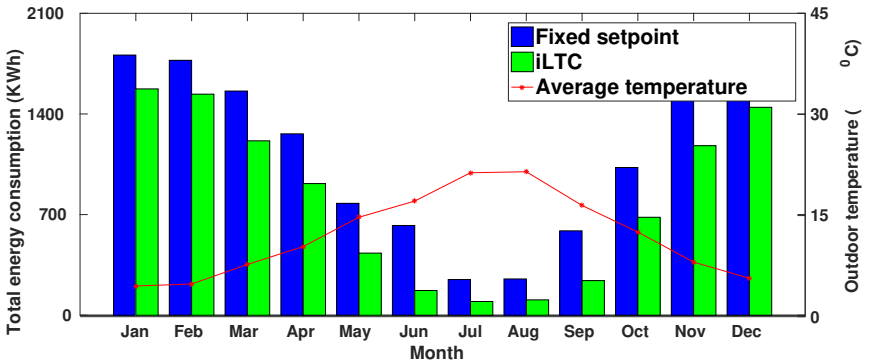
feeling when the light intensity was adjusted due to either insufficient light intensity or excess light intensity. 22% feedback received indicated amicable feeling at certain time periods. A closer look revealed that these are due to the variation in comfort preference of the user and also due to sudden fluctuations in natural light perceived at the user work-desks. This change is preference was further considered to adapt the user preference models accordingly.

Finally, to evaluate the efficiency of the light controller, we compared iLTC with two switching mechanisms. (i) Fixed switching – where the light units are turned on to the maximum levels when the outdoor light intensity drops beyond a certain threshold. This threshold is decided when either of the users face light deficiency. (ii) Stepwise switching – where the light units are turned on with varying brightness with the variation of outdoor light intensity. We tested these strategies on the data sets for a sunny day and a cloudy day, where on the cloudy day indoor light intensity was insufficient for the occupants almost throughout the day. The total energy consumption by all the light units is shown in Fig. 4.15 when various switching strategies are used. All the strategies considered include occupancy detection before turning on the lights. iLTC based switching consumes the least energy as compared to other strategies. This is mainly due to the individual control of brightness level at each light unit. Table 4.1 shows the total reduction in energy consumption by iLTC as compared to fixed and stepwise switching mechanisms.

To compare the energy consumption of the HVAC, we adopted the energy-temperature correlation model $P = |\frac{\lambda}{M}(t_i - t_o)|$ as described in [131], where P is the amount of energy consumed by the HVAC system in one second, λ is the conductivity of a particular room, M is the efficiency of the HVAC system, and t_i and t_o are the indoor and outdoor temperatures respectively. For a particular room, λ and M are constant. So, the HVAC energy consumption is mainly dependent on the difference in set-point temperature and outdoor temperature. Additional details of the HVAC system such as duct type, radiation/convection, air re-circulation is not considered as it varies from one HVAC system to another and also dependent on the building characteristics. Our objective is to show the potential energy savings by finding the optimal set-point to maximize user comfort and minimize energy consumption. Thus we use a simpler energy model based on difference between indoor set-point and outdoor temperature as described in [131].



(a) Delhi, India



(b) Delft, Netherlands

Figure 4.16: Comparison of energy consumption by the HVAC for fixed set-point technique and iLTC. Considering the total yearly consumption, iLTC is 39% and 27% less expensive based on the outdoor temperature in two cities.

For our evaluation, we fixed the values of λ and M to be 70.5 J/s.K and 0.14, respectively (from [131]). We consider two rooms at two different parts of the world – (i) Delft, the Netherlands, and (ii) Delhi, India. We collected the yearly weather data for these two cities from public repositories [8]. Also there were two occupants in these rooms in both the places with comfort range spanning from 21.75° C to 26.5° C (Fig. 4.4b), and from 18.75° C to 22.25° C (Fig. 4.4c). iLTC sets the room temperature based on the common comfort range of the occupants and the outdoor temperature, whereas a fixed temperature set-point strategy selects a fixed temperature for rooms irrespective of the comfort preferences of the occupants. However, we assume that the fixed set-points also vary between 21° C to 23° C from winter to summer months. Both the strategies employed occupancy detection before selecting a set-point.

The total energy consumption of the HVAC on a monthly basis is shown in Fig. 4.16. In most of the cases, iLTC incurs significantly less energy consumption than the fixed set-point strategy. However during the winter season in India, iLTC induces more energy consumption. This is because the lower bound of the common comfort range is 21.75° C , where the fixed-point strategy sets the temperature to 21° C . But, when yearly basis energy consumption is calculated, iLTC outperforms fixed-point strategy in terms of lesser energy consumption. In India, the total yearly energy consumption is 6032 kWh and 9881 kWh for the two methods, respectively. On the other hand, in Netherlands, they are 13129 kWh and 9595 kWh, respectively. Thus, iLTC reduces energy consumption by 39% and 27% in the respective cities.

Discussion

As demonstrated in the previous sections, the iLTC system can decide a set-point to reduce energy consumption and to maximize the comfort level for all the co-occupants in a shared space. However, there are a few challenges that need to be addressed: (i) The light intensity values collected from the smartphones of the users may vary due to the heterogeneity of the sensors used by different manufacturers. Hence, the data collected needs to be calibrated to derive accurate user-comfort preferences. Data calibration can be easily performed by comparing the sensed data with baseline sensor data. (ii) In some scenarios, there may not be any common comfort range between the co-occupants in a shared space. It could even be discontinuous when more than two occupants share the space. iLTC then determines a set-point to save energy and also minimize the average discomfort for all the co-occupants. (iii) The efficiency of traditional BEMS can be very low, when there is frequent user movement. Our iterative approach in iLTC for controlling the actuators ensures that frequent movement of users does not affect the overall comfort drastically. (iv) The HVAC model utilized here shows the energy saving considering only the temperature difference between outdoor and indoors, however sophisticated simulation tools can be employed to derive detailed energy savings by considering other building parameters such as duct type, air re-circulation, zone thermal storage, etc. iLTC can take any given model and set the operating point; (v) Since iLTC measures the current light intensity and temperature at a specific space, it is agnostic with respect to building type and the surrounding spaces. It learns the comfort preferences and decides on energy-optimal set-points. Hence iLTC can also be used in large shared spaces with multiple occupants; (vi) User involvement in iLTC is minimal where new users can join and leave the system freely.

Conclusions

We developed an indoor environment controlling system called **iLTC** that offers automated HVAC and lighting control at the room level trying to match individual

user preferences. Instead of choosing a conservative set-point for the actuators that can provide nominal comfort to the occupants in a shared space, iLTC decides a set-point that can be energy optimal while tuning the settings to cater to the comfort levels of all co-occupants. The system learns preferences of each individual based on human perception of comfort through the developed smartphone App. We developed a comprehensive comfort representation function from a few comfort indicators using the collected data, and reduced explicit human intervention. We leveraged the light sensor in the users' smartphones to monitor the received light at their desk in addition to a single reference light sensor for all the users in the building. Thus iLTC reduces the deployment and management costs of multiple light sensors. Results show that iLTC set-point selection can reduce energy consumption up to 39% and 60% by the HVAC and lighting systems, respectively, compared to the fixed set-point mechanism. We evaluated iLTC with 21 participants housed in multiple rooms and our qualitative user evaluation shows over 78% of the participants felt comfortable with the deployed iLTC system. iLTC showcases the utility of the DIAT reference architectural model in Chapter 2. By utilizing the layered concept of DIAT, iLTC not only achieves interoperability among heterogeneous components of the systems, it also provides a scalable solution.

5

Routing for Virtualized IoT

The virtual sensing framework (VSF) achieves energy-efficiency and extends the network lifetime by switching-on only a subset of nodes at any time instant (node-scheduling) and putting the remaining nodes to sleep. In addition to sensing coverage, the node-scheduling scheme must also ensure that (i) the network stays connected, and (ii) the time needed to wake-up the complete protocol stack after sleeping is minimized. We present *Sleeping Beauty*, a highly-efficient data collection protocol that aids node-scheduling schemes in both aspects.

Sleeping Beauty uses a slotted and tightly synchronized communication primitive, where a node keeps its radio off for most of the time, except in the slots when it needs to participate for successful communication. Further, an efficient neighbor-discovery mechanism is included that provides partial, but sufficient topology information (potential parents) to avoid network partitions. Furthermore, Sleeping Beauty employs a novel, yet simple clock-offset estimation technique that maintains highly-accurate time synchronization over long radio-off periods (i.e., less than 500 μs deviation even after 45 min of sleeping). This minimizes time wasted in resynchronizing the network in between data collection rounds. Through experiments on two different testbeds, we verified that Sleeping Beauty decreases the duty cycle up to a factor of 3 compared to state-of-the-art techniques, while achieving similar delivery ratios.

Introduction

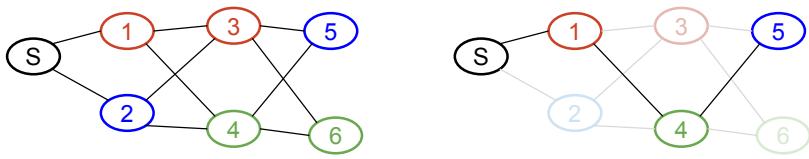
Even after more than a decade of research, energy-efficient communication –and data collection in particular– is still a *holy grail* within the community. Part of

Parts of this chapter have been published as – Sarkar *et al.*,"Sleeping Beauty: Efficient Communication for Node Scheduling", IEEE MASS'16, October 2016 [103].

the challenge is that typical WSN deployments include more nodes than needed to accurately sense the phenomena of interest. This redundancy safe guards, on the one hand, against failing nodes and links, but on the other hand leads to inefficient use of energy. *Node-scheduling* schemes address the latter aspect by keeping only a (minimal) subset of nodes active at any instant. They do so by selecting a representative from each so-called *common sensing group*. A common sensing group can be formed when (i) a common target or area is covered by a set of nodes [21, 76], or (ii) the data generated by the nodes show high degree of correlation such that sensed data for multiple nodes can be reconstructed using data from a single node [44, 117]. In either case, representative node(s) from each group needs to be active during the data collection round. Based on the application scenario, only one or multiple representative (such as k-coverage) nodes are selected active. By alternating duties across rounds, nodes save energy and extend the network lifetime.

As an example, Fig. 5.1a shows a network of six nodes, where only a subset of active nodes is sufficient under the common sensing group regime (indicated by color). Fig. 5.1b shows one such combination. However, if nodes {3, 4, 5} are selected instead of {1, 4, 5}, all groups are still represented, but the network would be disconnected from the sink making it impossible for the application to receive the sensed data.

Given the common sensing groups and network topology of a WSN, the selection of a minimal set of active nodes that guarantees both (a) the connectivity of active nodes to the sink, and (b) the representation of each group by active node(s), is shown to be NP-hard [17, 100]. Thus a number of heuristic algorithms have been proposed in the literature [11, 127] to select close to optimal sets of active nodes satisfying the above constraints. After selecting the active nodes for a round, usually by the centralized sink node, the sensed data from these nodes still need to be collected *efficiently*. That is the main challenge we address in this chapter.



(a) (b)

Figure 5.1: (a) A deployment with multiple correlated sensor groups (denoted by colors), and (b) coverage of the entire monitoring region by a subset of connected active nodes.

Motivation

Different node-scheduling strategies have been adapted based on application requirements, e.g., k-coverage [11], point-coverage [17], spatial correlation [44], etc. In

general, node-scheduling is done either at the application layer and an underlying routing protocol is assumed to ensure data delivery [11, 17] or a joint scheduling and routing is performed [21]. There are some limitations of the existing strategies. In the first case, the routing protocol has to select some additional nodes as relay apart from the chosen active nodes in order to keep the network connected. If the number of relay nodes are not kept minimal, it can diminish the overall efficiency. In the second case, as the cross-layer solution is tightly coupled to the particular scheduling scheme, it cannot be reused for other scheduling strategy without significant adaptation. Thus there is a need for a generic data-collection algorithm that ensures highly-efficient network operation irrespective of the node-scheduling policy.

Solution approach and challenges

In this chapter, we present Sleeping Beauty, an energy-efficient communication protocol for node-scheduling scenarios. It uses a slotted and tightly synchronous operation among the nodes, and deploys a pull-based mechanism to collect data from every active node when the sink requires it (based on application requirement). This ensures a highly compact radio-on time of the nodes and increases radio-off time in-between two sensing rounds. The efficient operation of Sleeping Beauty is rooted in the fast-flooding mechanism provided by Glossy [37]. Like LWB [36], which is also based on Glossy, Sleeping Beauty contains a central part controlling what happens in every slot. In order to be flexible and efficient, the control mechanism runs at the sink and has been designed to address the following challenges. First, as the inefficiency of multi-hop communication in WSN caused by retransmissions and per-hop delays, repercussion of these factors need to be restrained. Moreover, in every communication slots, only those nodes that provide routing progress need to be involved while maintaining minimal retransmission and delay. Second, as the network topology varies over time (e.g., due to link quality fluctuations), the sink should account for this and ensure that a valid path exists from any active node to the sink at all times. The challenge is to provide neighbor discovery at minimal cost. Third, since the efficiency of Sleeping Beauty stems from time-triggered operations, a tight synchronization among the nodes needs to be maintained, even when nodes go to sleep for long times. If not, what was gained in node-scheduling will be lost in trying to synchronize again. In particular, standard node-scheduling schemes assume that once nodes wake-up they can instantly join the network. In reality, however, waking up the complete protocol stack can be time consuming. Thus, zero/minimal overhead should be ensured when a node rejoins the network after long periods of radio inactivity.

Contributions

Sleeping Beauty gains efficiency by tackling the challenges mentioned above. Specifically, we improve state-of-the-art with these contributions:

Efficient data collection for node scheduling
We provide the design and evaluation of a communication protocol that combines fast flooding and on-off scheduling to support scenarios where only a small, representative set of nodes need to report their data to the sink. We do so by separating the concerns of node selection and data collection. In that respect Sleeping Beauty is an efficient, slot-based communication substrate supporting different node selection schemes.

Efficient neighbor discovery
Neighborhood discovery is a required service for many WSN applications. We propose a synchronized strobing mechanism that helps every node to learn about its neighborhood efficiently, that is, with minimal radio activity. This is achieved by having nodes focus only on potential parents and update information instead of discovering it every time from scratch. This aids node-scheduling algorithms in coping with network connectivity and dynamics.

Accurate and long-lasting time synchronization
Much of Sleeping Beauty's efficiency stems from the fact that operations are tightly synchronized. To counter the drifts of the hardware clocks in nodes, Glossy (who pioneered this style of communication) utilizes frequent network-wide floods, taking up a significant amount of energy. To avoid this overhead we propose a simple clock estimation algorithm with which nodes – even after 45 min – are still synchronized within an error bound of 500 μs.

We substantiate our claims by evaluating the performance of Sleeping Beauty on two public testbeds –Indriya [29] and FlockLab [66]– and on a local setup. We compare against two Glossy-based communication protocols LWB [36] and Forwarder Selection [18], showing overall reductions in energy consumption, as well as providing detailed insights into the inner working of our protocol.

Background

In this section, we review state-of-the-art data collection protocols with respect to their limitations for node scheduling. The Collection Tree Protocol (CTP) [40] is a robust data collection protocol that builds and maintains a spanning tree. It provides a good packet delivery ratio for many deployments, but breaks down under increased traffic and network dynamics. To address the latter shortcoming, Landsiedel *et. al* [61] proposed an opportunistic data collection protocol, called ORW, which routes packets to the first available node from a set of possible forwarders. Although ORW outperforms CTP in many aspects, it shares the overhead of maintaining routing metrics, which compromises efficiency, especially when sets of nodes are switched on and off.

Recently, a number of synchronous all-to-all communication protocols have emerged as highly efficient as that of the traditional tree-based protocols [36, 37, 60]. Most of these protocols utilize Glossy's fast flooding mechanism, which we will describe in quite some detail as Sleeping Beauty is also based on it. In Glossy [37], a special node, called the initiator (sink node), starts the flooding in a time-triggered fashion. The overall flooding time is very small compared to traditional flooding [64], as Glossy embraces a phenomenon called constructive interference (CI). When more than one node transmits the same content exactly at the same time (within a tolerable time difference, usually half of a symbol period), the signals constructively interfere at the receiver. As a result, the receiver can successfully decode the symbols. In Glossy, each node turns on its radio just before the start of a new flood, and after the initiator sends the packet, all its first-hop neighbors receive the packet. Then they forward the packet immediately (with the same processing and switching delay), which causes constructive interference at the second-hop nodes from the initiator. The second-hop nodes then forward the packet immediately too. The process continues as a ripple effect and the whole network is flooded within a couple of milliseconds. Nodes turn off their radio immediately after the flooding. Note that as a result of participating in the flood, the nodes implicitly become synchronized with the initiator the moment they receive the packet.

The Low-power Wireless Bus (LWB) protocol [36] has turned the Glossy floods into a generic all-to-all communication primitive by adding a centralized component that creates a global schedule specifying who may initiate a flood (i.e., send data) in which slot. By having all nodes participate in every flood, LWB is completely topology agnostic and voids the need for obtaining/maintaining routing information, allowing it to function as a robust and efficient communication substrate. For data collection, however, it is not necessary to relay the messages at all nodes, providing room for optimization. For example, Carlson *et al.* [18] describe a *forwarder* selection mechanism for LWB that is quite effective in only involving a small set of nodes between source and sink. For node scheduling, however, this solution is inadequate as (i) nodes cannot go to sleep for extended periods of time without losing synchronization, and (ii) the schedule reserves slots for the inactive nodes, causing all nodes to waste energy by idling. These considerations prompted us to design a new protocol.

A first look at Sleeping Beauty

We begin our description of Sleeping Beauty with a conceptual overview depicted in Fig. 5.2. The sink coordinates all actions by periodically flooding the network with sync packets. Once a node has received such a sync packet it may join the network by requesting its own slot in the global schedule. It can then start reporting its sensed data to the sink every inter-packet interval (I). However, it may only do so if it is one of the active nodes determined by the sink every scheduling interval

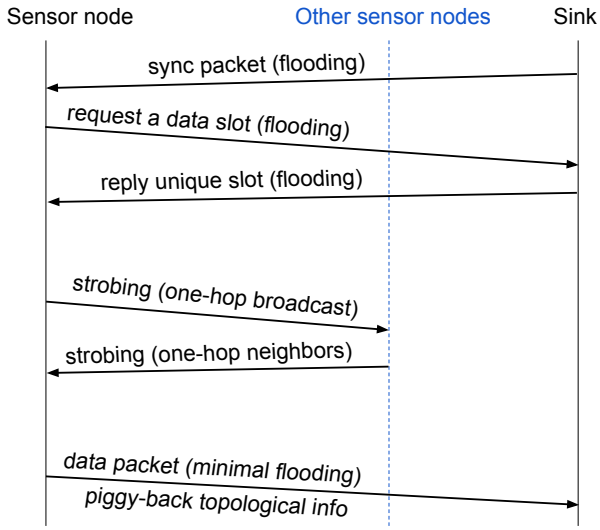


Figure 5.2: Conceptual overview of Sleeping Beauty.

(S). The sink disseminates this set and an accompanying schedule along with the sync packets. The selection of the connected set of active nodes is based on basic one-hop neighbor information (possible parents) collected by the nodes through localized strobing. Unlike the sync slot, where all nodes participate in the flooding, only the set of active nodes participate in the flooding during the data slots. This minimal flooding ensures higher energy efficiency as compared to other Glossy-based protocols.

Design goals

The overarching objective is to enable inactive nodes to sleep for prolonged periods of time, while utilizing Glossy's fast and efficient flooding primitive. This amounts to the following two design goals. **First**, we want an efficient neighbor discovery mechanism to muster partial, but sufficient topological information. In particular, we are interested in providing lists of potential parents to the active node selection algorithm running at the sink so that any node scheduling scheme can be applied on the network. **Second**, we desire a node synchronization scheme that allows nodes to be offline for extensive periods of time while maintaining the desired level of tight synchronization. That is to say that the requirement of frequently sending synchronization packets by a Glossy-based protocol is to be replaced by a method that runs at much larger (and even irregular) intervals.

Architecture

The operation of Sleeping Beauty is divided into two major phases. In the first phase, called *bootstrapping*, nodes join the network by synchronizing with the sink and requesting for data slots. During this phase nodes also estimate their own clock drift parameters, and survey the local neighborhood to identify high-quality links with nodes that can provide routing progress to the sink. This topological information is relayed to the sink by piggybacking with the data packet. In the second phase, called *steady-state*, the sink selects a new set of active nodes in every scheduling period. Once the selection is made and communicated to all nodes in the network, only the selected will stay active until the next scheduling round. Once the round finishes, all nodes wake-up again and topological information is refreshed to accommodate for any changes in network conditions, before the sink selects the next set of active nodes.

To achieve efficient data communication, Sleeping Beauty requires nodes to perform various secondary communication activities, e.g., neighbor discovery, offset estimation, etc., as outlined in Section 5.5. These secondary activities can lead to significantly-higher energy consumption by the nodes if they are not done efficiently. Therefore, the sink defines a superframe that reduces the average duty cycle by carefully managing the radio on time of all nodes.

Superframe as a building block

To cater to various types of activities, a superframe in Sleeping Beauty may consists of four different types of communication slots, as shown in Fig. 5.3. A slot is of sufficient length to flood a packet from an initiator node (who starts the flooding within this slot) to all other nodes, including time for processing the contents at the receiver(s). The total duration of a superframe is equal to 1 s and I s during the bootstrapping and steady state, respectively.

Synchronization slot

This is the most important slot in a superframe, and it functions as the header of the superframe. This slot is used for (re)synchronizing the nodes with the sink. Thus only the sink can initiate transmission of a sync packet, and there is only one sync slot in a superframe. Though synchronization is performed based on the reception time of the sync packet, the content of this packet specifies the lengths of the other slots (avoiding unnecessarily fixing a large slot time). Additionally, a sync packet also disseminates the list of active nodes that are selected based on a node scheduling policy. How a sync packet is processed to decide the active set of nodes is discussed in Section 5.5.1.

Request/Reply (RR) slots

Every node needs to acquire a data slot to send its data to the sink. RR slots are used to request a data slot (odd numbered RR slot) and subsequent granting (even

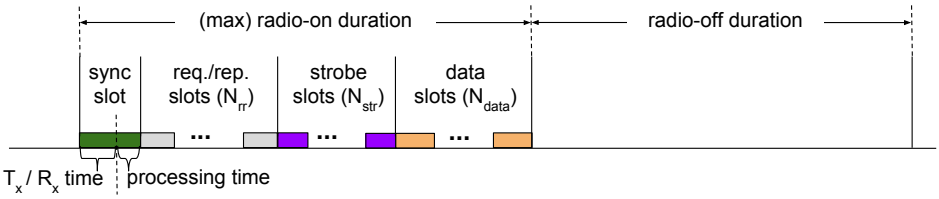


Figure 5.3: Superframe (SF) structure that contains a radio-on and radio-off duration. Note that not all parts depicted need to be present (cf. Fig. 5.4).

numbered RR slot) If two nodes send requests in the same slot, most of the times the sink is able to receive a single request message successfully either due to the capture effect [63] or one of the contending nodes is at least one hop closer to the sink compared to the others.

5

Strobe slots
These slots are used to notify the presence of a node to its neighboring nodes as described in Section 5.4.2. The number of strobe slots is equal to the total number of nodes in the network to rule out collisions.

Data slots
Every node is assigned a unique data slot to deliver its data, which it obtained using the RR slot. Additionally, a list of potential parent nodes that were discovered using the strobe packets is also piggybacked in the data packet. This provides a partial, but sufficient topological view of the network to the sink.

Bootstrapping

During bootstrapping nodes perform various learning activities, which are utilized for operational optimization and duty cycle reduction in the long run. Even during bootstrapping, nodes start reducing radio-on time as soon as they complete the required level of learning. Moreover, the duration of bootstrapping is very insignificant (only a couple of minutes) compared to that of the steady state, which can last for months if not years. The following three major tasks are performed during bootstrapping.

Node joining

As every communication is time triggered and synchronous, every node needs to synchronize itself with respect to the sink before starting any transmission. Thus, after a node is powered on, it keeps listening for sync packets. As soon as the node receives such a packet, it synchronizes with the network and learns about the superframe structure. As mentioned earlier, a node requests for a data slot and gets a reply from the sink in the odd and even RR slots, respectively. Once a

node has obtained a slot, it stops sending any further requests, and completes its joining procedure. However, it keeps participating in all other RR slots to help with delivering the request/reply messages to the intended recipients.

It is clear that if the number of RR slots is low and the number of requesting nodes is large, then it will take a long time before every node gets a dedicated data slot. On the other hand, if there are more RR slots, more nodes can get their request granted within a short period of time. However, this implies that nodes will be wasting a significant amount of energy during the RR slots. Sleeping Beauty uses a dynamic approach to decide the number of RR slots.

Deciding the number of Request/Reply slots

At the beginning, Sleeping Beauty starts with 48 RR slots (the maximum that fits within one second). Thus, within a second a maximum of 24 nodes can be allotted a data slot. When most of the nodes are allotted a slot, many unused RR slots cause unnecessary energy consumption at the nodes. Sleeping Beauty notes the number of used RR slots within the current superframe, and uses this information to adjust the number of RR slots in the next superframe. After some time, the number of RR slots is reduced to 2. By that time, most of the nodes, if not all, would have successfully acquired a data slot. The sink then decides to move to the steady state (see Section 5.5).

Building a partial view of the network

To gather a partial, but sufficient topological view of the network at the sink, nodes piggyback a list of potential parents with the sensed data. The ETX metric is used to identify these potential parents from the beacons transmitted in the strobe slots. To avoid any collisions among nodes, a separate strobe slot is allocated to each node as follows.

When a node is assigned to transmit in the t^{th} data slot, it shall use the $(t+1)^{th}$ strobe slot to send its strobes (the first strobe slot is reserved for the sink). In its slot, a node sends a fixed number of consecutive strobe packets, which contain the current ETX value of the transmitter. A neighboring node estimates the link quality between the sender and itself based on the number of received strobe packets and updates its own ETX value. Based on the ETX values of all neighbors, a receiver node compiles a list of potential parent nodes. Unlike CTP, where a node chooses only one parent exclusively based on the best (minimum) ETX value, Sleeping Beauty maintains a list of parents. If a long list of parents would be reported with the data packets, a significant amount of energy would be spent by the nodes for delivering larger payloads. Therefore, only a small list of parent nodes with better ETX values is reported to the sink. This way the sink can gather partial, but sufficient topological information about the network. From our empirical evaluation, see Section 6.7, we inferred that a list of 5 nodes is sufficient for 97% cases.

Clock-offset estimation

Due to unstable clocks of the small embedded devices involved, a node can experience a significantly-high clock offset with respect to the reference node (i.e. the sink) within a small period. To rectify such clock offsets, the common method is to send synchronization packets more often and individual nodes should put in extra efforts, such as keeping the radio on for a longer duration to receive the synchronization packets. This induces significantly-higher energy consumption by the nodes. In Sleeping Beauty, during bootstrapping, frequent sync packets are sent so that sufficient data points can be collected to estimate the clock offset. Once in steady state the clock offset is rectified by the node itself without receiving sync packets as will be described in Section 5.6.

During bootstrapping, every node in the network remains active and senses data every inter-packet-interval (I s) as determined by the application. Therefore the superframe contains as many data slots as the number of nodes in the network. The bootstrapping process is ended T s after the last node joins the network. The timeout T should be set such that the sink has sufficient time to acquire the topological view of the network and that the regular nodes can gather enough data points to estimate their clock offset.

Steady-state operation

The beginning of the steady state is marked by selecting a set of active nodes. The sink disseminates the list of active nodes using the sync packet. It selects a new set of active nodes every scheduling period, which amounts to $S * I$ s (Fig. 5.4). Note that the protocol is flexible enough to cope with any value of S and I as sought by the application.

Periodic active node selection

Since active node selection is outside the scope of this chapter, we used an existing node scheduling algorithm on top of Sleeping Beauty to demonstrate how it can be integrated with such methods. The common sensing groups are assumed to be known or can be formed at runtime based on the sensed data. Then the active nodes are selected by means of the algorithm described in [100]. The list of active nodes is disseminated using a bitmap in the sync packet. If, upon receiving a sync packet, a node finds a zero at the bit position of its data slot, it goes to sleep (dormant node). The active nodes continue to wake-up every I s to send their data without requiring any sync packet (Fig. 5.4). Note that active nodes keep their radio on in their assigned data slot, to transmit their own data, as well as in the data slots associated with the other active nodes, to forward the data from them.

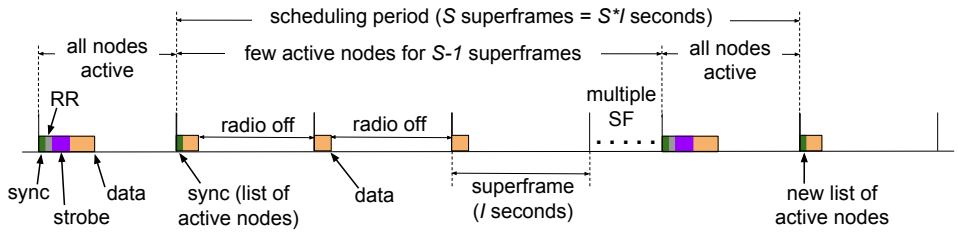


Figure 5.4: Periodic strobe slots and new set of active node dissemination using various superframe structures in the steady state.

After S superframes, the sink reconsiders the selection of the active nodes. Since the quality of the links may have changed during this long time period ($S * I$ s), the last superframe of the series includes strobing slots to reassess the links (see Fig. 5.4). To obtain a complete picture of the (changed) network topology all nodes participate in the link quality assessment procedure. Hence, non-active nodes can sleep for only S-1 superframes.

Parent list update

During bootstrapping, every node listens to every strobe slot to infer its neighbors, and compute its ETX to the sink. The same procedure could be followed in the steady state as well, but would waste a lot of energy in the case of large multi-hop networks that comprise many more nodes than neighbors. Therefore, a node records any neighbor seen during bootstrapping in a bitmap. This bitmap is then used to selectively listen to the strobing slots of potential parents during steady state. This 'smart strobing' optimization significantly reduces the energy consumed for link quality assessment, see Fig. 5.10a for details. To ensure a high Packet Reception Rate (PRR), nodes sort the list of parents based on ETX and picks the top 5 (lowest ETX) as the potential parents (cf. Fig. 5.10b).

Clock-offset correction

Active nodes wake up in every superframe, allowing them to stay synchronized easily. Dormant nodes, in contrast, run the risk of waking up out of phase due to drifts in their clocks when rejoining the network for the link quality assessment after $(S-1)I$ s. The drift could be countered by waking up early, but that would waste energy, so Sleeping Beauty adopts a correction procedure based on estimating the clock-offset parameters, as discussed in next.

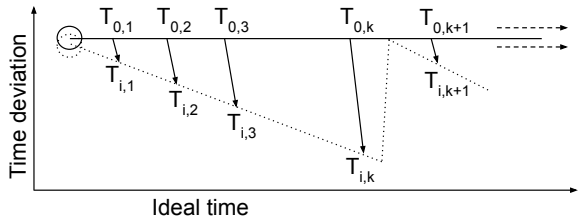


Figure 5.5: Aperiodic one-way communication between sink (solid line) and sensor node (dotted line), and the effect of iterative clock correction at the sensor node at the k^{th} time instant.

Low-cost clock-offset estimation

We consider the scenario shown in Fig. 5.5, where the sink (solid line) transmits packets to the sensor nodes. For the sake of simplicity, we illustrate a single-hop communication link. However, in the case of multi-hop scenarios, given the known additive transmission delay between the hops, a similar illustration holds. Note that time deviation can either be positive or negative with respect to the sink.

At the k^{th} transmission, the sink node transmits the reference time $T_{0,k}$, which is received by node i at its local time $T_{i,k}$. The sink node transmits a packet to the local node i at every δT s for a total time duration of ΔT, during which K transmissions occur. It is worth noting that the communication between the nodes is not necessarily at a fixed interval, which allows us to dynamically vary the duty cycle of communication δ between the nodes as per our requirement.

Least squares estimator

All the clocks are inherently non-linear, however given sufficiently low *Allan* deviation for a short period of time, the clock model could be linearized to fit a first-order model [89]. Under the assumption that the sink node is our reference node, the corresponding local time at node i is,

$$T_{i,k} = (1 + \dot{\phi}_i)T_{0,k} + \phi_i, \tag{5.1}$$

where $\{\dot{\phi}_i, \phi_i\}$ are the frequency offset and phase offset of node i [106]. Under ideal conditions, $\{\dot{\phi}_i, \phi_i\} = \{0, 0\}$ and subsequently $T_{i,k} = T_{0,k}$, however in practice these clock errors are prevalent and the challenge is to estimate and correct the local clock at node i appropriately. Furthermore, the time deviation at node i with respect to the reference clock is,

$$\epsilon_{i,k} \triangleq T_{i,k} - T_{0,k} = \dot{\phi}_i T_{0,k} + \phi_i. \tag{5.2}$$

Now, collecting all the K transmissions and using (5.2), the unknown clock

coefficients $\boldsymbol{\theta}_i \triangleq [\dot{\phi}_i, \phi_i]$ can be estimated by solving for,

$$\hat{\boldsymbol{\theta}}_i = \arg\min_{\boldsymbol{\theta}_i} \|\mathbf{A}_K\boldsymbol{\theta}_i - \boldsymbol{\epsilon}_i\|_2^2 \tag{5.3}$$

$$= \left(\mathbf{A}_K^T\mathbf{A}_K\right)^{-1}\mathbf{A}_K^T\boldsymbol{\epsilon}_i = \mathbf{G}_K^{-1}\mathbf{b}_{i,K}, \tag{5.4}$$

where $\mathbf{A}_K = [\mathbf{t},\ \mathbf{1}_K]$, $\mathbf{b}_{i,K} = \mathbf{A}_K^T\boldsymbol{\epsilon}_i$, $\mathbf{G}_K = \mathbf{A}_K^T\mathbf{A}_K$, $\mathbf{1}_K$ denotes a column vector of K ones and the measurement vectors are of the form,

$$\mathbf{t} = \left[T_{0,1} \quad T_{0,2}, \ldots, T_{0,K}\right], \tag{5.5}$$

$$\boldsymbol{\epsilon}_i = \left[\epsilon_{i,1} \quad \epsilon_{i,2}, \ldots, \epsilon_{i,K}\right]. \tag{5.6}$$

Here $\hat{\boldsymbol{\theta}}_i$ is an estimate of the true clock parameters and (5.3) has a feasible solution provided $K \geq 2$ [90]. Observe that the matrix $\mathbf{G}_K$ is dependent only on the time stamps from the reference sink node $\mathbf{t}$. Hence, in case the (possibly varying) polling-interval δt is known advance, then the inversion $\mathbf{G}_K^{-1}$ can be estimated offline and stored locally, at the cost of more memory.

5

Iterative Least Square update

For a set of K time measurements, the number of multipliers to solve the least squares (LS) solution of (5.3) is $\mathcal{O}(2K)$, where the inversion of the Grammian matrix $\mathbf{G}_K$ is the most expensive operation. However, since the measurements arrive sequentially, the proposed least squares estimator can be solved row-iteratively [104]. Let $\mathbf{a}_k^T$ and $\boldsymbol{\epsilon}_{i,k}^T$ denote kth row input (during the k^{th} transmission) of $\mathbf{A}$ and $\boldsymbol{\epsilon}_i$, respectively. Then, solving (5.3) for $2 < k \leq K$ is equivalent to iteratively solving $\mathbf{G}_k^{-1}\mathbf{b}_{i,k}$, where the k^{th} update is given by,

$$\mathbf{G}_k^{-1} \triangleq (\mathbf{G}_{k-1} + \mathbf{a}_k\mathbf{a}_k^T)^{-1}$$

$$= \mathbf{G}_{k-1} - \frac{\mathbf{G}_{k-1}^{-1}(\mathbf{a}_k\mathbf{a}_k^T)\mathbf{G}_{k-1}^{-1}}{1 + (\mathbf{a}_k^T\mathbf{G}_{k-1}^{-1}\mathbf{a}_k)}, \tag{5.7}$$

$$\mathbf{b}_{i,k} = \mathbf{b}_{i,k-1} + \mathbf{a}_k^T\boldsymbol{\epsilon}_{i,k}. \tag{5.8}$$

The initial estimate at $k = 2$ is obtained by solving the 2×2 linear system $\mathbf{G}_2^{-1}\mathbf{b}_{i,2}$, where

$$\mathbf{G}_2 = \mathbf{A}_2^T\mathbf{A}_2 = \begin{bmatrix} G_{1,1}, & G_{1,1} \\ G_{2,1}, & G_{2,2} \end{bmatrix}, \tag{5.9}$$

$$\mathbf{G}_2^{-1} = \frac{1}{\det(\mathbf{G}_2)}\begin{bmatrix} G_{2,2}, & G_{1,2} \\ -G_{1,2}, & G_{1,1} \end{bmatrix}, \tag{5.10}$$

$$\mathbf{b}_{i,2} = \mathbf{A}_2\boldsymbol{\epsilon}_{i,2}. \tag{5.11}$$

As the iterative-LS update is very inexpensive in terms of memory and CPU cycles, it can be implemented on any embedded device. We integrated the above clock-offset estimation technique in our Sleeping Beauty implementation.

Performance Evaluation of Sleeping Beauty

We evaluate the performance of Sleeping Beauty based on the overall energy consumption by the nodes in a WSN. In this regard, we study the reduction in duty cycle of the nodes over a longer period. Additionally, we monitor the performance in terms of packet reception ratio (PRR).

Implementation details

Sleeping Beauty is implemented on the Tmote Sky platform using the Contiki operating system. We use the core functionalities of Glossy to obtain the precise clock synchronization using fast flooding. We further adopted from our previous implementation of LWB and FS-LWB [96]. As Glossy's implementation is bound to the Tmote Sky platform, the current implementation of Sleeping Beauty is also available for this platform. However, Glossy can easily be ported to other platforms as discussed by its developers. As Sleeping Beauty does not have any platform-specific component, it becomes directly usable on any other platform where Glossy will be ported to.

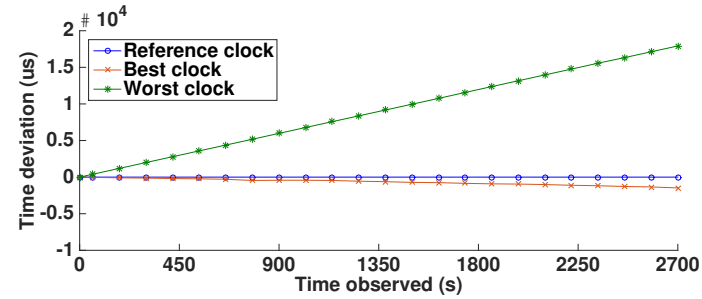
There are a few configuration parameters in Sleeping Beauty that can be tuned based on the application's requirements. Without loss of generality, we use the following values during our evaluation – the inter-packet interval I is set to 10 s, the scheduling interval S to $10I$ s (i.e., 100 s), and the time-out period T to end bootstrapping is set to 120 s.

Evaluation platforms

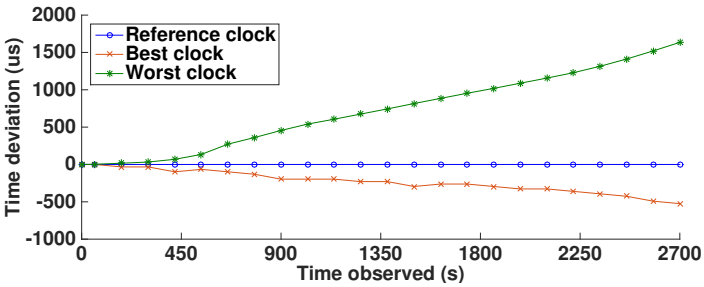
We present the results of a study carried out on the Indriya [29] and FlockLab [66] testbeds along with experiments using some local nodes in our lab. Indriya is a sensor network with a total of 139 nodes spanning three floors of a building at the National University of Singapore. Our experiments were conducted on 80 TelosB nodes. FlockLab is a wireless sensor network (WSN) testbed at the Swiss Federal Institute of Technology, Zurich in Switzerland and currently it has 32 Tmote Sky devices. We performed two separate sets of experiments: one to evaluate the clock-offset estimation and another to evaluate the overall performance of Sleeping Beauty.

Accuracy of clock-offset estimation

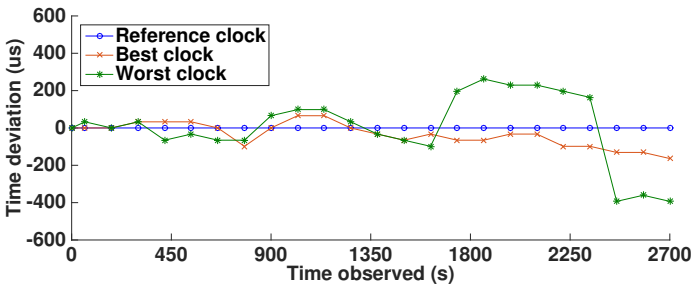
First, we discuss the results related to the clock-offset estimation and correction. As the clocks on different nodes behave differently, we present the clock behavior of two nodes that showed the minimum (best) and maximum (worst) offset over time with respect to the reference node (sink) in Fig. 5.6. The clock of the first node is running slower than the reference node as the observed deviation is on the negative side. The clock of the second node instead is running faster than the reference node (Fig. 5.6a).



(a) clock behavior with respect to the reference clock



(b) offset estimation error with fixed parameters



(c) offset estimation with parameter updates

Figure 5.6: Behavior of two clocks (best and worst) with respect to the reference clock.

Using 120 training samples, the clock-offset parameters are estimated, where sync packets are sent every second. In the evaluation phase, sync packets are sent at various (longer) intervals. Before receiving the next sync packet, the node estimates the clock offset and adjusts its clock. Later, in steady state, when a sync packet is received the offset is calculated, if any. This offset denotes the estimation error. Note that during the evaluation, clocks are always readjusted with respect to the reference clock after receiving a sync packet.

Fig. 5.6b shows the estimation error for various synchronization periods. For each synchronization period, multiple measurements were performed and an average value

is plotted in the figure. From this data, it is clear that the estimation error increases when the synchronization period increases. This behavior is due to the non-linear nature of the clock and the small set of the training data. The estimation error can be contained if the offset parameters are also updated during the evaluation phase after receiving each sync packet. Fig. 5.6c shows an improved clock correction and the estimation error is smaller compared to Fig. 5.6b. However, even in this case, the estimation error increases with respect to the synchronization period, though at a much lower rate.

Glossy maintains high synchronization accuracy among the nodes by transmitting sync packets frequently enough such that the clock offset of the nodes remain within a 500 μss bound. Based on the experiments on Indriya, we discovered that some of the nodes become asynchronous even if a sync packet is sent as often as every 10 s. Thus, we assumed that to maintain synchronization for all the nodes sync packets need to be sent in 5 s intervals. However, if the application sends data at larger intervals, a lot of sync packets will be exchanged just to keep the network synchronized. Using our clock-offset estimation technique, a sync packet can be sent once in every 45 min, while the clock offset will be within the predefined threshold of 500 μs (Fig.5.6c).

Sleeping Beauty's performance

Next we study the performance of Sleeping Beauty. As the number of common sensing groups in a WSN can vary significantly, we experimented with different sets of groups, and assumed that only one active node from each group is sufficient. Specifically, we experimented with four different group settings such that there are the following percentages of active nodes: 6.25%, 12.5%, 25%, and 50%. That means, there are respectively 4, 8, 16, and 40 groups on Indriya. In case of FlockLab, there are 2, 4, 8, and 16 groups.

We compare Sleeping Beauty (SB) with two state-of-the-art protocols – LWB [36] and FS-LWB [18]. As the radio is the most significant energy consuming activity of a node, we compare the radio-on time (duty cycle) with both these protocols. Figures 5.7 and 5.8 show the radio-on time and packet reception rate averaged across all nodes of the Indriya and FlockLab testbeds using: (i) LWB; (ii) FS-LWB, and (iii) Sleeping Beauty for the four different numbers of source nodes. In all but one case Sleeping Beauty outperforms (FS-)LWB in terms of duty cycle, up to a factor of 3, while achieving comparable packet reception rates. The savings in energy consumption are most pronounced when the least number of source nodes are active. This matches intuition as only then Sleeping Beauty can put nodes to sleep for prolonged periods. The observed PRRs show that most of the data is delivered at the sink; only on the Indriya testbed some packets are lost. There is, however, no clear winner amongst the three protocols. Depending on the number of source nodes, Sleeping Beauty looses more or fewer packets than (FS-)LWB. We conjecture that the un-

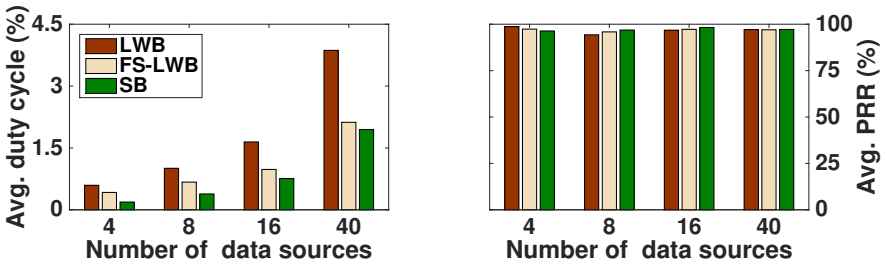


Figure 5.7: Experiments on Indriya: average duty cycle per node over a long period and average packet reception ratio.

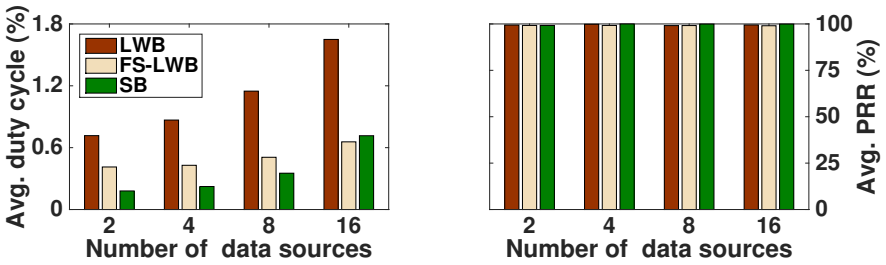
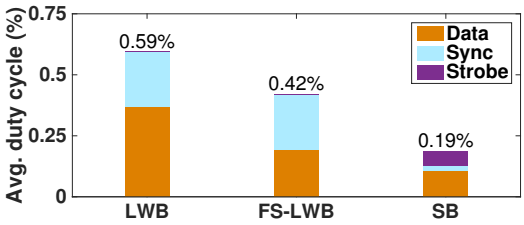


Figure 5.8: Experiments on FlockLab: average duty cycle per node over a long period and average packet reception ratio.

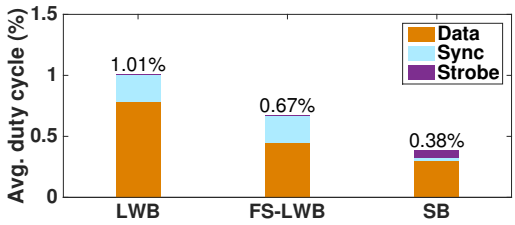
derlying cause for the drop in PRR is the constructive interference (CI) mechanism, whose effectiveness depends on the number of simultaneous transmitters. With too few transmitters nothing is gained; with too many, a slight mis-alignment at the symbol granularity leads to packet loss. Depending on the number of active sources, a different protocol strikes the CI sweet spot.

To gain a deeper understanding of how Sleeping Beauty can provide the same service (PRR) while consuming less energy, Fig. 5.9a shows the average radio-on time broken down into the main activities of the protocols (handling data, sync, and strobe packets). In LWB, even if there are a few data sources and few data slots associated with them, all nodes participate in all data slots. FS-LWB reduces the time spent in handling data packets, by using only a subset of the nodes in a data slot associated with a particular data source. Because of the more refined topology information collected by Sleeping Beauty, it is capable of putting even more nodes to sleep, reducing the average time spent on handling (forwarding) data even further. For example, when 8 source nodes are active, LWB spends 0.78 percentage points on handling data packets, while FS-LWB reduces that to 0.44 and Sleeping Beauty only requires 0.30 percentage point.

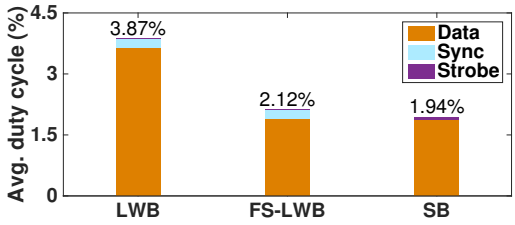
Besides gaining efficiency by flooding less data, Sleeping Beauty also benefits greatly from the improved synchronization method. LWB and FS-LWB spend about 0.23 percentage point of their duty cycle on handling synchronization packets to



(a) 4 source nodes



(b) 8 source nodes

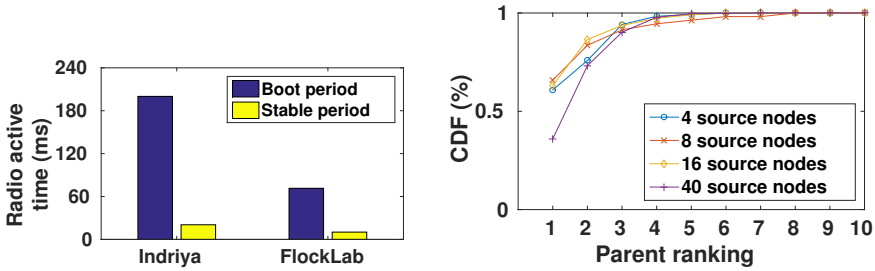


(c) 40 source nodes

Figure 5.9: Experiments on Indriya: average radio-on time for nodes for various operations of Sleeping Beauty (SB).

maintain Glossy's $\leq 500\mu$s timing requirements. Sleeping Beauty on the other hand, spends only 0.02 percentage points on handling sync packets. This coarsely matches the ratio of injection rates, with (FS-)LWB sending out sync packets once every 5 s, and Sleeping Beauty synchronizing once every 100 s.

Sleeping Beauty has some extra overhead (0.06 percentage points) in the form of strobe packets, which are are used to collect topology information. This amounts to a total synchronization cost of 0.08 percentage points, which still compares favorably to that of (FS-)LWB (0.23). The importance of the 'smart strobing' policy, which only updates ETX values for potential parent nodes (cf. Section 5.5.2), becomes clear when comparing the overhead of Sleeping Beauty during bootstrapping and steady state. Fig. 5.10a shows that listening to all nodes in the network (bootstrapping) is 7 to 10 times more expensive than just monitoring the potential parent nodes (steady state) on FlockLab and Indriya, respectively. This factor matches with the



(a) (b)

Figure 5.10: (a) Strobe overhead during bootstrapping and steady state. (b) Active parent distribution across the ranked list.

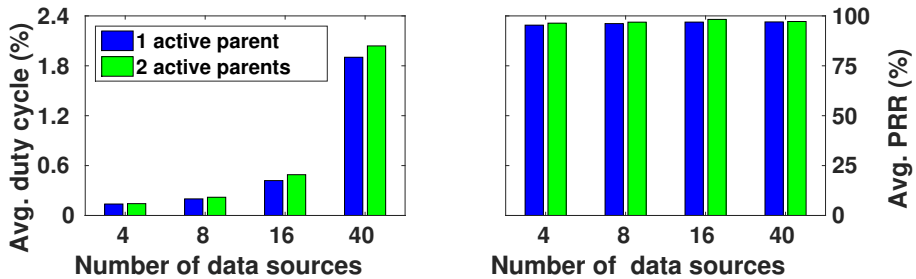


Figure 5.11: Comparison of average radio-on time and PRR when one and two parent nodes are selected active.

ratio of the total number of nodes in the testbed over the average number of parent nodes: $\frac{32}{4.5} = 7$ for FlockLab and $\frac{80}{8} = 10$ for Indriya.

The collected neighborhood information is reported back to the sink, but only partially to limit the overhead. Sleeping Beauty orders the neighbors based on their ETX values and sends out the top-X entries with the shortest routes to the sink. It is important to report the right number of potential parents as, on the one hand, the overhead is directly proportional to it, and on the other hand, the sink can make better decisions on which nodes to activate if it has more complete knowledge of the topology. To study this trade off we conducted an experiment in which we configured nodes to report the top-10 list with potential parents. We then monitored which parents were then selected by the sink for the next scheduling round. Fig. 5.10b shows the cumulative distribution of the selected, ranked parents for different numbers of active sources on the Indriya testbed. In 97% of the cases the sink selected 5 or fewer parents to become active, prompting us to conclude that lists of 5 parents provide sufficient topology information for the sink to build an efficient data-collection overlay.

Knowing that the overall efficiency of Sleeping Beauty is mainly dictated by the number of active nodes, it is tempting to construct overlays where each (source) node has only 1 active parent. However, such sparse overlays are likely to be susceptible to packet loss induced by errors on the wireless channel. As the underlying fast-flooding mechanism relies on exercising all available links, it could be advantageous to activate a few redundant (parent) nodes. To study the PRR-redundancy trade off, we experimented with selecting one or two parent nodes. Fig. 5.11 shows the difference in PRR and duty cycle for both cases on Indriya with different numbers of active source nodes. To our surprise the PRR is hardly affected. Studying the single-parent overlays in detail, we noted that in many cases selecting the best parent of one node, as a side effect, also implied activating the 2nd or 3rd best parent of other nodes. The effect of selecting 1 or 2 parents is a bit more pronounced for the duty cycle, but still limited, showing that either option is fine.

5

Conclusions

To provision against adverse network conditions and node failures, WSN deployments typically contain redundant nodes. Node-scheduling exploits this feature by limiting the number of active nodes to achieve energy-efficient network operation without violating the coverage requirements of the application. However, it is not sufficient to just limit the number of active nodes; active nodes should form a *connected* (sub)network. There exists a large body of work on selecting a connected subset of nodes covering the complete deployment. These optimized node-scheduling techniques, however, are generally not applicable to real-world deployments as they require complete topological information, which is difficult and expensive to obtain accurately.

We presented Sleeping Beauty, an energy-efficient communication protocol that operates with partial topological information, yet outperforms state-of-the-art flooding-based protocols (LWB and FS-LWB) in node-scheduling scenarios. Sleeping Beauty accomplishes this by including (i) an efficient neighbor-discovery mechanism that enables the selection of a minimal, but connected set of active nodes, and (ii) a simple, but elegant clock-offset estimation technique that allows nodes to sleep for a long time without the need for explicit resynchronization. The latter is important for allowing the use of efficient time-triggered flooding a la Glossy. Our time synchronization technique can be used in any application. We compared the performance of Sleeping Beauty with state-of-the-art protocols on two public testbeds (Indriya and FlockLab), and showed that the same performance (PRR) can be achieved at a fraction of the energy consumption. We recorded a factor of three reduction for the best case scenario with 5 % active nodes.

6

Real-time Communication

Due to the low deployment and monitoring cost, many Internet of Things (IoT) applications use Wireless Sensor-Actuator Networks (WSANs) typically favoring energy efficiency (lifetime) over latency. However, delay is an important aspect not to be neglected in, for example, process automation, factories of future, and industrial applications in general. These applications require highly reliable data delivery with strict real-time guarantees. At the same time, since batteries are labor-intensive to replace, energy efficiency is an important aspect when targeting WSANs. In particular, the classical real-time approach of flow scheduling is hard to implement with energy-constrained nodes.

In this chapter, we present Rapid, a real-time communication protocol that guarantees timely data delivery with – (i) low latency by scheduling a maximum number of end-to-end flows within a short time span, (ii) highly energy-efficient network operations, and (iii) reliable data delivery. Using a smart parallelization technique, Rapid achieves simultaneous transmissions while ensuring guaranteed data delivery. This reduces the average duty-cycle of the nodes and makes it a superior communication protocol in terms of energy efficiency as compared to state-of-the-art sensor network communication protocols. By combining multiple routing strategies, Rapid not only simplifies the schedulability problem, but it is also able to accommodate more flows within a time span as compared to current standard like WirelessHART. Specifically, Rapid offers delay and duty cycling reduction of 2.2 and 2.8 times, respectively over LWB and FS-LWB.

Introduction

Many IoT installations have stringent requirements on reliability in terms of time-bound guaranteed data delivery. For example, the process industry has installed

more than 40 million sensor and actuator devices around the world operating feedback control loops in the time scale of tens to hundreds of milliseconds [105]. Since IoT devices are typically battery operated, there is an equally strong requirement on energy efficiency to keep operational costs down. As classic WSN routing and MAC protocols trade-off performance for energy efficiency, there is a need for a new set of real-time communication protocols that strike a (different) balance between latency and energy consumption meeting the demands of IoT applications.

Duty cycling is the standard approach to reduce energy consumption by periodically putting sensor devices to sleep, thereby extending their lifetime to years vs. days when running them flat out [19]. To keep design complexity low, many WSN-specific MAC –medium access control– protocols apply duty cycling in an asynchronous fashion. This, however, impacts end-to-end latency as packets will be delayed at each hop through the network to wait for the next node to wake up. Therefore, real-time protocols employed slot-based synchronous communication instead, where a slot is defined as a small time period to forward a data packet from one node to another. For example, the WirelessHART standard uses a TDMA-based medium access control with 10 ms slots [110]. Energy saving can be achieved if a node activates its radio transceiver in a slot if it is supposed to transmit or receive in that slot, where a global scheduler decides sender/receiver pair for each slot. That in itself introduces two problems: (i) the schedule must be computed, which is non-trivial, and (ii) resources are wasted when nodes have no data to report/forward in their slots. Let us elaborate.

To craft an efficient route (i.e., a staggered set of slots) between a source (sensor or sink) and a destination (sink or actuator) node the network topology must be known. Collecting, and keeping an up-to-date status of links between neighboring nodes is quite a challenge as environmental factors can have detrimental effects on packet reception rates, and it may fluctuate heavily. Worse, scheduling multiple flows (i.e., source-destination pairs) is known to be an NP-hard problem [94], and it becomes even more difficult when each flow needs to be completed within a time-bound [93]. We will take a radical approach and avoid flow scheduling altogether.

Another issue is that most of the existing real-time protocols are aimed at periodic traffic (flows) where sensor nodes report their data at regular intervals [85]. However, many IoT applications involve event-driven scenarios, where sensors are read out periodically, but data is reported only when a significant event is detected. For example, a PIR sensor reporting an event when a person enters a room and the lights need to be switched on. Mapping such a traffic pattern to periodic reporting leads to a large waste of resources (bandwidth and energy). An alternative approach is to assign a few open slots to handle event-based traffic. Any node who detects an event, reports its data using these slots. When multiple events are detected, nodes contend within the open slots. As a result, collisions may occur leading to increased reporting delays. In the worst case, that is, many events and few available slots,

the nodes may repeatedly collide with each other without being able to deliver their data, let alone on time. That forces protocol designers to over provision by a large margin.

Approach In this chapter we describe Rapid, a real-time communication protocol that effectively tackles the above-mentioned scheduling and over-provisioning challenges. Rapid combines synchronous, slot-based communication at the link level with advanced flooding and clustering at the network level in a cross-layer approach.

Clustering To limit over provisioning and enhance the efficiency, Rapid employs a hierarchical approach in which nodes report to cluster-heads, who aggregate data from all members into a single packet that is forwarded (flooded) to the controller. This reduces the number of data packets within the network as only the aggregated packets need to travel multiple hops, which in turn reduces the number of slots in the global schedule.

Capture Data collection within a cluster is short ranged, which allows for spatial reuse of slots when seen from the network perspective. Because of the capture effect [63], which dictates that a node will be able to receive the packet sent from the closest source with high probability, Rapid opportunistically orchestrates all clusters to operate in parallel speeding up the data collection process considerably. Here retransmissions are allowed to enhance reliability for safety critical applications.

Constructive interference Rapid employs constructive interference (CI) to quickly flood the aggregated data through the network. Here we capitalize on the success of Glossy [37], which proved that by having nodes broadcast the same message at exactly the same time, nodes in range perceive the interfering radio signals as one and the same message. This use of constructive interference supports fast flooding of information to all nodes in the network, eliminating the need for routing. As each cluster-head is directed to flood out its message in turn –in one slot– the flow scheduling problem is effectively eliminated.

To demonstrate the feasibility of Rapid's novel design leveraging the benefits of clustering, capture and CI, we implemented it for the Contiki operating system and tested it on two testbeds (Indriya [29] and FlockLab [66]). Rapid was demonstrated to achieve up to 2.2 and 2 times lower delay, while consuming up to 2.8 and 3.8 times lesser energy as compared to the state-of-the-art low-power wireless bus (LWB) [36] protocol, which is a flow-scheduling protocol on top of Glossy.

Contributions: The main contributions of Rapid can be briefly summarized as follows:

1. Bounded latency: Rapid provides bounded communication latency without explicitly solving the complex flow-scheduling problems. We prove that total

latency in Rapid is bounded by $\Omega(\sqrt{n})$ and $\mathcal{O}(n)$ slots, where the total number flows in the network is n.

2. On-the-fly-clustering: To the best of our knowledge, Rapid is the first protocol to employ on-the-fly clustering for real-time IoT applications.

3. Parallel operation: Through spatial slot reuse, Rapid achieves parallel communication, which ensures that a large number of flows can be successfully delivered within a given short time period. In other words, more flows can meet their deadline than with the conventional (sequential) communication.

4. Energy efficiency: Rapid efficiently supports both periodic and event-driven traffic (through aggregation). It provides the same delay bound for both types of traffic, while energy waste for event-driven traffic is kept minimal. Moreover, the energy efficiency of Rapid is (much) higher than of existing data collection protocols.

Related Work

Rapid builds upon the foundations laid by energy-efficient communication developed for WSNs and the low-latency protocols developed for wireless sensor actuator networks (WSANs). We will succinctly review the most relevant work from both these areas as the amount of literature is vast. Moreover, as much of Rapid's efficiency is due to the use of clustering, we will also briefly address that topic.

Communication protocols for WSNs are geared towards duty-cycling at the MAC layer [15, 30, 33], and data collection at the routing layer [40, 61, 87]. The former generally sacrifice latency and predictability for a reduction in energy consumption, while the latter are too specific in routing traffic only to one (or few) edge nodes to be used for IoT applications. Even recent protocols, like RPL [115] and ORPL [32], that do support any-to-any routing perform poorly in a real-time context [48].

WirelessHART is the most prominent MAC protocol for WSANs [109]. The standard defines a TDMA structure, but leaves open the scheduling of the slots [110]. In practice, WirelessHART is often combined with source routing [45] or graph routing [65]. Source routing is straight forward to implement, but struggles with changing link qualities as routes need to be recomputed. Graph routing is more robust as it includes alternative routes in the schedule, but pays its price in terms of latency and energy consumption [65].

To ensure time-bounded data delivery, end-to-end node scheduling is exercised hand-in-hand with routing. As node scheduling is an NP-hard problem [94], a number of heuristic solutions have been proposed in the literature. For example, a mathematical model for joint routing and link scheduling is proposed in [108]. A system with schedule construction for time-critical data delivery is developed in [85]. As, scheduling all flows in a network within a certain time-bound is a challenge,

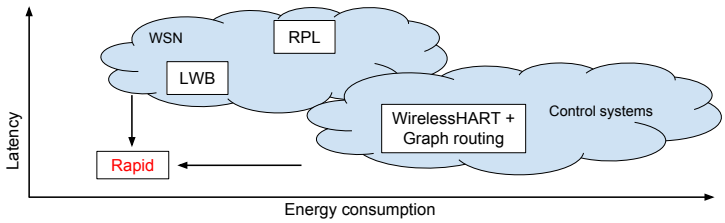


Figure 6.1: Positioning of Rapid in the design space of energy-efficient and real-time protocols.

schedulability analysis under graph routing is studied in [93]. The common drawback of these solutions is their computational complexity.

An alternative approach is to forgo routing altogether and operate the multi-hop network as a shared bus delivering all data to all nodes. The corner stone of this approach is Glossy's fast flooding mechanism that exploits constructive interference [37]. The LWB protocol [36] overlays a TDMA structure on top in which nodes take turns in initiating such an efficient flood. The reduction in complexity (doing away with routing) outweighs the overhead of flooding the whole network in each slot. LWB works remarkably well outperforming traditional tree-based routing by quite some margin. However, it does not scale well to large networks as the average latency grows linearly with the number of nodes in the network. Though LWB is shown to be efficient compared to state-of-the-art data collection protocols, such as CTP [40], RPL [115], and BCP [77], the inherent all-to-all communication pattern of LWB introduces a significant amount of energy overhead on the nodes.

Fig. 6.1 shows how the various protocols fit in the design space for energy-efficient real-time communication protocols spanned by the two dimensions – latency and energy consumption. Rapid has been carefully designed to capitalize on the best practices from the WSN and control domains. The combination of a slot-based approach and (selective) flooding by means of clustering yields a solution that results in both a lower latency and a lower energy consumption.

The usage of clustering has been successfully proposed before [51, 128]. However, most of the existing research does not consider real-time data delivery. An exception is the work by Deng *et al.*, who propose a cluster-based data collection mechanism for delay-sensitive WSANs [28]. Their method, however, cannot quash the NP-hard scheduling problem, it assumes that the topology of the network is known beforehand. Thus they cannot be extended to IoT scenarios involving actuator nodes. We will show that Rapid suffers from none of these drawbacks.

Overview

We begin our description of the Rapid protocol with a simple conceptual overview (see Fig. 6.2). Let us assume that the nodes sense periodically and that the interval

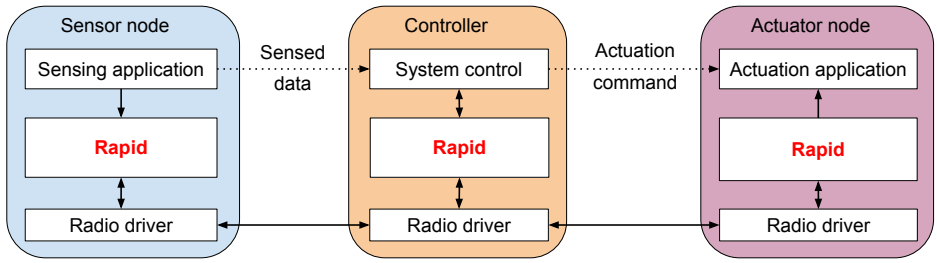


Figure 6.2: System model of Rapid.

is predefined by the IoT application at hand. Each sensor node decides to transmit data based on a simple threshold-based filter. In other words, a node only transmits a data packet if it observes a significant change in its surroundings and thus in the sensed data. If this threshold is zero, the traffic will be periodic since all sensed data is reported to the controller. Based on the received data, the controller issues commands to the actuators. Rapid delivers the filtered data and the actuation commands reliably over a multihop wireless network, within the time constraints set by the application. We next discuss the design goals and the employed architecture.

Design goals

The overall objective of Rapid is to provide a real-time communication primitive that supports both periodic and event-driven traffic while ensuring high energy efficiency. This amounts to the following design goals.

Goal 1: support data collection from as many nodes (flows) as possible within the given latency constraints. This implies a slot-based approach since it provides predictability as opposed to asynchronous approaches.

Goal 2: reduce the average latency as much as possible. This hints at maximizing the amount of communication happening in parallel in the network by means of spatial reuse.

Goal 3: handle topology changes in a timely manner. This rules out running complex, and time consuming flow scheduling algorithms at the central controller (or sink node). That implies the use of advanced topology-agnostic flooding primitives, but without the associated scaling problems.

Goal 4: make the protocol dynamic such that it can effectively support event-based applications. This implies that the protocol should adapt at the local (cluster) level instead of at the global controller.

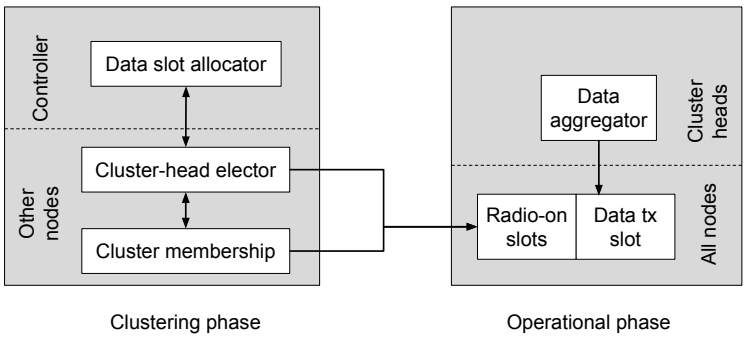


Figure 6.3: Components of Rapid grouped in two phases.

Architecture

To provide real-time communication, Rapid adopts a TDMA approach in which slots are carefully allocated to maximize parallel execution (spatial reuse of time slots) and minimize energy consumption. The resulting transmission schedule is compact, yet comprises retransmissions and path diversity to account for the ever-changing wireless environment. Fig. 6.3 shows the key building blocks of Rapid grouped in two different phases: *clustering* and *operational*. At the start, Rapid operates in global mode to solicit an election process in which a set of cluster heads is determined such that all remaining nodes are only one-hop away. Once the clusters are formed Rapid enters the operational phase in which available data is streamed from the sensor nodes via the cluster heads to the global controller, but not necessarily only through cluster heads. Controller determines the necessary actions and sends out commands (if any) to the actuators in the network.

The motivation for the clustered architecture is that it supports two important optimizations: (i) data can be aggregated at the cluster heads *in parallel*, and (ii) the aggregated data can be sent in one shot (slot) to the controller, reducing the schedule length (i.e., latency) considerably. As a secondary advantage, commands too can be aggregated in a single message, and distributed to the actuators in parallel. The cluster size is an important parameter, which is quite often dictated by practical concerns. The collective data of all cluster members should fit in the maximum payload supported by the underlying radio technology (e.g., 127 bytes in case of IEEE 802.15.4 compliant radios). In the following, we discuss how the clustered architecture is instrumental in realizing the design goals outlined before.

Parallel intra-cluster operations: The clusters are formed such that member nodes are in direct proximity of their cluster head. Consequently, clusters are spatially separated, and can therefore operate in parallel; the capture effect addresses interference by members from different clusters transmitting at the same time. During this intra-cluster communication, members take turns in delivering their data to

the cluster head for aggregation; each member is assigned its own slot avoiding any possible contention. To fight possible external interference in the wireless channel and loss of packets, cluster members retransmit their data up to two times.

Selective flooding: Unlike intra-cluster communication, cluster heads need to send packets to the controller over a multihop network. Doing so reliably makes it even harder. Fortunately, the clustering/aggregation results in only a few packets being sent: n/c vs. n in a network of n nodes and c clusters. That provides room for spending a little more time per packet. Rapid adopts Glossy-style flooding, in which a packet flows as a (guided) wave from a cluster head to the controller *within one slot*. Unlike Glossy, cluster heads only intend to reach one specific node. By being selective in which nodes propagate the wave, path diversity can be provided while most nodes can be put to sleep to conserve energy. A detailed description of the cluster formation and corresponding slot allocation will be provided in Section 6.5.1.

Adaptation: To efficiently support event-based traffic, Rapid allows cluster members to forgo sending data to their cluster head. That adaptive behavior results in aggregated messages of variable length. Sending a message to the controller is a significant cost as multiple nodes are involved. However, these "relay" nodes are only woken up in vain when none of the members has anything to report, which is unlikely to happen for larger clusters.

Achieving bounded latency

Due to the retransmissions within a cluster, and the path diversity between the cluster heads and controller, sensor data is delivered with high probability. Rapid also does it fast, as the name suggests. Exactly how fast Rapid handles this depends on the number of clusters c. In step one (aggregation) n/c members may send their data. In the next step c cluster heads forward their data to the controller, who then sends the actuation commands back into the network in a single slot. That amounts to a total of $n/c + c + 1$ slots. As the number of clusters varies from 1 (a clique network) to $n/2$, the length of the schedule is bounded to $\mathcal{O}(n)$ and $\Omega(\sqrt{n})$, respectively. As remarked earlier practical constraints on the maximum payload limit the number of members in a cluster, so the degenerated case of clique topology will not apply. Thus, the schedule length is typically in the order of $\sqrt{n}$ slots. Note that this compares favorably to *flow scheduling* on top of Glossy, which requires at least n slots, one for each node to report its data.

BASICS

Rapid operates in a time-triggered fashion in which each node activity is mapped onto a slot. In that sense it mimics WirelessHART, whose operation is driven by a superframe consisting of multiple slots specifying what needs to happen and when. For reference, Table 6.1 shows how the frame structure of Rapid and WirelessHART

Table 6.1: Comparison of various system parameters between WirelessHART and Rapid.

Parameters	**WirelessHART**	**Rapid**
superframe length	max. 15 s	flexible
sync duration	1.5 s	20 ms
sync technique	FTSP [75]	Glossy [37]
slot length	10 ms	10 ms and 20 ms
# of slots	max. 1350	flexible

compare. The main difference is that Rapid uses two kinds of slots: unicast slots for local and intra-cluster communication, and selective flooding for global, network-wide communication.

The maximum duration of a superframe can be of 15 s and the maximum number of slots allowed in a superframe is 1350. They can be used for hop-by-hop communication between a sender and a transmitter. Note that a slot can be utilized by multiple sender-transmitter pairs simultaneously. The controller finds the global schedule of which slot will be used by which node pair. These schedules are calculated based on the topological information of the network.

Local slots

In the operational phase, sensors first have to send their data to the respective cluster heads. This is best done by traditional unicast as it provides reliability through acknowledgements (ACKs) and retransmissions. Our implementation targets IEEE 802.15.4 compliant radios, where a packet transmission and its ACK take about 3 ms. To account for interference on the wireless channel, from external sources or neighboring clusters, we budget for a maximum of two retransmissions, which is within a slot length of 10 ms (cf. Table 6.1). Three transmissions are sufficient in most of the cases as will be demonstrated in Section 6.7.

Global slots

Once data is aggregated at the cluster heads, it needs to be sent across multiple hops to the controller. We use fast flooding to avoid the complications of routing, and its robustness through path diversity. Each multi-hop data transfer is handled by one flood in one global slot. Each flood propagates as a wave through the network. The node that initiates the flood starts by broadcasting a packet to its direct neighbors, who immediately rebroadcast it to their neighbors. Glossy [37] has shown that by tight synchronization –less than half a symbol period– the signals of multiple concurrent transmissions interfere constructively at the receiving nodes in range resulting in higher power. This allows these 2nd-hop nodes to decode the original (duplicated) packet successfully. They rebroadcast the packet to the next round of neighbors, and so on.

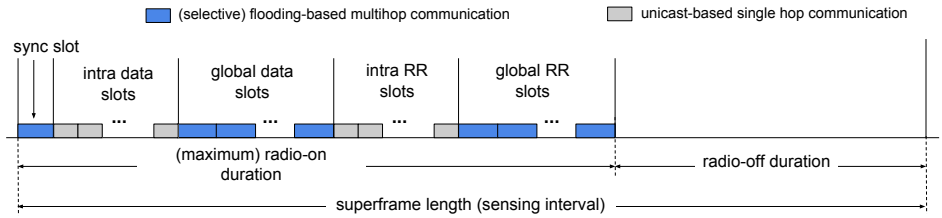


Figure 6.4: Structure of the superframe as the building block of slotted communication in Rapid.

Since the packet transmissions happen back-to-back over multiple hops, the time needed for a complete network-wide flood need not be large. We follow the WirelessHART standard and restrict the network diameter to 5 hops, effectively allowing Rapid to operate with 20 ms slots. As the global communication in Rapid is mostly directed towards the controller, there is little point in forwarding packets in all the directions. This observation prompted us to implement a selective flooding mechanism in which only nodes "en route" to the controller participate. Selective flooding is implemented by creating a wide path towards the destination with a small deviation on either side of the direction of flood. This also increases the flood's resilience to external interference. Nodes off the wide path may put their radio to sleep to conserve energy. Details of the node selection will be provided in Section 6.5.1, in which Rapid's complete slot-scheduling mechanism is discussed.

Rapid superframe

Fig. 6.4 shows the structure of the superframes used by Rapid. A node turns its radio on only if it needs to participate in a particular slot, i.e., either it is the source node or it is part of the route of the flow assigned in that slot. The aim is to make a node participate in a minimal number of slots to save as much energy as possible, while ensuring end-to-end data delivery of all flows in the network. The duration of a superframe matches the required sensing frequency of the application, with most of the frames being spent in sleep mode (see Fig. 6.4, which is not drawn to scale). Next we describe the purpose of the various slot types in the Rapid superframe.

Synchronization slot

As all communication in Rapid is time triggered and synchronous, a global synchronization needs to be maintained within the network. WirelessHART uses the flooding time synchronization protocol (FTSP) [75] as the synchronization mechanism, and allocates 1.5 s for synchronization [69]. Rapid, on the other hand, uses Glossy-style flooding-based synchronization, which only takes 20 ms to sync the whole network. Every superframe starts with a sync slot. Apart from synchronization, which is performed based on the reception time of packets and hop count from the controller, the content of the sync packet indicates the layout of the superframe.

Intra-cluster data slot
These slots are used to deliver sensed data from cluster members to their respective cluster heads. Each member within a cluster has a separate slot assigned to it. Nodes in different clusters can simultaneously communicate reusing these intra-cluster slots. Transmissions by the nodes at the border of a cluster may interfere with other clusters. Thanks to the capture effect, we see fewer packets being lost. The number of intra-cluster slots is decided based on the maximum number of cluster members, which in turn is decided based on the maximum payload size of the aggregated data. A superframe contains intra-cluster slots only in the operational phase. The number of intra-cluster slots is set to zero during the clustering phase which is indicated in the sync packet. Their assignment is discussed in Section 6.5.1.

Global data slot
The global data slots are used to deliver aggregated sensed data from the cluster heads to the controller as well as the commands from the controller to the actuators. Like intra-cluster data slots, global data slots are also used in the operational phase only. The assignment of global slots and the number of global slots in a superframe is discussed in Section 6.5.1 and Section 6.5.2, respectively.

Request/reply (RR) slot
Request/reply (RR) slots are used to form clusters within the network. At the same time, they are also used to acquire a data slot from the controller if the node is a cluster head or from a cluster head if the node is a cluster member using the global RR slots and intra-cluster RR slots, respectively. Like data slots, they also use flooding and unicast. However, RR slots are not exclusively assigned to a node. Thus there is contention among the nodes to send a request. In the next section, we discuss how RR slots are utilized in cluster formation.

Protocol description

As mentioned earlier, Rapid has two phases – clustering and operational. All nodes need to synchronize themselves with the controller immediately after joining the network. Thus, after a node is powered on, it keeps its radio on until it receives a sync packet from the controller. As soon as it receives this packet, it learns the superframe structure and adjusts its radio-on time accordingly. Subsequently, the node starts acquiring a data slot (if it is a sensor node) in the clustering phase.

Clustering phase

In this phase, every node performs three major tasks: (i) decide a role for itself, i.e., either become a cluster head or a member, (ii) acquire a data slot to deliver its data, and (iii) determine in which global data slots to participate to help in routing/flooding packets to/from the controller.

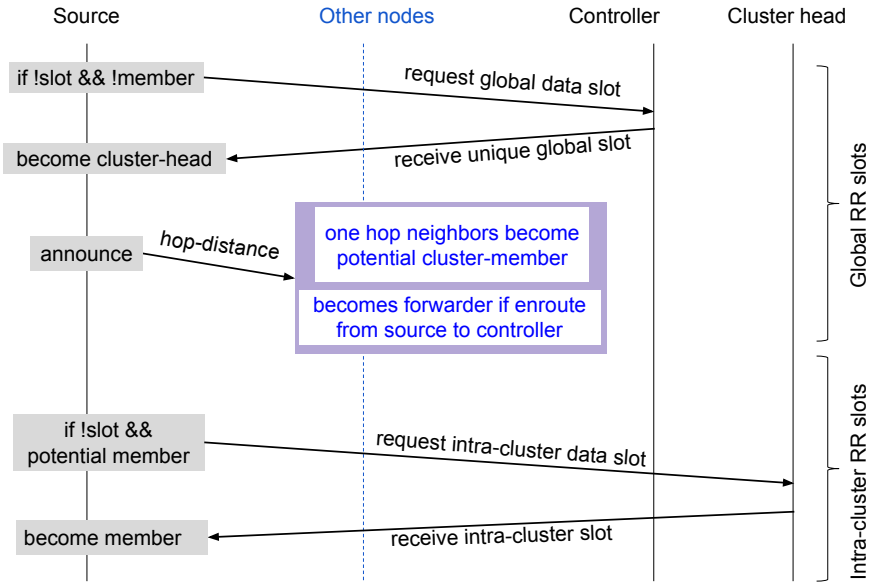


Figure 6.5: Message exchange among the nodes during cluster formation, which is intertwined with data slot assignment.

Election of cluster heads and assignment of global data slots

Fig. 6.5 provides an overview of the cluster-formation process. At the beginning of the clustering phase, a superframe contains only global Request/reply (RR) slots (apart from a sync slot), and nodes have neither cluster role nor a data slot. Thus, everyone requests for a global data slot from the controller in the first (1^{st}) available RR slot. Note that collisions are (usually) resolved by the capture effect, ensuring that only one request reaches the controller. Note that in the beginning there are more collisions, but that will reduce sharply over a few slots when nodes start succeeding (and stop contending).

Once a successful request has been received, the controller sends back a reply containing a unique slot index, which the requesting node can use in subsequent superframes to report its (aggregated) data. As this reply may need to travel multiple hops it is sent in the next (2^{nd}) RR slot, so it will be flooded throughout the network (all global slots are flooding based). After receiving that reply, the requester node becomes a cluster head. It announces this fact in the next (3^{rd}) RR slot to its immediate neighbors, who become potential cluster members, and the "upstream" nodes who need to aid in (selectively) flooding the data from the cluster head to the controller. After this, the next round starts with all remaining nodes requesting a global data slot. The process terminates once all nodes in the network have either been appointed as cluster head or adopted potential member status.

It is clear that if there are only a few global RR slots in every superframe, it will take a large number of rounds before every node can acquire a data slot. To solve this issue, Rapid uses as many global RR slots as can possibly fit within a superframe. After a couple of unutilized global RR slots, the number of global RR slots is reduced to the minimum (i.e., *three* slots for request/reply/announcement) to reduce energy waste. This switch is accompanied with a change in the superframe structure, which will now include intra-cluster RR slots in addition to the minimal set of three global RR slots (used for nodes joining late and/or upgrading to cluster-head status).

Registering cluster membership and assignment of intra-cluster data slots
Potential members record a list of all neighboring cluster heads along with their received signal strengths (RSS). This allows them to become member of the cluster they are closest to, providing better protection against inter-cluster interference (see below in Section 6.5.2). In the first intra-cluster RR slot, potential members send a request to their cluster head to allocate them an intra-cluster data slot. As before, collisions are anticipated to be resolved by the capture effect, and the cluster head sends back a reply to the winning (strongest) member with its slot index. As only one slot can be handed out at a time, nodes have to repeatedly send in requests. In case of a higher number of members in a cluster, randomizing member requests could be applied to reduce the collisions, however this aspect has no effect on the Rapid protocol. If the maximum number of cluster members has been reached the head responds with a *negative* reply, and the potential member can then switch to another cluster head with the next-best RSS value. When a node has exhausted the list of neighboring cluster heads, it must become a cluster head itself and starts requesting the controller for its own global data slot.

Participation in global data slots
Though intra-cluster data transmission occurs over a single-hop using unicasting, the cluster heads send the aggregated data to the controller through multiple hops using selective flooding. Thus, a number of intermediate nodes need to forward the packet within the same global data slot. Unlike LWB, Rapid restricts the number of nodes that participate in each global data slot in order to reduce overall energy consumption. Nodes follow a distributed approach to determine whether or not they should become a forwarder in a particular global data slot. We follow approach similar to Carlson et al. [18].

The idea is to widen the shortest path between a node and the controller by using all paths of equal length[1] connecting the two. A requesting node (n), upon receiving a global data slot from the controller, notes the hop count (h) from the relay-count field in the packet header (as per virtue of the original Glossy protocol).

[1]For extra reliability, paths with (an) additional hop(s) can also be included.

Since the global RR slots are flooding based, every other node also receives the reply message, and notes its hop count from the controller (h_c). Next, when node n announces itself as a cluster head, it attaches its hop-distance h from the controller to the announcement message. All other nodes receive this message and record h as well as their hop distance (h_n) to node n. This allows them to check if they are en route to the controller. If $h_n + h_c \leq h$ then they add the global data slot of node n to their list of active slots.

Operational phase

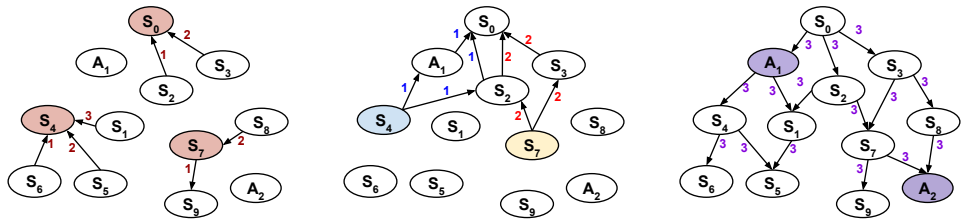
From the above description one can see that the clustering process is highly efficient and its length is customizable during the deployment, i.e., proportional to the number of nodes in the network. The operational phase starts after the clustering phase, and is initiated by the controller adjusting the layout of the superframes. The intra-cluster RR slots are dropped, the global RR slots are set to the minimum, and the appropriate number of intra-cluster and global data slots are included, as well as slots for disseminating actuation commands.

Intra-cluster data collection

As mentioned earlier, cluster members use intra-cluster data slots to send their data to their respective cluster heads through unicasting (see Fig. 6.6a). Though the slots are unique among the members of the same cluster, slots are reused across clusters. Even if there are simultaneous transmissions in their vicinity, cluster heads can successfully receive the packets from their cluster members due to capture. When a packet is received correctly, the cluster head immediately sends an ACK. If no ACK is received, the cluster member retransmits the packet after a small timeout period within the same slot (up to 2x). If the retransmission prove in vain the packet is discarded.

Data collection and delivery of actuation commands

As mentioned earlier, global data slots are used by the cluster heads for data collection (Fig. 6.6b) and by the controller for delivery of actuation commands (Fig. 6.6c). To report the aggregated sensed data, only a subset of nodes participates in forwarding the packets using selective flooding. However, all nodes forward packets containing actuation commands such that multiple destinations (actuators) can be reached within the same slot. Fig. 6.7 shows the difference between unicasting and (selective) flooding in the intra and global slots, respectively.



(a) intra-cluster data delivery (slot 1,2,3)

(b) aggregated sensed data delivery (slot 1,2)

(c) actuation command delivery (slot 3)

Figure 6.6: Packet routing in the network during operational phase.

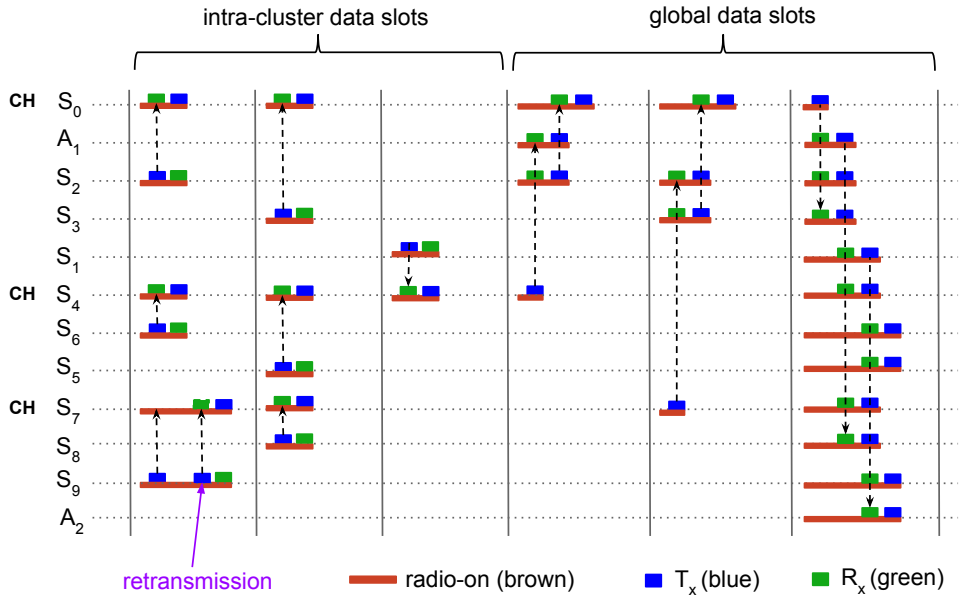


Figure 6.7: Slot-based communication – showing node activity in the data slots of the superframe during operational phase.

When none of the cluster members detect any event in a sensing round, the cluster head has no data to report. To save energy, the cluster head sends a dummy packet so that participating nodes can switch-off their radio immediately after forwarding the packet. Without a dummy packet nodes keep listening unnecessarily for the whole slot as the exact moment a flooding ripple passes by is source/destination/ interference dependent. Dummy packets are also used when no actuation is required. Thus some more energy is saved because of this technique.

Deciding the number of global data slots

The number of global data slots is typically one slot higher than the number of cluster heads to accommodate actuation commands. Like sensed data, multiple

actuation commands are aggregated in one packet and sent to multiple actuators in the same global data slot. If there are more actuation commands than can fit within a single packet, overflow slots are used. Each actuation packet therefore contains a flag signaling if it is the last in the pipeline or not.

Implementation details

In this section we describe some important parameter settings of Rapid. Rapid is implemented using the Contiki operating system [31]. As mentioned earlier, it utilizes constructive interference for effective communication. Glossy protocol uses constructive interference based flooding for synchronizing the network. Thus, we used the core functionalities of Glossy with some adaptation to obtain network-wide time synchronization, and as a basis for route-free multihop communication. Glossy uses a combined yet simple routing and MAC layer with a simple communication model. Every node has a common understanding of time as well as the starting time of a flood. The initiator node starts the flood by transmitting the packet, and every other node simply transmits the packet after receiving it. Glossy ensures a fixed switching delay between receiving and transmitting packets in order to ensure constructive interference. On the other hand, the design of Rapid involves various other types of communication while maintaining the simple design principle.

Without loss of generality, we fixed the message size for intra-cluster communication to 4 bytes of sensor-data payload and 4 bytes of headers (2-byte source and destination addresses), so 8 bytes in total. For inter-cluster communication the message size depends on the number of cluster members n, which we varied over 2, 4, and 8, yielding packet sizes of $(n+1) \times (4+2) + 2 + 2$ bytes to accommodate the payload of the members and head (with source ID), the destination address, and a length field (22, 34, and 58 bytes respectively). To ensure reliable data delivery without end-to-end control (to reduce latency), Rapid forms clusters such that the link quality between a cluster head and its member nodes is sufficiently high. We used an RSS value of -75 dBm as the threshold to filter out good links. This setting, in combination with up to two retransmissions within a cluster, proved to achieve good packet reception rates across a range of different topologies (as reported in the next section). Only in the case of very sparse networks the chosen threshold had a noticeable impact on performance as many clusters were created with just a few, or even zero, members due to a lack of quality links, which compromises aggregation efficiency.

The code footprint of Rapid is very small. The implementation added about 900 lines of C code on top of the existing implementation of Glossy. The compiled firmware is just about 267 kB as compared to 250 kB and 258 kB for Glossy and LWB, respectively.

Though the current implementation of Rapid is targeted for the Tmote sky platform with CC2420 radio, it can easily be adapted to work on other types of

devices. One of the biggest hurdles is to achieve constructive inference by ensuring multiple simultaneous transmissions, i.e., keeping the switching delay from receive to transmit mode constant across multiple devices. The developers of Glossy already described how it can be ported to other radio platforms, and indeed a number of CI-based systems have emerged running on other radios, e.g., CC430 is used in [18], CC2530 is used in [92], etc. Porting Rapid to these platforms should be no more complicated.

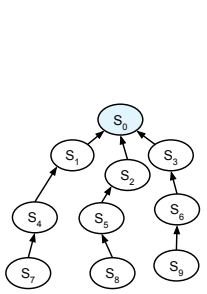
Evaluation

We evaluated Rapid both in simulation, to provide repeatable experiments suitable for cross protocol comparison, and on a real-world testbed, to study resilience to external interference and other practicalities. For the simulations we used the Cooja software that comes with the Contiki operating system. The real-world experiments were carried out on two publicly-available testbeds, i.e., Indriya [29] and FlockLab [] along with some testing on local nodes in our laboratory. Our experiments are conducted on 97 and 32 TelosB nodes from Indriya and FlockLab, respectively. As the testbeds do not contain any actuators, some of the devices are also marked as actuators from these testbeds (10 and 4 respectively). Since the testbed measurements involved a lot of randomness (e.g., interference from 802.11 traffic during working hours) we ran different protocols back-to-back to ensure they endured similar conditions.

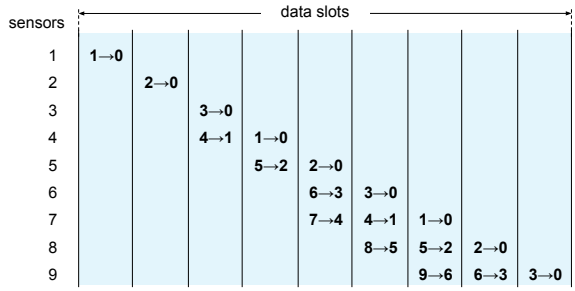
The testing code on top of Rapid was set to mimic a control application in which each node periodically reads out a sensor and reports it to the central controller. The sensing frequency has been fixed at $\frac{1}{10}$ Hz, and the deadline was set to 1 s. That effectively leads to a Rapid superframe of 10 s. A more frequent sensing (smaller superframe) can lead to higher data granularity but at the expense of increasing the duty cycle.

Analytical evaluation

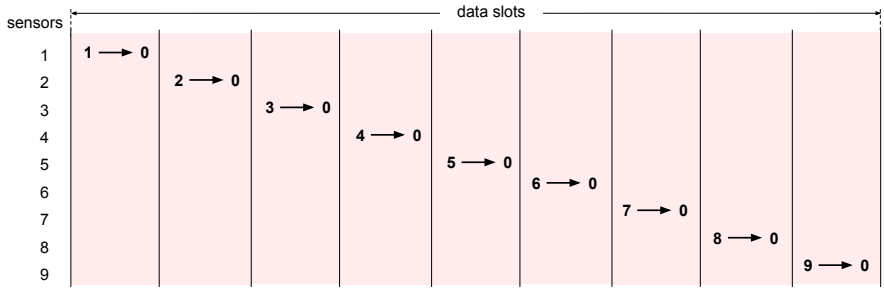
The lack of an open implementation of any prevalent real-time routing protocol restricted our options for performing a one-on-one comparison with Rapid. Before providing a detailed study based on the actual implementation –to thoroughly evaluate Rapid vis-á-vis latency– we therefore considered a *hypothetical real-time protocol (HRP)* that packs the data packets tightly to avoid any wastage of slots. Let us assume that HRP combines source routing with optimal flow scheduling as described by Pottner *et al.* [85]. For the sample 9-flow (10-node) network shown in Fig. 6.8a, the optimal HRP scheduler finds the shortest possible schedule to complete all the flows as shown in Fig. 6.8b. As messages are not aggregated at least 9 slots are needed. HRP manages to do this by scheduling the remaining transfers in parallel. Thus, the total time required is 90 ms considering a slot length of 10 ms according



(a) Topology



(b) Optimal source routing, completion in 90 ms



(c) LWB, completion in 180 ms

Figure 6.8: Sample 9-flow network with schedules for optimal source routing (HRP) and LWB. Slot length for HRP and LWB is 10 ms and 20 ms, respectively.

to the WirelessHART standard. In practice, completion will take longer as the flow scheduling problem is NP-hard and one needs to resort to heuristics. Moreover, for resilience one may prefer graph routing over source routing, adding even more slots to the schedule.

To consider a realistic case for comparison, we turn to LWB, which is the closest state-of-the-art routing protocol in WSNs that offers data delivery with lowest latency and high energy efficiency. Though LWB was *not* specifically designed for real-time communication, it can be used as such by mapping each flow to a global flood (slot). LWB therefor requires 9 slots (like HRP), but these slots are 20 ms long to ensure the data can be flooded across multiple hops. Thus it takes LWB a total of 180 ms to complete all the flows (Fig. 6.8c), which is twice as long as for HRP.

Rapid combines the unicast slots from HRP and network-wide floods from LWB through its clustering approach, see Fig. 6.9. The maximum number of cluster members was set to 3, leading to 3 clusters headed by nodes S_0, S_4, and S_9 with 3, 2 and 2 members respectively. Consequently Rapid needs 3 intra-cluster slots, followed by 2 global data slots to forward the aggregated data from remote clusters S_4 and S_9 to the controller S_0. Thus, the total time required to complete the flows

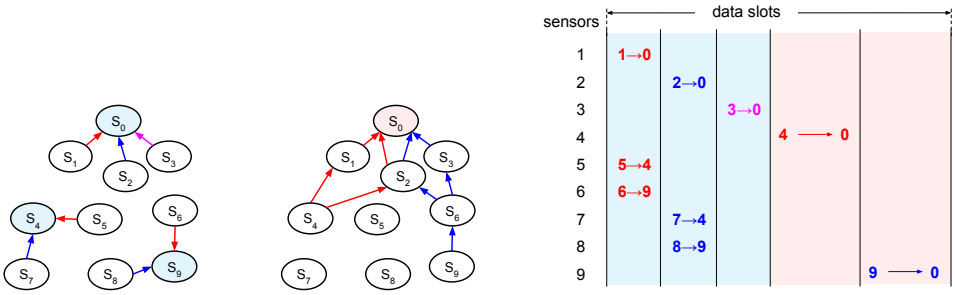


Figure 6.9: Sample 9-flow network with 3 Rapid clusters and selective flooding, completing in just 70 ms. The length of a intra-cluster and a global slots are 10 ms and 20 ms, respectively.

is 70 ms ($3 \times 10 + 2 \times 20$), where the length of an intra-cluster and a global slot is 10 ms and 20 ms, respectively.

The key observation is that Rapid outperforms both HRP and LWB. Rapid beats HRP because it aggregates data allowing it to use fewer slots, while offering additional resilience through its use of flooding. Rapid beats LWB because it exploits parallelism at a local level (spatial reuse) allowing it to use shorter and fewer slots.

6

Simulation results

To study the performance of Rapid in more detail we evaluate three aspects: (i) delay bound, (ii) energy efficiency, and (iii) data-collection reliability. In particular we are interested in the associated metrics of the total time (latency) taken for scheduling all flows, the average duty cycle of the nodes, and the packet reception ratio (PRR). As studying various parameters under various topologies is not possible in a static testbed, we first performed a simulation-based study. We considered three different network topologies of 50 nodes, in which the average node degree was set to 2, 4 and 8. We did so by changing the total deployment area, while keeping the total number of nodes (flows) fixed.

We compared the performance of Rapid with LWB and its successor Forwarder-Selection LWB (FS-LWB), which achieves higher energy efficiency in data collection scenarios by limiting the set of participating nodes in a flood [18]. Note that Rapid also uses such forwarder selection when cluster heads report the aggregated data to the sink. Fig. 6.10 shows the total time (in blue) and number of slots (in yellow) required to complete the 50 flows in the network. Note that for LWB and FS-LWB, the numbers remain the same irrespective of the network density as each flow uses one global slot (and the number of flows is constant). Hence, we only plot one bar. For Rapid, however, the increase in node degree raises the number of members per cluster, which in turn increases the scope for parallel communication. The net effect is that Rapid uses more intra-cluster slots (for members) and fewer global slots (for cluster heads) as the breakdown shows. This translates into a lower number of total

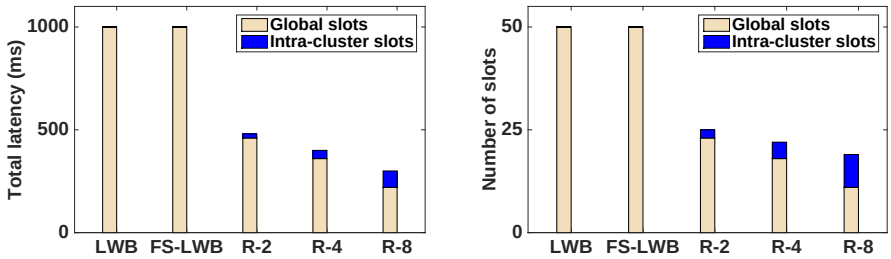


Figure 6.10: Comparison of total time and number of slots required to complete 50 flows for different node degrees.

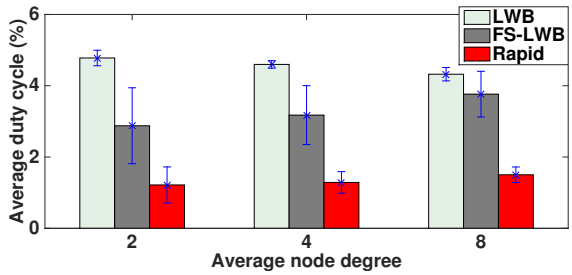


Figure 6.11: Average duty cycle in a 50 node network.

slots per superframe (more parallelism) leading to a lower latency. Note that this effect is amplified by intra-cluster slots taking only 10 ms vs. 20 ms for global slots.

Rapid excels not only in terms of overall latency, it also reduces the energy consumption of the network. Fig. 6.11 shows a comparison of the average duty cycle of the nodes. For LWB and FS-LWB, unlike with latency, the average duty cycle of the nodes changes when the density of the network increases. Recall that the density was increased by shrinking the deployment area, which causes nodes to be located closer to the controller. For LWB this results in Glossy floods completing faster as fewer hops need to be traversed. This leads to (slightly) lower duty cycles. This effect also applies to FS-LWB, but at the same time the increased density leads to more redundant paths between a node and the controller. The effect of involving more nodes per flood is stronger than the reduction in hops, causing FS-LWB's energy consumption to go up with increasing density. For Rapid a third factor comes into play. Its shift from global (flooding) to local (unicast) communication counters the reduction in efficiency of the selective forwarding optimization. The overall effect is that Rapid's efficiency is almost insensitive to network density.

Table 6.2: Packet reception ratio vs. node degree.

Node degree	**LWB**	**FS-LWB**	**Rapid**
2	100%	99%	98.4%
4	100%	99%	97.3%
8	100%	100%	99.6%

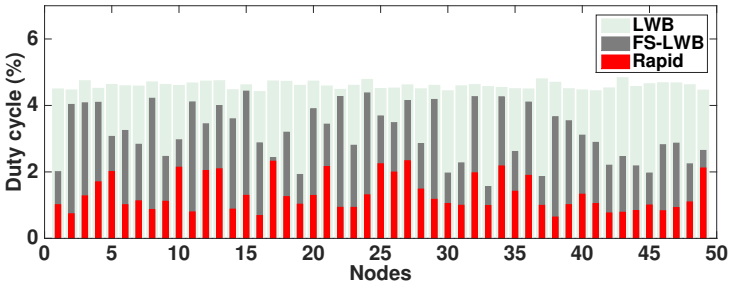


Figure 6.12: Duty cycle of individual nodes in a 50 node network (node degree of 4).

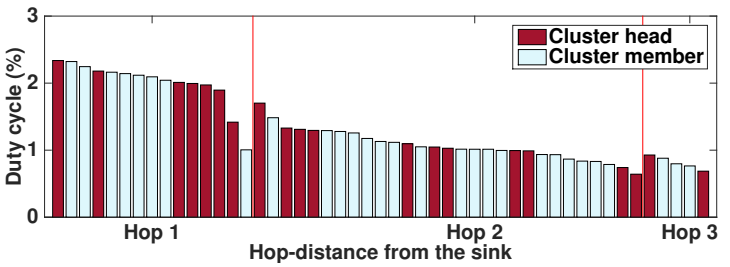


Figure 6.13: Duty cycle of Rapid nodes, sorted by hop-distance from the controller.

The final performance metric to consider is the average PRR, which is listed in Table 6.2. LWB achieves a 100% PRR due to a high degree of redundancy (all nodes participate in all floods). FS-LWB does slightly worse, especially for low density networks as the number of redundant paths is limited, causing about 1 % of the packets being lost. Rapid suffers from the same effect, and looses up to 3 % of the packets in the worst case. Closer inspection of the results revealed that this extra packet loss is incurred during intra-cluster communication. We conjecture that this is due to the capture effect failing to resolve all interference from communication in neighboring clusters. Further research is needed to study if careful cluster formation or advanced retransmit policies can bring Rapid's reliability in line with (FS-)LWB. Alternatively, more intra-cluster slots can be used to reduce the amount of parallel communication.

To gain a deeper understanding of the efficiency of the protocols, we have analyzed the duty cycle of individual nodes. Fig. 6.12 shows the duty cycle of the 50 nodes for the three protocols. According to expectations LWB shows hardly any difference between the nodes as they all participate in all slots. FS-LWB and Rapid, on the other hand, do show significant fluctuations. The selective forwarding scheme only involves a subset of the nodes, with those close to the sink being active for all (global) slots and those at the edge only in case of sending their own data. Note that the maximum duty cycle for FS-LWB (by Node 15) is roughly the same as for LWB, so the network lifetimes, that is until the first node dies, are the same. In that respect Rapid does much better (i.e. doubles the lifetime) due to its cluster-based approach.

Fig. 6.13 provides additional insight by plotting the duty cycle of Rapid for members (blue) and cluster heads (purple) sorted by hop distance to the controller. Two important observations can be made. First, nodes close to the controller spend more energy than nodes 2 or 3 hops away, because of the selective forwarding optimization. Second, cluster heads consume only a little more than cluster members. That somewhat surprising result follows from the low number of members per cluster. Note that the maximum number of cluster members was set to 4, yet only 30 out 50 nodes became a cluster member (and not $40 = 4/5 \times 50$) due to solitary nodes becoming cluster heads. Apparently, the requirement for having a strong link with the head of a cluster pruned away too many options for edge nodes, who are relatively poorly connected to begin with. Having few members per cluster automatically reduces the overhead, putting the duty cycle of the cluster heads in line with that of the members.

Testbed results

To validate the findings from the Cooja simulations, we conducted real-world tests on the Indriya and FlockLab testbeds with 97 and 32 nodes, respectively. On Indriya, we used node 1 as the sink and the other 96 nodes as sensors, of which 10 nodes served the dual role of actuator, totaling 106 flows in the network. Similarly, FlockLab has 35 flows in total (31 sensors and 4 actuators). For the testbed experiments, we used a maximum of 8 intra-cluster slots. The results reported in Table 6.3 and Table 6.4 summarize the outcome of over 100 hours of experimentation. The performance numbers confirm that Rapid provides a multi-fold latency reduction compared to LWB and FS-LWB, while ensuring significantly higher energy efficiency. Specifically, Rapid achieves a latency reduction with a factor of 2.2 and 2 over Indriya and FlockLab, respectively.

The average duty cycle of the nodes depends on the total number of flows in the network, and utilization of intra-cluster slots. On Indriya Rapid is 2.8 and 1.6 times more energy efficient compared to LWB and FS-LWB, respectively. On FlockLab the efficiency is even higher at 3.8 and 1.9 times, respectively. The reason is that

Table 6.3: Performance comparison of various protocols on the Indriya testbed with 106 flows (96 sensors and 10 actuators).

	LWB	**FS-LWB**	**Rapid**	**Rapid-X**
Total latency (ms)	2120	2120	980	980
Avg. duty cycle (%)	8.11	4.81	2.92	5.23
PRR (%)	98.3	96.5	95.7	97.9

Table 6.4: Performance comparison of various protocols on the FlockLab testbed with 35 flows (31 sensors and 4 actuators).

	LWB	**FS-LWB**	**Rapid**	**Rapid-X**
Total latency (ms)	720	720	360	360
Avg. duty cycle (%)	3.91	1.95	1.02	1.88
PRR (%)	99.9	99.5	97.1	98.8

the percentage of global slots in FlockLab is lower compared to Indriya.

The harsh conditions on the testbed reflect in the packet reception ratios, which are all lower than recorded in simulation. Even LWB looses a fraction of packets, while in simulations it lost none. FS-LWB and Rapid do slightly worse and loose about 3-4 % of packets. Note however that these numbers are comparable to the 95 % average reliability of flow delivery reported by Lu *et al.* using graph routing [69]. To improve the reliability of Rapid, we have created a special version (named Rapid extra, or Rapid-X for short) in which we disable forwarder selection and resort to full Glossy floods (like LWB). The immediate benefit is an improvement in PRR. On Indriya, Rapid-X even surpasses the PRR of FS-LWB. The remaining packet loss, especially on FlockLab in comparison to LWB, is due to collisions in the intra-cluster slots. Of course there is no free lunch. Rapid-X's improved PRR goes at the expense of its energy efficiency, which is similar to FS-LWB (but with a much shorter latency). Wether or not the improved PRR of Rapid-X outweighs the cost in terms of energy efficiency depends on the application at hand.

So far we have discussed the total time required to complete all flows. An important question is what will happen if the deadline is too small to accommodate all flows. The flow completion pattern over time is shown in Fig. 6.14. It is clear that the majority of the flows can be completed in about half of the time (80% complete in 540 ms). All the members belonging to the controller's cluster complete real quickly during the intra-cluster phase. Then, with each global slot a set of flows complete together as the aggregated message contains the data of all members belonging the cluster head that was allocated to that slot. The number of flows varies per global slot as the topology dictates the cluster formation. In particular, towards the end slots are taken by solitary cluster heads who failed to join any cluster and are forced to use a global slot just for themselves.

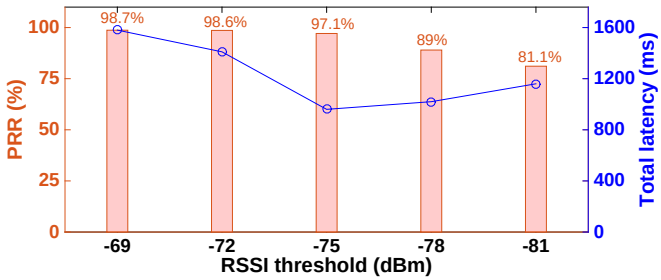


Figure 6.15: The impact of RSSI threshold on latency and reliability (Indriya testbed).

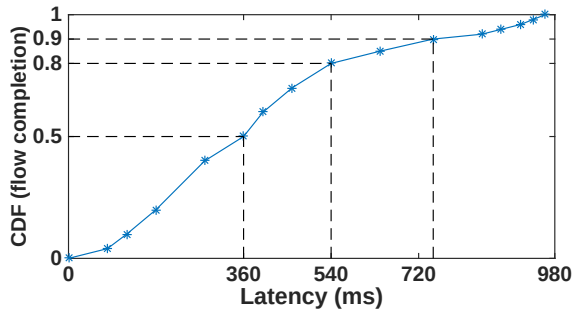


Figure 6.14: Cumulative distribution showing the number of flows that are completed over time (Indriya testbed).

Recall that a node decides to join a cluster only if it has a good link to the respective cluster head. Rapid uses the RSSI to discriminate good from bad links by means of a threshold set to -75 dBm. Changing the threshold will impact the cluster formation process. If we tighten the threshold, the number of cluster members will be lower leading to more clusters, hence, a longer schedule and latency goes up. In return the reliability will increase as the links are of better quality. If we relax the threshold, the reverse will happen with schedules completing faster and PRR becoming worse. To study the exact trade off, we have run a number of experiments with different thresholds on the Indriya testbed, see Fig. 6.15. Observe that, as expected, the PRR decreases when relaxing the threshold from -69 dBm to -81 dBm. The latency, however, does initially go down, but then raises again once the threshold is relaxed beyond -75 dBm. A detailed inspection of this surprising result revealed that this behavior is the consequence of a safety mechanism built into Rapid. When a cluster member fails to transmit in two successive superframes, i.e. it does not receive an ACK from its cluster head, it leaves the cluster and acquires a global slot. This does keep the PRR under control, but at the expense of additional latency (an extra global slot). Without this fail-safe mechanism the latency does decrease further, but the PRR drops to unacceptable low values for typical IoT

applications. Therefore Rapid's default threshold setting of -75 dBm is at the sweet spot of lowest latency and marginal packet loss.

Discussion

In this section we step back and discuss Rapid in a broader context. While developing the protocol, we did not consider all possible application scenarios. In the following, we discuss the usage of Rapid *vis-á-vis* different scenarios and also how Rapid can be adapted in such cases.

In the evaluation we assumed a maximum time bound of 1 s and showed that more than 100 flows can be scheduled within such a deadline. The design of Rapid provides the flexibility to set the deadline as required by the application without any design or implementation changes. Of course there is a limit on how many flows can be scheduled within a given time bound. From the real-time perspective, the benefits of Rapid are two-fold (i) it can schedule more flows within a given time bound, and (ii) it provides guaranteed delay bound with high accuracy. When there are only a few flows in the network, it becomes easy to accommodate all of them within the required time bound. In such cases, not only Rapid, but any other protocol would meet the deadline for all the flows. However, Rapid will still be effective as it provides significantly higher energy efficiency than any other existing protocol.

Rapid assumes a fixed priority for all the flows in the network. Thus, it targets to complete as many flows as possible within a given time. However, in many applications, flows may have different priorities. In such cases, high priority tasks need to be scheduled earlier, and if time permits only then low priority tasks need to be scheduled. To accommodate more flows, the low priority flows can be assigned a slot in every, say s, superframes in a round-robin fashion. A suitable scheduler on top of Rapid can tackle such priority-based scheduling while the communication mechanism remains the same.

In case of event-driven applications, a fixed schedule is assigned similar to periodic applications to ensure a fixed delay and guaranteed delivery. If the events are very sparse and there are more flows to schedule, a single global slot can be assigned to multiple cluster heads. Whenever, a particular cluster head has data from any of its members, it contends within its assigned slot to deliver the packet. As the events are sparse, the communication would be successful in most of the cases. In case of event burst, further study is required to device a proper solution.

Conclusions

Real-time communication is a desired feature of many IoT applications in view of industrial acceptance. Though a number of energy-efficient routing protocols exist in the WSN domain, timely data delivery is often not the prime focus. On the other

hand, in the control systems, a number of routing technique exist primarily focusing on the latency of data delivery with energy efficiency as a secondary concern. Moreover, reliability of the data delivery also need to be ensured.

We presented Rapid, a highly energy-efficient communication protocol that provides real-time guarantees. It uses a cross-layer approach – slot-based, synchronous communication at the link layer and flooding and clustering at the network layer. Using on-the-fly clustering, we split the data collection in two levels. This allows us to achieve parallel operations leading to higher energy-efficiency and lower latency at the same time. We compared Rapid with source and graph routing to explained how it behaves. The performance of Rapid compared with LWB and FS-LWB on two public testbeds showed that similar PRR can be achieved in significantly lower time and with lower energy consumption. We recorded a 2.8 and 3.8 times reduction in average duty cycle of the nodes at 2.2 and 2 times reduction in data delivery latency compared to LWB on the two testbeds. These benefits are primarily achieved by the use of fast flooding for multihop communication, which eliminates explicit routing and hop-by-hop scheduling for each flow. We gained further by adapting a selective flooding mechanism to tackle the inefficiency of flooding the whole network.

7

Conclusions

Excellence is a continuous process and not an accident.

A. P. J. Abdul Kalam

Now at the end of this thesis, let us take a holistic view and look back at the contributions so far. The vision of this thesis is not providing a few *smart choices* in communication amongst the devices or enabling *smart services*. The idea is to move from the localized and domain specific applications that are in silos to an all compassing vision of *Virtualization of the IoT*. We started off with the virtualization of devices and provided a holistic architecture. Then we went on to showcase a simple application that used our architecture. To enable smart applications, a large number of devices are deployed under the purview of IoT. Thus we delved deep into the communication substrate and optimization therein to connect these devices. We note that virtualization helps in such connectivity too. Since IoT is not only an enabler of communication to/from devices from a global perspective it should also include intelligence so as to take humans out of the system as much as possible. The devices should be autonomous and talk to each other even when they are offering different IoT applications. Indeed virtualization helped in this task too. We now argue how these small but important steps cater to virtualization of the IoT.

We provide the evidence as to how the following benefits of virtualization could be drawn from our contributions in this thesis: (i) tackling device heterogeneity through a unified view, (ii) reusing and sharing the IoT devices smartly across multiple applications and reduce infrastructure cost, and (iii) achieving operational optimization of the resource-constrained devices. Through the above accomplishments,

this thesis advocates virtualization to tackle the challenges related to large-scale expansion of IoT. Let us recapitulate our journey so far.

Recapitulation

We shall look at the contributions of this thesis through the prism of virtualization. To start with we briefly mention the three main outcome of the research presented in this thesis:

- The feasibility and usability of IoT virtualization has been demonstrated to enable cooperation between IoT devices resulting in operational optimization of the devices and taking humans out of the control loop.
- We have achieved a significant reduction of data traffic within a network without violating the deployment goals with respect to coverage and accuracy.
- We have provided a scheme to guarantee real-time communication in a multi-hop WSAN without requiring to solve the NP-hard node-scheduling problem.

To this aim, this thesis addressed a number of conceptual and system level challenges that are lay ahead of virtualizing the IoT. In particular, we provided a virtualization framework for IoT and enhancement of WSANs as an enabler of IoT through virtualization. We recap them below.

Virtualization framework for IoT: IoT devices will have a diverse range of capabilities and functionalities. Managing these heterogeneous devices such that they all become part of the IoT ecosystem is a challenge. The vision of IoT takes the human out of the control loop as much as possible. This is only possible through seamless and smart interaction among these heterogeneous devices.

Through virtualization of IoT, we envision to tackle most of these challenges. Existing architectural models for IoT usually target a specific application scenario. Each application vertical operates in its own silo. IoT reference architecture should act as a proponent of IoT virtualization such that intra-domain and inter-domain cooperation among IoT devices becomes a reality to get most out of these IoT deployments. We proposed a distributed layered architecture – Distributed Internet-like Architecture for Things (DIAT). The lowest layer of DIAT, called the virtual object (VO) layer, acts as the basis of virtualizing the IoT. DIAT not only offers interoperability among the devices through a unified interaction mechanism, it also helps to tackle scalability issues of large number of devices and offers higher degree of optimization. As mere connectivity is not the only goal of IoT, DIAT acts as a placeholder for cognitive functions that can provide smartness to any IoT application, which is enabled by a composite virtual object (CVO) layer in DIAT.

The feasibility and usability of DIAT were showcased using an IoT application. To this end, we developed iLTC, an automated **i**ndoor **L**ighting and **T**emperature

Controller (iLTC). It controls the HVAC and lighting systems at room level based on individual user preferences. Through virtualization iLTC ensures easy cooperation among various entities of the system. Though there is some user intervention, it is kept minimal and only during the training period. The system uses the layered modular design as proposed by DIAT, which makes it scalable with the number of users and rooms.

Overall data traffic reduction in WSAN: WSNs are one of the key components in the IoT ecosystem. As the embedded devices are often battery operated, energy efficiency is always the most sought after feature in WSNs. Since the radio transceiver is one of the major energy-consuming components in a wireless embedded device, efforts are made to keep the radio transceiver in low-power mode as much as possible without affecting the data delivery. A popular approach is to reduce the overall amount of data traffic within a deployment by exploiting correlation among the nodes. Existing approaches assume *a priori* and static correlation among the nodes. However, based on the sensed data collected from various deployments, we found that this assumption is not realistic. Thus, the scope of data traffic reduction as well as operational optimization is limited with the existing methods.

Through virtualization of WSNs, we showed how we achieve a higher operational optimization. We introduced the virtual sensing framework (VSF) that utilizes the concept of virtual objects (VO) from DIAT. It reconstructs sensor data for all the nodes in a deployment while the actual measurement is done using only a subset of them. We have utilized the inherent correlation amongst the sensor data without having: (i) any *a priori* knowledge of the statistics of the data; (ii) the locations of the sensor nodes and, (iii) the type of the physical parameters observed. A case in point is predicting temperature with a light sensor within a tolerable error bound.

Virtual sensors (against each of the real sensor nodes) are the building blocks of VSF. The prediction technique of the virtual sensors adapts to the changes in the sensor data. Using VSF's activity reduction technique, we have achieved a significant improvement in energy savings compared to other similar techniques, while maintaining sufficiently high accuracy of the sensor data. Our maximal sleeping node policy can reduce the overall energy consumption of a WSN. However, the formulated minimum active node scheduling problem is shown to be an NP-hard problem. Thus, we provided a heuristic algorithm to find the minimum number of active nodes at any instance. This way VSF achieves a significantly higher energy savings for the overall network by reducing the amount of traffic. But, it also ensures that the quality of the sensed data is maintained within a tolerable error bound. We note here that a virtual sensor is not bound to any one IoT application/deployment.

Efficient routing for node-scheduling: Node-scheduling exploits over-provisioning of nodes that are typically employed to counter adverse network conditions and node

failures. Based on the application scenarios, different node scheduling strategy is needed, e.g., k-coverage, point-coverage, spatial correlation, etc. For example, VSF significantly reduces the overall traffic within a network by exploiting the inherent correlation among the nodes. However, it is not sufficient to just limit the number of active nodes; active nodes should form a *connected* (sub)network.

We presented Sleeping Beauty, an energy-efficient communication protocol that operates with partial topological information, yet outperforms state-of-the-art protocols (LWB and FS-LWB) in node-scheduling scenarios. Sleeping Beauty accomplishes this by including (i) an efficient neighbor-discovery mechanism that enables the selection of a minimal, but connected set of active nodes, and (ii) a simple, but elegant clock-offset estimation technique that allows nodes to sleep for a long time without esynchronization. Our time synchronization technique can be used in any application other than node-scheduling. We compared the performance of Sleeping Beauty with state-of-the-art protocols on two public testbeds (Indriya and FlockLab), and showed that the same performance (PRR) can be achieved at a fraction of the energy consumption.

Routing for timely data delivery: Neither the traditional data collection protocols nor the routing protocols for node-scheduling are suitable for applications where sensed data or actuation commands need to be delivered within time bound, e.g., industrial automation, automatic lighting control, etc. Since these protocols trade off performance for energy efficiency, real-time applications demand a specialized communication protocol that strikes a balance between latency and energy consumption.

Though duty cycling is an efficient approach to reduce energy consumption of a node, it seriously affects end-to-end latency as nodes have different wake/sleep schedule at different hops. Thus, real-time communication protocols use slot-based synchronous communication. Energy is saved by having the nodes become active only during those slots in which they need to transmit or receive data by means of a global schedule that determines the sender/receiver pair for each slot. However, scheduling multiple flows is known to be an NP-hard problem, where each flow needs to be completed within a time bound. The question is then how a real-time communication protocol solves an NP-hard scheduling problem along with providing real-time guarantees at minimal energy expense.

We presented Rapid, a highly energy-efficient communication protocol that provides real-time guarantees. It uses a cross-layer approach – slot-based, synchronous communication at the link layer and flooding and clustering at the network layer. Using on-the-fly clustering, we split the data collection in two levels. This allows us to achieve parallel operations leading to both higher energy-efficiency and low latency together. We compared Rapid with source and graph routing to explain how it performs. We showed that Rapid significantly lowers latency and energy

consumption for similar PRR comparing with LWB and FS-LWB. These benefits are primarily achieved because of the use of fast flooding based multihop communication, which eliminates explicit routing and hop-by-hop scheduling for each flow. This avoids solving the NP-hard flow-scheduling problem. We gained further by tackling the inefficiency of flooding the whole network by adopting selective flooding involving a small subset of the network.

Future work

This thesis took a few steps forward towards the vision of virtualizing IoT by revisiting some of the challenges of traditional WSANs and provided improved solutions. However, to realize this vision we still need to solve some more challenges, which were slightly away from the purview of this thesis. Some of the important open challenges are mentioned below.

VSF considers the correlation among nodes in a short time interval. However, applying the concept of VSF over longer duration will reap larger benefits in the cases where seasonality is present in the data. As VSF assumes that the change in correlation among the nodes is relatively low, a further study is necessary to check the performance of VSF and its successive improvements if the correlation changes rapidly. Devising newer version of VSF is necessary to cater to such scenarios.

The efficient data collection protocols, presented in this thesis, take advantage of the network-wide fast flooding mechanism, where addresses of the individual nodes are of little importance. However, IoT devices will be equipped with global addresses (like IPv6) for global access. As the fast flooding mechanism works with a simple addressing scheme, the performance of these protocols (especially end-to-end latency) need to be studied thoroughly under IP-based addressing scheme. While the routing protocols in this thesis can work with heterogeneous hardware that consumes energy at different rates, we tested only on testbeds with nodes having similar energy consumption profiles. Further deployment studies are required to stress test our protocols in various scenarios, where devices will spend different amount of energy for their operations. As devices cooperate among themselves for data delivery, care should be taken such that all the devices spend energy proportionately. A global optimal usage of device resources is a hard problem, where priority of applications, deployment difficulties, priority of data, etc., need to be considered.

As many IoT applications require a time-bound data delivery, we developed Rapid that provides a real-time guarantee for a large number of flows along with higher energy efficiency. It assumes a fixed priority for all the flows. In practice, different flows can have different priority. How to adapt Rapid for such scenarios needs to be studied. Moreover, Rapid assumes a relatively good and stable channel condition and operates on a single channel. However, some of the operating channels of IEEE 802.15.4 radios may incur high external interference. A further improvement of Rapid is required in terms of channel blacklisting and channel switching.

Security is another important aspect that needs to be addressed extensively in IoT systems. Existing security protocols for resource-rich devices are inefficient for IoT devices due to the resource constraints as well as their nature of frequent small bursts of traffic. A new set of security mechanisms needs to be developed that are not only energy efficient, but also ensure the highest level of security. Moreover, the security mechanisms should not cause excessive delay in packet delivery for real-time applications.

Epilogue

While listing the contributions of this thesis, we can observe that virtualization, not limiting to a device, can indeed work across different IoT applications and it can be easily applied to other verticals. It is also evident that with multiple devices participating, significant resource optimization could be achieved through virtualization. Various protocols we proposed and implemented for connecting multiple devices can support each other independent of their primary applications. We thus enable not only virtualization of devices but virtualizing the IoT itself. While we agree that we have not solved all the challenges/issues in virtualizing the IoT, we do claim that we have taken major steps forward.

Bibliography

[1] "BUTLER," http://www.iot-butler.eu/, [Online].

[2] "COMPOSE," http://www.compose-project.eu, [Online].

[3] "Ericsson Mobility Report: On the Pulse of the Networked Society [Online]," http://www.ericsson.com/res/docs/2016/ericsson-mobility-report-2016.pdf, accessed: 3-June-2016.

[4] "FIRE," http://www.ict-fire.eu/, [Online].

[5] "iCore: Empowering IoT Through Cognitive Technologies," http://www.iot-icore.eu/, [Online].

[6] "IoT-A," http://www.iot-a.eu/, [Online].

[7] "Measurement and Instrumentation Data Center," http://www.nrel.gov/midc/, [Online].

[8] "Meteorologisk Institutt," http://www.yr.no/, [Online].

[9] "Monash Set-Points trial," http://fsd.monash.edu.au/environmental-sustainability/environmental-issues/set-points-faqs, [Online].

[10] "U.S. Energy Information Administration," http://www.c2es.org/technology/overview/buildings, [Online].

[11] H. M. Ammari and S. K. Das, "Centralized and clustered k-coverage protocols for wireless sensor networks," *Computers, IEEE Transactions on*, vol. 61, no. 1, pp. 118–133, 2012.

[12] L. Atzori, A. Iera, and G. Morabito, "SIoT: Giving a social structure to the internet of things," *Communications Letters, IEEE*, vol. 15, no. 11, pp. 1193–1195, 2011.

[13] C. Basu, J. J. Caubel, K. Kim, E. Cheng, A. Dhinakaran, A. M. Agogino, R. Martin *et al.*, "Sensor-based predictive modeling for smart lighting in grid-integrated buildings," *Sensors Journal, IEEE*, vol. 14, no. 12, pp. 4216–4229, 2014.

[14] S. Beier, T. Grandison, K. Kailing, and R. Rantzau, "Discovery services-enabling RFID traceability in epcglobal networks." in *COMAD*, vol. 6, 2006, pp. 214–217.

[15] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in *Embedded Networked Sensor Systems*. ACM, 2006, pp. 307–320.

[16] D. Caicedo, A. Pandharipande, and G. Leus, "Occupancy-based illumination

control of LED lighting systems," *Lighting Research and Technology*, vol. 43, no. 2, pp. 217–234, 2011.

[17] M. Cardei and J. Wu, "Energy-efficient coverage problems in wireless ad-hoc sensor networks," *Computer communications*, vol. 29, no. 4, pp. 413–420, 2006.

[18] D. Carlson, M. Chang, A. Terzis, Y. Chen, and O. Gnawali, "Forwarder selection in multi-transmitter networks," in *Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–10.

[19] R. C. Carrano, D. Passos, L. Magalhaes, and C. V. Albuquerque, "Survey and taxonomy of duty cycling mechanisms in wireless sensor networks," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 1, pp. 181–194, 2014.

[20] A. Castellani, N. Bui, P. Casari, M. Rossi, Z. Shelby, and M. Zorzi, "Architecture and protocols for the internet of things: A case study," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*. IEEE, 2010, pp. 678–683.

[21] A. Chamam and S. Pierre, "On the planning of wireless sensor networks: energy-efficient clustering under the joint routing and coverage constraint," *Mobile Computing, IEEE Transactions on*, vol. 8, no. 8, pp. 1077–1086, 2009.

[22] C.-T. Cheng, C. K. Tse, and F. Lau, "A clustering algorithm for wireless sensor networks based on social insect colonies," *Sensors Journal, IEEE*, vol. 11, no. 3, pp. 711–721, 2011.

[23] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong, "Approximate data collection in sensor networks using probabilistic models," in *IEEE Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*. IEEE, 2006, pp. 48–48.

[24] R. Cristescu and M. Vetterli, "On the optimal density for real-time data gathering of spatio-temporal processes in sensor networks," in *IPSN'05*. IEEE, 2005, p. 21.

[25] J. Crowcroft, M. Segal, and L. Levin, "Improved structures for data collection in wireless sensor networks," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 1375–1383.

[26] G. Dai and Y. Wang, "Design on architecture of internet of things," in *Advances in Computer Science and Information Engineering*. Springer, 2012, pp. 1–7.

[27] R. J. De Dear, G. S. Brager, J. Reardon, F. Nicol *et al.*, "Developing an adaptive model of thermal comfort and preference/discussion," *ASHRAE transactions*, vol. 104, p. 145, 1998.

[28] X. Deng and Y. Yang, "Cluster communication synchronization in delay-sensitive wireless sensor networks," in *Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on*. IEEE, 2013, pp. 36–43.

[29] M. Doddavenkatappa, M. C. Chan, and A. L. Ananda, "Indriya: A low-cost, 3d wireless sensor network testbed," in *Testbeds and Research Infrastructure. Development of Networks and Communities*. Springer, 2011, pp. 302–316.

[30] A. Dunkels, "The contikimac radio duty cycling protocol," 2011.

[31] A. Dunkels, B. Grönvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *29th Annual Conf. on Local Computer Networks*. IEEE, 2004, pp. 455–462.

[32] S. Duquennoy, O. Landsiedel, and T. Voigt, "Let the tree bloom: Scalable opportunistic routing with orpl," in *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2013, p. 2.

[33] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis, "Design and evaluation of a versatile and efficient receiver-initiated link layer for low-power wireless," in *Embedded Networked Sensor Systems*. ACM, 2010, pp. 1–14.

[34] V. L. Erickson and A. E. Cerpa, "Occupancy based demand response HVAC control strategy," in *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*. ACM, 2010, pp. 7–12.

[35] ——, "Thermovote: participatory sensing for efficient building hvac conditioning," in *Proceedings of the 4th ACM Workshop on Embedded Systems for Energy-Efficient Buildings*. ACM, 2012, pp. 9–16.

[36] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele, "Low-power wireless bus," in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. ACM, 2012, pp. 1–14.

[37] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with Glossy," in *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*. IEEE, 2011, pp. 73–84.

[38] A. D. Galasiu and J. A. Veitch, "Occupant preferences and satisfaction with the luminous environment and control systems in daylit offices: a literature review," *Energy and Buildings*, vol. 38, no. 7, pp. 728–742, 2006.

[39] A. Ghahramani, F. Jazizadeh, and B. Becerik-Gerber, "A knowledge based approach for selecting energy-aware and comfort-driven HVAC temperature set points," *Energy and Buildings*, vol. 85, pp. 536–548, 2014.

[40] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM conference on embedded networked sensor systems*. ACM, 2009, pp. 1–14.

[41] I. Gronbaek, "Architecture for the internet of things (IoT): Api and interconnect," in *Sensor Technologies and Applications, 2008. SENSORCOMM'08. Second International Conference on*. IEEE, 2008, pp. 802–807.

[42] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *Infor-*

mation Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposium on. IEEE, 2004, pp. 1–10.

[43] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio, "Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services," *Services Computing, IEEE Transactions on*, vol. 3, no. 3, pp. 223–235, 2010.

[44] H. Gupta, V. Navda, S. Das, and V. Chowdhary, "Efficient gathering of correlated data in sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 4, no. 1, p. 4, 2008.

[45] S. Han, X. Zhu, A. K. Mok, D. Chen, and M. Nixon, "Reliable and real-time communication in industrial wireless mesh networks," in *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2011 17th IEEE.* IEEE, 2011, pp. 3–12.

[46] S. He, J. Chen, D. K. Yau, and Y. Sun, "Cross-layer optimization of correlated data gathering in wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 11, no. 11, pp. 1678–1691, 2012.

[47] J. Hoebeke, G. Holderbeke, I. Moerman, M. Jacobsson, V. Prasad, N. CEMPAKA WANGI, I. Niemegeers, and S. Heemstra De Groot, "Personal network federations," *Proceedings of the 15th IST Mobile & Wireless Communications Summit 2006*, 2006.

[48] T. Istomin, C. Kiraly, and G. P. Picco, "Is RPL ready for actuation? a comparative evaluation in a smart city scenario," in *Wireless Sensor Networks.* Springer, 2015, pp. 291–299.

[49] H. Jiang, S. Jin, and C. Wang, "Prediction or not? an energy-efficient framework for clustering-based data collection in wireless sensor networks," *Parallel and Distributed Systems, IEEE Trans. on*, vol. 22, no. 6, pp. 1064–1071, 2011.

[50] C. Joo and N. B. Shroff, "On the delay performance of in-network aggregation in lossy wireless sensor networks," *Networking, IEEE/ACM Transactions on*, vol. 22, no. 2, pp. 662–673, 2014.

[51] D. Karaboga, S. Okdem, and C. Ozturk, "Cluster based wireless sensor network routing using artificial bee colony algorithm," *Wireless Networks*, vol. 18, no. 7, pp. 847–860, 2012.

[52] E. Karasabun, I. Korpeoglu, and C. Aykanat, "Active node determination for correlated data gathering in wireless sensor networks," *Computer Networks*, vol. 57, no. 5, pp. 1124–1138, 2013.

[53] R. M. Karp, *Reducibility among combinatorial problems.* Springer, 1972.

[54] A. Kelman, Y. Ma, and F. Borrelli, "Analysis of local optima in predictive control for energy efficient buildings," *Journal of Building Performance Simulation*, vol. 6, no. 3, pp. 236–255, 2013.

[55] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Wireless sensor networks for structural health monitoring," in *Proceedings*

of the 4th international conference on Embedded networked sensor systems. ACM, 2006, pp. 427–428.

[56] L. Klein, J.-y. Kwak, G. Kavulya, F. Jazizadeh, B. Becerik-Gerber, P. Varakantham, and M. Tambe, "Coordinating occupant behavior for building energy and comfort management using multi-agent systems," *Automation in Construction*, vol. 22, pp. 525–536, 2012.

[57] M. Kovatsch, S. Mayer, and B. Ostermaier, "Moving application logic from the firmware to the cloud: Towards the thin server architecture for the internet of things," in *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*. IEEE, 2012, pp. 751–756.

[58] A. Kusiak, M. Li, and F. Tang, "Modeling and optimization of HVAC energy consumption," *Applied Energy*, vol. 87, no. 10, pp. 3092–3102, 2010.

[59] A. H.-y. Lam, Y. Yuan, and D. Wang, "An occupant-participatory approach for thermal comfort enhancement and energy conservation in buildings," in *Proceedings of the 5th international conference on Future energy systems*. ACM, 2014, pp. 133–143.

[60] O. Landsiedel, F. Ferrari, and M. Zimmerling, "Chaos: Versatile and efficient all-to-all data sharing and in-network processing at scale," in *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2013, p. 1.

[61] O. Landsiedel, E. Ghadimi, S. Duquennoy, and M. Johansson, "Low power, low delay: opportunistic routing meets duty cycling," in *Information Processing in Sensor Networks (IPSN), 2012 ACM/IEEE 11th International Conference on*. IEEE, 2012, pp. 185–196.

[62] G. M. Lee and J. Y. Kim, "The internet of things—a problem statement," in *Information and Communication Technology Convergence (ICTC), 2010 International Conference on*. IEEE, 2010, pp. 517–518.

[63] K. Leentvaar and J. H. Flint, "The capture effect in FM receivers," *Communications, IEEE Transactions on*, vol. 24, no. 5, pp. 531–539, 1976.

[64] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks," in *NSDI'04*. USENIX Association.

[65] B. Li, Y. Ma, T. Westenbroek, C. Wu, H. Gonzalez, and C. Lu, "Wireless routing and control: a cyber-physical case study," in *ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS'16)*. ACM/IEEE, 2016.

[66] R. Lim, F. Ferrari, M. Zimmerling, C. Walser, P. Sommer, and J. Beutel, "Flocklab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems," in *Information Processing in Sensor Networks (IPSN), 2013 ACM/IEEE International Conference on*. IEEE, 2013, pp. 153–165.

[67] X.-Y. Liu, Y. Zhu, L. Kong, C. Liu, Y. Gu, A. V. Vasilakos, and M.-Y. Wu,

"CDC: Compressive data collection for wireless sensor networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 26, no. 8, pp. 2188–2197, 2015.

[68] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, and X. Li, "Does wireless sensor network scale? a measurement study on greenorbs," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 10, pp. 1983–1993, 2013.

[69] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen, "Real-time wireless sensor-actuator networks for industrial cyber-physical systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1013–1024, 2016.

[70] C. Luo, F. Wu, J. Sun, and C. W. Chen, "Compressive data gathering for large-scale wireless sensor networks," in *Proceedings of the 15th annual international conference on Mobile computing and networking*. ACM, 2009, pp. 145–156.

[71] S. Madden, "Intel Berkeley research lab data [Online]," http://db.csail.mit.edu/labdata/labdata.html, 2004, accessed: 30-Nov-2013.

[72] D. Madigan, E. Einahrawy, R. P. Martin, W.-H. Ju, P. Krishnan, and A. Krishnakumar, "Bayesian indoor positioning systems," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 2. IEEE, 2005, pp. 1217–1227.

[73] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM, 2002, pp. 88–97.

[74] F. Marcelloni and M. Vecchio, "A simple algorithm for data compression in wireless sensor networks," *Commun. Lett., IEEE*, vol. 12, no. 6, pp. 411–413, 2008.

[75] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 39–49.

[76] S. Mini, S. K. Udgata, and S. L. Sabat, "Sensor deployment and scheduling for target coverage problem in wireless sensor networks," *Sensors Journal, IEEE*, vol. 14, no. 3, pp. 636–644, 2014.

[77] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, "Routing without routes: The backpressure collection protocol," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*. ACM, 2010, pp. 279–290.

[78] D. Moss and P. Levis, "BoX-MACs: Exploiting physical and link layer boundaries in low-power networking," *Computer Systems Laboratory Stanford University*, pp. 116–119, 2008.

[79] Moteiv, "Sheet, Tmote Sky Data," 2004. [Online]. Available: http://moss.csc.ncsu.edu/~mueller/rt/rt11/readings/projects/g4/datasheet.pdf

[80] S. Nambi, C. Sarkar, R. V. Prasad, and A. Rahim, "A unified semantic knowledge base for iot," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. IEEE, 2014, pp. 575–580.

[81] H. Ning and Z. Wang, "Future internet of things architecture: like mankind neural system or social organization framework?" *Communications Letters, IEEE*, vol. 15, no. 4, pp. 461–463, 2011.

[82] D. Pan, A. H.-y. Lam, and D. Wang, "Carrying my environment with me in iot-enhanced smart buildings," in *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*. ACM, 2013, pp. 521–522.

[83] R. Paulson, C. Basu, A. M. Agogino, and S. Poll, "Inverse modeling using a wireless sensor network (wsn) for personalized daylight harvesting." in *SENSORNETS*, 2013, pp. 213–221.

[84] D. Pisharoty, R. Yang, M. W. Newman, and K. Whitehouse, "ThermoCoach: Reducing home energy consumption with personalized thermostat recommendations," in *Proceedings of the 2nd ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, 2015, pp. 201–210.

[85] W.-B. Pöttner, H. Seidel, J. Brown, U. Roedig, and L. Wolf, "Constructing schedules for time-critical data delivery in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 3, p. 44, 2014.

[86] R. V. Prasad, C. Sarkar, V. S. Rao, A. R. Biswas, and I. Niemegeers, "Opportunistic service provisioning in the future internet using cognitive service approximation," in *28th WWRF Meeting, Athens, Greece*, 2012.

[87] D. Puccinelli, S. Giordano, M. Zuniga, and P. J. Marrón, "Broadcast-free collection protocol," in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. ACM, 2012, pp. 29–42.

[88] Z. Quan, W. J. Kaiser, and A. H. Sayed, "A spatial sampling scheme based on innovations diffusion in sensor networks," in *Proceedings of the 6th international conference on Information processing in sensor networks*. ACM, 2007, pp. 323–330.

[89] R. T. Rajan, M. Bentum, and A.-J. Boonstra, "Synchronization for space based ultra low frequency interferometry," in *IEEE Aerospace Conference*, 3 2013, pp. 1–8.

[90] R. T. Rajan and A.-J. van der Veen, "Joint ranging and synchronization for an anchorless network of mobile nodes," *IEEE Transactions on Signal Processing,*, vol. 63, no. 8, pp. 1925–1940, 4 2015.

[91] A. S. Rao, M. P. Georgeff *et al.*, "BDI agents: From theory to practice." in *ICMAS*, vol. 95, 1995, pp. 312–319.

[92] V. S. Rao, M. Koppal, R. V. Prasad, T. Prabhakar, C. Sarkar, and I. Niemegeers, "Murphy loves CI: Unfolding and improving constructive interference in wsns," in *Computer Communications, IEEE INFOCOM 2016-The*

35th Annual IEEE International Conference on. IEEE, 2016, pp. 1–9.

[93] A. Saifullah, D. Gunatilaka, P. Tiwari, M. Sha, C. Lu, B. Li, C. Wu, and Y. Chen, "Schedulability analysis under graph routing in wirelesshart networks," in *Real-Time Systems Symposium (RTSS), 2015 IEEE 36th*. IEEE, 2015.

[94] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-time scheduling for wirelessHART networks," in *Real-Time Systems Symposium (RTSS), 2010 IEEE 31st*. IEEE, 2010, pp. 150–159.

[95] S. Santini and K. Romer, "An adaptive strategy for quality-based data reduction in wireless sensor networks," in *INSS 2006*, 2006, pp. 29–36.

[96] C. Sarkar, "LWB and FS-LWB implementation for sky platform using contiki," arXiv preprint - https://arxiv.org/pdf/1607.06622.pdf, 2016.

[97] C. Sarkar, A. Nambi, R. V. Prasad, A. Rahim, R. Neisse, and G. Baldini, "DIAT: A scalable distributed architecture for iot," *IEEE Internet of Things Journal*, vol. 2, no. 3, pp. 230–239, 2015.

[98] C. Sarkar, V. S. Rao, and R. V. Prasad, "No-sense: Sense with dormant sensors," in *Communications (NCC), 2014 Twentieth National Conference on*. IEEE, 2014, pp. 1–6.

[99] C. Sarkar, V. S. Rao, R. V. Prasad, A. Rahim, and I. Niemegeers, "A distributed model for approximate service provisioning in internet of things," in *Proceedings of the 2012 international workshop on Self-aware internet of things*. ACM, 2012, pp. 31–36.

[100] C. Sarkar, V. S. Rao, R. Venkatesha Prasad, and K. Langendoen, "SleepRoute: Assured sensing with aggressively sleeping nodes," in *Mobile Ad Hoc and Sensor Systems (MASS), 2014 IEEE 11th International Conference on*. IEEE, 2014, pp. 237–241.

[101] C. Sarkar, V. S. Rao, R. Venkatesha Prasad, S. Narayan Das, S. Misra, and A. Vasilakos, "VSF: An energy-efficient sensing framework using virtual sensors," *IEEE Sensors Journal*, vol. 16, no. 12, pp. 5046–5059, 2016.

[102] C. Sarkar, A. U. N. S.N., and V. Prasad, "iLTC: Achieving individual comfort in shared spaces," in *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, 2016, pp. 65–76.

[103] C. Sarkar, R. Venkatesha Prasad, R. T. Rajan, and K. Langendoen, "Sleeping Beauty: Efficient communication for node scheduling," in *Mobile Ad Hoc and Sensor Systems (MASS), 2016 IEEE 13th International Conference on*, 2016, pp. 56–64.

[104] L. L. Scharf, *Statistical signal processing*. Addison-Wesley Reading, MA, 1991, vol. 98.

[105] G. Schmidt, "Getting the Most Out of Your WirelessHART System [Online]," http://www.phoenixcontact.com/, accessed: 30-April-2016.

[106] E. Serpedin and Q. M. Chaudhari, *Synchronization in Wireless Sensor*

Networks: Parameter Estimation, Peformance Benchmarks, and Protocols, 1st ed. NY, USA: Cambridge University Press, 2009.

[107] M. Shan, G. Chen, D. Luo, X. Zhu, and X. Wu, "Building maximum lifetime shortest path data aggregation trees in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 11, no. 1, p. 11, 2014.

[108] P. Soldati, H. Zhang, and M. Johansson, "Deadline-constrained transmission scheduling and data evacuation in wirelessHART networks," in *Control Conference (ECC), 2009 European*. IEEE, 2009, pp. 4320–4325.

[109] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, and M. Nixon, "WirelessHART: Applying wireless technology in real-time industrial process control," in *IEEE RTAS*, 2008, pp. 377–386.

[110] W. Specification, "75: TDMA data-link layer," *HART Communication Foundation Std., Rev*, vol. 1, 2008.

[111] P. Spiess, S. Karnouskos, D. Guinard, D. Savio, O. Baecker, L. Souza, and V. Trifa, "SOA-based integration of the internet of things in enterprise services," in *Web Services, 2009. ICWS 2009. IEEE International Conference on*. IEEE, 2009, pp. 968–975.

[112] L. Tan and N. Wang, "Future internet: The internet of things," in *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, vol. 5. IEEE, 2010, pp. 376–380.

[113] J. R. Taylor and E. Cohen, "An introduction to error analysis: the study of uncertainties in physical measurements," *Measurement Science and Technology*, vol. 9, no. 6, p. 1015, 1998.

[114] J. A. Torkestani, "An adaptive energy-efficient area coverage algorithm for wireless sensor networks," *Ad hoc networks*, vol. 11, no. 6, pp. 1655–1666, 2013.

[115] N. Tsiftes, J. Eriksson, and A. Dunkels, "Low-power wireless IPv6 routing with contikirpl," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*. ACM, 2010, pp. 406–407.

[116] D. Tulone and S. Madden, "PAQ: Time series forecasting for approximate query answering in sensor networks," in *Wireless Sensor Networks*. Springer, 2006, pp. 21–37.

[117] L. A. Villas, A. Boukerche, H. A. De Oliveira, R. B. De Araujo, and A. A. Loureiro, "A spatial correlation aware algorithm to perform efficient data collection in wireless sensor networks," *Ad Hoc Networks*, vol. 12, pp. 69–85, 2014.

[118] J. C. Vischer, "The effects of the physical environment on job performance: towards a theoretical model of workspace stress," *Stress and Health*, vol. 23, no. 3, pp. 175–184, 2007.

[119] B. Wang, "Coverage problems in sensor networks: A survey," *ACM Computing Surveys (CSUR)*, vol. 43, no. 4, p. 32, 2011.

[120] N. Wangi, R. V. Prasad, M. Jacobsson, and I. Niemegeers, "Address autoconfiguration in wireless ad hoc networks: Protocols and techniques," *Wireless Communications, IEEE*, vol. 15, no. 1, pp. 70–80, 2008.

[121] M. Weiser, "The computer for the twenty-first century (pp. 94-100)," *Scientific American, September Issue*, 1991.

[122] Y.-J. Wen and A. M. Agogino, "Wireless networked lighting systems for optimizing energy savings and user satisfaction," in *Wireless Hive Networks Conference*. IEEE, 2008, pp. 1–7.

[123] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *Internet Computing, IEEE*, vol. 10, no. 2, pp. 18–25, 2006.

[124] M. Wright, "Philips Lighting questions proper light-level standards for office workers," http://www.ledsmagazine.com/, 2015.

[125] L. Xiang, J. Luo, and A. Vasilakos, "Compressed data aggregation for energy efficient wireless sensor networks," in *Sensor, mesh and ad hoc communications and networks (SECON), 8th IEEE conference on*, 2011, pp. 46–54.

[126] X. Xu, R. Ansari, A. Khokhar, and A. V. Vasilakos, "Hierarchical data aggregation using compressive sensing (hdacs) in wsns," *ACM Transactions on Sensor Networks (TOSN)*, vol. 11, no. 3, p. 45, 2015.

[127] Y. Yao, Q. Cao, and A. V. Vasilakos, "EDAL: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for wireless sensor networks," in *IEEE MASS*, 2013, pp. 182–190.

[128] S. Yoon and C. Shahabi, "The Clustered AGgregation (CAG) technique leveraging spatial and temporal correlations in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 3, no. 1, p. 3, 2007.

[129] Y. Yoon and Y.-H. Kim, "An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks," *Cybernetics, IEEE Transactions on*, vol. 43, no. 5, pp. 1473–1483, 2013.

[130] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 3, no. 4, pp. 366–379, 2004.

[131] Y. Yuan, D. Pan, D. Wang, X. Xu, Y. Peng, X. Peng, and P.-J. Wan, "A study towards applying thermal inertia for energy conservation in rooms," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 1, 2013.

[132] L. Zhang, A. H.-y. Lam, and D. Wang, "Strategy-proof thermal comfort voting in buildings," in *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, 2014, pp. 160–163.

[133] Q. Zhao and M. Gurusamy, "Lifetime maximization for connected target coverage in wireless sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 6, pp. 1378–1391, 2008.

Acknowledgments

This thesis is a direct by-product of me spending some phenomenal time around some awe-inspiring people over last four and half years. However, I believe there are many other people that I came across since my mortal inception in this world, who also inspired me, and thus contributed in bits and parts to this thesis. I want to take this opportunity to sincerely thank all of them.

First and foremost, I want to thank my two supervisors, Dr. R. Venkatesha Prasad (VP) and Prof. Dr. K.G. Langendoen (Koen) to guide me in this entire process. Though 'guidance' seems to be a limiting word as compared to their contributions, I am using its meaning to the fullest in this context. Being my daily supervisor, I interacted with VP every day, exploiting the meaning of 'daily' to the max. I thank him for bearing with me for the countless late evening (actually any time of the day) discussions and brainstormings, which often lead to heated debate. He was patient enough to listen to my long arguments, while carefully filtering my implausible points without negation. He fed me on numerous occasions, which kept my brain and stomach full and happy. On the other hand, Koen helped me to analyze problems in a structured way. He taught me that it is not sufficient to understand a problem myself, but one should also express it methodically. His approach to call a spade a spade, motivated me to go deeper into a problem. His bottom-up approach towards juvenile ideas helped me mature as a researcher. In that process, I also learned a few things on how to help others in their research.

A number of other colleagues and friends, who were not my official supervisors, played that role and provided a helping hand whenever needed. Sometimes they even reached out to me without an explicit call for help. So, a big thanks to Akshay, Vijay, Kishor, Venkat, and Andreas. Actually, I am hugely indebted to all the members of the embedded software group. And my buddy Mihir (Dr. Shah now), without your food and support my brain could not work.

Though I came across many excellent teachers, I would especially like to thank Nandita kakimoni, Bindu pisi, Gopal kaku, Chandan dadamoni, Bimal Basak, Prof. Avijit Kar, Dr. Kameswari Chebrolu, and Dr. Stephan Rein. At times, I might not have been a good learner, but their ability to encourage a student helped me acquire new knowledge and feel the joy of learning.

Not to mention, family and friends contributed (and still counting) to my life selflessly. So a mere 'thank' is not enough to express my gratitude. My baba (father), maa (mother), didibhai (sister), and grandparents are always by my side. My grandfather was my first teacher, and covered a great extent of my childhood.

My father always told me to stay focused without concentrating too much on the small gains and failures. On the other hand, my mother always encouraged on 'do-it-yourself' that helped to face every challenge in my life. My sister is always there as a backup and saved my back on countless occasions. Last but not least, whatever I achieved should be attributed to my wife, Anuradha. She always stood by my side, especially when I lost faith in myself. Sometimes she gave me the confidence (overconfidence!) to pursue what I thought would never be possible by me. In my third year of Ph.D., when I was really struggling, she kept my dream floating by providing the moral boost.

People often say the Dutch are so direct that others can be offended. My friends also say the same thing about me. People say the Dutch spend their money very carefully. My friends say, I do not like to spend my money at all. No wonder, I enjoyed my stay in the Netherlands. Since I spent my childhood in a small town, I do not like big cities, especially the ones we have in India. Delft was the ideal place for me to live in. Yet, the time has arrived to move on in life. I heartfully thank Delft for being a wonderful host.

Chayan Sarkar
Delft, November 2016