UNIVERSIDADE DO ALGARVE

Data Storage system for Wireless Sensor Networks

David Sacramento

Master Thesis in Computer Science Engineering

Work done under the supervision of: Prof.ª Noélia Correia

2015

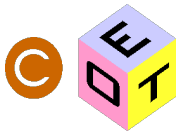
Statement of Originality

Data Storage system for Wireless Sensor Networks

Statement of authorship: The work presented in this thesis is, to the best of my knowledge and belief, original, except as acknowledged in the text. The material has not been submitted, either in whole or in part, for a degree at this or any other university.

Candidate:

(David Sacramento)



NETWORKING

Work done at Research Center of Electronics Optoelectronics and Telecommunications (CEOT)

Abstract

Wireless sensor networks (WSNs) are starting to have a high impact on our societies and, for next generation WSNs to become more integrated with the Internet, researchers recently proposed to embed Internet Protocol (IP) v6 into such very constrained networks. Also, Constrained Application Protocol (CoAP) and Observe have been proposed for RESTful services to be provided. CoAP can be used not only between nodes on the same constrained network but also with other nodes in the Internet. The latter is possible if CoAP is translated to Hyper Text Transfer Protocol (HTTP).

With CoAP/Observe, a client may store notifications in it and use a stored notification, as long as it is fresh, to serve other clients requesting it, without contacting the origin server. However, the use of CoAP/Observe for clients to be notified of resource changes, together with the possibility of incorporating proxies, raise optimization issues when multiple requests for observation are present in a network. These optimization issues involve registration steps, energy saving, consistency and caching. Registration steps must be carefully planned (which proxies to use), while keeping data consistency and timeliness, so that energy is saved.

In this thesis, a careful planning of registration steps for data/notification storage is proposed. That is a framework for the planning of such registration steps, together with the aggregation and scheduling of notification deliveries at proxies, so that energy saving is maximized, bandwidth is used efficiently and the overall delay is reduced. This framework can be applied to different types of applications. A heuristic approach is also proposed for solutions to be obtained faster.
Keywords: Wireless sensor networks, CoAP, Observe, energy saving, aggregation, scheduling.

Resumo

À medida que os sensores vão diminuindo (em tamanho), juntamente com a possibilidade de os organizar em redes, um grande número de dados poderão ser recolhidos, agregados e partilhados. Estas redes de sensores sem fios começam a causar um grande impacto na nossa sociedade. Até já se fala numa nova revolução industrial chamada *Internet of Things* (IoT). Assim, dispositivos ligados à Internet poderão ser acedidos e geridos a qualquer hora e em qualquer lugar.

Atualmente, e no contexto das *Smart Cities*, estas redes de sensores sem fios poderão desempenhar um papel importante na sociedade e indústria assim como em tarefas do dia a dia. Aplicações comuns para estas redes são: deteção de incêndios florestais, monitorização de poluição no ar ou simples captura de dados em que a possibilidade de os obter em tempo real, desde que integrados com a Internet, é uma grande vantagem.

Já existem múltiplas redes, para domínios específicos, desenvolvidas para a IoT. No entanto, a existência de diferenças na pilha protocolar faz com que sejam necessários dispositivos que assegurem a tradução e transição entre os diferentes protocolos utilizados. Por esta razão, e tendo em mente o sucesso e ubiquidade conseguida pelo protocolo IP, começou a surgir um movimento de convergência em direção a uma pilha protocolar totalmente baseada em comunicações sobre o protocolo IP. Desta forma, o grupo *Internet Engineering Task Force* (IETF) iniciou o processo de standardização de *IPv6 over Low-Power Wireless Area Networks* (6LoWPAN) para que as próximas gerações destas redes, com recursos tão limitados, estejam mais integradas com a Internet. Surge assim, e mais uma vez, o protocolo IP a garantir a interconectividade e heterogeneidade entre redes de sensores sem fios. Como complemento, foi criado um novo grupo de trabalho chamado *Routing Over Low-power and Lossy-network* (ROLL) focado nas questões de roteamento em redes de recursos limitados e baixo consumo energético. Através deste grupo foi então criado um novo protocolo de roteamento em redes IP *Routing Protocol for Low Power and Lossy Networks* (RPL), especificamente desenhado para estas redes de recursos limitados.

Outros protocolos e respetivas extensões, tais como CoAP e Observe, foram

propostos para servirem de base a serviços baseados na abordagem *Representational State Transfer* (REST). Para além de poder ser utilizado entre nós na mesma rede, também é possível a comunicação entre qualquer outro nó presente na Web uma vez que REST, por ser um pilar da Web, permite uma integração direta, desde que exista uma tradução entre os protocolos CoAP e HTTP. No contexto de comunicação *Machine-to-Machine* (M2M), CoAP também propõe o formato *CoRE Link Format* de forma a que clientes possam listar os recursos disponíveis em servidores.

CoAP e Observe, ao suportarem *caches* e *proxies*, são naturalmente capazes de fornecer serviços altamente escaláveis e eficientes. No entanto, a natureza destas redes leva a que as notificações em *cache* sejam mais voláteis e que deixe de ter interesse em tê-las em *cache* sempre que a representação do recurso seja alterada. Em contraposição com a Web, utilizando HTTP, em que a representação dos recursos é alterada com menos frequência e uma representação mais antiga desses recursos poderá ser utilizada. Por esta razão, o grande potencial na utilização de *caches* e *proxies* está no planeamento e otimização dos passos de registo e transmissão das notificações de forma a obter uma utilização eficiente da largura de banda e reduzir o consumo de energia - algo de grande preocupação neste tipo de redes.

Nesta tese, é proposto um criterioso planeamento dos passos de registo de forma a armazenar as informações/notificações. Isto é, uma *framework* para o planeamento dos, acima referidos, passos de registo que emprega técnicas de agregação e agendamento de entrega de notificações nos nós intermediários de forma a maximizar a poupança de energia, utilizar a largura de banda de forma eficiente e que o atraso global, na rede, seja reduzido. Para além desta *framework* também é proposta uma abordagem heurística de forma a obter soluções mais rapidamente. Por definição, a abordagem heurística resulta de uma técnica utilizada de forma a resolver um problema mais rapidamente que os métodos que obtêm o ótimo. No caso desta tese, o ótimo resulta de otimização matemática. Ao utilizar esta técnica existe sempre um *trade-off* de vários fatores - tempo de execução, precisão ou integridade. Nos fatores de precisão e integridade, naturalmente a abordagem heurística será sempre menos eficiente que o método de otimização matemática. Já no caso do tempo de execução, a heurística dá-nos, normalmente, soluções mais rapidamente.

A extensão ao protocolo CoAP - Observe - permite que nós cliente possam observar recursos de nós servidores e manter uma representação correta destes recursos atualizada. Assim, sempre que a representação de um recurso seja alterada, o cliente será notificado desta alteração desde que mantenha o interesse em continuar a observar o dito recurso. Dependendo da aplicação

em causa, poderá fazer mais sentido obter uma atualização da representação do recurso num intervalo de tempo pré-definido ao invés de apenas quando a representação do recurso for alterada. Tal também é possível com o Observe.

Assim, através do protocolo CoAP/Observe um cliente poderá armazenar notificações na sua *cache* e utilizar uma notificação armazenada, desde que tenha uma versão atual, de forma a disponibilizá-la a outros clientes que a requisitem, sem ser necessário estabelecer ligação ao servidor de origem.

No entanto, a utilização de CoAP/Observe para que clientes possam ser notificados da alteração nos recursos, juntamente com a possibilidade de incorporar nós intermediários, levantam questões de otimização quando existirem múltiplos pedidos para observação na rede. Estas questões envolvem o planeamento dos passos de registo, poupança de energia, consistência dos dados e *caching*. Os passos de registo terão de ser cuidadosamente planeados (quais nós intermediários a utilizar) ao mesmo tempo que é necessário garantir a consistência dos dados e o *timing* mais adequado de forma a reduzir o consumo de energia na rede.

Termos chave: Redes de sensores sem fios, CoAP, Observe, poupança de energia, agregação.

Acknowledgements

I would like to express my gratitude to my supervisor Prof.ª Noélia Correia for the patience, support, guidance and knowledge that she provided during my research and writing of this thesis.

To CEOT, that provided me the facilities needed to produce and complete my thesis.

Finally, I would like to thank my family, friends and especially Helena for the motivation and personal support during this journey.

Contents

List of Figures

List of Tables

List of Publications

D. Sacramento, G. Schütz and N. Correia: Aggregation and Scheduling in CoAP/Observe Based Wireless Sensor Networks, IEEE International Conference on Communications (ICC), June 2015.

D. Sacramento, G. Schütz and N. Correia: Dynamic Aggregation and Scheduling in CoAP/Observe Based Wireless Sensor Networks, IEEE Internet of Things Journal. (submitted)

Nomenclature

6LoWPAN	IPv6 over Low-Power Wireless Area Networks
AS-WSN	Aggregation and Scheduling in WSNs
CoAP	Constrained Application Protocol
CoRE	Constrained RESTful Environments
DTLS	Datagram Transport Layer Security
HTTP	Hyper Text Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IoT	Internet of Things
IP	Internet Protocol
M2M	Machine to Machine
MAC	Media Access Control
QoS	Quality of Service
RAM	Random Access Memory
REST	Representational State Transfer
ROLL	Routing Over Low-power and Lossy-networks
RPL	Routing Protocol for Low Power and Lossy Networks
ROM	Read Only Memory
RTT	Round Trip Time
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
WSN	Wireless Sensor Network

CHAPTER 1

Introduction

1.1 Motivation and goals

As sensors become increasingly smaller and cheaper, together with the possibility of organizing them into networks, a whole range of data can be collected, aggregated and shared [1]. This new wave, called IoT, is seen by many as a new industrial revolution. Devices connected to the Internet will be able to be reached and managed at any time and any place. This will change classic solutions in many fields, e.g. production processes will end up having a decentralized production control and trigger a paradigm shift in manufacturing.

Many special-purpose networks have been built for the IoT. However, differences in the protocol stack beyond the physical and link layers required protocol translation at gateway devices. For this reason, and due to the success and ubiquity of the IP technology, a convergence towards an all IP-based communication stack started to emerge [1, 4]. The Internet Engineering Task Force (IETF) started the standardization process of 6LoWPAN enabling IPv6 over very constrained networks [5]. The IP protocol emerges, therefore, as the glue to interconnect heterogeneous wireless networks. Besides this, an IETF working group called ROLL was created that focuses on routing issues for low power and lossy networks. The main outcome of such group is RPL, an IP routing protocol designed for low power and lossy networks [6].

Also within IETF, the Constrained RESTful Environments (CoRE) working group has been focusing on the development of a resource-oriented application framework so that data can be stored, retrieved and manipulated using a client-server protocol [1]. CoAP, a Web application transfer protocol intended to provide RESTful services in constrained nodes and networks, is the main result of such working group. In the REST model clients exchange resource representations with a server that is the authority for resource representations in its namespace. A representation may capture the current state of a resource or an intended state for a resource [7]. An extension to

CoAP, called Observe, has been proposed in [8]. Observe gives clients the ability to observe resource changes. When compared to HTTP, CoAP and Observe greatly reduce overhead in implementation complexity, bandwidth requirements, latency and help to increase reliability because a publish/subscribe model is used [1]. With HTTP the clients would have to periodically poll the server to receive updates of a resource being monitored, and the server would have to respond to every request even if there is no new data or the moment is not adequate [9]. This does not happen with CoAP/Observe.

With CoAP/Observe, a client may store notifications in it and use a stored notification, as long as it is fresh, to serve other clients requesting it, without contacting the origin server. This means that the support for caches and proxies, incorporated in Observe, allows registrations and transmission of notifications, from different clients and servers, to be carefully planned so that energy saving and bandwidth utilization is optimized. This avoids energy depletion of nodes and increases network lifetime, with a positive overall impact on delay. In this thesis, a careful planning of registration steps for data/notification storage is proposed. That is, a framework to plan registration steps through proxies, and also the aggregation and scheduling of notification deliveries at such proxies, so that maximum energy saving is achieved, bandwidth is used efficiently and the overall delay is reduced. This tool can be used offline by network managers or can be incorporated at a manager node responsible for the management of observation requests.

More specifically, the contributions of this thesis are:

- Mathematical formalization for the Aggregation and Scheduling in WSNs (AS-WSN) problem for both monitoring and machine-to-machine applications as an energy saving solution;
- As the formalization can be hard to solve or can become impractical as the WSN or number of requests for observation increase, a heuristic approach for the same problem stated above was developed as a faster solving approach;
- Proposal for incorporation of aggregation/scheduling decisions into CoAP nodes.

1.2 Related work

Contributions around CoAP and Observe are emerging and are expected to increase in a near future. In [2] an evaluation of CoAP compared to HTTP

is done. The performance evaluation is done in terms of energy consumption and response time at motes. Concerning security, significant efforts have been done to use Datagram Transport Layer Security (DTLS) to protect CoAP [12]. The experience with the authorization framework OAuth is also being investigated for IoT devices to access each other's resources and grant specific authorization flows [1, 13].

Concerning the way observations can be done, in [9] a modification to Observe is proposed to enable Quality of Service (QoS) support for timeliness. Authors argue that the server should be able to prioritize the delivery of notifications to nodes requiring it, and classify notification into critical and non-critical. Clients demand for a certain degree of QoS and servers may accept or negotiate. In [11] an alternative to normal resource observation, called conditional observation, is introduced. This is because many state changes are not significant to clients, resulting in waste of resources. With conditional observation the client, instead of deciding whether to use a value or not, would be able to tell the server the criteria for notification. This is done through the insertion of a condition option that extends the Observe option.

Here we are also concerned with transmission scheduling, as in [9], but the application of our contribution has a wider coverage. Aggregation and scheduling choices are done for all arriving observation requests, and optimized transmission plans always emerge. Priority to one of such transmission plans can be given, if it includes critical notifications. Our contribution can be applied to periodic and conditional observations, as discussed in the following chapters.

1.3 Thesis organization

The first chapter of this work clarifies the motivation for the work presented and states related work.

Chapter 2 provides background information about WSN, sensors, what are considered to be RESTful web services and an overview about the protocol stack involved in this work.

On chapter 3 the problem at hand is presented along with its definitions, motivation and a brief description of the applications dealt with.

Chapter 4 contains a mathematical formalization that can be seen as the first approach to solve the problem for a monitoring application.

Chapter 5 is a more comprehensive version than chapter 4, featuring an improved mathematical formalization able to deal with two different applica-

tions - monitoring and M2M - and a heuristic approach for the problem.

Finally, in chapter 6 results for both monitoring and M2M applications of the framework are discussed and some conclusions are drawn.

CHAPTER 2

Wireless Sensor Networks

2.1 Sensor Nodes

Sensor nodes are low power devices equipped with one or more sensors, a processor, memory, a power supply, a radio, and/or an actuator (see Figure 2.1) . Various types of sensors (mechanical, thermal, biological, chemical, optical . . .) can be attached to the node in order to measure properties of the environment.

Since the sensor nodes have limited memory and are typically deployed in difficult-to-access locations, a radio is implemented for wireless communication to transfer the data to a base station.

Battery is the main power source in a sensor node. Secondary power supply that harvests power from the environment such as solar panels may be added to the node depending on the appropriateness of the environment where the sensor will be deployed. Depending on the application and the type of sensors used, actuators may be incorporated in the sensors.

A WSN typically has little or no infrastructure (see Figure 2.2). It consists of a number of sensor nodes (few tens to thousands) working together to monitor a region to obtain data about the environment. Sensor nodes may be deployed in an ad hoc manner into the field. [16]

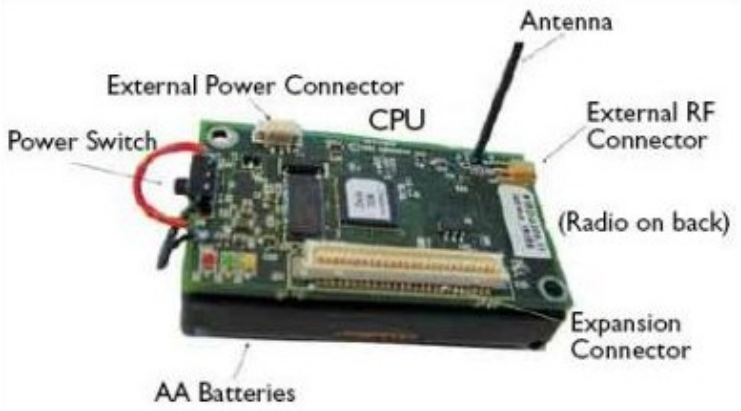


Figure 2.1: A typical sensor node.

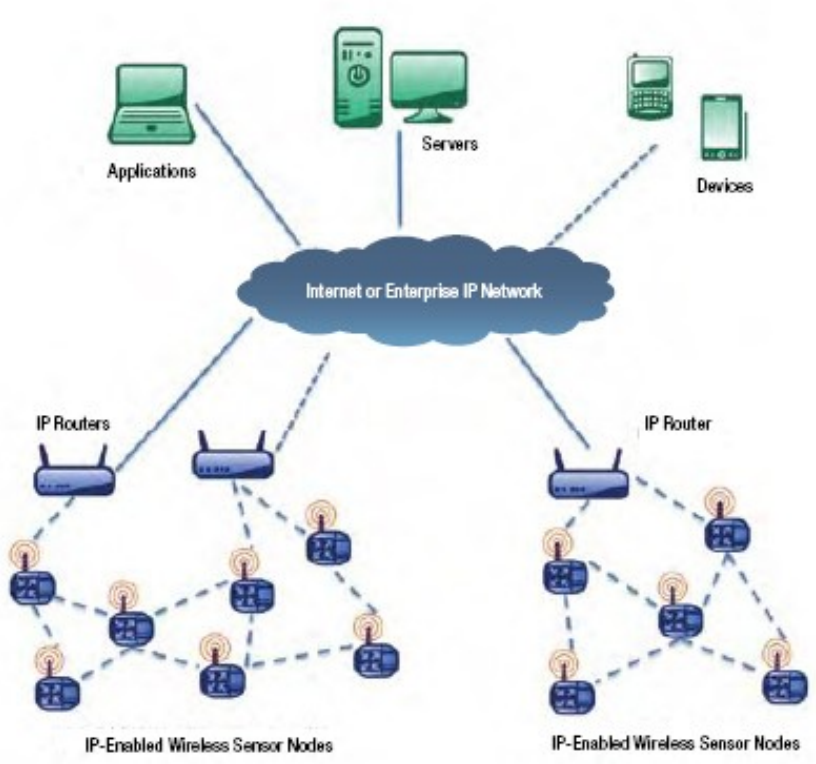


Figure 2.2: Representation of IP-enabled WSN's connected to the Internet.

2.2 Protocol Stack

As depicted in Figure 2.3, the protocol stack involved in WSNs consists of several layers. Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 protocol is responsible for the physical and Media Access Control (MAC) layers, focusing on low-cost and low-speed communication between devices leading to lower power consumption and featuring low transfer rates - up to 250kbit/s.

IPv6/6LoWPAN is present in network and adaptation layers. Bellow, on subsection 2.2.1, an overview of this standard is shown.

Unlike Transmission Control Protocol (TCP), User Datagram Protocol (UDP) is connectionless and, therefore without a connection state to be maintained, so memory size/usage is not much of an issue. And because a UDP transaction requires only two UDP datagrams, one in each direction, load on the network is minimized, further reducing response times and energy consumption. For those reasons, UDP is a natural choice for the transport layer on such constrained networks.

Finally, CoAP is the protocol used on the application layer and further discussion about CoAP can be read on subsection 2.2.2.

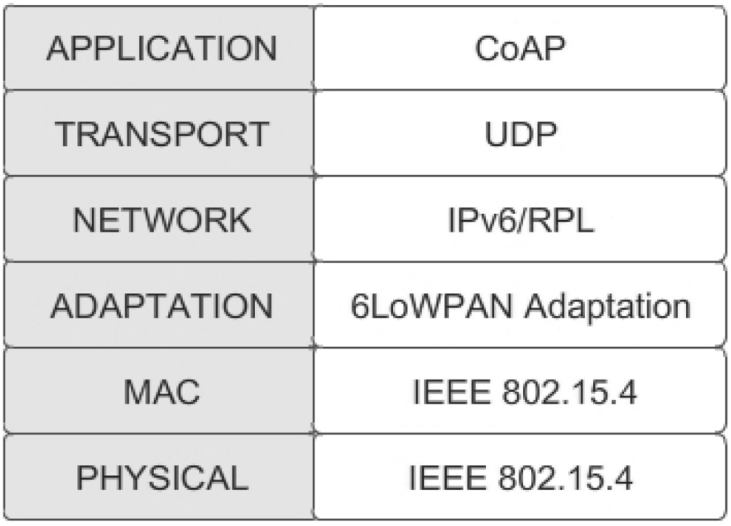


Figure 2.3: Representation of the protocol stack commonly used in WSNs.

2.2.1 6LoWPAN

6LoWPAN enables IPv6 packets communication over an IEEE 802.15.4 based network. Low power devices can communicate directly with IP devices using IP based protocols. Using 6LoWPAN, low power devices have all the benefits of IP communication and management. 6LoWPAN standard provides

an adaptation layer, new packet format, and address management. Because IPv6 packet sizes are much larger than the frame size of IEEE 802.15.4, an adaptation layer is used. The adaptation layer carries out the functionality for header compression. With header compression, smaller packets are created to fit into an IEEE 802.15.4 frame size. Address management mechanism handles the forming of device addresses for communication. 6LoWPAN is designed for applications with low data rate devices that require Internet communication. [16]

2.2.2 CoAP

CoAP is a specialized web transfer protocol for use with constrained nodes and constrained (e.g., low-power, lossy) networks. Before getting into the details of CoAP, RESTful web services are first discussed. This will allow for a better understanding of CoAP, a web application protocol intended to provide RESTful services in constrained nodes and networks.

RESTful web services

REST can be described as an architectural style for distributed hypermedia systems consisting of a set of constraints - the web's constraints. REST is the underlying architecture style of the Web, so applying REST principles means direct integration to the Web. Rather than focusing on functions, RESTful Web services use resources as the main abstraction.

Fielding's [22] five major constraints:

Uniform interface

- Each resource is identified in request using a Uniform Resource Identifier (URI) and these resources are conceptually separate from the representations that are sent to the client;
- Clients deliver state via body contents, query-string parameters, request headers and the URI. Servers deliver state via body content, response codes and response headers.

Stateless

- The server holds no memory or server-side storage of the client's state.

Cacheable

- Clients can cache responses improving scalability and performance.

Client-Server

- Separation of concerns between client and server. Clients are not concerned about how the data is stored on the server and servers are not concerned with the client's state.

Layered system

- Clients cannot tell whether it is connected to the end server or to an intermediary. This can improve scalability by enabling load-balancing.

CoAP details

As previously said, CoAP is a web transfer protocol for constrained nodes and networks. Therefore, the nodes often have 8-bit micro-controllers with small amounts of Read Only Memory (ROM) and Random Access Memory (RAM), while constrained networks such as 6LoWPAN often have high packet error rates and a typical throughput of 10s of kbit/s. The protocol is designed for M2M applications such as smart energy and building automation.

CoAP provides a request/response interaction model between application endpoints, supports built-in discovery of services and resources, and includes key concepts of the Web such as URIs and Internet media types. CoAP is designed to easily interface with HTTP for integration with the Web while meeting specialized requirements such as multicast support, very low overhead and simplicity for constrained environments. [7]

CoAP seeks to apply the same application transfer paradigm and basic features of HTTP to constrained networks, while maintaining a simple design and low overhead. Unlike HTTP, CoAP uses UDP as transport protocol. This choice would enable CoAP to have a low impact on the limited bandwidth of the 802.15.4 wireless links. However, since UDP is an unreliable protocol, CoAP has to implement its own mechanisms in order to guarantee reliability to those applications that use it.

CoAP proposes that a server should support the URI "/.well-known/core" that any client can query. A server, specifically designed for resource discovery must listen on the default port 5683.

When a client queries the server well known URI, the response contains a list of URIs. The URIs follow the CoRE Link Format [21]. The content-type of the response that contains the list of resources in CoRE Link Format is "application/link-format".

The response to a GET request on /.well-known/core, on a node running CoAP server will look something like this:

```
<sensors/temp>;sz=512;title=Temperature Sensor;ct=50
```

From this response, we have the information that this particular node can respond to a request to the URI “sensors/temp” - named for humans as “Temperature Sensor” -, the response has, at maximum, 512 bytes and the content-type of the payload is 50 (application/json). This way, any sensor node can obtain a list of available resources for each other nodes.

2.2.3 CoAP Observe

For clients to be able to observe resources, and keep their representations updated over time, the Observe extension to CoAP has been recently proposed. This extension adds a mechanism that enables CoAP clients to ”observe” resources on CoAP servers. Each time the representation of the resource is changed, the client is updated as long as it keeps its interest on observing such resource [8]. In Observe:

- An observer/client must register separately to each subject component. A subject component is a resource in the namespace of a CoAP server. Each subject component manages its list of registered observers. A client cannot register more than once for the same subject component.
- When registering, the observer/client may request periodic or threshold activated notifications. Usually periodic notifications are sent in non confirmable CoAP messages while threshold activated notifications are sent in confirmable CoAP messages. When using Observe such CoAP messages include an Observe option. In requests for threshold activated notifications a set of conditions, that might activate the sending of notifications, can be defined [11].

As illustrated in Figure 2.4, an observer can register its interest at an intermediate node, a proxy, allowing multiple registration steps/hops in large scale networks where the subject is away from the observer [9]. Concerning timeliness, a maximum age value is inserted in notifications to indicate its validity.

Registration and Notification Procedures

Whenever an observer is interested in a subject/resource, an extended GET request is either sent to the server having such resource in its namespace or

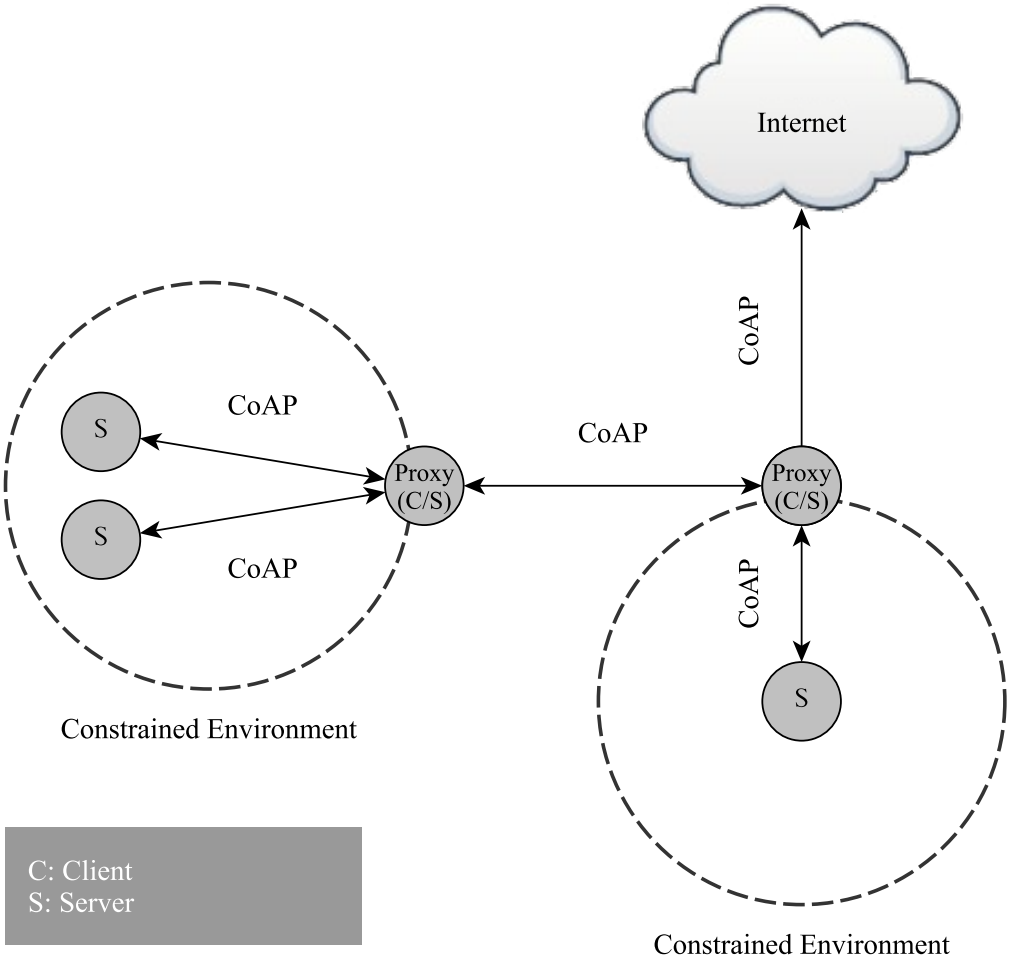


Figure 2.4: CoAP/Observe based WSN. Proxies can be used for scalability.

a proxy is used as an intermediate. The server, or proxy, will register the client as an observer, so that the client starts to receive notifications, and responds with an extended response. Extended requests/responses are CoAP requests/responses with an Observe option inside them. In Figure 2.5 the registration process using a proxy as intermediate is illustrated. As previously said, multiple proxies in cascade can be used for scalability. This model allows notifications to be kept in cache until they no not expire, which is controlled by the max-age CoAP option.

As shown in Figure 2.5, the 'Observe:' option value is basically a sequence number, of interest for responses. There is also a 'Token:' option used to match multiple notifications with a subject registration. Observation denials from server and observation deletes from observers can be issued when the Observe option is removed from messages.

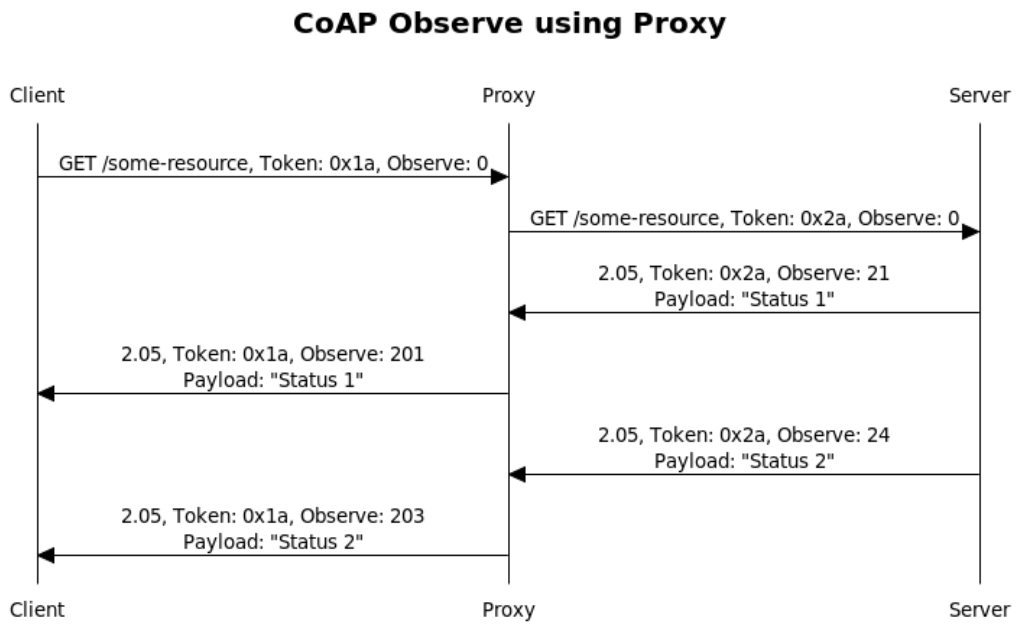


Figure 2.5: Illustration of client registration through a proxy.

Optimization Issues

The observing possibilities of CoAP/Observe raise optimization issues when there are multiple notification requests in a network. Such optimization issues involve the registration steps to be done, energy saving, consistency and caching. Usually the major concerns include:

- Energy - Proxies can be used as intermediate nodes for scalability purposes. A proxy server will act on behalf of clients registered in it, requesting resource notifications, as a client, to subject components at CoAP servers. Depending on the maximum age value at notifications from multiple/different subject components, a single aggregated transmission may be done, containing all data, toward a client or, instead, multiple transmissions toward different destinations can be scheduled for transmission in batch, avoiding multiple wakeups. This increases energy saving.

- Consistency - Resource state observed by clients should be close to the actual resource state at the server. A maximum age is inserted in notifications that indicates the maximum allowed out of sync. That is, when the age of a notification reaches this limit then the resource representation cannot be used. This means that local proxy decisions, e.g. wait for expected notification, with the purpose of aggregating/scheduling multiple transmissions must have the maximum age into consideration.

- Registrations and caching - Different registration processes may share intermediate proxies in order to avoid multiple inquiries to servers, and take the best of caching. A client cannot register more than once for the same subject component, meaning that a proxy acting as client must aggregate requests whenever the resource target is the same.

For congestion control it is advised not to send more than one non-confirmable notification per Round Trip Time (RTT) to a client, in average, and to wait for the acknowledgement of a single confirmable message. For congestion control, or to improve energy and bandwidth usage, registrations may be aborted and re-established in a different way.

The framework proposed in this work takes these issues into consideration. For each arriving observation request the best registration steps is found, together with aggregation and scheduling for notification deliveries, that is expected to maximize energy saving and bandwidth utilization, increasing the lifetime of nodes and reducing the overall delay.

CHAPTER 3

Aggregation and Scheduling at the Application Layer

3.1 Motivation and Definitions

CoAP and Observe are suitable to provide high scalability and efficiency through the support of caches and proxies. The nature of resources in WSNs, however, causes notifications in cache to be of no interest to others whenever the resource state changes, contrarily to the Internet using HTTP in which resources change less dynamically and an old resource representation still serves. For this reason, we believe that the great potential of caches and proxies is in optimizing registration steps and transmission of notifications, for efficient bandwidth utilization and energy saving, of major concern in highly constrained environments. This requires planning which proxies to use, and how to aggregate/schedule the transmission of notifications, for each arriving observation request.

In the following discussion it is assumed that each observation request relates to a single subject. Also, in order to consider different types of applications, it is assumed that there is a set of sources (one or more WSN nodes) that can provide notifications of a subject, denoted by $\mathcal{N}^S$, and there is also a set of destinations (one or more WSN nodes), denoted by $\mathcal{N}^D$. Notifications of an observation request can be forwarded toward one of the destinations in $\mathcal{N}^D$. Hence, $\mathcal{N}^S \subset \mathcal{N}$ and $\mathcal{N}^D \subset \mathcal{N}$, where $\mathcal{N}$ includes all nodes of the WSN (edge routers connected to the Internet also). In a M2M application $\mathcal{N}^D$ would have a single node inside the WSN, while in a monitoring application $\mathcal{N}^D$ will include the edge routers connected to the Internet, meaning that one of such routers can be used to forward data to any client outside the WSN. If many nodes can provide the subject being requested (subject is in its namespace or in its cache) then $\mathcal{N}^S$ includes these nodes.

Notifications generated at a server node must reach one of the destinations, using or not other internal nodes as proxies. The goal, whenever a new

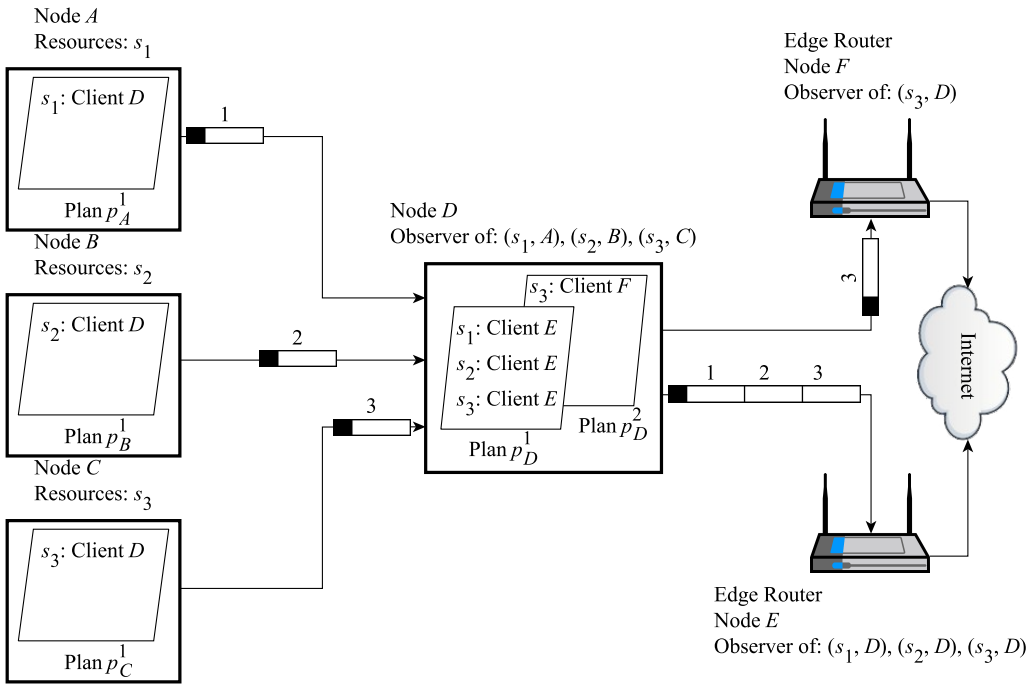


Figure 3.1: Illustration of node transmission plans.

request for observation arrives, is to determine the best registration steps, through proxies, for an adequate forwarding of notifications and best aggregation/scheduling, at such proxies, that will lead to resource optimization. In the following sections a mathematical formalization of such dynamic problem is discussed. To better understand this problem a few definitions must be introduced. For aggregation and scheduling of notification transmissions, each node will have transmission plans. Each plan includes a set of observations, whose notifications will be aggregated/scheduled when transmitted. Thus, a plan can be seen as an aggregated/scheduled transmission entity. This is illustrated in Figure 3.1.

Definition 1 (Node Transmission Plan) *A transmission plan at node n_i, denoted by $p_{n_i}^k$, includes a set of observation requests. The notifications of observation requests included in a transmission plan $p_{n_i}^k$ will be either aggregated or scheduled together for transmission.*

In Figure 3.1, node D is an observer of resources s_1, s_2 and s_3 at nodes A, B and C, respectively. While acting as a server, the transmission plan p_D^1 of node D, for example, includes observation requests for subject s_1, s_2 and s_3,

that must be sent to the same observer/client E, allowing aggregation. Note that a transmission plan may include different destination nodes in order to avoid multiple wake ups, as transmission of notifications would be closely scheduled. However, there might exist packet size or duty cycle constraints that limit the number of entries per transmission plan. Plan p_D^2 includes a single observation request that must be sent to observer F.

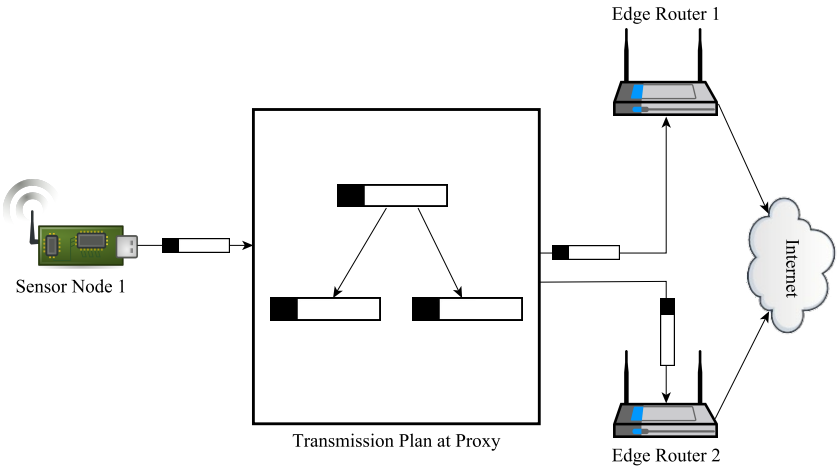
Each node transmission plan will have a transmission window associated with it.

Definition 2 (Transmission Window of a Plan) *Period, after all plan's data has been received, during which plan's data will be transmitted. Considering a plan at node n_i, denoted by $p_{n_i}^k$, the lower bound of such window, $W^L(p_{n_i}^k)$, indicates the moment when all notifications are available (arrived) for plan transmission to occur, considering no other delay besides transmission delay. The upper bound, $W^H(p_{n_i}^k)$, indicates the moment when all notifications have been sent out, considering transmission delay only.*

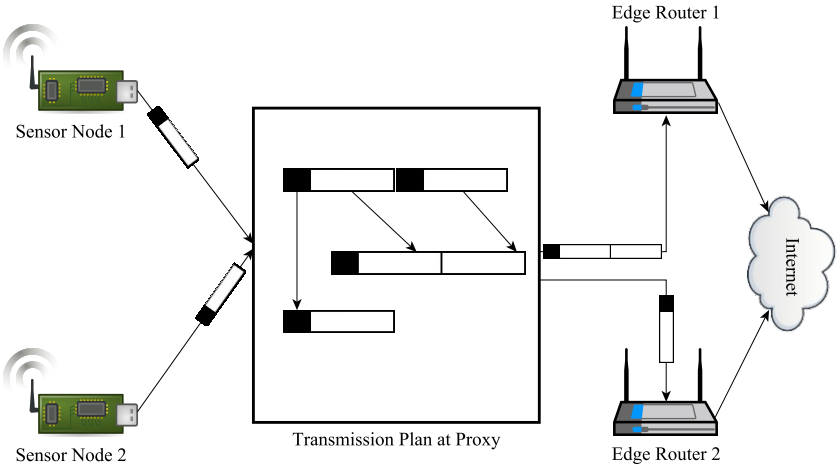
It is assumed that $W^L(p_{n_i}^k) \geq 0$, $W^H(p_{n_i}^k) \geq 0$, and that time evolves in one byte time steps. One byte time, denoted by T^{min}, refers to the time that one byte takes to be transmitted. With these definitions in mind, the problem under consideration can be defined as follows:

Definition 3 (AS-WSN Problem) *Given a WSN with a set of nodes, each node with a set of transmission plans, and considering a new arriving observation request, choose the subject server and determine the registrations to be done, at the chosen subject server and intermediate proxies, so that notifications reach one of the request's destinations while using efficiently the bandwidth and maximizing energy saving due to: i) sharing of registration steps; ii) aggregation of notifications; iii) scheduling of transmissions.*

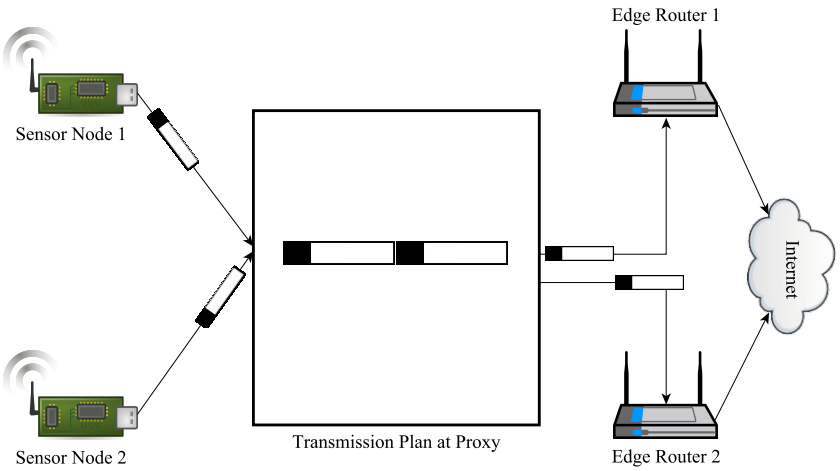
Figure 3.2a) illustrates sharing of registration (the proxy will register once at the server node, for requests arriving from edge routers 1 and 2), Figure 3.2b) illustrates aggregation of notifications (the proxy aggregates two notifications and sends a single packet to edge router 1), and Figure 3.2c) illustrates the scheduling of two transmissions (ie, the proxy waits for both notifications so that it can transmit them in a row when it wakes up).



a)



b)



c)

Figure 3.2: Illustration of scenarios: a) sharing of registration; b) aggregation of notifications; c) scheduling of transmissions.

Please note that a subject may be served by more than one node and also that a node may have different sensors. Proxies also end up having notifications in cache meaning that they can provide such notifications to others. Also note that, although higher benefits can be obtained when observation requests trigger the same frequency of notification transmissions, conditional and threshold activated observations will also benefit from such approach since notifications can meet for aggregation/scheduling somewhere in time.

3.2 Incorporating Decisions into Intermediate Proxies

After determining the best registration steps and aggregation/scheduling decisions, registration takes place. The bits of the 'Observe:' option value at registration, unused by default, can be used to mount the required registrations and build transmission plans. These bits can be used as follows:

- Bit 0: proxy indicator;
- Bit 1-3: transmission plan indicator;

For a single request for observation, one or more registration messages can be sent. Each registration message includes a Proxy-Uri option that makes reference to either a proxy or the final server. These are sent using confirmable CoAP messages and the order of their transmission is the same as the proxy chain registrations. As an example, for the sequence of registrations A-B-C-D illustrated in Figure 3.3, where A is an edge router (client), B and C are proxies and D is the sensor server node, the registration message sequence would be:

1. send message directed to C with 'Proxy-Uri:D' (D sends notifications to C; bit 0 set to 1 avoids registering A as client at C);
2. send message directed to B with 'Proxy-Uri:C' (C sends notifications to B; bit 0 set to 0 allows registering A as client at B, meaning that B will send notifications to A).

3.3 Applications

Having the previous notation in mind, the problem of planning registration steps, together with aggregation and scheduling, will be mathematically formalized for the following application cases:

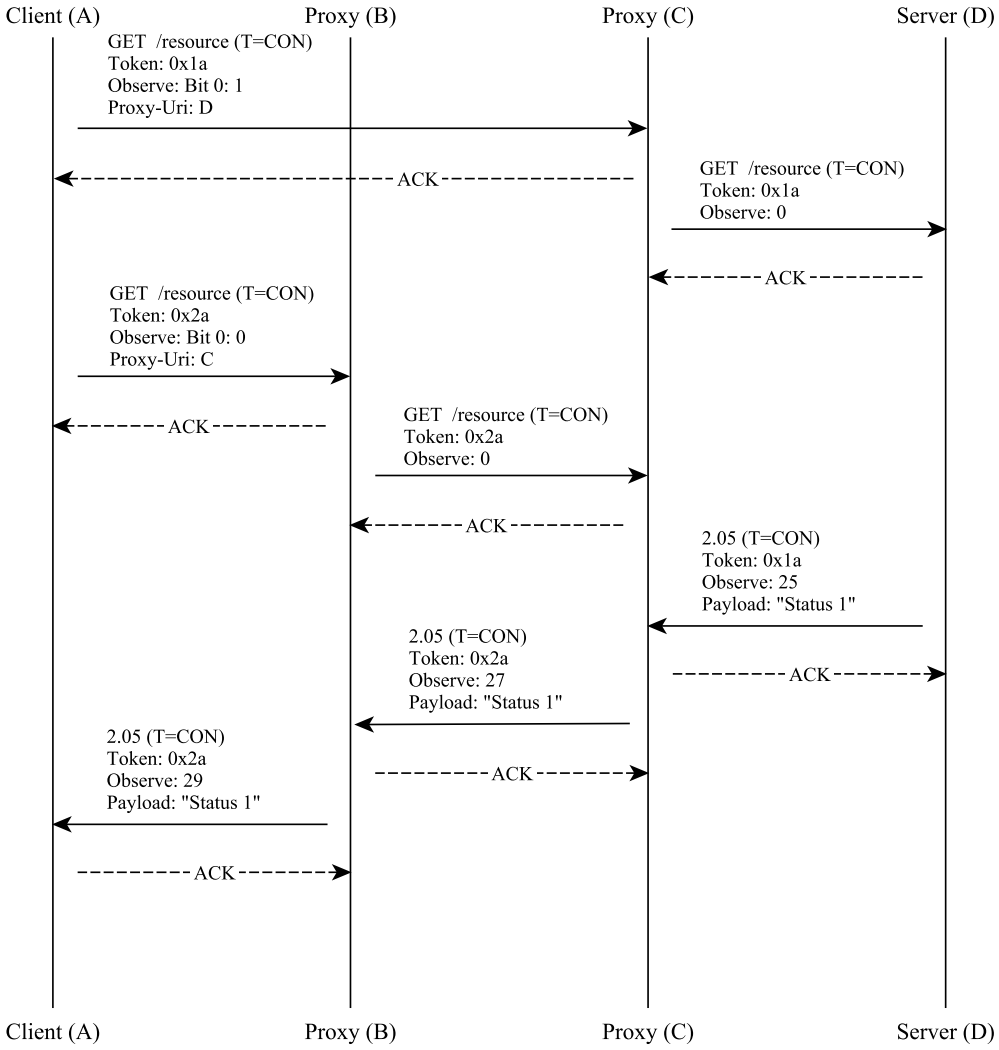


Figure 3.3: Illustration of a sequence of registrations.

Monitoring

This is a first scenario where aggregation/scheduling should be applied, in order to evaluate the effectiveness of using aggregation/scheduling for energy saving purposes. Notifications must flow to the Internet through a set of edge routers. Due to the success of the results obtained, a more general and improved framework has been developed that can be applied to any scenario of application, described next.

Monitoring and M2M

A framework version more general than the previous one, featuring a

mathematical formalization able to deal with two different applications - monitoring and M2M - and a heuristic approach for the problem.

CHAPTER 4

Monitoring Application

4.1 Problem Formalization

It is assumed that a new request for observation r' has just arrived, and that the subject/resource to be observed is given by $s(r')$. Therefore, registrations to be done at the source node and intermediate proxies need to be determined. Since a resource can be available at one or more WSN nodes, the best server must be found. The best edge router, through which notifications will flow into the Internet, is to be found also. This means that, although the first requests do not benefit from aggregation/scheduling, future requests will take advantage of such flexibility in choosing the source server and edge router. The following information is given:

$\mathcal{E}$	Set of edge routers to which notifications can be sent. Edge routers are connected to the Internet.
$\mathcal{N}$	Set of nodes internal to the WSN. A node $n \in \mathcal{N} \cup \mathcal{E}$ will have a set of already registered requests for observations, $\mathcal{R}(n)$, while acting as a client. Each $r \in \mathcal{R}(n)$ has a server source of notifications, $src(r)$, and resource $s(r)$.
$\mathcal{S}$	Set of subjects/resources available at the WSN. A resource $s \in \mathcal{S}$ has max age parameter $m(s)$.
b_s^n	One if node $n \in \mathcal{N}$ has resource s, zero otherwise.
$OWD_{n_j}^{n_i}$	One-way delay from n_i to $n_j \in \mathcal{N} \cup \mathcal{E}$; $\frac{RTT}{2}$ can be assumed.
$\mathcal{P}_n$	Set of existing transmission plans at node $n \in \mathcal{N}$. Each plan $p_n^k \in \mathcal{P}_n$ has a current lowest value for the transmission time window, $W^L(p_n^k)$, and a current highest value for the transmission time window, $W^H(p_n^k)$, both in T^{min} units. A plan p_n^k has also a set of registered observations, $\mathcal{O}(p_n^k)$, associated with it while n being a server. Each registered observation $o \in \mathcal{O}(p_n^k)$ has resource $s(o)$ and a set of clients/destinations to which the notification is sent, $\mathcal{D}(o)$;
B	Bandwidth in bytes. Thus, $T^{min} = \frac{1}{B}$.
L	Average notification packet length in bytes.
C	Duty cycle length in T^{min} units.
M	Maximum number of wakeups per node, for congestion control purposes.
G	Network density parameter.

The duty cycle C is used when accounting for energy cost. This is related with the fact that devices will turn on the entire period of C if a packet is to be transmitted [11]. It is assumed that there is always a plan in $\mathcal{P}_n$, $\forall n \in \mathcal{N}$, with no registered observations in it so that the new request can be processed individually if the transmission schedule of the other plans are restrictive. The variables used to make the planning for the new request are:

γ^n	One if node $n \in \mathcal{N}$ is the one that should provide the resource notifications, zero otherwise.
ζ^e	One if edge router $e \in \mathcal{E}$ is the one to which notifications should be sent, zero otherwise.
$\sigma_{p_{n_i}^k}^{n_i,n_j}$	One if notifications of new request flow to $n_j \in \mathcal{N} \cup \mathcal{E}$, coming from transmission plan $p_{n_i}^k$ of node $n_i \in \mathcal{N}$, in its way from source to an edge router.
$\kappa_{p_n^k}$	One if an extra packet transmission is required at transmission plan $p_n^k \in \mathcal{P}_n$, zero otherwise.
$\omega_{p_n^k}^L$	New lowest transmission time window value, in T^{min} units, in case the new request uses transmission plan $p_n^k \in \mathcal{P}_n$.
$\omega_{p_n^k}^H$	New highest transmission time window value, in T^{min} units, in case the new request uses transmission plan $p_n^k \in \mathcal{P}_n$.
λ^n	Number of wake ups (plans) at node $n \in \mathcal{N}$.

Basically, the request r' to observe resource $s(r')$ can give rise to extra packet transmissions if aggregation is not possible. This may happen at any of the multiple steps. The problem is formalized as follows:

– Objective function:

$$\text{Minimize} \sum_{n_i \in \mathcal{N}} \sum_{p_{n_i}^k \in \mathcal{P}_{n_i}} \sum_{n_j \in \{\mathcal{N} \cup \mathcal{E}\}: n_j \neq n_i} \sigma_{p_{n_i}^k}^{n_i,n_j} \times OWD_{n_j}^{n_i} \times \\ \times [1 - \left\lceil \frac{|\{o \in \mathcal{O}(p_{n_i}^k) : n_j \in \mathcal{D}(o)\}|}{|\mathcal{O}(p_{n_i}^k)|} \right\rceil] \tag{4.1}$$

With this function the shortest forwarding through the network and highest aggregation and scheduling of transmissions for energy saving is chosen. The energy cost of extra packet transmissions and/or wakeups will be accounted, after the optimizer finds the best solution, using the values found for κ and λ.

– Flow conservation law for the delivery of notifications using, if necessary, intermediate/proxy nodes:

$$\sum_{n \in \mathcal{N}} \sum_{p_n^k \in \mathcal{P}_n} \sum_{e \in \mathcal{E}} \sigma_{p_n^k}^{n,e} = 1 \tag{4.2}$$

$$\sum_{n_i \in \mathcal{N} \cup \mathcal{E}: n_i \neq n} \sum_{p_n^k \in \mathcal{P}_n} \sigma_{p_n^k}^{n,n_i} - \sum_{n_i \in \mathcal{N}: n_i \neq n} \sum_{p_{n_i}^k \in \mathcal{P}_{n_i}} \sigma_{p_{n_i}^k}^{n_i,n} = \gamma^n \times b_{s(r')}^n, \ \forall n \in \mathcal{N} \tag{4.3}$$

$$\zeta^e = \sum_{n \in \mathcal{N}} \sum_{p_n^k \in \mathcal{P}_n} \sigma_{p_n^k}^{n,e},\ \forall e \in \mathcal{E} \tag{4.4}$$

Constraints (4.2) and (4.3) ensure that notifications will flow toward an edge router using, if necessary, one or more intermediate/proxy nodes. At each node a single transmission plan is used. The source of notifications for r' is also determined, by constraints (4.3), and can only be a node having the resource being requested. Constraints (4.4) set the adequate ζ^e variable indicating the edge router to which notifications should be sent.

– Extra packet transmissions:

$$\kappa_{p_{n_i}^k} = \sum_{n_j \in \{\mathcal{N} \cup \mathcal{E}\}: n_j \neq n_i} \sigma_{p_{n_i}^k}^{n_i,n_j} \times [1 - \left\lceil \frac{|\{o \in \mathcal{O}(p_{n_i}^k) : n_j \in \mathcal{D}(o)\}|}{|\mathcal{O}(p_{n_i}^k)|} \right\rceil],\ \forall n_i \in \mathcal{N}, \\ ,\ \forall p_{n_i}^k \in \mathcal{P}_{n_i} \tag{4.5}$$

These determine the extra packet transmissions required, depending on the aggregation/forwarding done, over the multiple hops. If the new request is flowing through node n_i and plan $p_{n_i}^k$ then there will be no extra packet transmissions if the next node/hop, n_j, is already receiving notifications from n_i.

– Window control at nodes:

$$\omega_{p_{n_i}^k}^H \geq \omega_{p_{n_i}^k}^L + [W^H(p_{n_i}^k) - W^L(p_{n_i}^k)] + \kappa_{p_{n_i}^k} \times L \times T^{min},\ \forall n_i \in \mathcal{N}, \forall p_{n_i}^k \in \mathcal{P}_{n_i} \tag{4.6}$$

$$\omega_{p_{n_i}^k}^L \geq max_{n_j, p_{n_j}^l \in \mathcal{P}_{n_j} : \mathcal{X} \neq \emptyset} \{\omega_{p_{n_j}^l}^H\},\ \forall n_i \in \mathcal{N},\ \forall p_{n_i}^k \in \mathcal{P}_{n_i} \tag{4.7}$$

where $\mathcal{X} = \{o \in \mathcal{O}(p_{n_j}^l) : n_i \in \mathcal{D}(o)\}$, for the n_i, n_j, $p_{n_i}^k$ and $p_{n_j}^l$ under consideration. These constraints ensure that the time associated with the extra transmission components are included in time windows.

– Bandwidth limitation:

$$\sum_{p_n^k \in \mathcal{P}_n} (\omega_{p_n^k}^H - \omega_{p_n^k}^L) \leq \frac{1}{G},\ \forall n \in \mathcal{N} \tag{4.8}$$

where G is a network density parameter that can be adjusted according to the WSN. These constraints ensure that the bandwidth consumed by transmissions at a specific node does not exceed $\frac{1}{G}$, the available bandwidth reduced

by a factor related with the density, or degree of interference, of the network so that time division is assured.

– Max age accomplishment or out of sync avoidance:

$$\omega^H_{p^k_{src(r)}} \times G \leq m(s(r)),\ \forall e \in \mathcal{E},\ \forall r \in \mathcal{R}(e), \exists k : r \in \mathcal{O}(p^k_{src(r)}) \tag{4.9}$$

$$\omega^H_{p^k_n} \times G \leq \sigma^{n,e}_{p^k_n} \times [m(s(r')) - \Delta] + \Delta,\ \forall e \in \mathcal{E},\ \forall n \in \mathcal{N},\ \forall p^k_n \in \mathcal{P}_n \tag{4.10}$$

where Δ is a big value. Note that $src(r)$ in expression (4.9) is the node preceding the edge router, and not the source capturing the resource state, when notifications are being forwarded.

– Number of wake ups per node:

$$\lambda^{n_i} = \sum_{p^k_{n_i} \in \mathcal{P}_{n_i}} \left\lceil \frac{|\mathcal{O}(p^k_{n_i})| + \sum_{n_j \in \{\mathcal{N} \cup \mathcal{E}\} : n_j \neq n_i} \sigma^{n_i,n_j}_{p^k_{n_i}}}{\Delta} \right\rceil, \forall n_i \in \mathcal{N}$$

$$\lambda^{n_i} \leq M,\ \forall n_i \in \mathcal{N} \tag{4.11}$$

Constraints (4.11) determine the number of wake ups (non empty plans) at a node while the constraints in 4.11 limit the maximum number of wake ups on a node to M. This indirectly leads to congestion control.

– Non-negative assignments:

$$\gamma^n, \zeta^e, \sigma^{n_i,n_j}_p, \kappa_{p^k_{n_i}} \in \{0,1\}; \omega^H_{p^k_{n_i}}, \omega^L_{p^k_{n_i}} \geq 0;\ \lambda^n \in \mathbb{N}. \tag{4.12}$$

4.2 Information Update

For each arriving request the CPLEX is used to solve the previous mathematical problem. For each problem the following update steps must be performed after finding the solution:

1. For every node n_j acting as a client for r', insert new request r' into $\mathcal{R}(n_j)$, together with its info $src(r')$ and $s(r')$. That is, this is done for any n_j such that $\sigma^{n_i,n_j}_{p^k_{n_i}} \neq 0 \wedge \kappa_{p^k_{n_i}} \neq 0, \forall n_i, \forall p^k_{n_i}$;
2. For every plan $p^k_{n_i}$ used by r', where node n_i acts as a server and $\kappa_{p^k_{n_i}} \neq 0$, insert/update observation o where subject $s(o) = s(r')$ and update $\mathcal{D}(o)$;

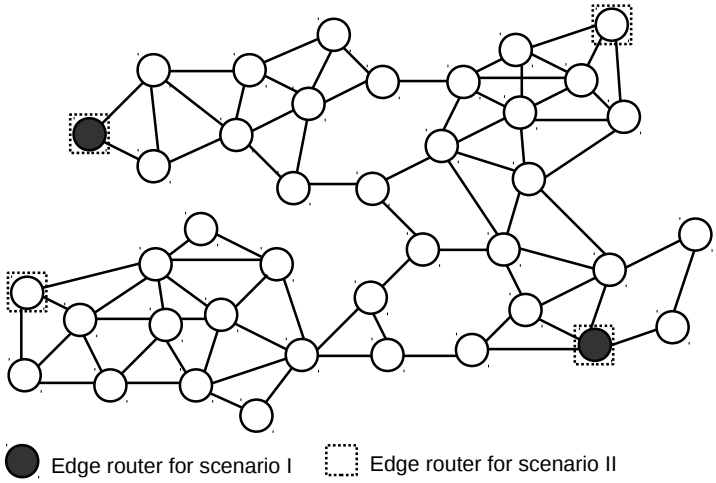


Figure 4.1: Network topology under test.

3. Update $W^H(p^k_{n_i})$ according to $\omega^H_{p^k_{n_i}}$.
4. Update $W^L(p^k_{n_i})$ according to $\omega^L_{p^k_{n_i}}$.
5. For every node n_i, if $|\mathcal{P}_{n_i}| = \lambda^{n_i}$ then there is no empty plan. Insert new empty plan in $\mathcal{P}_{n_i}$.

4.3 Performance Evaluation

4.4 Network Setup

The network under test is shown in Figure 4.1. This network has 40 nodes randomly generated using the weighted proximity algorithm in [15]. Two network scenarios were created, as illustrated in Figure 4.1:

I : Two edge routers;

II : Four edge routers.

For each network scenario, the energy consumption and average transmission delay per node are plotted considering two cases: i) one subject per node; ii) two subjects per node. Subjects are always unique.

Concerning the values for the OWD between nodes, the number of hops between the nodes is assumed for simplicity. Table 4.1 summarizes the remaining network parameters used, based on [14].

Table 4.1: Network Parameters: Monitoring Application

Parameter	Value
Bandwidth (B)	64000 bytes
Byte time (T^{min})	$\frac{1}{B} = \frac{1}{64000}s$
Packet length (L)	30 bytes
Network degree (G)	10
Duty cycle (C)	8000
Maximum wakeups (M)	8
Max age of all subjects ($m(s)$)	1s

4.5 Accounting of Energy Consumption

For energy consumption calculation it is assumed that nodes use 24mW in wake mode, 52.2mW when transmitting and 1.278mW in sleep mode, values for Z1 motes taken from [11]. Thus, the following per byte time energy consumption values can be adopted:

$E^T = 81.6 \times 10^{-8}\mathbf{J}$

$E^W = 37.5 \times 10^{-8}\mathbf{J}$

$E^S = 2.0 \times 10^{-8}\mathbf{J}$

The devices are assumed to use a listening protocol with a duty cycle of 0.125s (value for contikiMAC LPL), as in [11], leading to a duty cycle of $C = \frac{0.125s}{T^{min}} = 8000$ in byte times. For the maximum number of wakeups per node the upper bound $M = \lfloor \frac{B}{C} \rfloor = 8$ is used.

After the update steps done in Section 4.2, the total energy currently being consumed at the network can be accounted as follows. The number of byte times in transmission mode associated with plan $p^k_{n_i}$ of node $n_i \in \mathcal{N}$ is given by:

$$x(p^k_{n_i}) = |\{\mathcal{D}(o) : o \in \mathcal{O}(p^k_{n_i})\}| \times L \tag{4.13}$$

while the number of byte times in wake mode is:

$$y(p^k_{n_i}) = max\{C - x(p^k_{n_i}), 0\}. \tag{4.14}$$

The number of byte times in sleep mode for the whole node n_i will, therefore, be given by:

$$z(n_i) = B - \sum_{p^k_{n_i} \in \mathcal{P}_{n_i}} [x(p^k_{n_i}) + y(p^k_{n_i})] \tag{4.15}$$

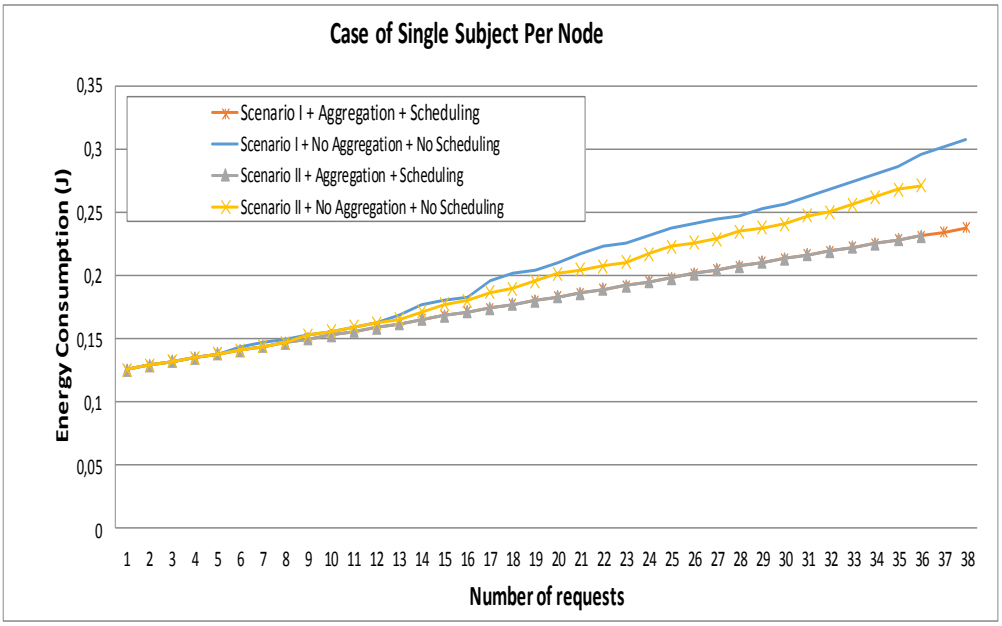


Figure 4.2: Energy consumption for the case of single subject per node.

meaning that the total energy consumption at the WSN can be calculated using:

$$E^{TOTAL} = \sum_{n_i \in N} \sum_{p^k_{n_i} \in \mathcal{P}_{n_i}} [x(p^k_{n_i}) \times E^T + y(p^k_{n_i}) \times E^W] + \\ + \sum_{n_i \in N} z(n_i) \times E^S \tag{4.16}$$

4.6 Analysis of Results

The plots include results obtained using the mathematical formalization discussed, where aggregation and scheduling is a concern, and results for a modified version of this mathematical formalization, where aggregation and scheduling are not considered. In this modified version the factor at the end of expressions (4.1) and (4.5) are removed. This means that an extra packet transmission will always exist at any node for any notification arriving.

Concerning the energy consumption results, for the cases of a single resource per node and two resources per node, shown in Figures 4.2 and 4.3, it is possible to observe that aggregation and scheduling can save energy a lot when compared with the results for no aggregation/scheduling. This is more pronounced in scenario I (two edge routers) than in scenario II (four edge routers) because in this last case there are more routing alternatives and, therefore, ways of going through less hops, saving energy. However, the ag-

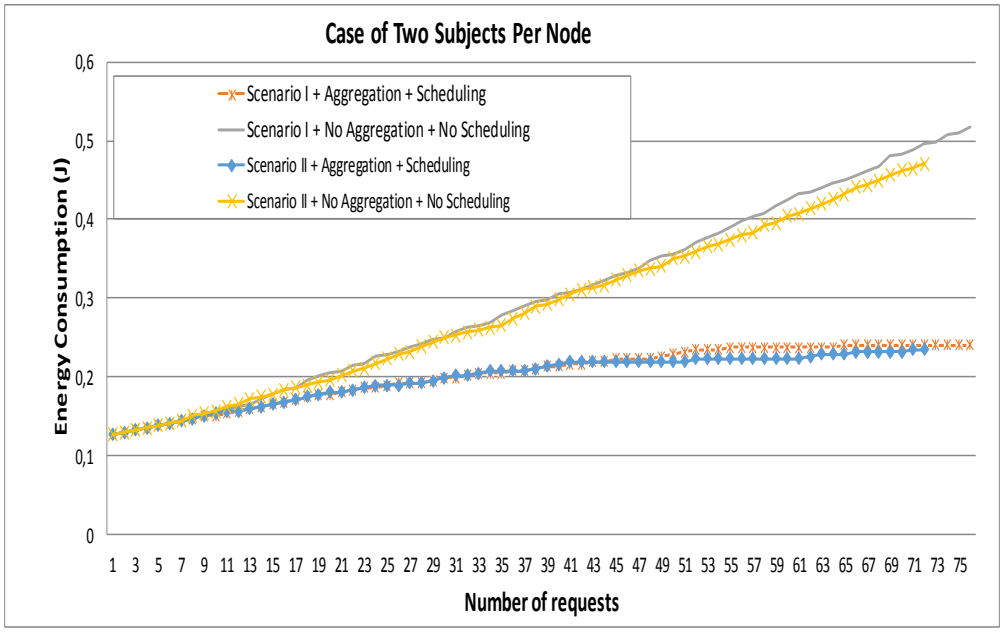


Figure 4.3: Energy consumption for the case of two subjects per node.

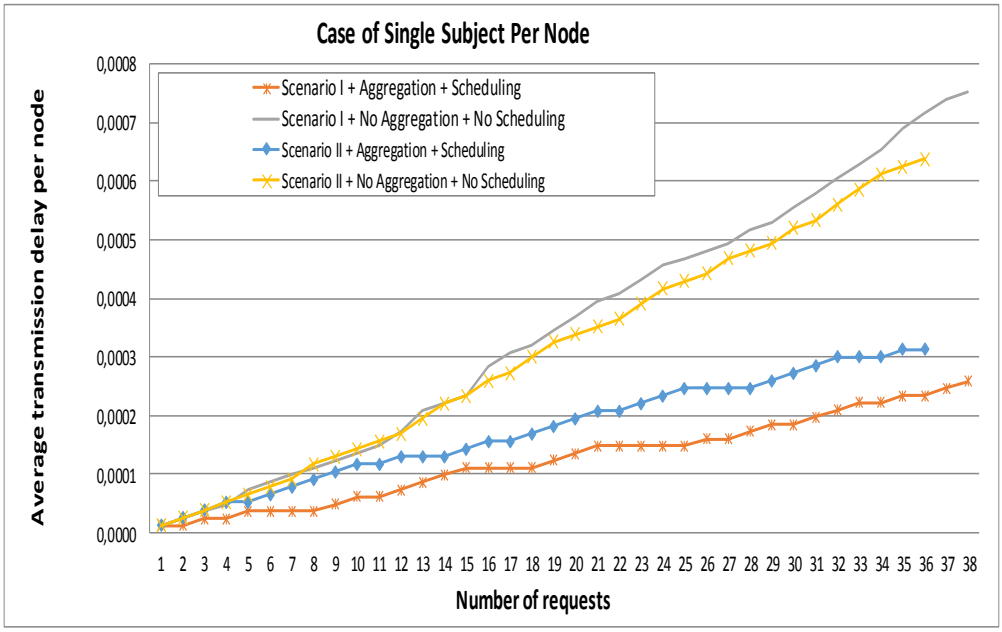


Figure 4.4: Transmission delay for the case of single subject per node.

gregation/scheduling approach is always better in terms of energy saving and becomes even better in the case of having more than one resource per node, as shown in Figure 4.3. This is so because notifications, regarding different sensors of a specific node, are aggregated at the source requiring no extra packet transmission. For the case of more than one resource per node the difference between scenarios I and II decreases.

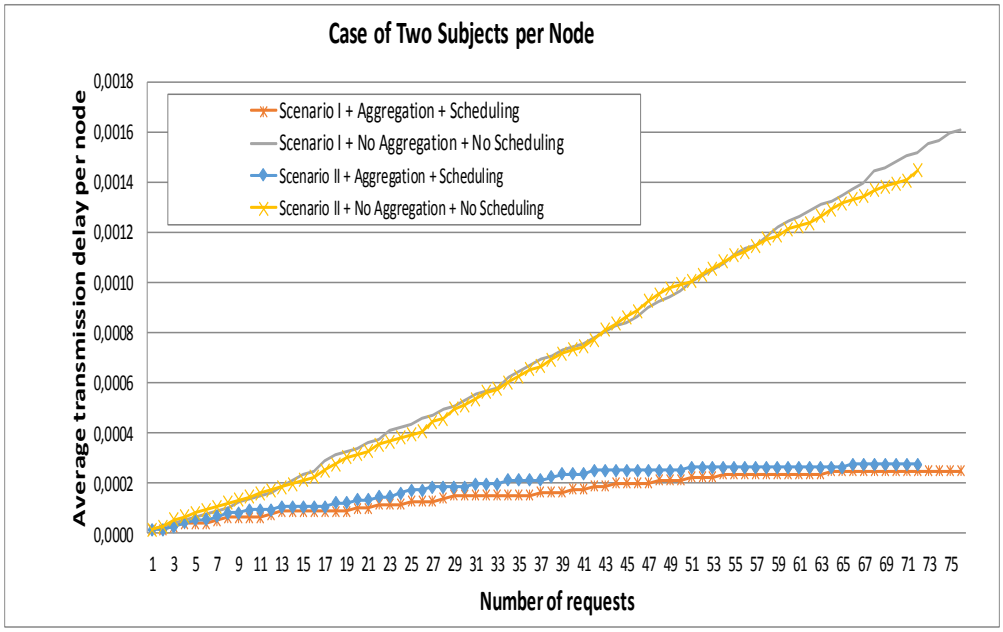


Figure 4.5: Transmission delay for the case of two subjects per node.

The average transmission delay results, for the cases of a single resource per node and two resources per node, are shown in Figures 4.4 and 4.5. These plots are in agreement with what has just been said. Transmission delay is always higher when no aggregation is done, as more packets need to be transmitted, and in scenario I this is more pronounced since more packet transmissions are required. The difference between scenarios I and II decreases when there are two resources per node. Please note that these delays include transmission delay only and not queueing or propagating delay.

Another relevant conclusion is that the energy waste and transmission delay tend to stabilize for the case of two resources per node. This happens because, as time progresses, new requests will have a higher probability of finding a source node, or intermediate nodes, able to make aggregation. The higher the number of sensors at nodes, the earlier the stabilization occurs.

CHAPTER 5

Monitoring and M2M Applications

5.1 Problem Formalization

In the following discussion $\mathcal{S}(n)$ denotes the set of subjects that node $n \in \mathcal{N}$ has in its namespace, where s is an element of such set. Node n is able, therefore, to act as a server sending notifications of any subject in $\mathcal{S}(n)$. While acting as a client/observer, n can also have a set of ongoing observations that will be denoted by $\mathcal{O}(n)$, where o is an element of such set, being able to cache notifications of such observations. Each $o \in \mathcal{O}(n)$ has a server/proxy source of notifications, $srv(o)$, and subject $sub(o)$. When acting as a server, node n will use transmission plans to send notifications. A plan at node n, denoted by p_n^k, has a set of observations registered in it, denoted by $\mathcal{O}(p_n^k)$. An observation $o \in \mathcal{O}(p_n^k)$, besides $srv(o)$ and $sub(o)$, has also a set of clients/destinations to which the notification must be sent, $\mathcal{D}(o)$.

It is assumed that a new request for observation r has just arrived, and that the subject to be observed is given by $sub(r)$. This subject is available at a set of nodes denoted by $\mathcal{N}^S = \{n \in \mathcal{N} \backslash \mathcal{N}^D : sub(r) \in \mathcal{S}(n) \vee sub(r) = sub(o), \exists o \in \mathcal{O}(n)\}$, and notifications should be sent to one of the nodes in the destination set, denoted[1] by $\mathcal{N}^D$. That is, $\mathcal{N}^S$ includes nodes with subject in its namespace and nodes with subject notifications in its cache, and the best source from this set needs to be found. The best destination, if many destinations are defined, is to be found also. Note that, although the first requests do not benefit from aggregation/scheduling, future requests will take advantage of such flexibility in choosing the source and destination. The overall required information is the following:

[1]Note that we are avoiding the use of $\mathcal{N}^S(r)$ and $\mathcal{N}^D(r)$, for a cleaner notation, but these sets change according to the request.

$\mathcal{N}$	Set of nodes of the WSN, edge routers included. A node n can have one or more subjects in its namespace, or no subject at all, given by $\mathcal{S}(n)$, and a subject $s \in \mathcal{S}(n)$ has max age parameter $m(s)$. A node $n \in \mathcal{N}$ will have a set of ongoing observations, $\mathcal{O}(n)$, while acting as a client.
b_s^n	Binary value indicating if node $n \in \mathcal{N}$ has subject s either in its namespace, $s \in \mathcal{S}(n)$, or in its cache, $\exists o \in \mathcal{O}(n) : sub(o) = s$.
$OWD_{n_j}^{n_i}$	One-way delay from n_i to $n_j \in \mathcal{N}$. Such information can be captured at the application layer as $\frac{RTT}{2}$. For simplicity a direct match with the number of physical link hops will be assumed.
$\mathcal{P}(n)$	Set of existing transmission plans at node $n \in \mathcal{N}$. Each plan $p_n^k \in \mathcal{P}(n)$ has a current lowest value for the transmission time window, $W^L(p_n^k)$, and a current highest value for the transmission time window, $W^H(p_n^k)$, both in T^{min} units. A plan p_n^k has also a set of ongoing observations associated with it, $\mathcal{O}(p_n^k)$, while n acting as a server;
L	Average notification packet length in bytes.
G	Network density parameter.

It is assumed that there is always a plan in $\mathcal{P}(n)$, $\forall n \in \mathcal{N}$, with no registered observations in it so that the new request for observation can be processed individually if the other plans are not able to accommodative more requests. Congestion control, referred in Section 2.2.3, will be achieved through parameter G, as it will become more clear next. Note that input information will change as requests arrive and corresponding registrations are established within the network. The variables used to plan how to accommodate the new request are:

γ^n	One if node $n \in \mathcal{N}$ is the one to provide the subject notifications, zero otherwise.
ζ^n	One if node $n \in \mathcal{N}^D$ is the destination to which notifications will be sent, zero otherwise.
$\sigma_{p_{n_i}^k}^{n_i,n_j}$	One if notifications will flow to $n_j \in \mathcal{N} \backslash \mathcal{N}^S$, coming from transmission plan $p_{n_i}^k$ of node $n_i \in \mathcal{N} \backslash \mathcal{N}^D$, in its way from source to destination.
$\kappa_{p_n^k}$	Extra load at transmission plan $p_n^k \in \mathcal{P}(n)$, $n \in \mathcal{N}$, due to the accommodation of the new request.
$\omega_{p_n^k}^L$	New lower bound value for the transmission window of plan $p_n^k \in \mathcal{P}(n)$, $n \in \mathcal{N}$, in T^{min} units.
$\omega_{p_n^k}^H$	New upper bound value for the transmission window of plan $p_n^k \in \mathcal{P}(n)$, $n \in \mathcal{N}$, in T^{min} units.
λ^n	Number of wake ups at node $n \in \mathcal{N}$. This is the same as the total number of plans.

The values found for κ and λ variables, after solution is found by the optimizer, will allow energy cost to be accounted as discussed in Section 5.4.2.

The new request can give rise to the following extra load, in number of transmitted packets, at each forwarding step:

- 0 - No extra packet is required. Notifications of subject being requested are already present at the transmission plan's node (namespace or cache) and being sent, although toward another final destination;
- δ - Aggregation with other ongoing notifications (of other subjects) can be done if the next hop is the same. The δ value will be an aggregation overload factor due to the insertion of extra data into the packet. This is a fraction of a packet, i.e, $0 < \delta < 1$;
- 1 - An extra packet is required. Notifications of the subject being requested are not available at the transmission plan's node and no aggregation with other ongoing notifications is possible, meaning that a new packet must be created and transmitted.

This happens at every registration step, which has a transmission plan associated with it. When an extra packet is required there is still the possibility of finding a good scheduling, to reduce the number of wake ups. This is considered at the information update step discussed in Section 5.3. The AS-WSN problem is formalized as follows:

– Objective function:

$$\text{Minimize} \sum_{n_i \in \mathcal{N} \setminus \mathcal{N}^D} \sum_{p^k_{n_i} \in \mathcal{P}(n_i)} \sum_{n_j \in \mathcal{N} \setminus \mathcal{N}^S : n_j \neq n_i} \sigma^{n_i,n_j}_{p^k_{n_i}} \times \\ \times OWD^{n_i}_{n_j} \times [1 + \Delta(n_i, p^k_{n_i}, n_j, r)] \tag{5.1}$$

With this function the shortest forwarding through the network and highest saving of extra byte transmissions, for efficient bandwidth utilization and energy saving, is chosen. Concerning the Δ weight function, this is defined by:

$$\Delta(n_i, p^k_{n_i}, n_j, r) = max\{\chi(p^k_{n_i}, n_j) \times \delta, 1 - \chi(p^k_{n_i}, n_j)\} \times (1 - b^{n_j}_{sub(r)}) \tag{5.2}$$

where χ is a function that indicates if aggregation is possible, or not, at the transmission from n_i to n_j. This is defined by:

$$\chi_(p^k_{n_i}, n_j) = \left\lceil \frac{|\{o \in \mathcal{O}(p^k_{n_i}) : n_j \in \mathcal{D}(o)\}|}{|\mathcal{O}(p^k_{n_i})|} \right\rceil \tag{5.3}$$

– Flow conservation law for the delivery of notifications using, if necessary, proxy nodes:

$$\sum_{n_i \in \mathcal{N} \setminus \mathcal{N}^D} \sum_{p^k_{n_i} \in \mathcal{P}(n_i)} \sum_{n_j \in \mathcal{N}^D} \sigma^{n_i,n_j}_{p^k_{n_i}} = 1 \tag{5.4}$$

$$\sum_{n_i \in \mathcal{N} : n_i \neq n} \sum_{p^k_n \in \mathcal{P}(n)} \sigma^{n,n_i}_{p^k_n} - \sum_{n_i \in \mathcal{N} \setminus \mathcal{N}^D : n_i \neq n} \sum_{p^k_{n_i} \in \mathcal{P}(n_i)} \sigma^{n_i,n}_{p^k_{n_i}} \\ = \gamma^n \times b^n_{sub(r)}, \ \forall n \in \mathcal{N} \setminus \mathcal{N}^D \tag{5.5}$$

$$\zeta^{n_j} = \sum_{n_i \in \mathcal{N} \setminus \mathcal{N}^D} \sum_{p^k_{n_i} \in \mathcal{P}(n_i)} \sigma^{n_i,n_j}_{p^k_{n_i}}, \ \forall n_j \in \mathcal{N}^D \tag{5.6}$$

Constraint (5.4) ensures that notifications will flow toward one of the destinations in $\mathcal{N}^D$. Constraint (5.5) allows the use of intermediate proxy nodes, ensuring flow conservation. At each node a single transmission plan is used. The source of notifications for r is also determined, by constraints (5.5), and can only be a node having the subject being requested. Constraints (5.6) set the adequate ζ^{n_j} variable indicating the destination node to which notifications should be sent.

– Finding the extra load:

$$\kappa_{p^k_{n_i}} = \sum_{n_j \in \mathcal{N}\backslash\mathcal{N}^S : n_j \neq n_i} \sigma^{n_i,n_j}_{p^k_{n_i}} \times OWD^{n_i}_{n_j} \times [1 + \Delta(n_i, p^k_{n_i}, n_j, r)], \quad \forall n_i \in \mathcal{N}\backslash\mathcal{N}^D, \forall p^k_{n_i} \in \mathcal{P}(n_i) \tag{5.7}$$

These constraints determine the extra load, which depends on the forwarding and aggregation done over the multiple hops.

– Transmission windows:

$$\omega^H_{p^k_{n_i}} \geq \omega^L_{p^k_{n_i}} + [W^H(p^k_{n_i}) - W^L(p^k_{n_i})] + \kappa_{p^k_{n_i}} \times L \times T^{min}, \quad \forall n_i \in \mathcal{N}\backslash\mathcal{N}^D, \forall p^k_{n_i} \in \mathcal{P}(n_i) \tag{5.8}$$

$$\omega^L_{p^k_{n_j}} \geq max_{n_i, p^l_{n_i} \in \mathcal{P}(n_i) : \exists o \in \mathcal{O}(p^l_{n_i}) \wedge n_j \in \mathcal{D}(o)} \{\omega^H_{p^l_{n_i}}\}, \ \forall n_j \in \mathcal{N}\backslash\mathcal{N}^D, \ \forall p^k_{n_j} \in \mathcal{P}(n_j) \tag{5.9}$$

These constraints ensure that the time required for extra load to be transmitted is included in transmission windows. Note that W^H, W^L, ω^H and ω^L do not include time shifts for time division among transmissions. This is accounted through the use of a network density parameter, G, in the following constraints.

– Congestion control:

$$\sum_{p^k_n \in \mathcal{P}(n)} (\omega^H_{p^k_n} - \omega^L_{p^k_n}) \leq \frac{1}{G}, \ \forall n \in \mathcal{N}\backslash\mathcal{N}^D \tag{5.10}$$

where G is a network density parameter that can be adjusted according to the WSN. These constraints ensure that the bandwidth consumed by transmissions at a specific node does not exceed $\frac{1}{G}$, where $G = 1$ means full use of bandwidth. This is so because all the ω variables are in T^{min} units and T^{min} is the time that one byte takes to be transmitted, which is bandwidth dependent. The G value should be adjusted to avoid congestion control and should take into consideration that time division must occur when nodes are using the same frequency channel.

– Max age accomplishment or out of sync avoidance:

$$\omega^H_{p^k_{srv(o)}} \times G \leq m(sub(o)), \ \forall n \in \mathcal{N},$$

$$, \ \forall o \in \mathcal{O}(n), \exists p^k_{srv(o)} \wedge o' \in \mathcal{O}(p^k_{srv(o)}) : n \in \mathcal{D}(o') \tag{5.11}$$

$$\begin{aligned} &\omega^H_{p^k_{n_i}} \times G \leq \sigma^{n_i,n_j}_{p^k_{n_i}} \times [m(sub(r)) - \Theta] + \Theta, \\ &, \ \forall n_j \in \mathcal{N}^D, \ \forall n_i \in \mathcal{N} \backslash \mathcal{N}^D, \ \forall p^k_{n_i} \in \mathcal{P}(n_i) \end{aligned} \tag{5.12}$$

where Θ is a big value. Constraints (5.11) confirm the max age for all ongoing notifications, while constraints (5.12) confirm the max age for the new observation. The attempt to accomplish notification deliveries before max age expires, avoiding out of sync, is done through transmission windows. Note that $srv(o)$ in expression (5.11) can be a proxy or source node with the subject in its namespace.

– Number of wake ups per node:

$$\lambda^{n_i} = \sum_{p^k_{n_i} \in \mathcal{P}(n_i)} \left\lceil \frac{|\mathcal{O}(p^k_{n_i})| + \sum_{n_j \in \{\mathcal{N}\}: n_j \neq n_i} \sigma^{n_i,n_j}_{p^k_{n_i}}}{\Theta} \right\rceil, \ \forall n_i \in \mathcal{N} \backslash \mathcal{N}^D \tag{5.13}$$

$$\lambda^{n_i} \leq M, \ \forall n_i \in \mathcal{N} \tag{5.14}$$

Constraints (5.13) determine the number of wake ups (non empty plans) at a node while the constraints in (5.14) limit the maximum number of wake ups on a node to M. This indirectly leads to congestion control.

– Non-negativity assignment to variables:

$$\gamma^n, \zeta^e, \sigma^{n_i,n_j}_p \in \{0,1\}; \omega^H_{p^k_{n_i}}, \omega^L_{p^k_{n_i}}, \kappa_{p^k_{n_i}} \geq 0; \ \lambda^n \in \mathbb{N}. \tag{5.15}$$

5.2 Heuristic Approach

The previous formalization is hard to solve and can become impractical as the WSN or number of requests for observation increase. The mathematical formalization is, however, important for us to understand the problem, allowing for the development of a faster solving approach. Here we propose a heuristic algorithm that must run for each possible destination, $n_j \in \mathcal{N}^D$. For each run the heuristic will search for an admissible solution near the n_j under consideration and then increases the neighbourhood of search, if necessary. If an admissible solution is obtained for many $n_j \in \mathcal{N}^D$ then the best one (lowest cost) should be picked.

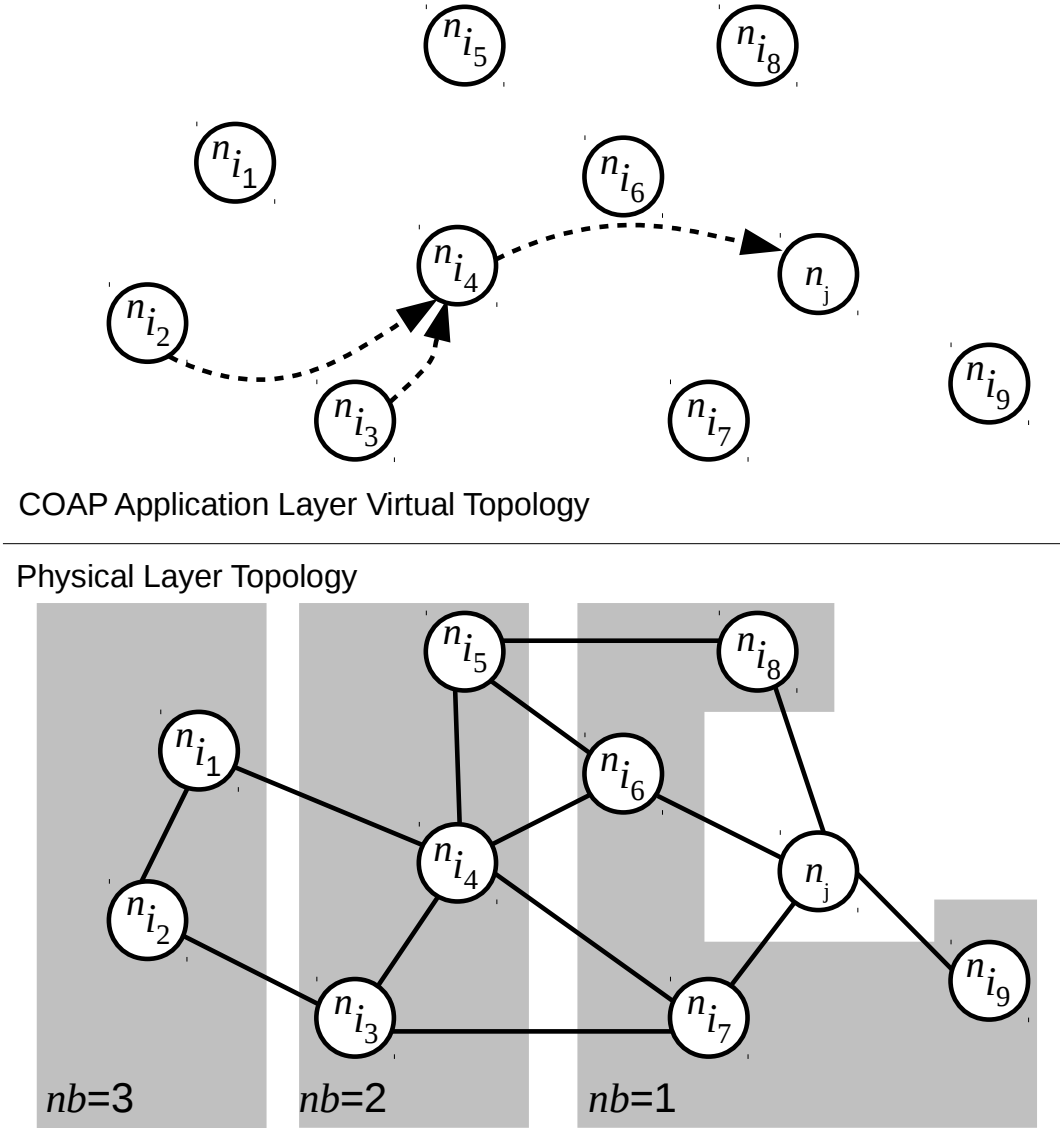


Figure 5.1: Illustration of physical topology, virtual topology and neighbourhoods.

When finding the solution for the new request, the heuristic must take into consideration previously registered observations. The example at Figure 5.1 (top) shows that node n_j has previously registered on proxy n_{i_4} that is receiving notifications from n_{i_2} and n_{i_3}. The proxy is sending an aggregate notification to n_j. The ongoing registrations will, therefore, form a virtual topology at the application layer. A virtual link (registration) may, however, travel through multiple hops at the physical layer and this must be taken into consideration by the heuristic when evaluating a possible registration step for the new request. This is done as follows. A node $n_i \in \mathcal{N} \backslash \mathcal{N}^D$ is inserted into a neighbourhood level according to its one-way delay to n_j, which for simplicity we assumed to correspond to the number of hops. That is, node n_i will be at

neighbourhood level $nb = OWD_{n_j}^{n_i}$, as shown in Figure 5.1. For an arriving request for observation, and considering a specific destination $n_j \in \mathcal{N}^D$, the heuristic will first look at nodes included in neighbourhood level $nb = 1$, then $nb = 2$, and so on, until a solution is found or all neighbourhoods have been searched. For a specific neighbourhood level the following two sets of nodes are processed in this order:

$\mathcal{H}^I$: Nodes with the resource (either at its namespace or at its cache) and sending notifications from previous requests to n_j. Such nodes already have the resource and can aggregate such info into other ongoing notifications being sent to destination n_j. The best solution is the pair $(n_i, p_{n_i}^k)$ providing the highest $\frac{1}{G} - \sum_{p_{n_i}^k \in \mathcal{P}(n_i)} (\omega_{p_{n_i}^k}^H - \omega_{p_{n_i}^k}^L)$ positive value (see expression (5.10)), where $\omega_{p_{n_i}^k}^H$ and $\omega_{p_{n_i}^k}^L$ are computed using expressions (5.8) and (5.9) and verify conditions (5.11) and (5.12). If a solution is found there will be a virtual link from n_i to n_j, otherwise set $\mathcal{H}^{II}$ is searched.

$\mathcal{H}^{II}$: Nodes without the resource but sending notifications from previous requests to n_j. Although not having the resource, unprocessed direct neighbours at next neighbourhood (another set denoted by $\mathcal{H}^{III}$) with the resource are inspected for a feasible solution. That is, for each element $\mathcal{H}^{II}$, and all its unprocessed direct neighbours in $\mathcal{H}^{III}$, the cost of the following possible feasible solutions is computed and, if many, the best is picked:

- Send notifications from a plan in $n_i \in \mathcal{H}^{III}$ directly to the destination node n_j (virtual link from n_i to n_j):
  - Making aggregation with other notifications already being sent to n_j;
  - Making no aggregation if no notifications of previous requests are being sent to n_j.
- Send notifications from a plan in $n_i \in \mathcal{H}^{III}$ to an intermediate node in $n_k \in \mathcal{H}^{II}$ (virtual links from n_i to n_k and from n_k to n_j):
  - Making aggregation with other notifications already being sent to n_k;
  - Making no aggregation if no notifications of previous requests are being sent to n_k.

If no feasible solution has been found for a specific neighbourhood level, then the level increases and search restarts. The details of the heuristic procedure are shown in Algorithm 1.

5.3 Managing Algorithm Decisions

For each arriving request, either the CPLEX[2] is used to find the optimal solution (mathematical formalization) or the heuristic will be used to find a solution that might not be optimal (both results will be under analysis in the following section). Then the following update steps must be performed:

- If heuristic is used, set appropriate $\sigma_{p_{n_i}^k}^{n_i,n_j}$ to 1 for each $(n_i, p_{n_i}^k, n_j)$ element returned by the heuristic. Set the corresponding $\kappa_{p_{n_i}^k}$ using expression (5.2), and set the remaining to 0. Finally set the corresponding $\omega_{p_{n_i}^k}^H$ and $\omega_{p_{n_i}^k}^L$ using expressions (5.8) and (5.9). This step is carried out by CPLEX if it is being used instead of the heuristic.

- For every node n_j acting as a client for r, insert new request r into $\mathcal{O}(n_j)$, together with its info $srv(r)$ and $sub(r)$. This is done for any n_j such that $\sigma_{p_{n_i}^k}^{n_i,n_j} \neq 0 \wedge \kappa_{p_{n_i}^k} \neq 0, \forall n_i, \forall p_{n_i}^k$.

- For every plan $p_{n_i}^k$ used by r, where node n_i acts as a server and $\kappa_{p_{n_i}^k} \neq 0$, reallocate if better scheduling is possible, and then insert/update observation o where subject $sub(o) = sub(r')$ and update $\mathcal{D}(o)$.

- Update $W^H(p_{n_i}^k)$ according to $\omega_{p_{n_i}^k}^H$.

- Update $W^L(p_{n_i}^k)$ according to $\omega_{p_{n_i}^k}^L$.

- Update subjects of each node n acting as client, that is, $\mathcal{S}(n)$.

- For every node n_i, if $|\mathcal{P}(n_i)| = \lambda^{n_i}$ then there is no empty plan. Insert new empty plan in $\mathcal{P}(n_i)$ if expression (5.14) is not violated.

For planning to become viable a set of manager nodes, which could incorporate an extra power source, should keep network information updated as discussed later.

Algorithm 1 - Heuristic for the AS-WSN problem.

/*neighbourhood level starts to 1, processed nodes includes destination nodes, and a $n_j \in \mathcal{N}^D$ is assumed*/
$nb = 1$
$pn = \mathcal{N}^D$
while $pn \neq \mathcal{N}$ **do**
  /*set of nodes with resource and transmitting to n_j*/
  $\mathcal{H}^I = \{n_i \in \mathcal{N} \backslash pn : OWD_{n_j}^{n_i} = nb \wedge n_i \in \mathcal{N}^S \wedge \exists o \in \mathcal{O}(p_{n_i}^k), p_{n_i}^k \in \mathcal{P}(n_i),\ where\ n_j \in \mathcal{D}(o)\}$
  while $\mathcal{H}^I \neq \emptyset$ **do**
    /*node with lowest congestion (constraints (5.10))*/
    $n_i = argmax_{n_i \in \mathcal{H}^I}\{\frac{1}{G} - \sum_{p_{n_i}^k \in \mathcal{P}(n_i)}(\omega_{p_{n_i}^k}^H - \omega_{p_{n_i}^k}^L)\}$
    $p_{n_i}^k = \{p_{n_i}^k \in \mathcal{P}(n_i) : (5.11), (5.12), (5.13), (5.14)$ **are met**$\}$
    if $\frac{1}{G} - \sum_{p_{n_i}^k \in \mathcal{P}(n_i)}(\omega_{p_{n_i}^k}^H - \omega_{p_{n_i}^k}^L) \geq 0 \wedge p_{n_i}^k \neq \emptyset$ **then**
      return $((n_i, p_{n_i}^k, n_j), cost = nb \times (1 + \delta))$
    end if
    $pn \leftarrow n_i$
    $\mathcal{H}^I = \mathcal{H}^I \backslash \{n_i\}$
  end while
  /*set of nodes without resource and transmitting to n_j*/
  $\mathcal{H}^{II} = \{n_i \in \mathcal{N} \backslash pn : OWD_{n_j}^{n_i} = nb \wedge n_i \notin \mathcal{N}^S \wedge \exists o \in \mathcal{O}(p_{n_i}^k), p_{n_i}^k \in \mathcal{P}(n_i),\ where\ n_j \in \mathcal{D}(o)\}$
  /*set of nodes in next neighbourhood with resource*/
  $\mathcal{H}^{III} = \{n_i \in \mathcal{N} \backslash pn : OWD_{n_j}^{n_i} = nb + 1 \wedge n_i \in \mathcal{N}^S\}$
  /*list of potential solutions starts empty*/
  $L = \emptyset$
  for each $(n_l \in \mathcal{H}^{II}, n_m \in \mathcal{H}^{III})$ pair **do**
    /*the following different solutions might exist*/
    if $\exists o \in \mathcal{O}(p_{n_m}^k), p_{n_m}^k \in \mathcal{P}(n_m),\ with\ n_j \in \mathcal{D}(o)$ **then**
      /*from n_m directly to n_j with aggregation*/
      $hopOne = \{(n_m, p_{n_m}^k, n_j) : \exists o \in \mathcal{O}(p_{n_m}^k), p_{n_m}^k \in \mathcal{P}(n_m),\ where\ n_j \in \mathcal{D}(o)\}$
      $hopTwo = NULL$
      $L \leftarrow (hopOne, hopTwo, cost = (nb + 1) \times (1 + \delta))$
    else
      /*from n_m directly to n_j without aggregation*/
      $hopOne = \{(n_m, p_{n_m}^k, n_j) : p_{n_m}^k \in \mathcal{P}(n_m)\}$
      $hopTwo = NULL$
      $L \leftarrow (hopOne, hopTwo, cost = (nb + 1) \times 2)$
    end if
    if $\exists o \in \mathcal{O}(p_{n_m}^k), p_{n_m}^k \in \mathcal{P}(n_m),\ with\ n_l \in \mathcal{D}(o)$ **then**
      /*hops (n_m, n_l) (n_l, n_j) with aggregation*/
      $hopOne = \{(n_m, p_{n_m}^k, n_l : \exists o \in \mathcal{O}(p_{n_m}^k), p_{n_m}^k \in \mathcal{P}(n_m),\ where\ n_l \in \mathcal{D}(o)\}$
      $hopTwo = \{(n_l, p_{n_l}^k, n_j) : \exists o \in \mathcal{O}(p_{n_l}^k), p_{n_l}^k \in \mathcal{P}(n_l),\ where\ n_j \in \mathcal{D}(o)\}$
      $L \leftarrow (hopOne, hopTwo, cost = (nb + 1) \times (1 + \delta))$
    else
      /*hops (n_m, n_l) (n_l, n_j), 2nd with aggregation*/
      $hopOne = \{(n_m, p_{n_m}^k, n_l : p_{n_m}^k \in \mathcal{P}(n_m)\}$
      $hopTwo = \{(n_l, p_{n_l}^k, n_j) : \exists o \in \mathcal{O}(p_{n_l}^k), p_{n_l}^k \in \mathcal{P}(n_l),\ where\ n_j \in \mathcal{D}(o)\}$
      $L \leftarrow (hopOne, hopTwo, cost = 2 + nb \times (1 + \delta))$
    end if
  end for

48: **while** $L \neq \emptyset$ **do**
$L' =$SmallestCostSubset(L)
50: /*find element with highest value in (5.10), and (5.11) (5.12) (5.13) (5.14) are met*/
$(hopOne, hopTwo, cost) =$MaxBandwidth(L')
52: **if** $(hopOne, hopTwo, cost) \neq NULL$ **then**
return $(hopOne, hopTwo, cost)$
54: **end if**
$L' = L' \backslash \{(hopOne, hopTwo, cost)\}$
56: **end while**
$pn = pn \cup \mathcal{H}^{II}$
58: $nb = nb + 1$
end while

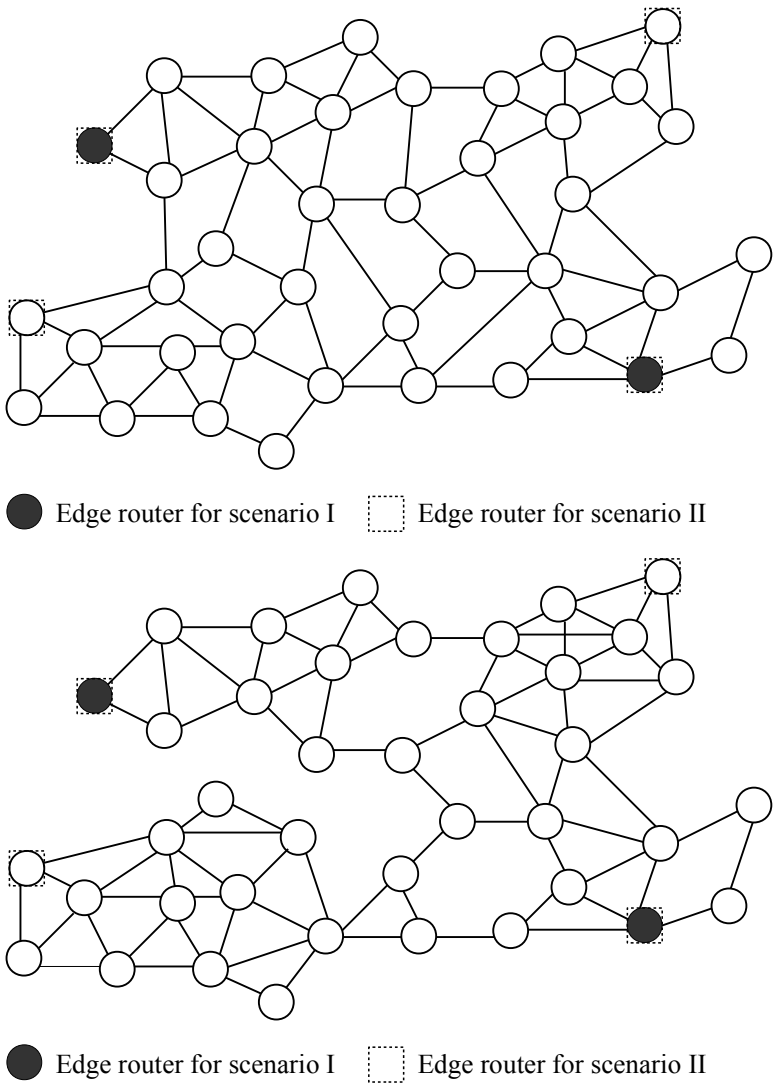


Figure 5.2: Network topologies under test: first has no congestion point while second includes a congestion point.

5.4 Performance Evaluation

5.4.1 Network Setup

The network topology under test is the one shown in Figure 5.2. This network has 40 wireless sensor nodes randomly generated using the weighted proximity algorithm in [15]. Two network scenarios were created for each network, as illustrated in Figure 5.2 :

I : 2 edge routers;

II : 4 edge routers.

Table 5.1: Network Parameters: Monitoring and M2M Applications

Parameter	Value
Bandwidth (B)	31250 bytes
Byte time (T^{min})	$\frac{1}{B} = \frac{1}{31250}s$
Packet length (L)	30 bytes
Network degree (G)	10
Duty cycle (C)	$\frac{0.125s}{T^{min}}$
Maximum wakeups (M)	8
Max age of all subjects ($m(s)$)	1s
Aggregation overhead factor (δ)	$\frac{1}{3}$

For each network scenario, the energy consumption and average transmission delay per node are plotted considering two cases: $CaseA$) one subject per node, all different; $CaseB$) 2 different subjects per node. That is, the number of available subjects at the network will be the same as the number of nodes, but in the second case a subject is available at two nodes.

Concerning the values for the OWD between nodes, the number of hops between the nodes is assumed for simplicity. Table 5.1 summarizes the remaining network parameters used, based on [14], [11] assuming 802.15.4 devices and contikiMAC LPL:.

5.4.2 Accounting of Energy Consumption

For energy consumption calculation it is assumed that nodes use 24mW in wake mode, 52.2mW when transmitting and 1.278mW in sleep mode, values for Z1 motes taken from [11]. Thus, the following per byte time energy consumption values can be adopted:

$E^T = 167.04 \times 10^{-8}$J

[2]IBM ILOG CPLEX Optimizer.

$E^W = 76.8 \times 10^{-8}\mathbf{J}$

$E^S = 4.09 \times 10^{-8}\mathbf{J}$

The devices are assumed to use a listening protocol with a duty cycle of 0.125s (value for contikiMAC LPL), as in [11], which leads to a duty cycle in byte times equal to $C = \frac{0.125s}{T^{min}} = 3906.25$. This duty cycle must be considered when accounting for energy consumption. For the maximum number of wakeups per node the upper bound $M = \lfloor \frac{B}{C} \rfloor = 8$ is used.

After the update steps done in Section 5.3, the total energy currently being consumed at the network can be accounted as follows. The number of byte times in transmission mode associated with plan $p^k_{n_i}$ of node $n_i \in \mathcal{N}$ is given by:

$$x(p^k_{n_i}) = |\{\mathcal{D}(o) : o \in \mathcal{O}(p^k_{n_i})\}| \times L \tag{5.16}$$

while the number of byte times in wake mode is given by:

$$y(p^k_{n_i}) = max\{C - x(p^k_{n_i}), 0\}. \tag{5.17}$$

The number of byte times in sleep mode for the whole node n_i is given by:

$$z(n_i) = B - \sum_{p^k_{n_i} \in \mathcal{P}_{n_i} : \mathcal{O}(p^k_{n_i}) \neq \emptyset} [x(p^k_{n_i}) + y(p^k_{n_i})] \tag{5.18}$$

meaning that the total energy consumption per second at the WSN can be approximated by:

$$E^{Total} = \sum_{n_i \in N} \sum_{p^k_{n_i} \in \mathcal{P}_{n_i} : \mathcal{O}(p^k_{n_i}) \neq \emptyset} [x(p^k_{n_i}) \times E^T + y(p^k_{n_i}) \times E^W] \times \\ \times AvgHops(\mathcal{O}(p^k_{n_i})) + \sum_{n_i \in N} z(n_i) \times E^S \tag{5.19}$$

where $AvgHops(\mathcal{O}(p^k_{n_i}))$ is the average number of hops travelled by the observations in $\mathcal{O}(p^k_{n_i})$, as energy waste will occur at intermediate non proxy nodes.

Regarding the average load per node, which will influence delay, this has been accounted using the following expression:

$$D^{Avg} = \frac{1}{|\mathcal{N}|} \times \sum_{n_i \in N} \sum_{p^k_{n_i} \in \mathcal{P}_{n_i}} |\{\mathcal{D}(o) : o \in \mathcal{O}(p^k_{n_i})\}| \times L \times AvgHops(\mathcal{O}(p^k_{n_i})) \tag{5.20}$$

5.5 Results and Analysis

5.5.1 Benefits of Using Aggregation and Scheduling

Results for Monitoring Application

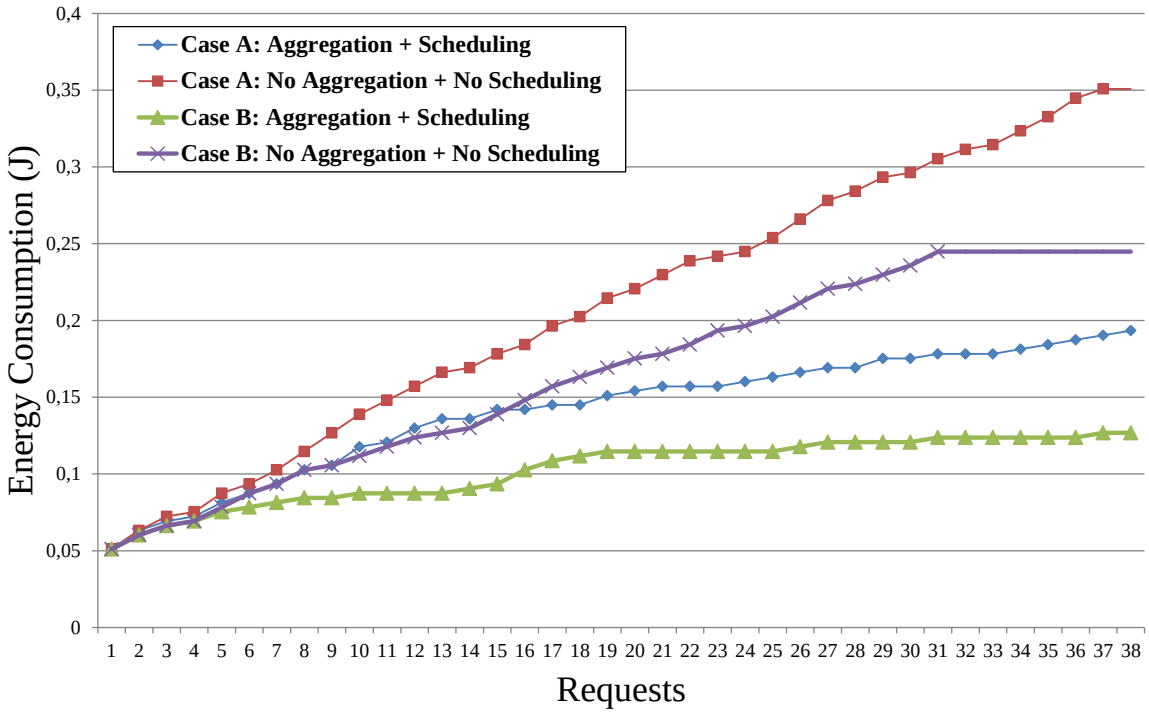


Figure 5.3: Energy consumption for monitoring in first topology, scenario I: aggregation/scheduling vs no aggregation/scheduling (optimal values).

In this section the use of aggregation and scheduling in a monitoring application context, where notifications are forwarded toward the edge, is analysed. To analyse the benefits of using aggregation/scheduling, when compared with a non aggregation/scheduling scenario, the energy consumptions and average loads per node have been plotted considering the optimal values obtained by the CPLEX optimizer. Figures from 5.3 to 5.6 relate to energy consumptions, for both topologies and considering scenarios I and II, while Figures from 5.7 to 5.10 show the average loads per node, also for both topologies and considering scenarios I and II, which will influence delay. The subject asked by a request is randomly generated, using an uniform distribution, and does not include previously requested subjects as these will be available at edge routers, and available to other clients is necessary. That is, subjects are requested only once. In case A, a subject is available at a single node, while in case B a subject is available at two nodes. Such subject to node assignment is done initially and does not change between simulations.

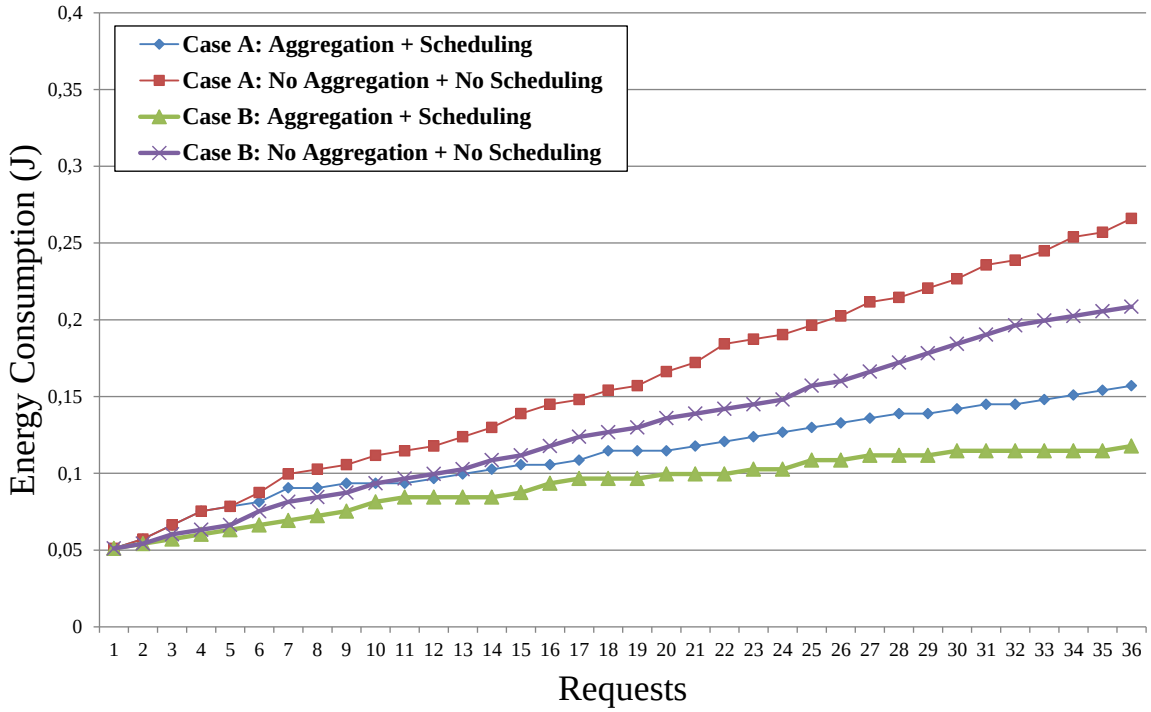


Figure 5.4: Energy consumption for monitoring in first topology, scenario II: aggregation/scheduling vs no aggregation/scheduling (optimal values).

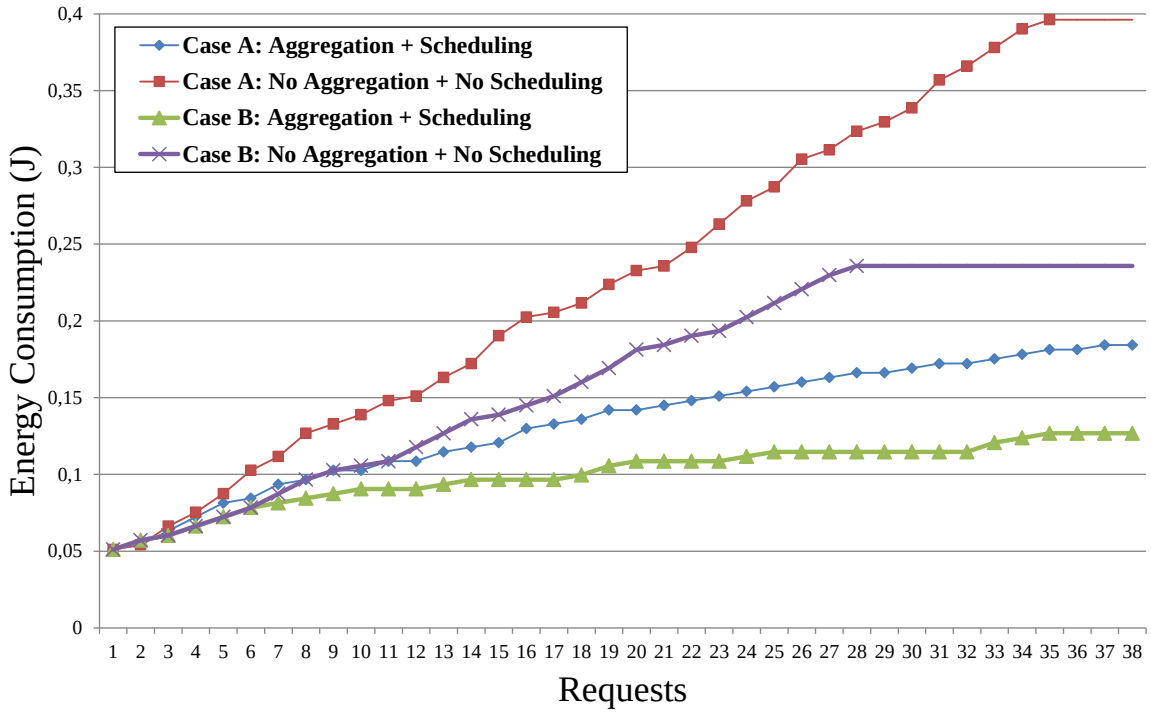


Figure 5.5: Energy consumption for monitoring in second topology, scenario I: aggregation/scheduling vs no aggregation/scheduling (optimal values).

From these plots we can observe that the use of optimal aggregation and scheduling always reduces energy consumption, the largest reductions reach-

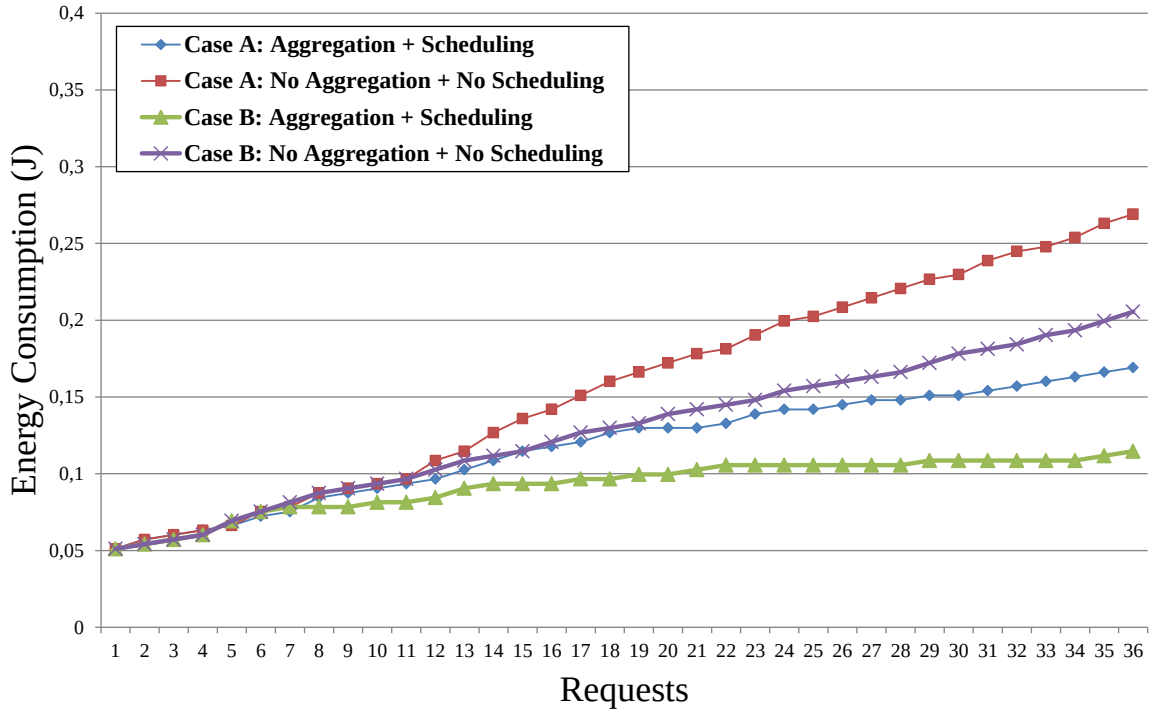


Figure 5.6: Energy consumption for monitoring in second topology, scenario II: aggregation/scheduling vs no aggregation/scheduling (optimal values).

ing more than 50%. This happens for scenarios I and II (using 2 and 4 edge routers, respectively) and one or two subjects per node (case A and B, respectively). This reduction is more pronounced for scenario I as less edge routers are available for packet forwarding, while in scenario II more edge routers mean more forwarding possibilities and the less consuming forwarding can be chosen even if optimal aggregation/scheduling is not being used. Please note that a solution might not be found, due to the congestion and max age constraints, after accommodating many requests.

Regarding the topology and scenarios, the topology with no congestion point is able to accommodate more requests than the topology with a congestion point, and scenario II also accommodates more requests than scenario I because more edge routers provide more forwarding choices. For the same number of accommodated requests, scenario II shows less energy consumption. Also, there is less energy consumption when subjects are available at two nodes, when compared with single server subjects. This means that the impact of optimal aggregation and scheduling strongly depends on the network topology configuration.

The plots showing the average load per node confirm that the use of aggregation/scheduling poses less overhead to the network, as expected. The load is higher when a single subject is available at every node (case A), when

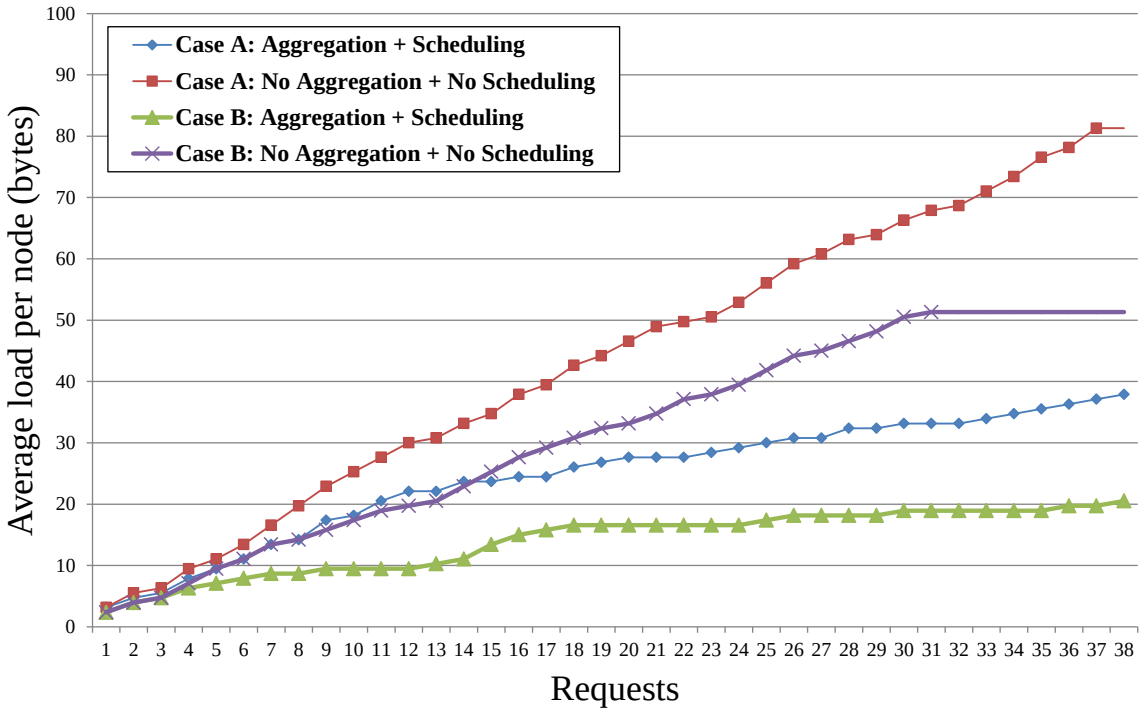


Figure 5.7: Average load per node for monitoring in first topology, scenario I: aggregation/scheduling vs no aggregation/scheduling (optimal values).

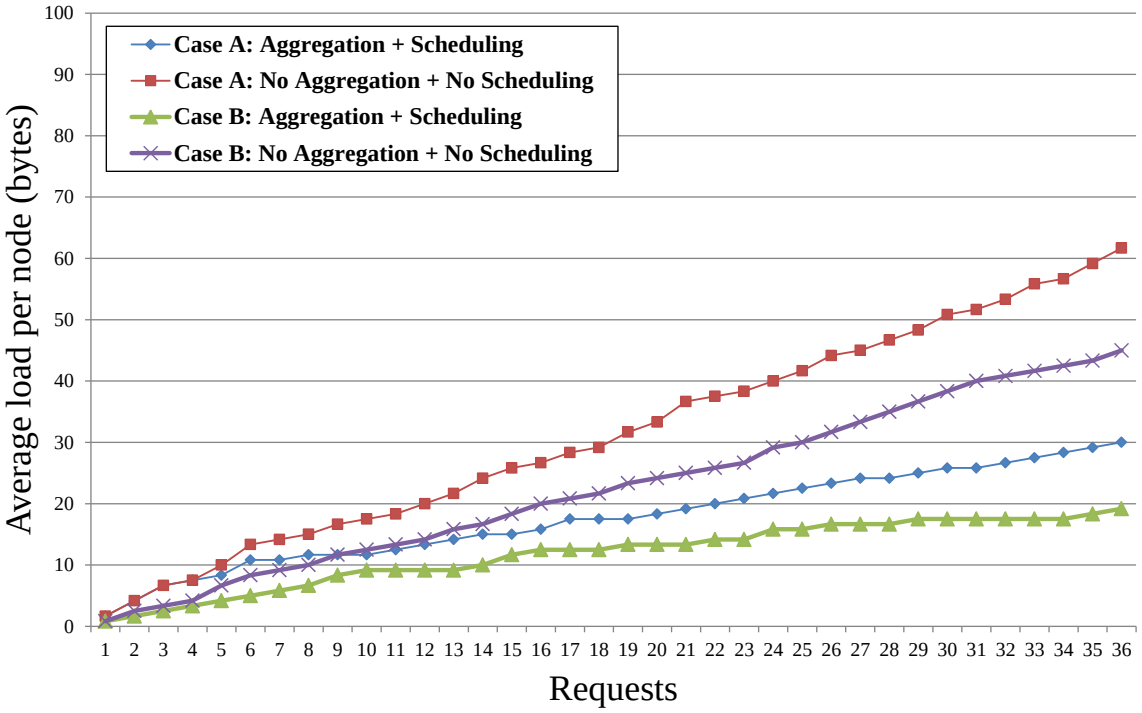


Figure 5.8: Average load per node for monitoring in first topology, scenario II: aggregation/scheduling vs no aggregation/scheduling (optimal values).

compared with two subjects per node. The topology with a congestion point also shows higher average load, in particular when there is a single subject

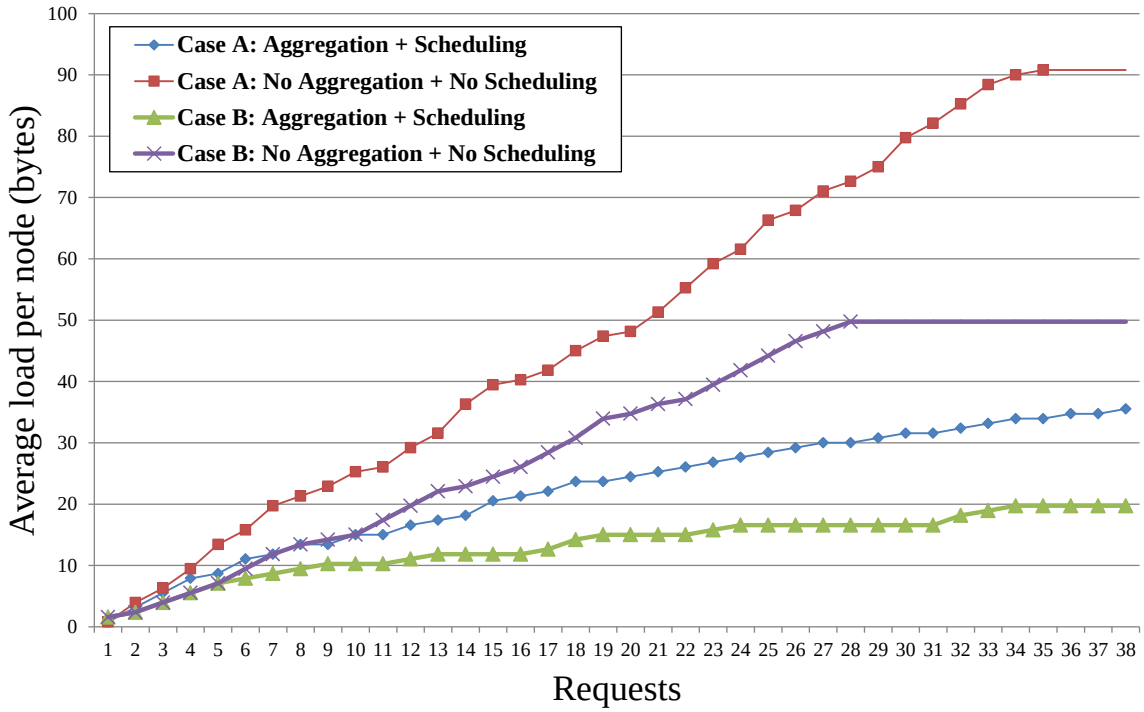


Figure 5.9: Average load per node for monitoring in second topology, scenario I: aggregation/scheduling vs no aggregation/scheduling (optimal values).

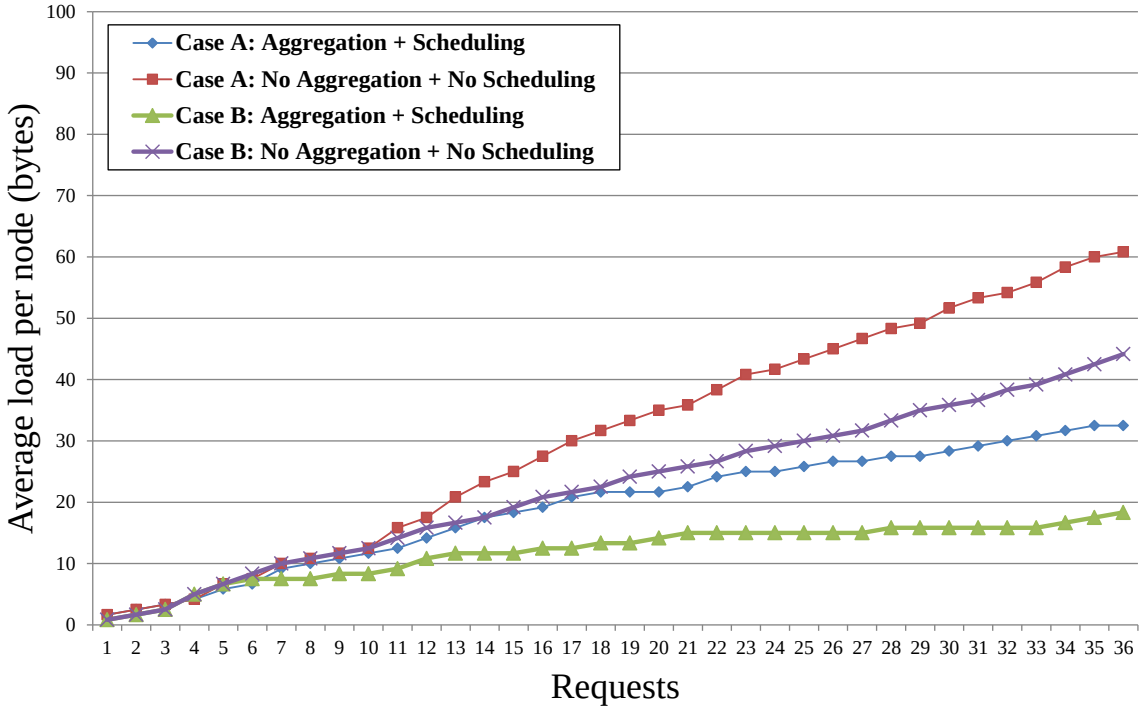


Figure 5.10: Average load per node for monitoring in second topology, scenario II: aggregation/scheduling vs no aggregation/scheduling (optimal values).

available per node in scenario I (2 edge routers) as less alternatives for aggregation/scheduling are available. Please note that the average load per node is

in bytes and not packets.

Results for M2M Application

In this section the use of aggregation and scheduling in a M2M application context, where the source and sink of notifications are both internal WSN networks (not edge routers), is analysed. The subject being asked, and client node requesting it, are randomly generated using an uniform distribution. Subjects are requested only once while the client node is chosen from the set of nodes not having the generated subject in its namespace or cache. The benefits of using aggregation/scheduling, when compared with a non aggregation/scheduling scenario, are also plotted considering the optimal values obtained by the CPLEX optimizer. Figures 5.11 and 5.12 relate to energy consumptions, for the two topologies, while Figures from 5.13 and 5.14 show the average loads per node, also for both topologies, which will influence delay.

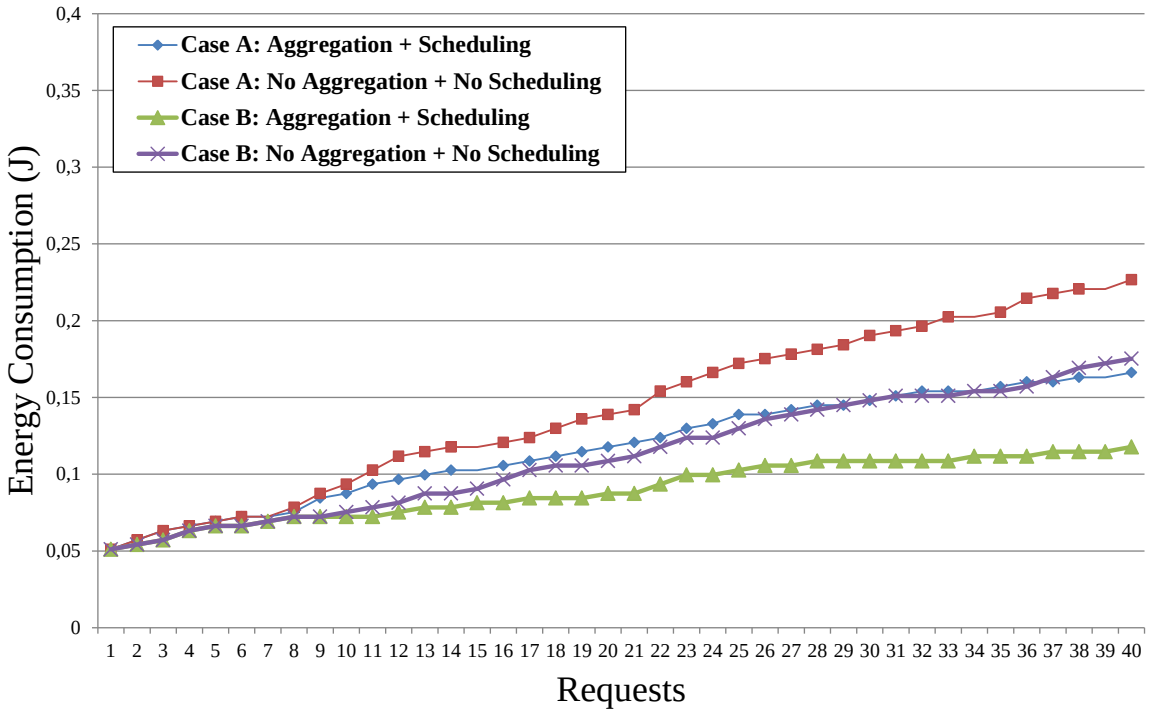


Figure 5.11: Energy consumption for M2M in first topology: aggregation/scheduling vs no aggregation/scheduling (optimal values).

M2M results reveal that the difference aggregation/scheduling and no aggregation/scheduling reduces, meaning that the potential for energy saving due to optimal aggregation and scheduling procedures is smaller. This is so because notifications are not being forwarded toward the same destination nodes. This issue strongly influences the number of requests that can be accommodated, which is more noticeable in case A where some dots are not

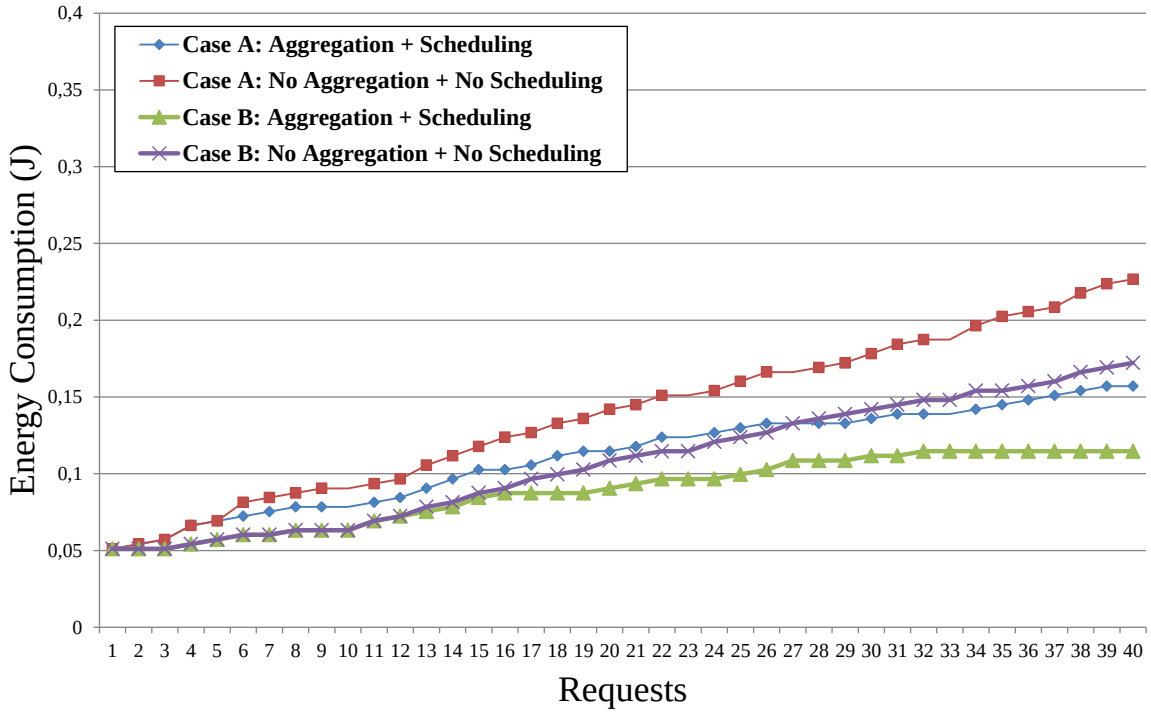


Figure 5.12: Energy consumption for M2M in second topology: aggregation/scheduling vs no aggregation/scheduling (optimal values).

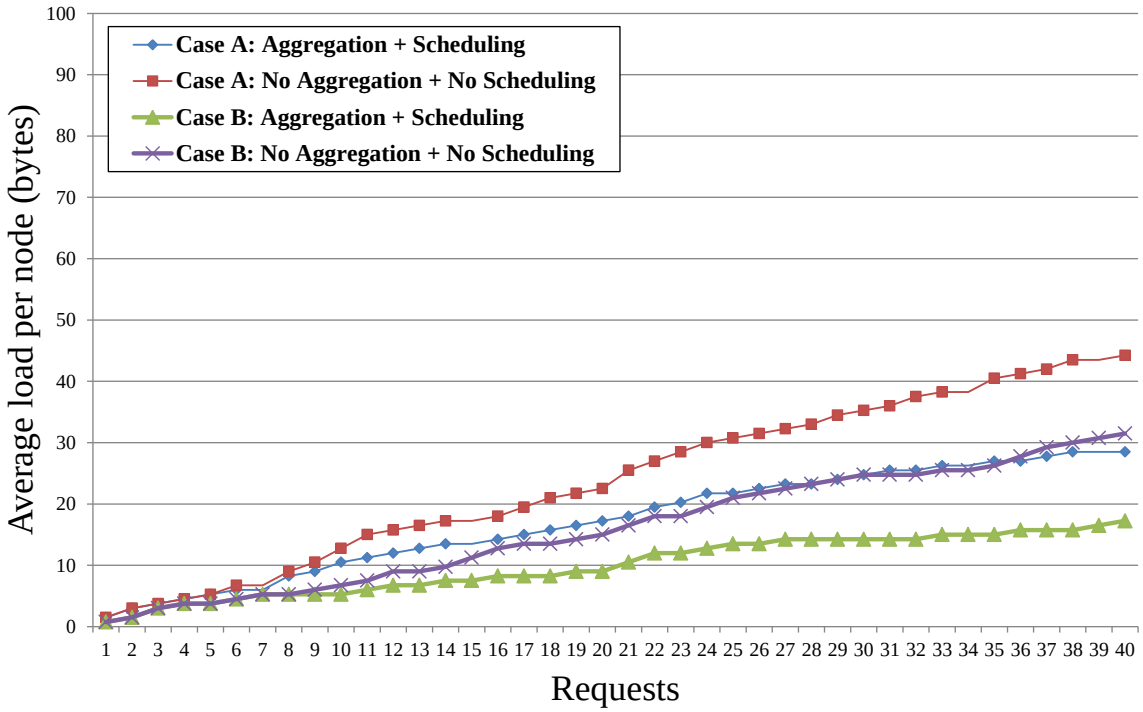


Figure 5.13: Average load per node for M2M in first topology, scenario I: aggregation/scheduling vs no aggregation/scheduling (optimal values).

present (request was not accommodated). For these topologies in particular, all requests were accommodated for case B since subjects are originally avail-

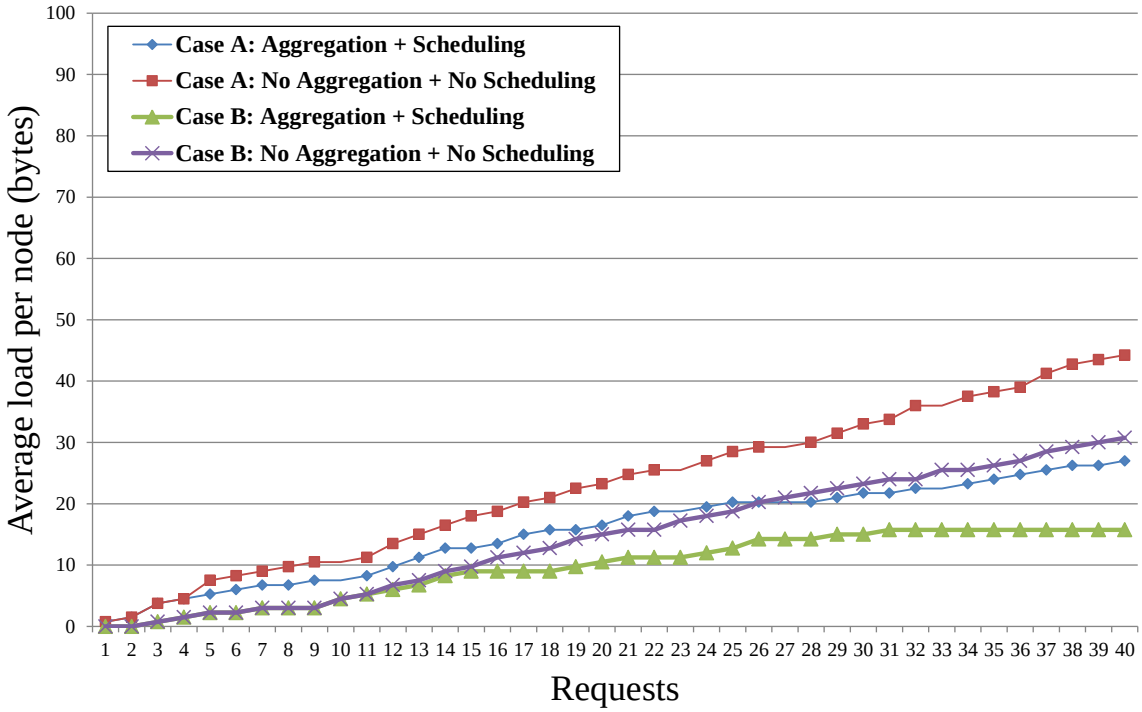


Figure 5.14: Average load per node for M2M in second topology, scenario I: aggregation/scheduling vs no aggregation/scheduling (optimal values).

able in more than one place.

5.5.2 Optimal vs Heuristic

Results for Monitoring Application

In this section the performance of the heuristic algorithm in a monitoring application context is analysed. For comparison the optimal results obtained by CPLEX, using aggregation/scheduling, are also plotted. Figures from 5.15 to 5.18 relate to energy consumptions, for both topologies and considering scenarios I and II.

Regarding the plots of the heuristic vs optimal values we can observe that the heuristic is able to get closer to the optimal in scenario II, while for scenario I the gap is higher. Also, while the closeness between the heuristic and the optimal does not change for cases A and B in the first topology, for the second topology having a congestion point the case A presents a higher gap. The heuristic was also able to apply aggregation/scheduling to all the arriving requests, and accommodate them, similarly to the optimal.

In general we can state that the performance of the heuristic algorithm used for aggregation and scheduling strongly depends on the network topology, and the gap against the optimal increases as the number of requests

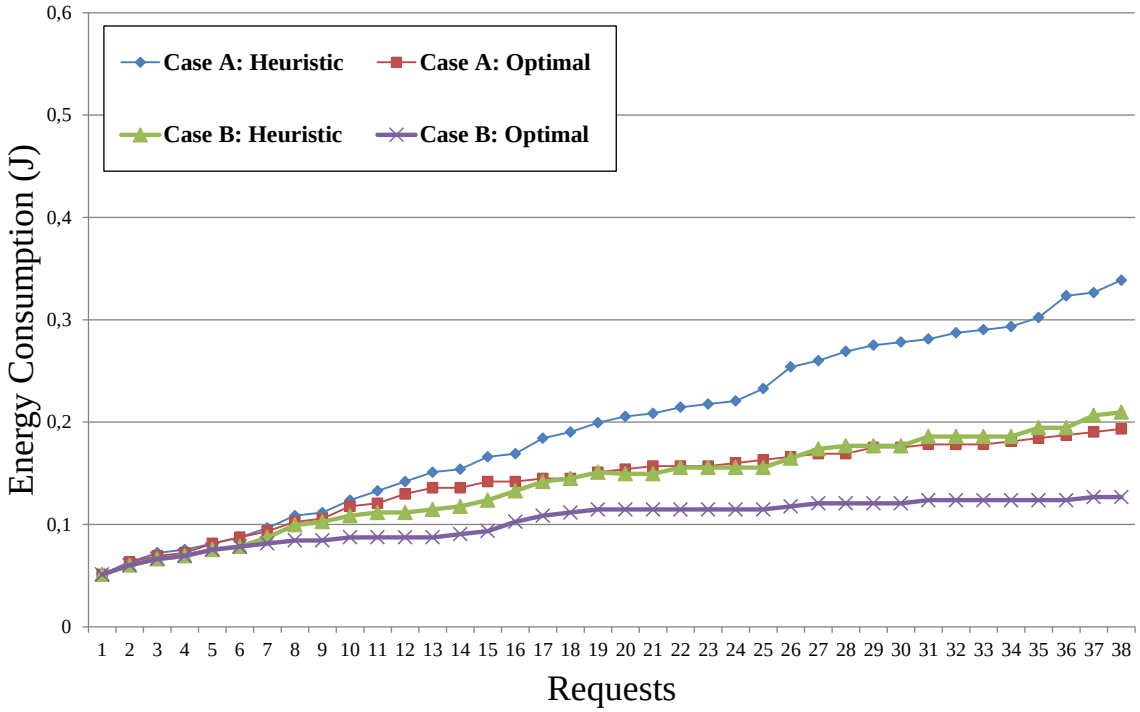


Figure 5.15: Energy consumption for monitoring in first topology, scenario I: optimal vs heuristic.

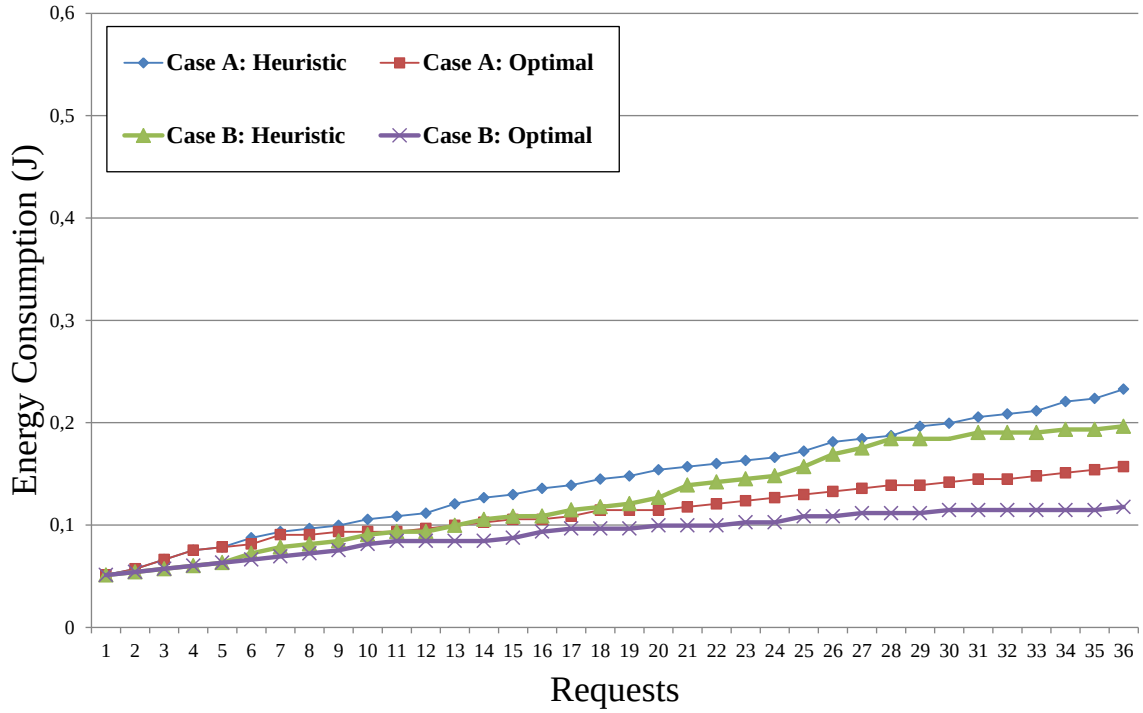


Figure 5.16: Energy consumption for monitoring in first topology, scenario II: optimal vs heuristic.

increases, ranging from 0% till 50%. This reveals that topology dependent mechanisms should be developed in the future for incorporation into the heuristic algorithm and further performance improvement.

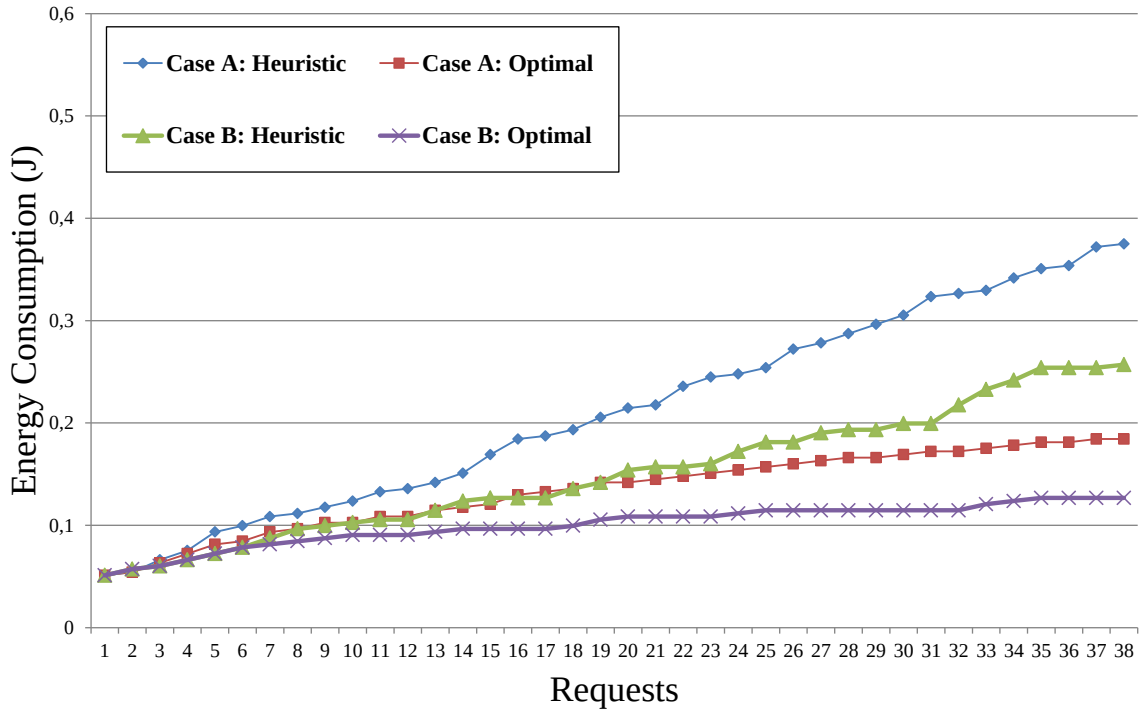


Figure 5.17: Energy consumption for monitoring in second topology, scenario I: optimal vs heuristic

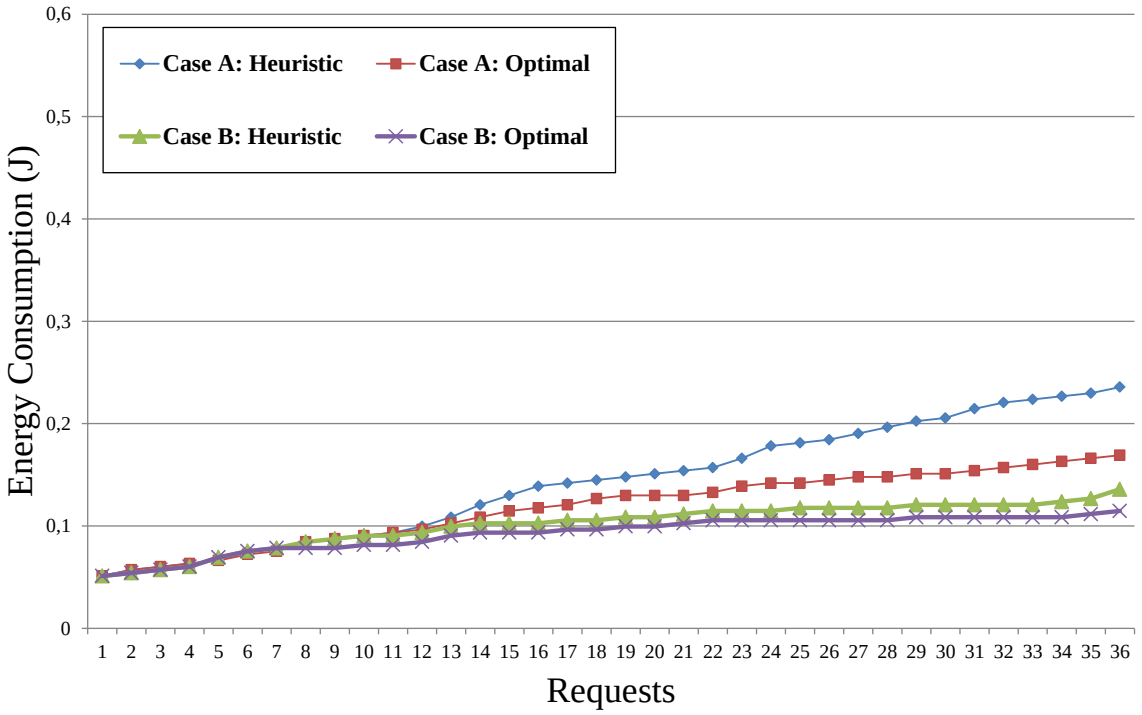


Figure 5.18: Energy consumption for monitoring in second topology, scenario II: optimal vs heuristic.

Results for M2M Application

In this section the performance of the heuristic algorithm in a M2M application context, where the source and sink of notifications are both internal WSN

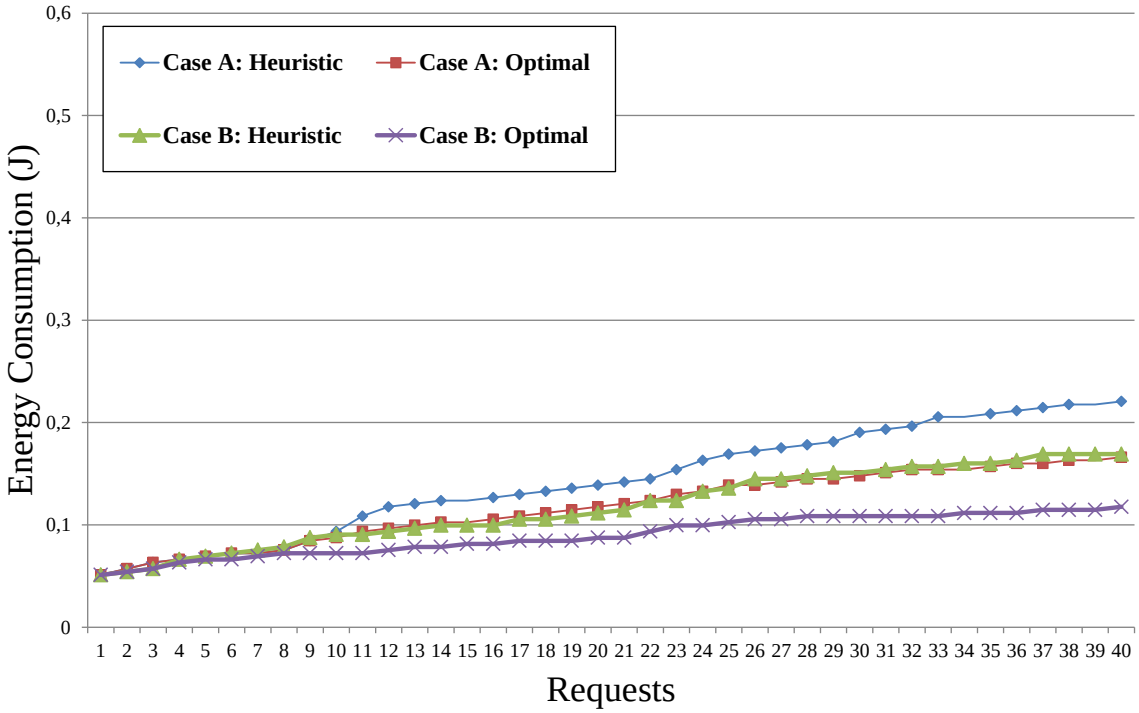


Figure 5.19: Energy consumption for M2M in first topology: optimal vs heuristic.

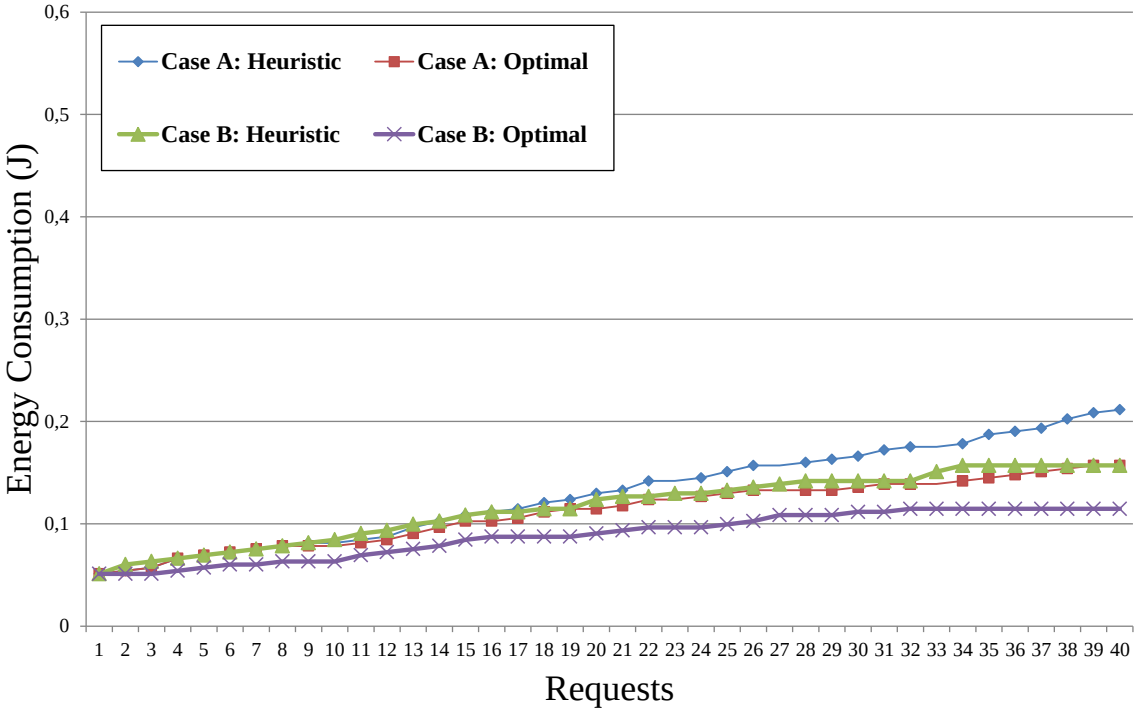


Figure 5.20: Energy consumption for M2M in second topology: optimal vs heuristic.

networks (not edge routers), is analysed. For comparison the optimal results obtained by CPLEX, using aggregation/scheduling, are also plotted. Figures 5.19 and 5.20 relate to energy consumptions, for the two topologies.

Results show that for the M2M the heuristic algorithm performs very sim-

ilarly for both topologies and is less sensitive to case A and B. That is the performance of the heuristic algorithm is less network topology dependent.

CHAPTER 6

Conclusions and Future Work

In this work a framework, heuristic included, is proposed for the planning of registration steps in CoAP/Observe based wireless sensor networks. These registration steps incorporate optimal aggregation and scheduling of notification deliveries at proxies, so that energy saving is maximized, bandwidth is used efficiently and the overall delay is reduced.

Results show that this can be applied to monitoring and M2M applications significantly reducing energy consumption and average load per node. After analysis of results for aggregation/scheduling and no aggregation/scheduling it is clear that there is a lot of potential for energy saving and reducing average load per node. On a monitoring application the energy savings can be as much as 50%. On a M2M application, the potential for energy saving decreased, mostly due to the fact that the notifications are not being forwarded to the same fixed destinations, but are still significant. Overall, performance results are network topology and application dependent.

As future work procedures that incorporate the possibility of aborting current requests, and their re-establishment in a different way, for energy saving purposes and bandwidth utilization improvement can be studied. Also, topology dependent mechanisms should be developed in the future for incorporation into the heuristic algorithm and further performance improvement.

References

[1] Sye Loong Keoh, Sandeep S. Kumar, and Hannes Tschofenig: Securing the Internet of Things: A Standardization Perspective, IEEE Internet of Things Journal, May 2014.

[2] W. Colitti, K. Steenhaut, N. De Caro, B. Buta and V. Dobrota: Evaluation of Constrained Application Protocol for Wireless Sensor Networks, IEEE Workshop on Local & Metropolitan Area Networks (LANMAN), 2011.

[3] L. Atzoria, A. Ierab and G. Morabito: The Internet of Things: A Survey, Computer Networks, Elsevier, Vol. 54, No. , 2010.

[4] Carsten Bormann, Angelo P. Castellani and Zach Shelby: CoAP: An Application Protocol for Billions of Tiny Internet Nodes, IEEE Internet Computing, Vol. 16, No. 2, 2012.

[5] N. Kushalnagar, G. Montenegro, and C. Schumacher: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals, RFC 4919, August 2007.

[6] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur and R. Alexander: RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, RFC 6550, March 2012.

[7] Z. Shelby, K. Hartke, and C. Bormann: Constrained Application Protocol (CoAP), draft-ietf-core-coap-18, IETF, 2013.

[8] K. Hartke: Observing Resources in CoAP, draft-ietf-core-observe-16, IETF, 2014.

[9] A. Ludovici, E. Garcia X. Gimeno and A. Calveras Auge: Adding QoS support for timeliness to the observe extension of CoAP, IEEE Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), October 2012.

[10] D. Sacramento, G. Schütz and N. Correia: Aggregation and Scheduling in CoAP/Observe Based Wireless Sensor Networks, IEEE International Conference on Communications (ICC), June 2015.

[11] Girum Ketema Teklemariam, Jeroen Hoebeke, Ingrid Moerman and Piet Demeester: Facilitating the Creation of IoT Applications Through Conditional Observations in CoAP, EURASIP Journal on Wireless Communications and Networking, 2013.

[12] Oscar Garcia-Morchon, Sye-Loong Keoh, Sandeep S. Kumar, Pedro Moreno-Sanchez, Francisco Vidal-Meca and Jan Henrik Ziegeldorf: Securing the IP-based internet of things with HIP and DTLS, ACM Conference on Security and privacy in wireless and mobile networks (WiSec), 2013.

[13] D. Hardt: The OAuth 2.0 Authorization Framework, RFC 6749, 2012

[14] Ying Jian, Shigang Chen, Shiping Chen and Yibei Ling: Fair End-to-end Bandwidth Distribution in Wireless Sensor Networks, IEEE International Conference on Communications (ICC) 2010.

[15] Furuzan Atay Onat and Ivan Stojmenovic: Generating Random Graphs for Wireless Actuator Networks, IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (June 2007), pp. 1–12.

[16] Jennifer Yick, Biswanath Mukherjee and Dipak Ghosal: Wireless Sensor Network Survey, Computer Networks, Elsevier 2010.

[17] Jun Zheng and Abbas Jamalipour: Wireless Sensor Networks, a Networking Perspective, Wiley 2009.

[18] Dogan Yazar: RESTful Wireless Sensor Networks, MSc Thesis - University of Uppsala 2009.

[19] M. Jazayeri, C. Huang and S. Liang: TinySOS: Design and Implementation of Interoperable and Tiny Web Service for the Internet of Things, SWE2012, 2012.

[20] T. Gao, D.Greenspan, M. Welsh, R.Juang, and A. Alm: Vital Signs Monitoring and Patient Tracking Over a Wireless Network, Proceedings of the 2005 IEEE, 2005.

[21] Constrained RESTful Environments (CoRE) Link Format, IETF RFC 6690, 2012.

[22] R. Fielding: Architectural Styles and the Design of Network-based Software Architectures, PhD Thesis, 2000