Mécanismes de traitement des données dans les réseaux de capteurs sans fils dans les cas d'accès intermittent à la station de base

Cosmin Dini

▶ **To cite this version:**

Cosmin Dini. Mécanismes de traitement des données dans les réseaux de capteurs sans fils dans les cas d'accès intermittent à la station de base. Autre [cs.OH]. Université de Haute Alsace - Mulhouse, 2010. Français. NNT : 2010MULH5092 . tel-00576919

HAL Id: tel-00576919

https://theses.hal.science/tel-00576919

Submitted on 15 Mar 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Année 2010

THÈSE

Présentée à

L'UNIVERSITÉ DE HAUTE ALSACE

Pour obtenir le

GRADE DE DOCTEUR
DE L'UNIVERSITÉ DE HAUTE ALSACE

Spécialité Informatique

Mécanismes de traitement des données dans les réseaux de capteurs sans fils dans le cas d'accès intermittent à la station de base

Data Management in Wireless Sensor Networks with Intermittent Sink Access

Par

Cosmin DINI

Soutenue le 21 Décembre 2010 devant le jury :

Directeur : Pascal LORENZ, Professeur, Université de Haute Alsace, France

Rapporteurs : Pascal URIEN, Professeur, Télécom ParisTech, France
Jaime LLORET, Professeur, Polytechnic University of Valencia, Spain

Examinateurs: Abdelhak GUEROUI, Professeur, Université de Versailles, France
Laurent PHILIPPE, Professeur, Université de Besancon, France

Acknowledgement

First, I would like to express my deep gratitude to my supervisor, Professor Pascal LORENZ, for his continuous support, guidance, and encouragement throughout the course of these three years. He helped me get the ideas across and clearly formulate the concepts presented here and provided me with the opportunity to discover and develop many interests.

My sincere thanks also go to the members of my dissertation committee: Professor Pascal URIEN, Professor Jaime LLORET, Professor Abdelhak GUEROUI, and Professor Laurent PHILIPPE for their insightful comments and constructive feedback.

Finally, I would like to thank my family for their invaluable support. Special thanks go to my wife, Oana, whose encouragement and understanding helped me fulfill my goal, and to our two children, Theodora and Isabella, who tried their best to be patient and understand when plans had to be changed.

Abstract

Wireless Sensor Networks have evolved as an alternative to wired networks fit for quick deployments in areas with limited access. New protocols have been devices to deal with the inherent scarcity of resources that characterizes such networks. Energy efficient network protocols are used for communication between nodes. Data collected by wireless nodes is transmitted at an energy cost and therefore carefully managed. The remote deployment of wireless networks opens the possibility of malicious attacks on the data and on the infrastructure itself. Security measures have also been devised, but they come at an energy cost. One item that has received little attention is the situation of the data sink becoming unreachable. The nodes still collect data as instructed and accumulate it. Under prolonged unavailability of the sink node, the storage space on sensor nodes is used up and collecting new data is no longer feasible. Our proposal for a prioritized data reduction alleviates this problem. The collected data is divided into data units who are assigned an importance level calculated in agreement with the business case. We have proposed data reduction primitive operations that reduce the needed space while only losing a limited amount of data resolution. A multi-node deployment opens the possibility for data load sharing between the nodes as well as redundancy. Algorithms were proposed to evaluate the potential gain of these approaches in relation to the amount of energy spent for data transfer. The proposed approach works well in coping with fixed size data storage by trimming the low interest data in a manner that data is still usable.

Sommaire

Les réseaux des capteurs sans fil sont considérés comme une alternative aux réseaux câblés afin de permettre l'installation dans des zones peu accessibles. Par conséquent, de nouveaux protocoles ont été conçus pour supporter le manque des ressources qui est spécifique à ce type de réseau. La communication entre les nœuds est réalisée par des protocoles spécifiques pour la gestion efficace de l'énergie. La gestion des données collectées par ces nœuds doit être également prise en compte car la communication entre les nœuds engendre un coût non-négligeable en termes d'énergie. De plus, l'installation de ce type de réseau dans des régions lointaines facilite les attaques sur la structure des réseaux ainsi que sur les données collectées. Les mesures de sécurité envisagées amènent des coûts d'énergie supplémentaires. Un aspect souvent négligé concerne le cas où un nœud ne peut pas communiquer avec la station de base (sink node) qui collectionne et traite les données. Cependant, les nœuds continuent à accumuler des informations en suivant les plans de collection. Si la situation continue, l'espace de mémoire (storage) diminue à un point où la collection de nouvelles données n'est plus possible.

Nous proposons des mécanismes pour la réduction contrôlée de données en considérant leur priorité relative. Les données sont divisées dans des unités auxquelles un niveau d'importance est alloué, en fonction des considérations d'utilité et de missions qui les utilisent. Nous proposons un ensemble de primitives (opérations) qui permettent la réduction d'espace de stockage nécessaire, tout en préservant un niveau raisonnable de

résolution des informations collectées. Pour les larges réseaux à multiple nœuds, nous proposons des mécanismes pour le partage de données (data load sharing) ainsi que la redondance. Des algorithmes ont été proposés pour évaluer l'efficacité de ces techniques de gestion de données vis-à-vis de l'énergie nécessaire pour transférer les données.

A travers des simulations, nous avons validé le fait que les résultats sont très utiles dans les cas à mémoire limitée (wireless nodes) et pour les communications intermittentes.

Table of Contents

List of Figures

List of Tables

List of Acronyms

WSN	Wireless Sensor Network
LEACH	Low Energy Adaptive Clustering Hierarchy
UNPF	Unified Network Protocol Framework
GPS	Global Positioning System
NTP	Network Time Protocol
FTSP	Flooding Time Synchronization Protocol
DC	Direct Current
CNS	Center at Nearest Source
STP	Shortest Path Tree
GIT	Greedy Incremental Tree
SQL	Structured Query Language
SRT	Semantic Routing Trees
GHT	Geographic Hash Tables
FAME-DBMS	Family of Embedded DataBase Management Systems
SE	Storage Engine
PDR	Prioritized Data Reduction
PDRE	Prioritized Data Reduction Engine
PDRC	Prioritized Data Reduction Controller
RDI	Recurring Data Instance
NRDI	Non Recurring Data Instance
DPI	Dots Per Inch
JVM	Java Virtual Machine
CBCS	Cluster Based Collaborative Storage
CH	Cluster Head
DCS	Data Centric Storage

Chapitre 1

Introduction

1.1 Motivation

Les réseaux sans fils bénéficient d'une autonomie de déploiement, car ils sont destinés pour des régions avec accès limité. Les nœuds capteurs sont équipés d'une source d'alimentation (limitée), de mécanismes de calcul, de mémoire (limitée) et d'un émetteur/récepteur sans fil. Bien que le rôle soit focalisé sur la collecte de données, une activité toute aussi importante est la gestion des ressources, en particulier celle de l'énergie et de la mémoire disponible.

La ressource la plus suivie est l'énergie. Les protocoles les plus utilisés ont comme une caractéristique principale la consommation minimale de l'énergie durant les opérations courantes. Pour des endroits qui ne sont pas accessibles, d'autres sources locale d'énergie ont été envisagées (solaires, thermales, ...).

Un aspect qui a été souvent négligé est la gestion des données en considérant l'espace limité de stockage d'un nœud. Ceci est particulièrement important dans les cas où la connexion avec la station de base (sink node) n'est pas fiable ou encore lorsque la connexion est interrompue. Ceci peut s'avérer une décision délibérée si la station de base arrête les opérations en suivant un plan, pour des raisons de dissimulation (surtout dans les zones non-sécuritaires).

1.2 Contribution de la Thèse

Nos travaux se situent dans le domaine de la gestion de données en considérant les contraintes de ressource mentionnées. Le problème est plus complexe lorsque les données collectées sont acheminées pour le traitement vers une station de base (sink node). Comme la station de base peut devenir inactive (défaillance ou planification), les données s'accumulent sur les nœuds. Notre contribution prévoit des mécanismes pour gérer les données, en particulier, en allouant un index lié à une collection de données afin de planifier de règles de réduction sans affecter (ou affecter très peu) l'information contenue dans ces données.

Nous introduisons deux types d'instance de données collectées: non-récurrent ou récurent. Une instance de donnée non-récurrente constitue la collection et le stockage d'une valeur d'un paramètre lorsqu'une condition est satisfaite. Une instance de donnée récurrente est représentée par un ensemble de valeurs d'un paramètre, en commençant lorsqu'une condition est satisfaite et en finissant lorsqu'une autre condition est satisfaite. Les instructions pour activer la collection (quoi ?, à quelle fréquence ?, quelle résolution ?, etc.) sont dérivées à partir des applications qui utilisent les données et sont dictées par les intérêts corporatifs.

Une fonction établie l'importance (importance factor) pour chaque unité de données. Elle prend en compte les dépendances d'une unité de données avec d'autres unités de données (valeurs, temps ...). Cette fonction permet de classifier les unités de

données par leur importance (ranking). Cette classification devient importante lorsque l'espace de mémoire disponible devient insuffisant et la réduction de données doit être invoquée.

La réduction de données peut se faire en annulant en totalité une partie de la collection de données. Une solution plus avantageuse est de diminuer la résolution ou l'intervalle d'échantillonnage (sampling interval). A cet égard, nous avons proposé des primitives qui réduisent une unité de données. Une application continue de ce processus assure plus d'espace pour les données avec une importance significative.

Nous proposons une solution complémentaire au niveau réseau, dans les réseaux multi-nœuds, où l'espace de stockage du réseau est vue comme une entité unique. Nous proposons des mécanismes pour un transfert contrôlé des données d'un nœud à un autre nœud en considérant le volume de données collectionnées et l'espace disponible, tout en ayant en vue l'effet de consommation d'énergie. Finalement, nous abordons le problème du sauvetage redondant pour les unités de données critiques et pour les nœuds classifiés à risque.

Une simulation est présentée comme un 'proof of concept', nous permettant de faire une évaluation sur les résultats de nos propositions et de développer des améliorations futures.

1.3 Structure de la Thèse

Le chapitre 1 introduit le sujet de la thèse, l'importance du sujet, ainsi que les contributions. Le chapitre 2 présente les principes de base des réseaux sans fil, leurs limitations et les principaux mécanismes et protocoles utilisés.

Le chapitre 3 se focalise sur la gestion de données, en présentant les algorithmes de base pour leurs traitements (agrégation, requêtes, sécurité....). Le chapitre 4 introduit un modèle de réduction de données sur un nœud. Il introduit également différents types de collection d'instances et des primitives afin de réduire le volume de données. Un cas d'étude utilisant la fonction de calcul d'une unité de données est présenté et discuté.

Le chapitre 5 étend ces concepts aux réseaux multi-nœuds. Nous introduisons des mécanismes pour le partage du stockage des données en considérant l'espace disponible, les priorités (par l'importance) et les fonctions de réduction. Une version utilisant la duplication des données est également proposée. Le chapitre 6 présente les résultats de nos propositions à travers un cas plus complexe. Les résultats confirment nos prévisions. Le chapitre 7 conclut sur nos contributions et détaille les sujets à étudier dans le futur.

Chapter 1

Introduction

1.1 Motivation

Wireless Sensor Networks are given a high level of autonomy for their operations as deployments are often in areas with difficult access. Sensor nodes are equipped with a power source, processing capabilities, memory, and a wireless transceiver. While performing data collection tasks, they are expected to maximize their lifetime by managing the use of their resources.

The main resource considered is the energy. Clearly, without energy, the node is useless. Protocols have been devised to handle the operations of the network while making the most of the limited amounts of energy. Proposals exist on how to harness additional energy from natural sources.

One item that has not received enough attention is the management of the limited storage space. This is especially relevant in cases where the sink node itself is unreliable or purposely unavailable. Mobile sink nodes are available and their very mobility makes it that the sink is not always in reach. Deployment in dangerous zones may forbid continuous operations of the sink node as a method of dissimulation.

1.2 Thesis Contribution

Our work is in the domain of data management under the restrictions imposed by the resource scarcity of Wireless Sensor Networks. We consider the case of data collection networks that relay all data to the sink node for off-site processing. As the sink node can fail, or become intentionally inactive, there data accumulates on the sensor node. Our proposal deals with this case by first formalizing how data is collecting and giving a methodology to rank data importance in view of applying reduction rules.

Data collection is divided into two types: non-recurring data instances and recurring data instances. A non-recurring data instance is the collection and storage of a sensed parameter value when a given condition is met. A recurring data instance is a collection of a parameter that starts when a condition is met and ends when another specified condition is met. The business case dictates what is being collected, how often, and with what resolution. In order to manage the data in blocks, as opposed to a stream, we propose a data unit production function that divides recurring data instances into data blocks.

The importance function is proposed in order to evaluate the importance of each data unit. This importance can be specified as depending on a wide array of available data in the node: time, values, other data, etc. Using this function, data units can be ranked in order of relevance. When available storage space reaches certain critical levels, a data reduction process is invoked.

Data reduction can be done by dropping entire data spans, but that is usually a poor choice. It may be better to decrease the resolution or the sampling interval of data

rather than completely erasing some of it. To this effect we proposed primitives that are applied to data units in order to reduce the data taken by an individual data unit. Applied continuously, this process ensures that data space is always available for high importance data at the expense of low importance data.

Finally, we propose an approach to extend the concept to multi-node deployments. Instead of data reduction, an option is to pool the total storage space that is available in the network. Nodes with high data output can transfer some data to the storage space of nodes with lower volume of data output. This comes at an energy cost. The issue of data redundancy is addressed where critical data or data generated by nodes in high risk areas needs to be present in multiple copies.

We show via simulation how data undergoes changes and moves between nodes of a network as part of the prioritized data reduction approach.

1.3 Structure of the Thesis

Chapter 1 introduces the topic of the thesis and the relevance of the issues tackled as well as the contribution of the thesis.

Chapter 2 presents the basic principles of WSNs, their limitations, and their intended use. Specific protocols are also presented.

Chapter 3 focuses on data handling in WSNs. It covers the basic algorithms of dealing with data, data aggregation, data search, and data security.

Chapter 4 introduces the model of data reduction focused on a single node. Collection instances are defined as well as primitives used in data size reduction. The importance function is introduced and a use case is presented.

Chapter 5 expands the concept to a multi-node deployment. The idea of shared data space is presented in the context of prioritized data reduction. The idea of redundant data copies is integrated into the proposed mechanism.

Chapter 6 presents the results of a use case built on all the newly introduced concepts. Results are presented and conclusions are drawn.

Chapter 7 summarizes the ideas proposed and presents several items in need of further research attention.

Chapitre 2

État de l'art

Sommaire

Dans ce chapitre nous présenterons les concepts liés aux réseaux sans fil qui sont vus comme des réseaux ad hoc déployés dans des régions avec un accès limité. Les nœuds de ces réseaux présentent des caractéristiques spéciales, telles que une capacité limitée de calcul, un espace limite pour stocker les données, plusieurs capteurs et des interfaces pour communique avec d'autres nœuds ou une station de base fiable. Il y a une diversité d'applications qui décrivent ces caractéristiques comme la santé, l'environnement, le militaire, etc.

Ce type des réseau utilise plusieurs protocoles, mais les deux plus sont: LEACH et UNFP. Les autres protocoles ont des utilisations plus criblées, comme la transmission des données, la collection des données, la localisation des capteurs, la synchronisation, etc.

L'énergie constitue le facteur majeur qui détermine le cycle de vie d'un capteur. Cependant, même pour les nœuds isolés, il y a des possibilités pour recharger les batteries, soit par des voies mécaniques, par lumière, par des sources thermales, etc.

Ce chapitre passe ensuite en revue les solutions actuelles et décrit les aspects qui restent ouverts.

Chapter 2

State of the Art

Summary

In this chapter we introduce Wireless Sensor Networks as ad-hoc networks often deployed in areas with restricted access. These nodes have limited computation capabilities, some memory space, sensing devices(s), and a transceiver to communicate with other nodes or with a reliable base station. The applications of such technology are diverse: environmental, military, health, etc.

Two main protocols are used to provide a structured organization in such a network: LEACH and UNPF. Additional protocols are implemented in such networks to support data disseminations, data gathering, localization, clock synchronization, etc.

A major limiting factor in the life of a sensor node is the energy reserve. This is carefully managed to last as long as possible. There are several possibilities for energy reserves to be replenished from natural sources: mechanical, light, thermal sources, etc.

2.1 Introduction

A Wireless Sensor Network (WSN) is a collection of network elements which work in collaboration to achieve tasks for which a standard network deployment is not practical. WSNs are prime candidates for scenarios where there is little to no infrastructure present either due to human inaccessibility or dangerous access conditions. The elements of such a network are designed to be very light weight from both hardware and software perspectives. Depending on design, the nodes have variable amount of processing capabilities, multiple types of memory, a wireless transmitter/receiver, and a power source. The great majority of the nodes are also equipped with some sort of sensing devices to measure parameters in the immediate area of the node [1].

WSNs are distributed systems. While developing technology to fit their requirements, one is tempted to look at other distributed systems and adapt the solutions to address the peculiarities of WSNs. Sensor networks are similar to regular networks in that data can originate at any point and can be routed to a final destination. There is traffic and data on top of which there are applications.

Unlike traditional networks, wireless sensor networks have shortcomings that dictate different priorities with respect to implementations. Except for the few cases where WSN nodes are equipped with energy gathering devices, the nodes start with a fixed allotment of power which must be efficiently used to prolong the life of the node. The links between nodes are wireless which makes them inherently unreliable. Node failure, which impacts routing paths, is higher in rough environments where WSNs are

regularly deployed. For these reasons, simple adaptations of current network protocols for use is WSNs is not feasible [2].

The remainder of this chapter presents the architecture of a WSN and protocols that have been devised to work around the constraints imposed by the scarcity of resources.

2.2 Architecture

The classical deployment architecture of a wireless sensor network consists of several sensor nodes and one sink node. The sensor nodes are in charge of collecting data at their location while the sink node provides an access path to the network via a reliable connection.

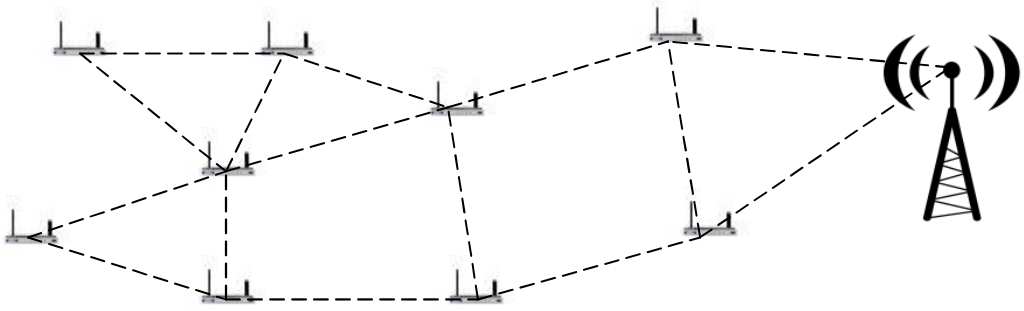


Figure 2.1. Deployment of a wireless sensor network

Figure 2.1 shows the generic deployment for a wireless sensor network. It reflects the fact that some nodes are within wireless transmission reach of each other via the

dashed lines, and that there is a multi-hop path for every node to communicate data to a central station with further relaying capabilities.

Depending on the nature of data collection that networks perform, there are variations that can be brought to this model. There are factors that affect the structure of a network compared to the layout presented above: sink redundancy, sink mobility, and sensor node mobility.

Sink redundancy is desirable when the sink node itself is not reliable or when there are very few nodes within wireless reach of the sink node. Since sensor nodes are prone to becoming faulty, having very few within reach of the sink node is a risk. In this case, reliability is not only dictated by the nature of the hardware deployed, but also by the conditions where they are deployed. For example, WSNs can be placed in combat zones where risk of hardware damage is very high.

Mobility is another design factor. Both the sink node and the sensor nodes can be mobile. Sink node mobility helps in situations where the deployment is in a very remote area, such as a jungle, where installing a base station is not feasible or perhaps not worth it due to a short term mission for the deployment. In such cases, the sink can be flown within contact range where either data can be retrieved for a full data acquisition, or queries can be performed on stored data. The nodes themselves can be mobile, but because they have such limited power, the mobility needs to come from external sources. We have encountered the case of nodes that float down a river and also the case of nodes attached to animals.

2.3 Application Domains

The areas of applications of sensor networks fall into two categories: those where reliable infrastructure is not available, and those where deploying reliable infrastructure is not feasible. In this section, we describe some scenarios where WSNs are the method of choice for input gathering. Additional details are presented in [3].

2.3.1 Surveillance

In combat zone situations, covert surveillance can be quickly deployed via wireless sensor networks. WSNs have the ability to form ad-hoc networks and can provide information collected by seismic, chemical, acoustic, and video sensors. Deployment opportunities are scarce, so the WSNs' approach of energy conservation and resilience to node loss can provide service for a long period of time. VigilNet is a system built to support such operations. It detects events such as presence of people, people with weapons, and vehicles. It also provides position information so that a tracking can be performed. Tripwires within VigilNet can activate additional sensors only when necessary thereby increasing their efficient use of limited power. Such a system can provide continuous service of near real-time data for 3 to 6 months [4].

2.3.2 Medical

For patients needing constant monitoring, it can be very expensive if the monitoring task is performed directly by medical workers. By using sensors attached to

the patient, monitoring can be achieved. At the same time, the bulky and wired monitoring devices are avoided. Small miniature on-body sensors can be used as well as external sensors such as cameras. Such a deployment can also benefit from a better ease of access to address failing equipment. There are already solutions available assisted living facilities based on sensor network technology. One solution is AlarmNet which can cover large-scale units. It integrates data collected from body sensors, sensors within the living space, as well as sensors attached to mobile units [5].

2.3.3 Environment

Environmental surveillance is effort intensive, and consequently costly, when done in person. Large areas cannot be covered with the same ease compared to deploying a WSN. Environments vary in landscape and features, but often they prove to be inaccessible. The situation is somewhat similar to combat situations, but for different reasons. When long term data collected from vast regions is needed, WSNs provide a feasible solution [6]. Similarly, monitoring animal migrations and movement patterns, WSN nodes can be attached to animals and allowed to roam. These nodes can move out of signal reach, but eventually make it back carrying with them relevant data [7].

2.3.4 Special cases

With some imagination and relaxation of the initial definitions of a wireless sensor network, the WSN technology can be adapted to some unexpected situations. In this section, we give a few examples of such situations.

Livestock monitoring is a fairly complex task when applied at a very large scale. This task is usually left up to the individual farms without mandatory reporting. As it turns out, cattle mobility can give serious hints about possible disease outbreak. It becomes interesting to monitor such activity. In [8], the authors propose a no infrastructure solution consisting of self-organizing nodes where data dissemination is supported by proactive caching.

Roaming profiling is still concerned with animal monitoring, yet it focuses on a very small set of individuals. This case is more tied in to the environmental observation case, but the fact that the possible mobility area is tens of square miles, creates new problems. There is a need for a large number of sink nodes to be able to make contact with the roaming sensor at an acceptable rate. In this case, the “sink” is a mobile trapping unit in charge of capturing the animal along with the sensor that it carries. Although not in the context of WSNs, such a case was recently shown on the Animal Planet show “The Trapper and The Amazon”.

Underwater wireless sensor networks are a special case because water absorbs a great amount of radio frequency waves. Unless positioning is very precise and nodes are linked via cables, communication is only possible at very short range, and even in that case, at a high energy cost. Acoustic transmissions are possible under water, but the cost

of hardware becomes a problem in that case. An acceptable solution for underwater sensors is to have mobile units that can move from sensor to sensor and perform data delivery which would otherwise happen via radio transmissions on land. The same device can then navigate to the sink node to make the data available. This works well in non real-time deployments [9][10][11].

2.4 Network Protocols

While specialized sensor networks call for specialized protocols, all sensor networks need to handle basic organizational, traffic, and data handling protocols. In this section, we introduce existing approaches to problems inherent to wireless sensor networks.

2.4.1 Architecture

WSNs can consist of very large number of nodes. For the purpose of energy conservation, these nodes need to be organized so as to function in an orderly fashion. The organizational decisions will later impart data flow through the WSN as well as between the WSN and the sink.

LEACH proposes a solution where the nodes form clusters. The clusters elect a head which is in charge of communication with the sink [12][13].

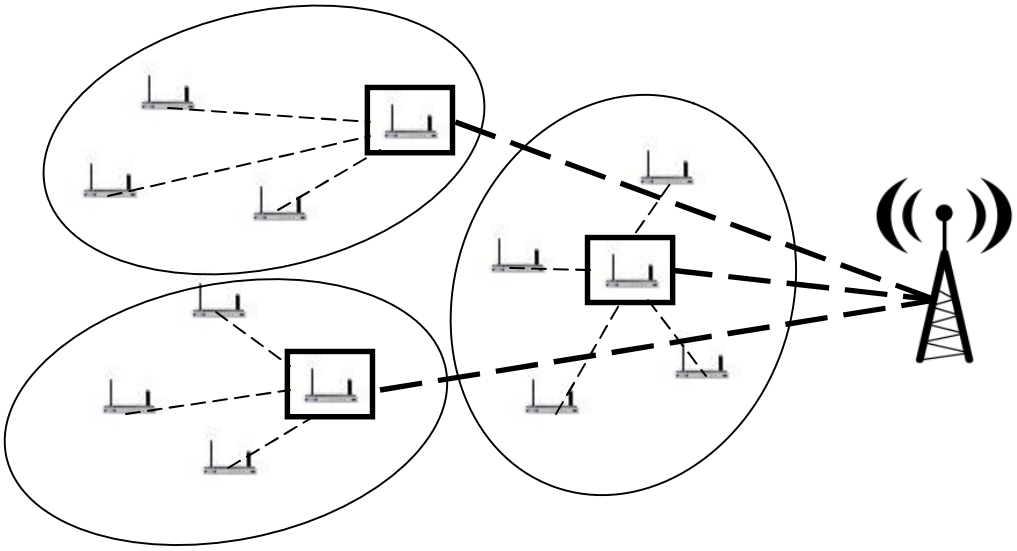


Figure 2.2. WSN organization with LEACH

Figure 2.2 depicts the conceptual organization of WSN nodes in clusters. Each cluster head is responsible for communication with the sink node on behalf of every node in the cluster. Clusters are formed on a geographical proximity basis. This gives non-head nodes in the cluster the option to reduce their transmit power when transmitting data to the cluster head. The head receives all data from the nodes in the cluster and has the opportunity to perform some data manipulation before sending it to the sink. Tasks such as data deduplication and data fusion are appropriate at this stage.

As a protocol, LEACH consists of two phases. In the first phase, the clusters are formed and the initial cluster head is elected. In the second phase, called the steady-state, the data collection and flow is operational. As the cluster head uses up considerably more

energy for transmissions, there is a periodical reiteration of the cluster head election process to even out the expense of being a cluster head.

LEACH performs well under stable conditions but, being location proximity oriented, is affected by mobility.

UNPF operates as a layered architecture. The WSN nodes that are in close proximity and can communicate directly with the sink are considered the one-hop layer. The rest of the network is divided in layers. A node that is far away from the sink will have to send data via multiple hops in order to arrive at the sink [14][15].

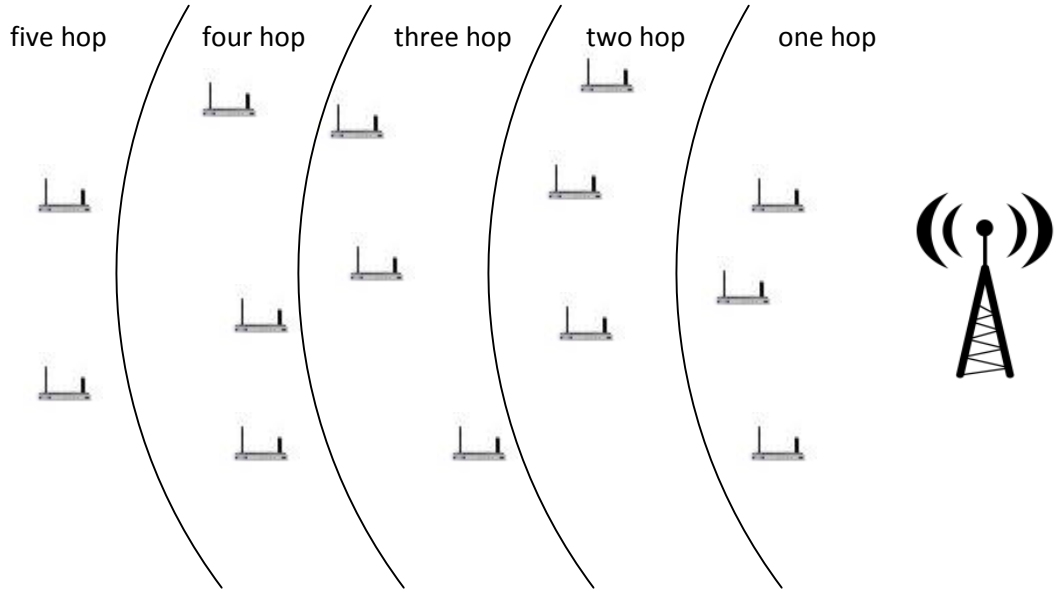


Figure 2.3. WSN organization with UNPF

Figure 2.3 shows the classification of WSN nodes with respect to transmit distance to the sink. In situations where all nodes have the same transmit power, the

division in hops closely maps to the geographical distance. In cases where the power differed from one node to another, we can have nodes further away that are fewer hops away from the sink.

The layers are established via a set of beacons. The first beacon is sent by the sink and can only reach the first layer nodes. These nodes send their ID to the sink node, at which point the sink node confirms them as first layer nodes. Following their confirmation, the layer one nodes send beacons to acquire the layer two nodes just like the sink node did to start off the process. Each node remembers the nodes in the next layer that it can use to relay data to the sink node. The beaconing process happens at regular intervals in order to account for possible changes in the network topology.

Overall UNPF has the advantage over LEACH in that it does not require any single node to transmit over long distances, hence saving power. It is also more resilient to node failure as each node contains several potential next hop nodes for transmission operations.

2.4.2 Data Dissemination

Data collected by a node can be of interest to the rest of the WSN. In order to relay data, or awareness of the existence of data, WNSs use data dissemination protocols. Several protocols are available each of them having advantages and drawbacks.

Flooding is a very simple protocol. Each packet of data that needs to be disseminated in the network is broadcast to all neighbors. The packet is tagged with the

maximum number of hops that it can go simply so that it is not broadcast forever. This type of algorithm is very easy to implement and easy to maintain. It is resilient to even the fastest changes in network topology. The drawback is in its overuse of resources. There are overlapping regions and redundant transmits of the same data to the same nodes, which results in unnecessary use of power [16][17].

Gossiping is a variation on Flooding. Instead of broadcasting the data to all its reachable neighbors, it randomly selects one to be the recipient of the message. Each node in turn sends it around until the number of hops it can go is met. In this case, there are no guarantees that the packet will actually make it to all the nodes in the network. The selection of the next recipient neighbor needs to be truly random. Improvements such as Smart Gossip have been brought to this protocol to decrease the overhead and make it more topology aware [18][19].

Rumor routing is proposed to handle the cases where there are queries for data happening. Flooding alone can be event flooding or query flooding. Rumor routing take a hybrid approach. Each node keeps a list of neighbors. When a node records new data, it adds it to the list of local events. At this point, the node has a random chance of generating an agent. The agent is very long lived and travels the network to propagate information residing on the node that created it. It updates every node on its path. If a node initiates a query and does not receive a reply via an agent, it will follow with a flood query request [20][21].

Sequential Assignment Routing is a tree based algorithm rooted at a central node which grows down the paths of most residual energy. Several trees can be created,

usually rooted at a node one hop away from the sink. Trees can overlap and different paths can provide variable energy and variable quality of service [22][23][24].

Directed diffusion uses named data and all queries are for named data without knowledge of central authority. The protocol is request driven with requests being placed by the sink. The source is found, and the intermediary nodes participate to relay the date to the sink. The path along the way is enforced. The protocol allows data propagation even in the absence of queries [25][26].

SPIN uses negotiations via three types of messages: ADV, REQ, and DATA. The messages are used to advertise presence of data, request data, and to tag data payloads respectively. The advertising of data is done via small messages which are less taxing on energy consumption than full data packets. This protocol also provides data fusion [27][28].

There are other protocols out there that dictate date dissemination, along with a large number of proposed improvements. Improvements often apply to specific use cases and don’t necessarily improve all relevant parameters.

2.4.2 Data Gathering

Data gathering algorithms are used to control the manner in which the sink node, and in consequence, the external world, gets to the data collected by the sensor network. There are conflicting requirements as to how these protocols should function as we want

to maximize the number of communication rounds before the network goes out of service while using as little energy as possible. Several proposals have been made.

Direct transmission is the naïve approach where any node gathering new data sends it to the sink. While easy to implement and not affected by network topology, the approach is very wasteful. Nodes that are far away from the sink don't take advantage of possibility to relay data and simply increase their radio power to be able to reach the sink.

PEGASIS operates by constructing a greedy chain and using it for data transmission. The data moves node to node and additional data is aggregated during the data migration. All data eventually arrives at a leader node which is in charge of relaying the data to the sink. There are two main drawbacks with this approach. There are increasing delays for nodes far away from the sink and having a single leader can create bottleneck problems [29][30].

Other than the examples above, there are several improvements that are proposed, each of them seeking to improve on specific parameters of these algorithms.

2.4.3 Additional Protocols

The main duty of wireless sensor networks is to sense and relay data. In addition to the organization and networking protocols, there are some additional factors that play an important role in WSN deployments.

Node localization is relevant to certain position sensitive protocols. In the case where nodes locations are pre-defined, this can be built into the nodes. For a smaller

deployment, GPS devices can be the answer, although they add considerable size. For large deployments, GPS devices are cost prohibitive. Nodes can key off of signal strength decay assuming they know the signal strength at the source node. They can also key off of message delay in radio transmission from other nodes [31]. For this to be achievable the next item is critical:

Clock synchronization is a problem in all distributed systems. It is known that clocks eventually drift over time and they need to be reset at regular intervals. NTP is the protocol of choice for internet connected devices, but it is too heavyweight for usage in WSNs [32]. A most complete solution is provided by FTSP [33][34] which is robust with regards to node failure and topology changes.

2.5 Autonomous WSNs

The field of autonomous systems is often times equated to the field of robotics. The studies usually focus on single entities or at the very least, entities that don't have a tight cooperation. WSNs are quite different in that nodes depend on each other to carry out tasks. So why put together autonomy and WSNs? In recent years, efforts have been made to give additional self-reliance to deployed wireless sensor nodes.

WSN protocols are already constructed with resource conservation in mind. While minimizing all usage and employing tactics such as load balancing, one still reaches a stage where deployed resources are being consumed. Additional self-reliance

can be gained by finding ways to compensate for the inherent limit on resources present in WSNs, by gathering resources to replenish what has been used.

Additional computational abilities (i.e. processing units) and additional data space (i.e. memory chips) are impossible to find readily available in deployment areas. The only target for replenishment is energy. There are several options as to where energy can be harnessed, which depends on the deployment environment:

2.5.1 Mechanical

Mechanical forces are used in conventional renewable energy sources mostly limited to waterfalls and wind. For WSNs, mechanical energy sources come in the form of vibrations and mechanical stress [35][36]. Extracting energy from a mechanical source involves a mass mounted on a spring mechanism. The mechanical acceleration induced by movement causes oscillations which can be converted to electricity via magnetic field or strain on a piezoelectric material [37][38]. Mechanical devices to harvest energy have a best yield at a specific frequency off of which yield greatly decreases. Matching the mechanical structure to the expected vibration frequency is essential.

For electromagnetic energy harvesting, the moving mass contains a coil which moves through a stationary magnetic field. As the coil moves through the magnetic field, a flow of electricity is induced.

Strain on a piezoelectric material by mechanical force causes charge separation producing an electrical field. The voltage produced varies with the strength of the

mechanical force applied. This idea has been considered in association with rain drops which provide the mechanical action [39][40].

Water flow is a viable source for anchored sensors in a moving water stream. For small scale devices, wind and water flow are not feasible [41].

2.5.2 Light

Indoors and outdoors light can be used by photovoltaic cells to produce electricity. The conversion happens at atomic level where the light rays strike a semiconductor surface, which captures part of the energy. This energy is enough to knock some electrons loose causing them to flow. An electric field causes the electrons to flow in specific direction. In theory, this is a feasible solution for sensors with reasonable access to sun light (for outdoors deployment) or appropriate spectrum (for indoors deployments).

Problems are encountered given the large surface of photovoltaic cells needed to produce a significant amount of electricity. Varying weather conditions affect the results of this technique [42][43].

2.5.3 Thermal Energy

Heat cannot be used to produce energy, but a flow across a temperature gradient can. The heat flow is accompanied by a charge flow as energy carriers migrate from high concentration to low concentration regions. This creates a voltage difference between the

cold and hot regions. Significant temperature gradients are required for this to create useful amounts of energy, usually 10C being the minimum usable temperature difference. For most small size WSN nodes, differences of 10C are not common. The solid state operation with no moving parts is an advantage to this technology, but the research targeted for micro components is only now starting to gain attention [44].

Thermophotovoltaics is a different approach to thermal energy capture. Instead of using heat gradients, it proposes the harnessing of energy from the photons emitted at temperatures above the photovoltaic device values [45][46].

2.5.4 Electromagnetic Energy

A wide wavelength spectrum is filled with transmissions from radio, TV, telephony, and wireless networks. The energy from these waves can be converted to useful energy. A specialized antenna can convert this to DC current. The conversion rate is very high although the actual energy content of these waves is very limited. The size of the necessary apparatus also poses a challenge for practical uses of this idea [47].

2.5.5 Human Body Energy

The human body is a perfect idea for providing energy especially for body sensors. There are several human body sources that can generate energy. Some of these ideas imply active involvement of humans with activities that are not part of routine: riding a bicycle or turning a wheel. Both of these tasks are regularly tapped into to power

a flashlight so there is enough potential power production in those instances. Other activities such as walking, arm movement, and breathing, are naturally occurring with some specific expected rate over a 24 hours cycle [48][49].

2.5.6 Energy Harvesting Issues

Additional items of concern are related to energy harvesting, regardless of the harvesting method used. The actual availability of the physical phenomenon to produce the energy is an issue. Communication standards between devices place requirements on power usage, especially related to the minimum usable power levels. Energy storage can be done with rechargeable batteries. In this case, additional power expenditure is incurred to operate the electronics that manage the charging profile of the batteries [50][51].

2.6 Conclusion

In this first chapter, we introduced the concept and presented use cases for wireless sensor networks. We presented algorithms used to organize the ad-hoc network into a hierarchy. Network protocols used to lay out data traffic paths were summarized. Additional protocols that deal with clock synchronization and geolocation were presented. Finally, the issue of autonomy was presented. Energy conservation being the key to WSN life, additional methods to harvest energy were presented.

The aim of a WSN is to collect data. In the next chapter, we present some of the aspects related to data. Many scenarios can arise where data manipulation is more

complex than a simple “collect and relay” approach. From the collection point to the sink, data can undergo many transformations: encryption, deduplication, compression, etc.

Chapitre 3

Données dans les Réseaux des Capteurs sans Fil

Sommaire

Ce chapitre présente les concepts de base pour la gestion de données collectées. Les réseaux de capteurs sans fil sont déployés pour les données dont la collecte s'avère difficile. Ainsi, certains protocoles sont conçus spécialement pour la gestion des données. D'une part, il y a des protocoles qui collectent des données et surveillent le trafic ; et d'autre part, des protocoles qui assurent la protection des données contre les attaques.

Afin de réduire l'énergie consommée pour la transmission des données, des algorithmes de compression sont utilisés, tout en assurant une perte tolérable ou minimale d'information. On peut rajouter aussi des algorithmes pour l'agrégation et la duplication de données afin obtenir des volumes plus petits à transférer.

Comme les modèles de trafic peuvent indiquer que certaines attaques altèrent les données, des algorithmes spéciaux détectent les anomalies dans l'architecture des réseaux. En même temps, la corrélation entre le type de trafic et les événements dans les réseaux peuvent rendre les réseaux vulnérables et ainsi les intrus peuvent obtenir l'information quant aux types de données collectées. Toute mesure de dissimulation induit des coûts en termes d'énergie. Une solution basée sur des délais aléatoires de transmission peut s'avérer alors satisfaisante.

Les données collectées par les capteurs peuvent être utilisées par le biais de requêtes spécialisées ou envoyées à la station de base pour un traitement plus laborieux

(off-line analysis). Il n’y a pas de solutions acceptables concernant le volume agrandissant de données lorsque la connexion entre un nœud de capteurs et la station de base est interrompue.

Chapter 3

Data in WSN

Summary

WSNs are deployed in order to gain access to data that otherwise would be difficult to collect. It follows that several protocols have been designed to address the issue of data handling. There are protocols that cover the aspects of data collection and traffic, and there are protocols to ensure that these operations occur in a secure manner without interference from potentially malicious attacks.

Lossless compression algorithms are used to reduce the amount of energy needed for transmissions. Data aggregation and deduplication algorithms are also employed to further compact the data for a less costly transmission.

Security can be compromised even if attackers do not get access to the data in the network. Traffic patterns can give hints about the architecture of the network. Correlation between traffic levels and events can lead to attackers gaining knowledge as to what type of information a WSN is collecting. Dissimulation measures come at the cost of additional energy being spent. Random delays can prevent correlation attempts.

Once data is collected by the network nodes, it can be made available to distributed query engines or it can be sent to the sink node for off-site analysis. There are currently no proposed solutions to deal with increasing data loads when the sink node is unavailable.

3.1 Introduction

Wireless sensor networks are all about data: sensing data and making it available. In this chapter, we review some of the changes that data can undergo and data base systems that support the necessary functions of a highly distributed database.

3.2 Data manipulation

There are two driving forces behind data manipulation in wireless sensor networks. First, the size of the data needs to be reduced simply to avoid moving around unnecessary amounts and thereby using up more energy than necessary. In these cases we have deduplicatios and aggregation. The data also needs to be protected. In remotely deployed networks, it's conceivable that malicious attacks can be mounted by instances seeking illegal access to the data or seeking to corrupt the gathered data. In this section, we review some of the changes that the data can go through.

3.2.1 Compression

Lossless data compression is meant to reduce the size of transmissions, and hence helps limit power usage. In [52], the authors introduce compressive wireless sensing. They propose that a deployed fusion center receive sensed information from several spatially distributed sensor nodes. Based on compressive sampling theory, a matching is proposed between the source and the communication channel. As a result, there is an

increase in latency of data but at the same time, a reduced size of the transmitted data. A variation of the same idea is presented in [53] where distributed compression is used.

[54] builds on the idea of a distributed compression. The proposed system computes random projections of the sensor data and distributes them in the network using a gossip type algorithm. The statistics are stored in several places in the network and they can be used to construct a good approximation of the data available on all nodes in the network.

3.2.2 Security

The resource limitations present in wireless sensor networks also affect the range of security measures that can be deployed on WSN nodes. Given that such deployments are generally not attended, security measures still need to be put in place. Attacks can seek to simply disrupt the operations of the WSN, or they can seek access to data and information from the network. In this section, we cover some of the aspects related to data security and privacy.

Common attacks against data are eavesdropping, traffic analysis, and camouflage [55][56][57]. Eavesdropping consists of simply listening to the data that is being sent from one node to another and collecting sensitive information. This information can compromise components related to the WSN as it can contain control information. Traffic analysis consists of statistical interpretation of packet counts. An increase in packet counts can show that the node in question has registered activity that it is meant to

monitor. With such analysis over time, specific node profiles are built and their tasks can be inferred. Camouflage is the technique by which one node inserts itself into a deployment, attract traffic, and then misdirect it.

There are solutions to these problems. Some solutions simply require protocol changes, others require additional resources. The use of the secure SPINS protocol prevents any meaningful eavesdropping [58]. Inserting additional bogus traffic can confuse the statistical collections for traffic analysis, but it comes at the extra cost of additional overhead traffic.

[59] presents an algorithm that one can use to minimize the effect of the security breach in a compromised network. The routing protocol proposed specifies the use of multiple routes to forward chunks of a single data. Even if certain routes are compromised by eavesdropping nodes, the adversary cannot get access to the entire information. If these chunks are part of an encoded data segment, the captured partial data yields no information as to the contents.

In [60], the author proposes two approaches on improving data security: camouflage and evasive data storage. The principle of camouflage is to decrease the odds that an adversary can tell between an active node and an inactive node. This can be done via fake traffic generation as well as delayed traffic generation in order to break the correlation between data traffic and events. An evasive data storage approach has as objective to dissimulate the exact location of long term data storage. It also seeks to dissimulate the identities of data aggregator nodes.

3.2.3 Dissemination

Two approaches to data dissemination protocols are presented in [62]. An acknowledgement based dissemination protocol is along the lines of a traditional unicast protocol. Data chunks are sent and confirmation of receipt is expected. In the case there is no confirmation within a time window, the unacknowledged data segment is retransmitted. Due to the nature of wireless media, a unicast can be picked up by any sensor within reach. Though not required to send acknowledgements, sensors other than the targeted one will most probably receive a copy of the relayed data. In case some of them have certain data pieces missing, they can request them once the channel is available for communication. To continue the dissemination of the data, each node having received new data will advertise it and will send it to reachable nodes [61].

Request-based data dissemination improves on the acknowledgement overhead associate with the above proposal. A node having data to disseminate will first broadcast knowledge of the data chunks that make up the data. The node will wait for a given time to give a chance to the neighboring nodes to express interest in the data. After the timer expires, the node starts broadcasting the chunks. Once the broadcast in complete, the node waits for re-requests that neighboring nodes make in order to get the data chunks that may have not been properly received [62].

3.2.4 Aggregation

In many instances, the end user of a wireless sensor networks only needs an aggregated form of data. They may be interested in minimum, maximum, and average

values for a specific parameter. In these cases, it makes sense to perform the computational tasks in the network - which cost little in the way of power - as opposed to sending entire sets of collected data to have the computation done at the sink node or beyond. Real time requirements from sink may pose a limitation on how much information can be aggregated before delivery [63].

In currently proposed solutions, trees are used to frame the data aggregation process:

Center at Nearest Source (CNS) is an approach where every node sends the information to the node closest to the sink. This node performs aggregation functions before relaying the data to the sink. **Shortest Path Tree** (SPT) is a solution where overlapping paths are merged to form trees. **Greedy Incremental Tree** (GIT) is built starting with the path from sink to the nearest source, to which the next nearest source is added to form the tree [64][65].

3.2.5 Time

Data collected by sensor networks can have both real-time and historical relevance. Historical data needs particular storage and indexing. As this can amount to a large size, it will be distributed across several nodes. In [66], the authors introduce a provenance-aware data storage approach which deal with indexing and accessing historical data.

In the case where real time data availability is needed, a different approach is needed Power consumption takes a second priority as it is better to perform for a limited lime at expected parameters than a longer time while providing unusable data. SPEED and RAP are two protocols dealing with real-time data delivery. They are based on a model that includes how close to real-time the data availability actually needs to be. Also factoring in distance and velocity of the packets, the data transfer is done so as not to miss the delivery deadline. Both protocols make extensive use of geographical protocols [67][68][69].

3.3 Query

Distributed databases are used to store the data for wireless sensor networks. This storage can be meant for in-network queries or simple storage for periodic relaying to the sink node.

TinyDB is a query engine that runs on sensor nodes and provides an SQL-like interface for data access. Once the data of interest is specified, the TinyDB engine manages the retrieval and aggregation of data which is then routed towards the initial requester using power conserving algorithms. Tasks are optimized via query batching and reordering of predicates. The aggregation stage of TinyDB makes used of Semantic Routing Trees (SRT) [70].

Geographic Hash Tables (GHT) provides a convenient method to map the location of data [71]. Using hash functions on data keys, data location is computed. A

perimeter of nodes is established in the area where the hash function pointed to for the data storage. These nodes will end up receiving a copy of the data to protect against potential node failures. The nodes and the query initiators share the same hash functions. This approach reduces the communication overhead in situations with large numbers of deployed nodes.

Another approach to a WSN database is Cougar [72]. Cougar assumes a centralized indexing of all data, although sensor clusters can provide a tiered aggregation method. Each sensor node implements a light weight database component to assist in the distributed querying process.

3.4 Sink failure

All wireless sensor networks depend on a sink node as their gateway for data transfer or incoming data queries. A question is what happens to data during times when the sink node is not available. Very limited research is available in this area, nevertheless, the subject has been considered in some occasions.

In [73], the authors propose a scheme that is based on having replicated data sinks. Under normal operating procedures, all sink nodes are available. Each sink is responsible to handle a specific number of sensor nodes. When a sink dies, the nodes that were in that sink's responsibility are simply handled by other sink nodes based on proximity considerations.

3.5 Conclusion

Data handling is an important aspect of wireless sensor networks. Considerable amounts of energy are spent on securing the data and dissimulating the locations where data is stored. Network wide query languages have been developed. However, very little attention has been given to the scenario where the sink node fails or is only intermittently available. Data gathering cannot stop, yet continuously accumulating data poses a problem for the nodes.

In the following chapters, we tackle the issue of data collection during sink node outages. We propose mechanisms that ensure optimal data survival and that ensure the data of most interest to the network operator is retained in its most fine grained state.

Chapitre 4

Réduction des Données: un Seul Nœud

Sommaire

Dans ce chapitre, nous proposons la réduction de données en considérant un index de priorité attaché à ces données. D'abord, nous classifions les données en deux catégories, soit récurant et non-récurant. Les données devant être réduites sont partagées en unités de données (en utilisant des fonctions spécifiquement introduites) qui peuvent être manipulées indépendamment.

La nature d'une telle unité peut avoir des liens avec d'autres unités. A chaque unité on associe un niveau d'importance. Lorsque l'espace de mémoire diminue sous un certain seuil, les unités avec le plus petit niveau d'importance sont soumises à la fonction réduction.

La réduction des unités est étroitement liée aux mécanismes internes corporatifs. Cependant, les primitives proposées permettent une réduction appropriée de la nature de l'application qui utilise ces données. L'idée de base est que, bien que la réduction décroît l'espace occupé, elle préserve la valeur de l'ensemble des données sans changement (minimum side effect).

Un cas d'étude permet de montrer les mécanismes proposés et la nature des résultats obtenus.

Chapter 4

Prioritized data reduction: single node

Summary

In this chapter we introduce the prioritized data reduction. Data collections are characterized as either recurring or non-recurring. The data reductions are further divided into manageable data units using a data unit productions functions.

Depending on the nature of the data they carry in relation to other data, each data unit is assigned an importance level. When the available data space is below a threshold, the least important data units are targeted for data reduction.

Data reduction is dictated by business case. The reduction process can employ any combination of several presented primitives. While the reduction decreases the space occupied by the data unit, it only minimally affects the relevant information that can be extracted from the data.

An example is given that shows how the mechanism is used and the nature of results that can be expected.

4.1 Introduction

As we have seen so far, wireless sensor networks have evolved into complex deployments where their nodes can have a full network protocol stack, database systems, etc. The main rationed resource in such a deployment is energy. Having power usage tightly managed ensures a long operational life for the node in cases where replenishments (either via recharge or battery change) are difficult or impossible.

The basic deployment of wireless sensor networks consists of sensing nodes as well as a relay node (i.e., sink), which collects sensory data to be relayed via a reliable network [74]. The sink node can become unreachable due to malfunction, scheduled uptime or, in the case of mobile sink nodes, due to being out of the sensor nodes' reach [75]. In addition, the sensor nodes may decide against relaying data for some period. In these cases, optimal use of sensor node memory space also becomes critical. In this section, we classify data types and establish a set of node level approaches that can be taken to make the most of limited data storage via a prioritized data reduction. We conclude that such a methodology enables the node to be useful by collecting data beyond the point where its data storage size would otherwise allow.

Much research has been devoted to optimizing the usage of limited resources in WSNs. In particular, energy has been a main focus. Very resourceful power sources have been suggested, nuclear energy included. Both routing and dissemination protocols have

been approached from the energy conservation standpoint [76][77]. In addition, some proposals target replenishing energy from sources such as sun, wind, water flow, etc. [78].

Other resources related to WSN deployment have received less attention compared to energy source and usage. In this chapter, the focus is data storage, which, just like energy, is limited. Unlike energy, which once used is gone, data storage space can be reclaimed by discarding existing stored data. Another difference between energy and data storage constraints is that one can propose ways of harnessing energy to prolong the life of a wireless node, but so far, there is no way of harnessing data storage space.

There are many examples outside of the WSN world where storage space is a factor. Many cases are outside of the technology world and include storage spaces, garages, warehouses, disk drives, kitchen drawers, video surveillance recording, etc. Unlike the WSN scenario, these cases allow for direct human intervention, i.e., additional storage space, although for a cost, can be achieved.

We consider a WSN node in cases where unloading the data is not possible at all times. The sink node, or the next hop routing node, may not be available at times. Referring to Figure 1, if the sink is not available, the nodes cannot relay their data. Similarly, if one node is unavailable, it cannot relay data from other nodes as part of a routing path. It is the responsibility of the node to use the storage space in order to hold the most relevant data until this data can be relayed at the expense of less relevant data. A set of business logic instructions that are deployed with the node provide the decision

making. The same principles can apply to sensor nodes that are deployed and later physically retrieved for data extraction without ever having to wirelessly transmit any data.

4.2 Related Work

Current work in WSN associated storage management revolves around energy efficiency in manipulating and querying the data [79][80], improving the characteristics of stored data [81][82][83], and making use of adjacent nodes in order to gain access to additional storage [84][85].

Norbert Siegmund *et al.* [81] propose FAME-DBMS to provide a robust data storage solution. This system ensures reliability and integrity of the data, and provides a customizable query engine. It answers to the requirements related to data retrieval more so than to storing data. In order to deal specifically with encryption, Joao Girão *et al.* [82] present TinyPEDS, an encryption data storage engine.

The energy usage is still relevant when focusing on storage and querying. Joon Ahn and Bhaskar Krishnamachari [79] evaluate the scalability of a WSN performance with respect to the distributed nature of data.

Kyungseo Park and Ramez Elmasri [80] evaluate several storage schemes in terms of where the data is stored, what types of routing protocols are most appropriate, and finally the impact that each storage approach has on energy usage.

Majid I. Khan *et al.* [84] present the problem of data persistence in a congested WSN scenario. The main point is that congested networks can drop packets, which in turn translates to a waste of energy equivalent to the cost of sending the dropped packets. The proposed approach involves clustering where cluster nodes can act as temporary buffers during congestion periods.

Current work on WSN related storage has been limited to data characteristics and management across several nodes. The case of a standalone node has not yet been considered.

4.3 Components of a WSN Node

In this section, we describe the conceptualized components performing the tasks - known as well as newly proposed - associated with a WSN node.

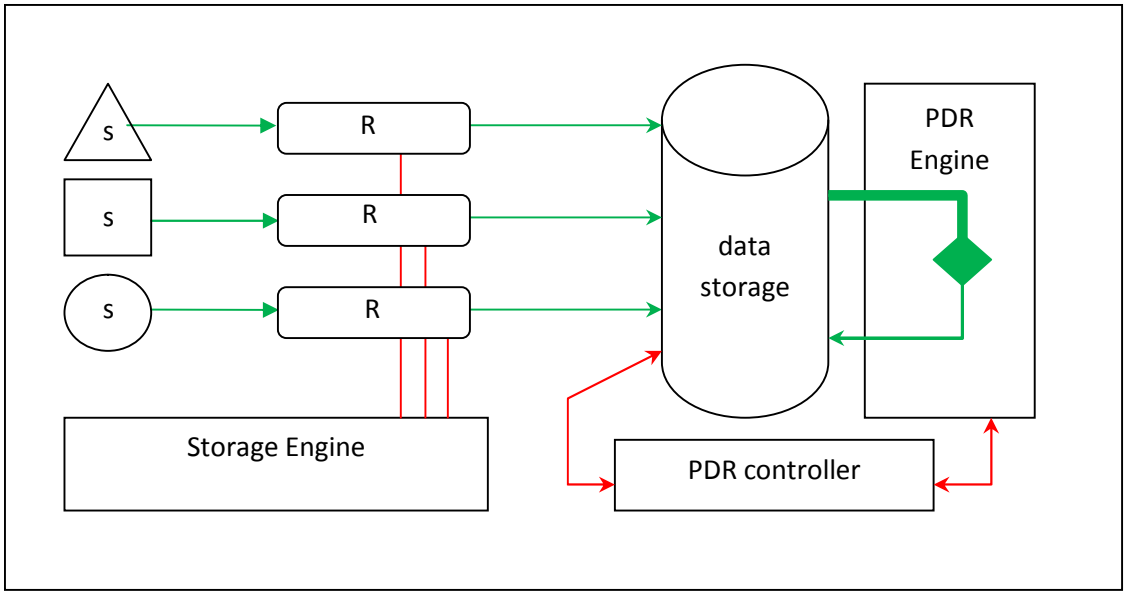


Figure 4.1. WSN node building blocks

There are several physical sensors s on a WSN node, each specialized in sensing a specific parameter: temperature, pressure, etc. Each of these physical sensors has **registers (R)**, which are updated to reflect a currently observed parameter vale. The **Storage Engine (SE)** is concerned with writing data to the node's storage. It makes no judgment as to the relevance or importance of the data itself. It simply follows data collection rules established by the business case and sends them to the node's permanent storage. At this time, there may enough space on the storage device in which case the data is simply recorded, or there isn't enough space at which point some data reduction occurs: either on the incoming data, existing data, or both. The **PDR Engine (PDRE)** contains all the data reduction rules, which are a direct reflection of the business case. They are not constantly

applied, but at specific times and with specific space recovery objectives as dictated by the PDR Controller. The **PDR Controller (PDRC)** is responsible for monitoring the state of the available storage, and, if dictated by the business case, triggers the PDRE to perform data reduction operations. Deciding what data to target and how much to reduce it is again subject to the business requirements.

This section focuses on the operations executed inside the PDR engine. For that matter, we first characterize how the SE operates.

4.4 A Model for Data Classification

Triggers or conditions are used to control data collection. These conditions trigger the SE to perform transfer operations from R into the data storage.

4.4.1 Conditions

We break down the conditions into two categories: value-based and sequence-based. Value-based conditions are evaluated as a whole and can immediately evaluate to TRUE or FALSE:

---

if ((temperature > α) && (humidity > β))

Figure 4.2. Value-based conditions

Sequence-based conditions are evaluated in sequence. We must establish a chain of TRUE evaluations in order for the entire condition to be considered TRUE. If one element in the sequence evaluates to FALSE, we pause there until the next round of evaluations:

```
sequence {
   s1: (temperature < 0C)
   s2: (temperature > 0C)
   s3: (temperature > 10C)
   s4: (temperature > 20C)
   s5: (temperature > 25C)
}
```

Figure 4.3. Sequence-based conditions

Sequences are useful in establishing trends. Even though one sequence item may temporarily be FALSE, once we arrive to s5 and it evaluates to TRUE, the sequence is said to evaluate to TRUE.

4.4.2 Data Collection

Data collections can be classified in two categories, based on the periodicity of the collection: there are samplings at defined intervals (recurring), or single shot samples (non-recurring).

4.4.2.1 Recurring Data Collection

Recurring data collections involve taking specific measurements at defined time intervals. One example of this type of data collection is temperature. We can specify such intervals at microseconds to hours and even less frequent. The sampling rate would depend on the exact use for the data collected and according to the business model. Not all recurring data sampling is enabled by default and continuously done throughout the live of the sensor. There are conditions that can trigger starting or stopping a series of such data collections.

As an example, we assume we are monitoring temperature on the side of a volcano in order to detect abnormally high values. We assume that baseline values are available. Under normal conditions, a few degrees difference warmer than prevailing temperatures may be acceptable, but once the temperature crosses a certain value (hinting of some sort of activity), it becomes interesting to start taking measurements of several factors: sound, land vibrations, gas composition, etc. When the ambient temperature returns to a specific value, it may not be of interest to sample a wide variety of parameters.

Recurring data sampling can be defined by a start condition, a stop condition, a sampled parameter, and a recurrence window.

start: (temperature > (baseline + Δ))
stop: (temperature <= baseline)
sample parameter: sound
recurrence: 1ms

Figure 4.4. Recurring data sampling definition

Once a collection has started, a second instance of the same collection cannot start even though the start condition evaluates as TRUE.

It serves no purpose to sample data any faster than the sensory devices can update registers holding the sensed data.

As a special case, the primitive values TRUE and FALSE can be used as a start condition and a stop condition respectively, and hence a continuous sampling is achieved.

We label as a *recurring data instance (RDI)* a recording of the entire set of data points from the collection start to collection stop, or to current time if the collection has not stopped.

Defining an RDI

RDI: (T_{start}, T_{end}, param, recurrence, resolution, compression), where:
T_{start}: start time
T_{end}: stop time
param: the parameter being collected

recurrence: how often the collection is done
resolution: how precise the stored value is
compression: boolean stating if the RDI has been compressed
e.g., RDI(2009/12/06 16:43:23, ongoing, temperature, 60 seconds, 0.01C, FALSE)

Figure 4.5. RDI definition

4.4.2.2 Non-Recurring Data Collection

Non-recurring data collection happens when a specific condition is met. It leads to a single value being stored every time the condition evaluated to true. Such conditions must be written as a sequence so as to avoid constant firing of the rule and hence leading to a constant parameter sampling.

```
sample condition:
   sequence {
      light < 50lx
      light > 10000lx
   }
sample parameter: temperature
```

Figure 4.6. Sample condition in non-recurring data

What the example above means is that we are sampling the temperature of a location after the sun has come up and is providing a specific light intensity. The reason we require a value increase from under 50lx to over 10000lx is to establish a trend.

If we simply state the above as :

---

sample condition: (light > 10000lx}
sample parameter: temperature

Figure 4.7. Counter example for a sampling condition

then we would have continuous temperature sampling once the light goes over 10000lx.

We label as a *non recurring data instance (NRDI)* a non-recurring stored data recording.

Defining an NRDI

NRDI: (T, param, resolution, compression), where:
T: recording time
param: the parameter being collected
resolution: how precise the stored value is
compression: boolean stating if the NRDI has been compressed
e.g., NRDI(2009/12/06 16:43:23, temperature, 0.01C, FALSE)

Figure 4.8. NRDI definition

4.5 Primitives for Space Optimization

In this section, we introduce primitives which are invoked inside the PDR Engine once the PDR Controller has identified data to be subjected to reduction.

4.5.1 Compression

Lossless data compression algorithms are widely available and used [85]. On a normal basis, compression and decompression cause little impact. On a sensor node, compressing certain portions of the data will yield available data space with no information loss. The loss is from a flexibility perspective. Once compressed, the data becomes a blob which should be treated as an atomic entity. The WSN node looses the capacity to discard partial data.

Such an approach is recommended for very critical data, and hence very important, that can never be discarded. Otherwise, it should be used for non-recurring data instances, or for portions of recurring data collections that can be dropped one whole section at a time. A parameterized compression is used to specify which span of a data instance should be compressed:

Compress[(a, b)](RDI) signals that only the data from time interval a to b is compressed, while the rest remains as initial.

The non-parameterized compression affects the entire RDI.

Usage on non-recurring data:

Compress(NRDI(2007/11/24 13:21:37, humidity, 0.01%, FALSE)) = NRDI(2007/11/24 13:21:37, humidity, 0.01%, TRUE)

Usage on portions of recurring data by first dividing the data instance several data instances:

Compress[(2008/11/11 12:00:00, 2008/11/12 12:00:00)](RDI: (2008/11/10 12:00:00, 2008/11/13 12:00:00, temperature, 3600 seconds, 0.01C, FALSE)) = RDI: (2008/11/10 12:00:00, 2008/11/11 12:00:00, temperature, 3600 seconds, 0.01C, FALSE) + Compress(RDI: (2008/11/11 12:00:00, 2008/11/12 12:00:00, temperature, 3600 seconds, 0.01C, FALSE)) + RDI: (2008/11/12 12:00:00, 2008/11/13 12:00:00, temperature, 3600 seconds, 0.01C, FALSE)

At this point, the second data instance can undergo compression and becomes:

RDI: (2008/11/11 12:00:00, 2008/11/12 12:00:00, temperature, 3600 seconds, 0.01C, TRUE)

while the first and third data instance retain all original data.

Figure 4.9. Usage of compression primitive

4.5.2 Thinning

For a non-recurring data instance, thinning involves simply discarding the collected data. For recurring data, thinning involves discarding a contiguous amount of data that corresponds to a time span of low importance in the case of recurring data instances. This can be used when the collection has a cyclic pattern and a long sampling period gives little additional insight when compared to a somewhat shorter period, or a period with gaps.

To better clarify, we can resort again to the temperature sampling example. Let's assume that we are sampling temperature every minute. This has been going on for five months. Depending on the business case, it may be acceptable, without any significant impact to data significance, to either discard data pertaining to the third operational month,

or to discard data pertaining to the third quarter of each operational month. Figure 4.10 shows the impact of thinning on a data sample.

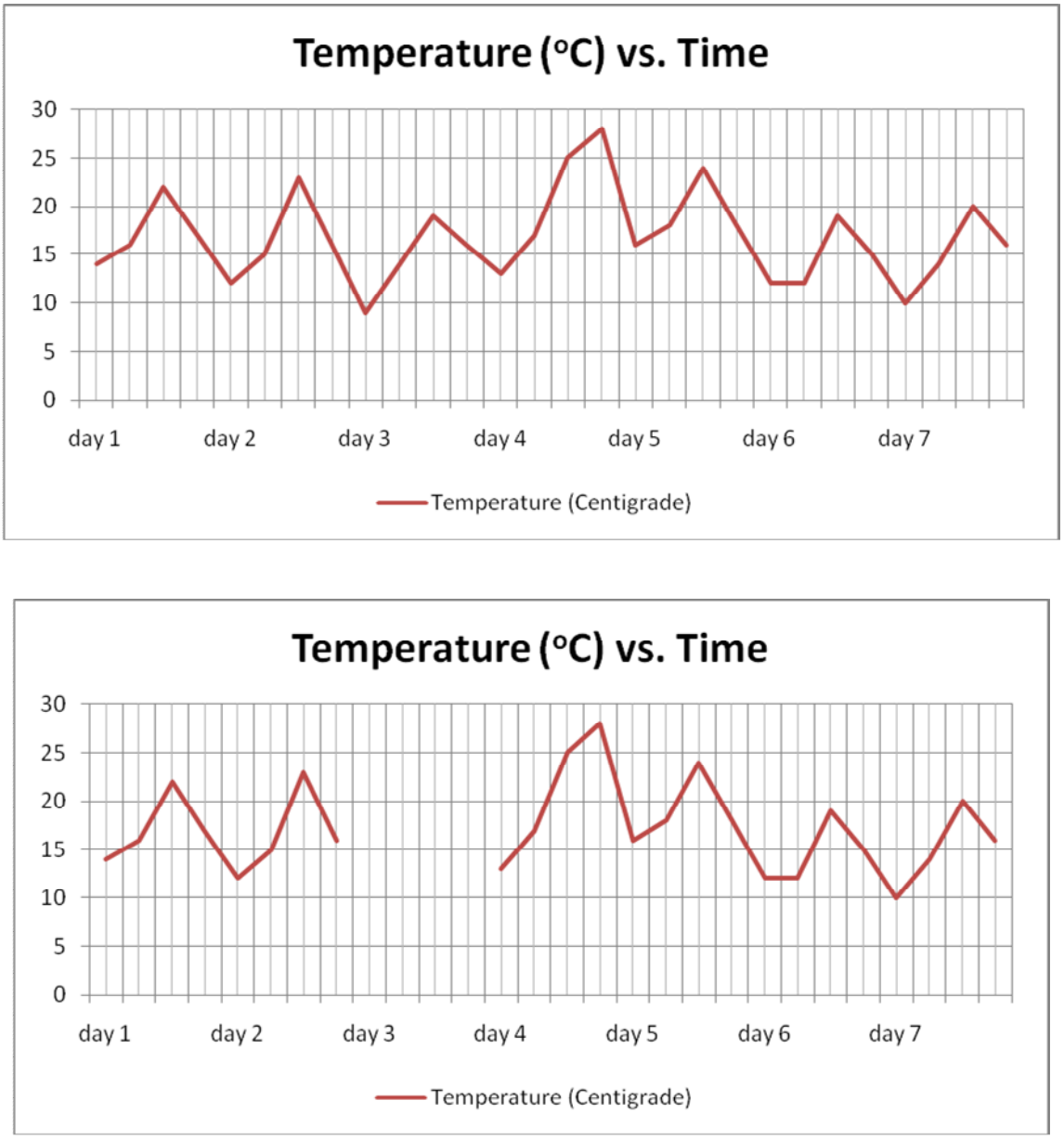


Figure 4.10. Impact of thinning on data top) initial data bottom) resulting data after thinning

Thin[(a, b)](RDI) signals that the data collected between time a and time b are dropped from the data instance.

Usage of thinning on recurring data:

Thin[(2008/11/12 12:00:00, 2008/11/13 12:00:00)] (RDI(2008/11/10 12:00:00, 2008/11/17 12:00:00, temperature, 6 hours, 1C, FALSE)) = RDI(2008/11/10 12:00:00, 2008/11/12 6:00:00, temperature, 6 hours, 1C, FALSE) + RDI(2008/11/13 12:00:00, 2008/11/17 12:00:00, temperature, 6 hours, 1C, FALSE)

Figure 4.11. Usage of thinning primitive

4.5.3 Sparsing

Sparsing can only be used recurring data collections. If the global pattern of fluctuation in the measurement is an important factor, then it is important not to lose entire spans of information. In such cases, the recurrence window can be widened by means of dropping values at regular intervals. The resolution suffers, but the overall pattern is conserved.

Looking at the same example as in 3.2, instead of dropping a full data set corresponding to day 3, we are going to double the sampling interval for days 2 and 3.

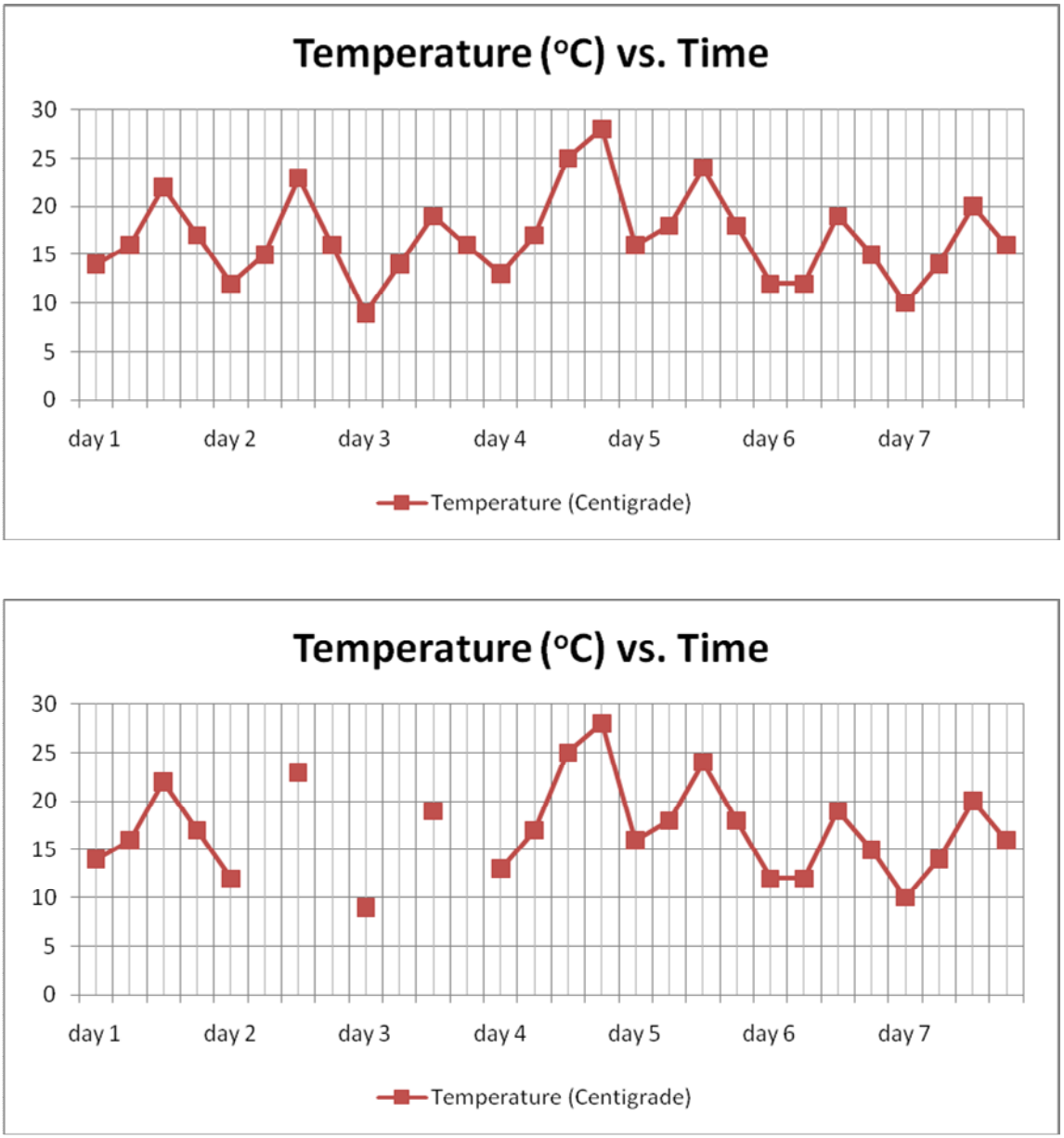


Figure 4.12. Impact of data sparsing top) on the left with initial data bottom) on the right after sparsing is applied

The syntax for sparsing is Sparse[(a,b,r,s)](RDI), which means that for the time interval between a and b, r entries are removed out of every s entries.

Usage of sparsing on recurring data:

Thin[(2008/11/12 12:00:00, 2008/11/14 12:00:00, 1, 2)] (RDI(2008/11/10 12:00:00, 2008/11/17 12:00:00, temperature, 6 hours, 1C, FALSE)) = RDI(2008/11/10 12:00:00, 2008/11/12 6:00:00, temperature, 6 hours, 1C, FALSE) + RDI(2008/11/12 12:00:00, 2008/11/14 6:00:00, temperature, 12 hours, 1C, FALSE) + RDI(2008/11/14 12:00:00, 2008/11/17 12:00:00, temperature, 6 hours, 1C, FALSE)

Figure 4.13. Usage of sparsing on recurring data

4.5.4 Grain coarsing

Data collections can be with very high precision, or can be with lower precision. For example, captured images can be anywhere from black and white to 24 bit images. To a lesser extent, this can be applied to numeric data which contains more precision in an 8 byte floating point representation versus a 2 byte integer representation.

Grain coarsing can be applied to both non-recurring and recurring data. Just like thinning and sparsing, we can opt to apply grain coarsing only to a specific interval of an RDI.

Because precision or resolution depends on the nature of the data, we give the example of an image whose resolution is measured in dpi. To specify a 50% decrease in DPI, the exact call would be GrainCoarse[(50%)](NRDI).

Usage of grain coarsing on an image:

GrainCoarse[(50%)](NRDI(2007/03/23 17:32:45, image, 300dpi, FALSE)) = NRDI(2007/03/23 17:32:45, image, 150dpi, FALSE)

Figure 4.14. Usage of grain coursing on an image

4.5.5 Range representation

In some cases we would like to eliminate most of the collected data for a specific parameter over a certain time range, yet still keep some hint of where values were for that range. In such an instance, we can keep for example the minimum value, the maximum value, the average, as well as the number of values used to compute the average and how far apart in time those values were. Note that the minimum and maximum values are not necessarily correct, but they are the largest and smallest value as far as available data samples.

A case where this can be misleading is for example a recurring data sampling that has undergone extensive sampling. In that case, it is plausible that many of the brief spikes and dips in values are lost.

The range primitive applied to an RDI produce a tuple: Range(RDI) = Tuple(T_{start}, T_{end}, Min, Max, Average). A data span that is represented as a tuple can be considered for further space optimization in the same manner as an NRDI.

4.5.6 Usage of primitives

The proposed space optimization methods are primitives. They are not the last word on a specific data collection. It should be noted that the same data can be subjected multiple times to data reduction, and each time a different mechanism can be deemed appropriate. While data morphs, its importance changes as well.

4.6 Use Case

We take the example of a single sensor node that is deployed in a dangerous climate area. The node is to collect several parameters over a long period of time. At the end, the node is either removed or a sink node is placed in proximity for a brief period in order to retrieve collected data.

The parameters that are to be surveyed are: temperature, barometric pressure, light, humidity, vibrations, and CO2 concentration. There are several recurring data collections as well as triggered single time collections.

We grade data importance from 0 to 1 as a continuous value. The value of 1 is reserved for critical data that under no circumstance should be reduced in size.

Critical data involves a full day's unaltered temperature and barometric pressure sampling every second on the first operational day, and every 10 minutes thereafter with importance 0.5. Except for the first day, recurring recordings of these two parameters are

subject to 50% sparsing while at the same time their importance increases by half the interval between current importance and 1.

Whenever a vibration amounting to 2^{nd} degree on the Richter scale, all parameters are collected for one hour at 5 second intervals and bear importance 1. This collection is followed by 24 hours of collections every minute with importance 0.8. This collection can be subjected to range representation based on 10 minute intervals, which data now becomes of importance 1.

With this data set and requirements, the simulation is allowed to proceed both with data reduction and no data reduction. Here is what is found in each case:

4.6.1 No data reduction

Under an approach of no data reduction, the storage space was used up in about 12 operational days. The following were found in the data storage:

- First day of temperature reading every second
- First day of barometric pressure reading every second
- Eleven days of temperature readings every 10 minutes
- Eleven days of barometric pressure readings every 10 minutes
- Data collections related to three vibration shocks: all parameters collected for one hour at 5 second intervals, and 24 hours every minute

4.6.2 With data reduction supported by proposed primitives

While allowing for the data reduction, here is what was found on the machine after 15 operational days:

- First day of temperature reading every second compressed
- First day of barometric pressure reading every second compressed
- Fourteen days of temperature readings every 10 minutes out of which the first six had been sparsed (i.e. are now every 20 minutes)
- Fourteen days of temperature readings every 10 minutes out of which the first six had been sparsed (i.e. are now every 20 minutes)
- Data collections related to five vibration shocks, two of which have been subjection to range representation

With all of the above in storage, there are still items that can be removed to make space for incoming data.

4.6.3 Comparison

Clearly the data reduction presents opportunity for storing more relevant data in the long run. There are some drawbacks regarding the resolution and recurrence of collected data, but the emphasis is placed on high importance data.

4.7 Freeing Space

Now that we have proposed procedures for data reduction, the question is when such data reduction should be performed. There are competing constraints to consider:

- We do not want to run the data reduction process too often
- We do not want to get overenthusiastic with data reduction as the sink may soon be available for integral data transmission

The problems faced here are similar to garbage collections in processes such as JVM. The added complexity is that, as opposed to JVM garbage collector, which collects true disposable garbage, the process we intend to run recovers space in exchange for some loss of lower importance data or loss of flexibility.

An assumption is made that while the storage recovery process is under way, the sensing devices have enough buffer space to store sensed data until the main processor is ready to take the data to the main storage unit.

4.7.1 The Pessimistic Approach

The main idea of a pessimistic approach in running a data reduction process is that the space available must always be able to accommodate the biggest possible data influx spike which cannot be dropped based on its importance. This rule can only be broken if the existing stored data has an importance level which makes it final and not subject to reduction.

Let us denote the total storage space with T, the occupied space by O, and the biggest possible influx spike by S. It follows that (T – O) >= S. Technically, we can get away with T – O = S. However, we risk fluctuating values of O which leads to a repeated

invocation of data reduction algorithm. The free space target depends highly on business logic and should be a value higher than S.

4.7.2 The Optimistic Approach

The optimistic approach keeps all data until it is time to store higher importance data. At that point, storage space will be made available by running a storage space recovery process while the incoming data is still in the sensing device buffers. For this approach, the assumption is made that data reduction is a quick process both in identifying the data subject to reduction, as well as the reduction process. In certain cases this may be true, and hence there is no need to have available space sitting unused just in case there is incoming data that needs to be stored.

4.8 Extending the Single Node Functionality

In this section, we take as a starting point the Prioritized Data Reduction (PDR) presented above and expand it to include a complete characterization of the data reduction mechanisms. The overall model of the sensor network under consideration is “dumb data collection”: collection is done by the nodes, processing is done offsite. At this point, data deduplication is not addressed as part of PDR; it is left as future work.

In [87], data importance was introduced as a continuous value assigned to collected data instances used in the data reduction process. The lowest value is 0 and the highest value is 1. In order to assign this value to collected data, the data needs to be divided in manageable pieces. Each such piece, called a *data unit*, has two associated values: *data importance* and *compensation factor*.

4.8.1 Data Units

Data collections can be both recurring and non recurring. The non recurring collections generate data that is stand alone and considered atomic. It makes sense to consider a non recurring data record as a single data unit.

Recurring data collections have several values over a potentially long period of time. Such a data needs to be divided in intervals that can be treated as a whole during the process of data reduction. We propose that such intervals be devised as contiguous time intervals. There is no need for the intervals to be of equal length. What the data units do is provide a contained set of data on which we can apply the primitives enumerated in Section 4.5.

For the sake of simplicity, we also assume that once a data unit has been delimited within a recurring data collection, it is no longer subject to further division or merger with other data units. There may be flexibility gains from allowing such operations, but this is left as future work.

The lifecycle of a data unit starts with a data collection, which can potentially go through several rounds of data reduction, and ultimately possibly dropped, assuming a long period of sink node unavailability. If at some point, the sink node is available, data is simply unloaded and space freed.

To assist in selecting what data unit to target for data reduction, each data unit is reflected by a data importance, denoted I. Once a data reduction is performed, this needs to be reflected in future importance computations. For this purpose, each data unit is assigned a compensation factor, K.

4.8.2 Data Importance

Data importance, denoted as I, is a value that numerically reflects the relevance of a data unit for the business case. The raw value of I is primarily used to rank data units in view of data reduction, but the exact magnitude of the difference does not necessarily carry a meaning.

START
WHILE ‘storage availability level’ is below a ‘threshold’
FIND ‘data unit ‘ with ‘lowest importance’
APPLY ‘data reduction’ to that data unit
ADJUST the compensation factor

RECOMPUTE ‘importance’ for any dependent data unit

Figure 4.15. High level PDRE process for data reduction

Figure 4.15 shows an abstraction of the process that happens within the PDRE when the controller triggers a data reduction operation. The reduction can occur in one or more iterations. In the first step, the data unit with the lowest importance is found and the corresponding reduction method is applied. As a consequence, the compensation factor is adjusted. All other data units whose importance factor depends on the reduced data need to have their importance level recomputed.

4.8.3 Compensation Factor

The compensation factor, K, is an importance modifier that reflects the data reductions that have already been performed on the specific data unit. The compensation factor, is defined as a percentage value between -100% and +100% and it is used in the following manner:

$$I = \begin{cases} \text{if } K < 0,\ (1+K)*I' \\ \text{if } K > 0,\ I' + K*(I_{max}-I) \end{cases} \quad (4.1)$$

where I is the data importance for a data unit, I’ is the resulting data importance post data reduction, and I_{max} is the maximum possible value for data importance. In our

example, we assume that data importance is continuous between 0 and 1; 1 is reserved for data that cannot be reduced under any circumstance.

To illustrate the application of the compensation factor, let's take an example where a data unit's importance value is evaluated to 0.75 before the compensation factor is applied.

- *for a compensation factor K = 40%*

  $$I = 0.75 + 0.40(1 - 0.75) = 0.85 \quad (4.2)$$

- *for a compensation factor K = -40%*

  $$I = (1 + (-0.40)) * 0.75 = 0.45 \quad (4.3)$$

4.8.4 Available Input

Determining data importance is in the hands of the business model. In this section, we identify input factors that can be used to establish the importance of a data unit.

- *age/collection time*

The data collection time is a relevant factor. Whether in a linear manner, exponential decay, or in some irregular manner, the collection time is relevant. For example, one may be interested to correlate rush hour to atmospheric CO_2 levels over a month's period. In such a case, the time dependency is rather irregular: the data collected

just before, during, and just after rush hour are more important than data collected on a week-end.

- *self values*

The very collected values can affect the importance of a data unit. For example, for any parameter that is collected, any value outside of an expected interval may need further investigation. Hence, additional importance can be assigned to data units containing such values.

- *other data units of same instance*

In cases where a parameter's cyclic aberrant values are suspected, the values of a data unit affect the importance of data in other data units of the same data instance.

- *other data instances*

Similar to the case above, if correlation is to be made between multiple parameters, then special values of interest in a data unit of one parameter add importance to data units taken around the same time for other parameters measured.

4.8.5 Summarizing the components of RDI and NRDI

In order to make the jump from a single node model to a multiple node deployment, we have identified additional functions to better characterize data handling. In this section, we will review the components added to the model described in [87].

- interval production function (for RDI only)

For a recurring data instance, this function is used to produce data units. This function can be rather simple, such as grouping every fixed number of measurements in data units, or it can be more complex resulting in the creation of data units covering variable time spans.

- default compensation factor

The compensation factor is used to reflect already performed data reduction. A default value needs to be specified.

- data importance function

A data importance function which computes the importance of a data unit, taking into account some or all of the factors listed in section A.

- data reduction operation function

A data reduction function which, given a data unit outputs a smaller sized data unit based on its internal use of reduction primitives. It also changes the compensation factor so as to reflect upon future data importance computations.

4.9 A Single Node Use Case

Given the above extensions to the single node model, we now consider an example and get into details as far as the specifications for data collection. The approach to data collection reflects human perception with respect to quantifying and formulating data importance. In order to validate such approaches, a preliminary step is to simulate the setup intended for deployment.

4.9.1 Car traffic and CO_2 concentration use case

The main objective of the deployment in the example is to observe data that can be used to correlate car traffic volume with atmospheric CO_2 concentration. There is a broad pattern that is expected, with more CO_2 production during rush hours where there is more traffic. There are also temporary spikes in the detected levels when a large vehicle passes, such as a large truck. These spikes need to be accounted for.

The deployment consists of many sensors installed in a very large geographical area, but all in proximity of a street. Some of the sensors may be able to reach other for

communication, but communication is very costly given the distance. The collection happens using a mobile base on a vehicle with. The frequency of data collection is not predetermined.

The sensors that are deployed have four sensory devices in addition to an internal clock: a gauge for CO_2 concentration, a sensor across the street to detect traffic levels, as well as to detect the presence of an oversized vehicle, a gauge for wind speed, and one for wind direction. The sensor is also able to take a panoramic photo and store the picture.

4.9.2 Data Collection Requirements

- Every 10 seconds, the CO_2 concentration is recorded continuously (data instance **A**)
- Matching the above, wind speed and direction is recorded continuously (data instance **B** and **C**)
- When a large car passes, the CO_2 concentration is recorded for 5 minutes at 1 second intervals (data instance **D**)
- Matching the above, wind speed and direction is recorded continuously (data instance **E** and **F**)
- If an unexpected spike in CO_2 is detected, a panoramic image is recorded (data instance **G**)
- The number of cars passing is recoded for every 10 second intervals (data instance **H**)

4.9.3 Data Reduction Specification

Each of the above data collections need to have specific interval production functions to generate data units, default compensation factor, data importance functions, and data reduction functions.

- data instances **A**, **B**, and **C**

The division in data units is done according to the time of day. The regions of higher interest (i.e. rush) are divided in smaller data units. That way, any data reduction algorithm applied to a data unit affects a smaller data interval.

Table 4.1. Data unit size for data instances A, B, and C

time	**label**	**data unit size**
9pm – 4am	night	10 minutes
4am – 6am	pre rush	2 minutes
6am – 9am	rush	30 seconds
9am – 11am	post rush	2 minutes
11am – 2pm	day	5 minutes
2pm – 4pm	pre rush	2 minutes
4pm – 7pm	rush	30 seconds
7pm – 9pm	post rush	2 minutes

For the purpose of defining compensation factor, the reduction function, and the data reduction function, we divide the collected data into four sections. The first three sections span 7 days each, and the fourth section covers the remaining time interval. The days being labeled from 0 (most recent), here are additional parameters for the data units:

*Table 4.2. Data handling parameters for instances **A**, **B**, and **C***

span	**K**	**data reduction**	**data importance**
day 0 – day 6	50%	25% trimming	set to 0.95
day 7 – day 13	25%	50% trimming	set to 0.85
day 14 – day 20	0%	75% trimming	set to 0.65
day 21 - end	-25%	75% trimming	set to 0.5

The use of K and the data reduction trimming primitive have been covered above. For the data importance, whenever a data unit moves from one span to another, its importance is set to a fixed value (as noted in the table). Only subsequent data reduction operations will affect this (via the compensation factor).

- data instances **D**, **E**, and **F**

In this case, we select a constant data unit interval for the entire span of the data collection. The data collection is rather shot, so the entire 5 minutes will be treated as a data unit.

For the remaining parameters of a data instance, we divide such instances into the 10 most recent, following 100 most recent, following 1000 most recent, and as a fourth, the remaining instances.

*Table 4.3. Data handling parameters for instances **D**, **E**, and **F***

span	**K**	**data reduction**	**data importance**
10 most recent	20%	50% sparsing	set to 0.9
next 100	20%	50% sparsing	set to 0.75
next 1000	20%	50% sparsing	set to 0.50
after1000	-20%	50% sparsing	set to 0.50

The same approach as above is used in the case of data importance.

- data instance **G**

This data instance is non-recurring. A recording of this data is a data unit. Similar to the case above, we divide the instances in the 5 most recent, the following 10 captures, and the rest of the captures:

*Table 4.4. Data handling parameters for instance **G***

span	**K**	**data reduction**	**data importance**
5 most recent	50%	20% resolution reduction	1
next 10	50%	20% resolution reduction	0.8
after1000	50%	20% resolution reduction	0.7

- data instance **H**

In this instance, we are dealing again with continuous data collection. In this case, counting the vehicles, we divide the collection in even data units. Just like in the first

example, the data units are placed in several time spans: three spans are 30 days long, and the fourth span contains the remaining days. The data units are 5 minutes long.

*Table 4.5. Data handling parameters for instance **H***

span	**K**	**data reduction**	**data importance**
day 0 – day 29	25%	50% grain coarsing	set to 0.80
day 30 – day 59	15%	50% grain coarsing	set to 0.70
day 60 – day 89	-15%	50% grain coarsing	set to 0.50
day 90 - end	-50%	50% grain coarsing	set to 0.40

4.9.4 Use Case Conclusions

The use case described is quite simple, yet it does the job of properly collecting the data. The model used needs the validation of intense simulation in order to validate the nature of the data it produces in various circumstances.

4.10 Data Dependency

In the example so far, there was no dependency between the different data collections with regards to data values from a concurrent data collection. In this section, we expand on data instance A from the previous section in relation to data instance B. As a reminder, data collection A is the CO_2 concentration sensed, while B is wind speed.

As a business case decision, we decide that certain atmospheric disturbances in the area of the sensor affect the relevance of the CO_2 concentration measurements. We are going to look at two aspects to determine dependency: (i) increase in wind speed and (ii) average wind speed within a data unit. We decide that wind acceleration beyond a fixed value *a* affects the importance of CO_2 measurements, while an average wind speed beyond a certain level *s* affects the importance of CO_2 measurements for four data units.

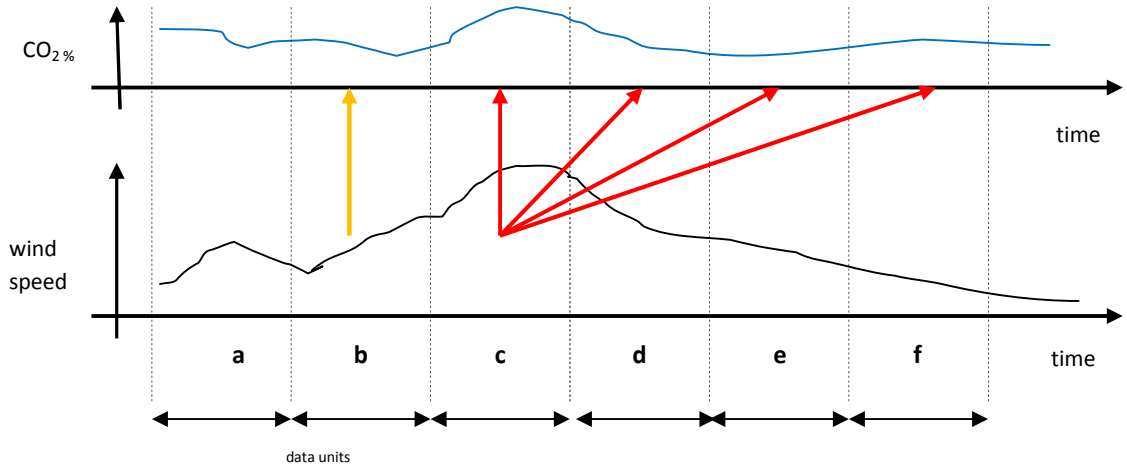


Figure 4.16. Data dependency across data units

In Figure 4.16, several data units, labeled from a to f, are shown with corresponding reading of CO_2 concentration and wind speed. Data units b and c on the wind speed graph are of interest as they affect the importance of CO_2 readings. In data unit b, the wind acceleration causes an effect on the corresponding concentration reading. In

data unit c, the average wind speed affects the data importance of four data units for concentration reading.

On a more specific note, here is a proposal for to compute the data importance mapped to a data unit of CO_2 concentration reading. We denote by *CO du* t_i - t_j the CO_2 concentration data unit between time ti and time tj, while *WS du* t_i - t_j refers to the wind speed data collection.

var tempI

if age(du t_i - t_{i+1}*) < 7 days*

tempI ::= 0.95

else if age(du t_i - t_{i+1}*) < 13 days*

tempI ::= 0.85

else if age (du t_i - t_{i+1}*) < 20 days*

tempI ::= 0.65

else

tempI ::= 0.50

if acceleration(WS du t_i - t_{i+1}*) > a*

*tempI ::= tempI * 50%*

if average(WS du t_i - t_{i+1}*) > s*

*tempI ::= tempI * 10%*

if average(WS du t_{i-1} *-* t_i*) > s*

*tempI ::= tempI * 20%*

if average(WS du t_{i-2} *-* t_{i-1}*) > s*

*tempI ::= tempI * 50%*

if average(WS du t_{i-3} *-* t_{i-2}*) > s*

*tempI ::= tempI * 80%*

I(COdu t_i *-* t_{i+1}*) :: = tempI*

Figure 4.17. Computing data importance with dependency

At this point, we can generalize the constraints on the data importance computation as pertaining to two categories: (i) internal constraints and (ii) external constraints.

External constraints are caused by factors over which input data has no effect. Such factors are data age and inherent interest in the data depending on the exact purpose of the data collection.

Internal constraints represent inter- and intra-data dependencies. In the case presented above, increased wind renders CO_2 concentration less relevant and less usable; there can be events which affect the importance of data at any point along the timeline.

4.11 General Architecture

In this section, we present the general building blocks of the data collection process, as well as the prioritized data reduction process. While these processes act on the same data, they run in parallel as uncoupled processes.

Figure 4.18 presents the control steps for the data collection. Figure 4.19 presents the steps involved in the data reduction process. The black arrows denote sequence of steps. The red arrows denote control.

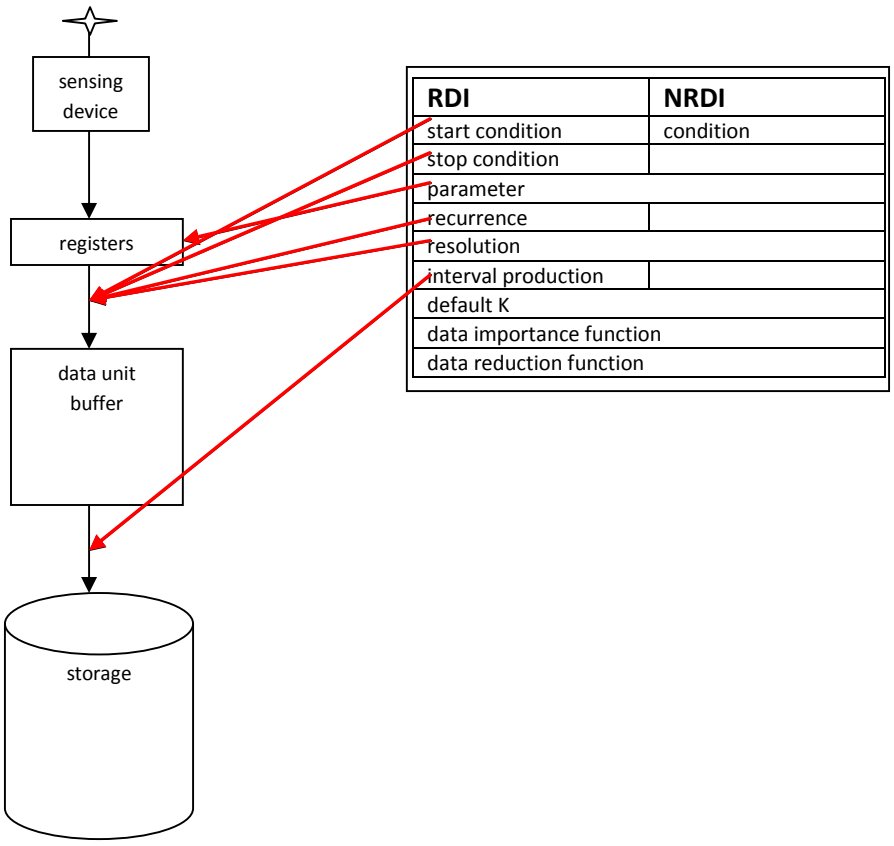


Figure 4.18. Storage management as per instance specifications

In Figure 4.18, we have the physical sensors which make actual measurements. They have associated registers where the data readings are placed. This happens whether data is being collected or not. At this point, depending on the running condition of the recurring data instances, the recurrence interval, and the resolution, the data is captured into a buffer. Enough data is accumulated to construct a full data unit. When complete, the data unit has its importance calculated and is placed into the storage. In case we are following an optimistic approach to data reduction, as described in [87], then we compute the data importance of the data unit in the buffer in order to compare with stored data. In case the data unit in the buffer is the one with lowest importance, then it will be the one to be reduced.

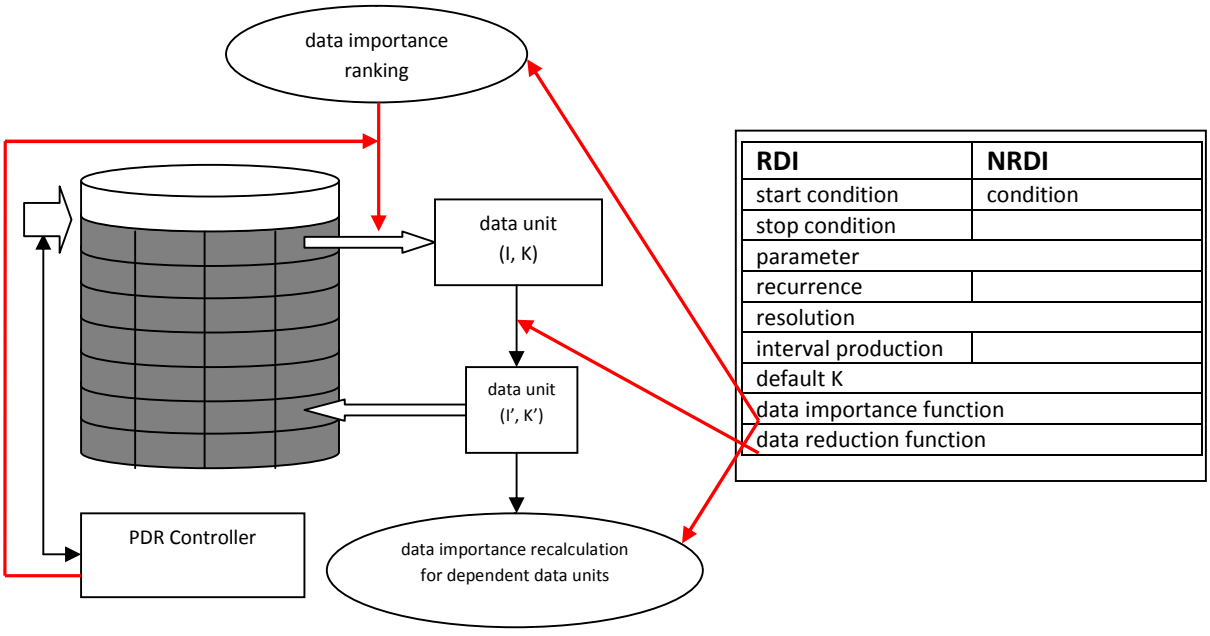


Figure 4.19. Data reduction based on data instance specifications

Figure 4.19 only shows a single sensing unit and a single data instance. In deployments, there can be several sensing units for different parameters, and several data instances that are collected. There can be different instances collecting from the same sensor, but with different specifications, such as resolution for example.

In Figure 4.19, the flow of data reduction processing is presented. The PDR Controller continuously monitors the amount of data in the storage. If the levels cross a given threshold, then data importance ranking is consulted to identify the least important data. This data unit maps to a specific data instance which carries a data reduction function as well as an importance computation function. After a reduction, the data unit is written back to memory.

4.12 Conclusion

In this chapter, primitives were proposed in order to deal with increasing amounts of data stored by a sensor node. The scope is to have the basic mechanisms to gracefully discard lower importance data, or lose some flexibility and data resolution, for the benefit of higher importance data. The results presented in the example are encouraging. With a more complex business case, the algorithms can get more complex, but the overall objective remains the same.

The internal operation of a sensor node consumes little energy compared to the requirements for transmission. Hence, the energy issue has been disregarded with respect to operations taking place without transmission.

We also presented a methodology for prioritizing data processing in WSNs. In the first part, primitives were introduced to support a data reduction approach. However, the general framework of the application of the primitives was not specified. This article identifies the modules and functions needed for a sustained data collection in situations of limited storage availability.

Related work was presented in relation to single node situations as well as collaborative approaches. In general, there are no specific solutions other than stating that one needs a systematic approach to data aging. Current strategies also don't address the problem from the perspective of sink node unavailability, but rather from the perspective of an intelligent WSN which gives the possibility of executing queries through the entire network.

In our case, the constant availability of the sink node is not a given, which reflects the reality in which sensor networks often operate. We have proposed that the data be divided into manageable data units. These data units are evaluated and ranked in terms of importance. When it is time to reduce the data load, the lowest importance data units are subjected to a specified data reduction function.

An example was shown with the use case of a very large area CO_2 monitoring WSN serviced by a travelling data sink node. Several data were collected: CO_2 concentration, car traffic, wind speed and direction. A complete solution was proposed using only external constraints on the importance of the data. One of the data collections was further considered with respect to internal data constraints.

Finally, a system architecture was shown. The proposed architecture shows the steps and control factors during a data collection, as well as the conditions and stages of prioritized data reduction. We conclude that these proposed methods offer an overall systematic approach to data reduction as a function of business case.

As future work, the presented approach can be extended via cooperation amongst nodes. One node has opportunity to negotiate storage on other sensor nodes. Data importance becomes a factor of the entire network. While the entire sensor network works together towards the data gathering task, there is competition amongst individual nodes for access to the storage available in the sensor network.

An additional item that adds interest for future work is the option to have heterogeneous sensor nodes, and hence the ability to design WSN data center nodes. As a consequence of a coordinated data storage approach, there is added communication, and hence, additional energy expense. Such an expense must be justified by benefits on the data storage optimization.

When going from the single node focus to the multiple node scenario, the positioning of the nodes is a factor. The distance matters for transmission power. If the

nodes are placed at predetermined positions, the preferred communication links and energy requirements can be pre computed. If the placement of nodes is non-deterministic, self-organization algorithms are needed to guide communication paths for access to other nodes' storage.

In conclusion, extending the idea from a single node to multiple cooperating nodes adds complexity to the computation of importance factor, energy expense to the process of data management, and to the organization of the network.

Several items remain open. A simulation model is needed to validate the choices that we make in the importance calculation function and the data reduction function. The parameters need to be adjusted as a function of expected input data and expected behavior.

The next step is the expansion of the model to several node systems. While some nodes may experience spikes in data production, other nodes may simply just collect low relevance monitoring data. It is beneficial to consider this case for collaborative storage. The context in which importance evaluations and reduction functions operate so far is specific to the node that has produced the data. When we consider sharing other nodes' storage space, data that is being housed on a different node needs to be sent with proper instructions for handling. Internal dependencies can no longer be enforced; they need to be rephrased or simply dropped.

As a final step for having a robust solution, redundancy needs to be addressed. We have assumed smooth operation of the nodes with no equipment loss or malfunctions. This

can very well be the case in a remote area. The degree to which redundancy is required and how importance of redundant copies is computed is left as future work.

The decisions taken by the node while handling data make no assumption regarding scheduled, probabilistic, or statistical availability of the sink node. This additional information can affect the manner in which data reduction is applied, as well as the storage occupancy level at which data reduction processes are triggered.

Overall, the proposed solution and framework for a single node case are now robust enough to provide flexibility for future extensions.

Chapitre 5

Réduction des Données: Multiples Nœuds

Sommaire

Dans le chapitre précédant, nous avons introduit la réduction de données en considérant un seul nœud. Dans un réseau à multiple-nœuds, la possibilité de déplacer les données est une alternative à la réduction (la relocalisation peut également être utilisée en parallèle avec la réduction).

La relocation des données implique inévitablement une dépense non-négligeable d'énergie. Les trois facteurs qui influencent cette consommation supplémentaire sont: le nœud source, le nœud destination et les données (en incluant aussi leurs dépendances avec d'autres données). Le nœud source doit conserver une énergie suffisante pour transmettre les données résidantes lorsque la station de base (sink node) devient disponible. Le nœud qui reçoit doit avoir assez d'espace disponible pour des données avec un index d'importance très bas (pour ne pas prendre la place de données transférées, car l'espace alloué est en compétition). Les données elles-mêmes doivent être auto-suffisantes, dans le sens où elles doivent toujours être associées avec tous les autres paramètres nécessaires pour le calcul de l'importance sur le nouveau nœud.

Un autre opération possible dans les réseaux à multiple nœuds consiste à avoir des copies redondantes afin de protéger la collection de données contre les défaillances d'un

nœud. Cependant, ces copies ont une particularité: leurs fonction d’importance ne doit pas dépendre d’autres données (c'est-à-dire rester sur le nœud d'origine).

Chapter 5

Prioritized data reduction: multiple nodes

Summary

The previous chapter approached the data reduction from the perspective of a single node. There are benefits from looking at a WSN deployment as a whole. There is an opportunity to relocate some data to less active nodes and spare it from a reduction process.

There are energy considerations to take into account in evaluating the potential benefit from spending some energy to relocate data. Three factors play a part in this: the source node, the receiver node, and the data itself. The source node needs to save enough energy to relay its data once the sink node becomes available. The receiver node needs to have space available, or at least have a lot of space occupied by low importance data. The data itself needs to be self-enclosed as far as parameters needed for importance computation.

Another aspect that becomes feasible in a multi node environment is having redundant copies of data to protect against potential node failures. These copies have their own particularities, a notable one being that their importance function cannot depend on other data.

5.1 Introduction

Several approaches and protocols have been proposed with regards to collaborative storage. By having a wider view of the deployment, such protocols attempt to address aspects such as alleviating situations, where certain nodes produce more data than others, deduplication of data that was collected unknowingly by more than one sensor, and ultimately, as a side effect of deduplication, less energy is needed to relay the data to a base station.

In [88], Tilak *et al.* propose a Cluster Based Collaborative Storage (CBCS) as a specific solution to collaborative storage. Several algorithm improvements in WSNs have this common approach of clustering nodes, electing cluster heads, and establishing an overall tree structure through the network. CBCS does this in the context of storage management. Nodes are grouped in clusters, based on geographical data. Cluster heads (CH) are elected, one per cluster; they have the task of aggregating all data from the cluster nodes. This improves power use efficiency as only the CH needs to further relay the data towards the sink node. Such an approach seems to place more emphasis on the energy saving rather than dealing with large amounts of data. Aggregating all the data from the cluster nodes on the CH storage space cannot be beneficial when we are trying to solve storage issues.

In Figure 5.1, a visual representation of clusters is shown. Most clusters are created with geographical proximity in mind in order to minimize energy expenditures on

communication. The designated or elected cluster heads, shown circled, handle the communication with a sink node, S, which is linked to a stable reliable network.

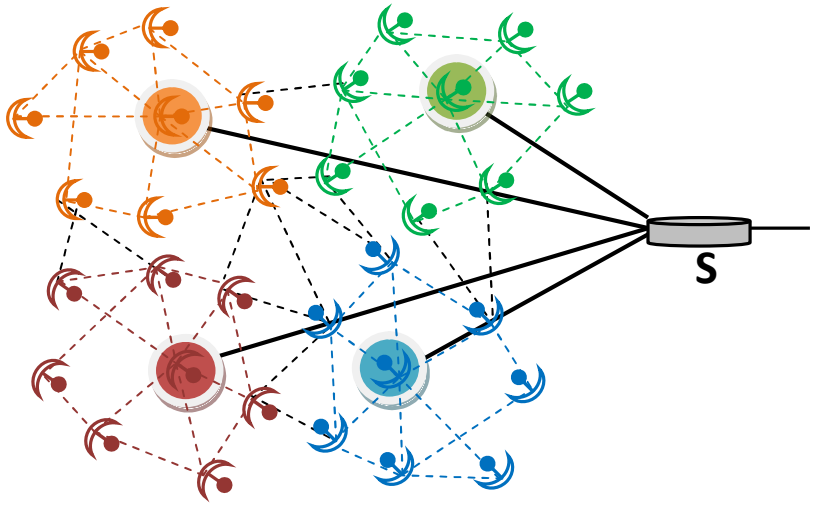


Figure 5.1. Clusters in a WSN

In [89], Shenker *et al.* propose a method, Data Centric Storage (DCS), to store data as identified by its name. Related data would in the end be stored either on the same node, or on neighboring sensor nodes. This would facilitate deduplication and also improve queries since data pertaining to a query would reside in the same proximity, therefore avoiding the need for queries to be run through large sections of the network. This approach is useful for WSNs that are designed to support searches, but does not apply to networks where data analysis is done offsite.

Siegmund *et al.* [90] address the issue of data integrity. Data redundancy is achieved in the network via the implementation of a new abstraction layer in the WSN. This layer can support the need for data redundancy. While robustness is of importance, there is very limited work in view of alleviating potential data overload in the network.

Collaborative storage presents challenges in terms of locating data during queries through the network. It also adds to the power requirements of the nodes. In the proposals published so far, there is no consideration for collaborative storage to mitigate memory limitations on some sensor nodes. There is also no effort to give a semantic to the stored data in order to assist with data age-out in the context of a node who is only storing the data without having produced it.

Having multiple nodes interact in an effort to better manage storage space gives the opportunity for two considerations: storage space sharing and redundancy. In this chapter, we address these concepts from the perspective of prioritized data reduction.

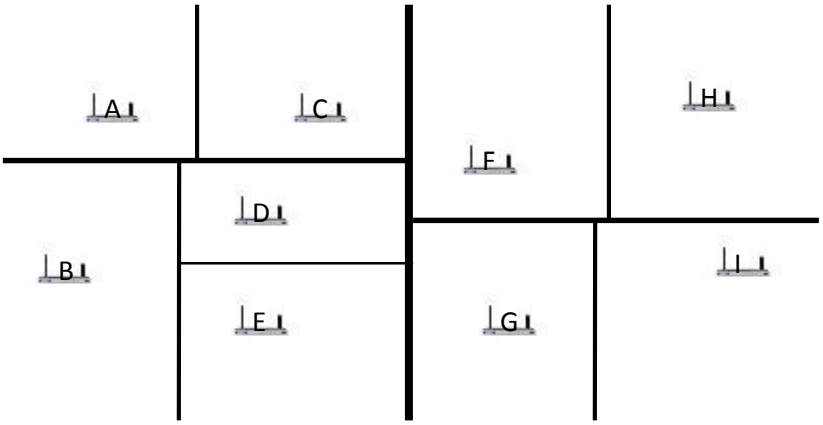
5.2 Storage space sharing

5.2.1 Load sharing

Most research related to sharing across several nodes the impact created by data refer to balancing out the data access hot-spots. Occasionally, data queried in a network will be accessed in a skewed manner resulting in a small set of nodes receiving a disproportionate amount of queries and hence taxing the energy storage of those specific

nodes. In order to deal with these cases, methods to replicate the data are presented as well as methods to divide the data across several nodes. In [91], the problem is approached from the perspective of k-d tree mapped data. In such a mapping, data storage is done independently of the node that produced the data. When one node becomes overwhelmed with respect to the amount of data that it needs to store, a tree rearrangement is done in order to balance the data load.

A k-d tree covers a two dimensional surface by recursively dividing the surface in two parts. At the point where we have a single node inside a parcel, that parcel no longer undergoes division and the node becomes the leaf of the path that created the parcel.



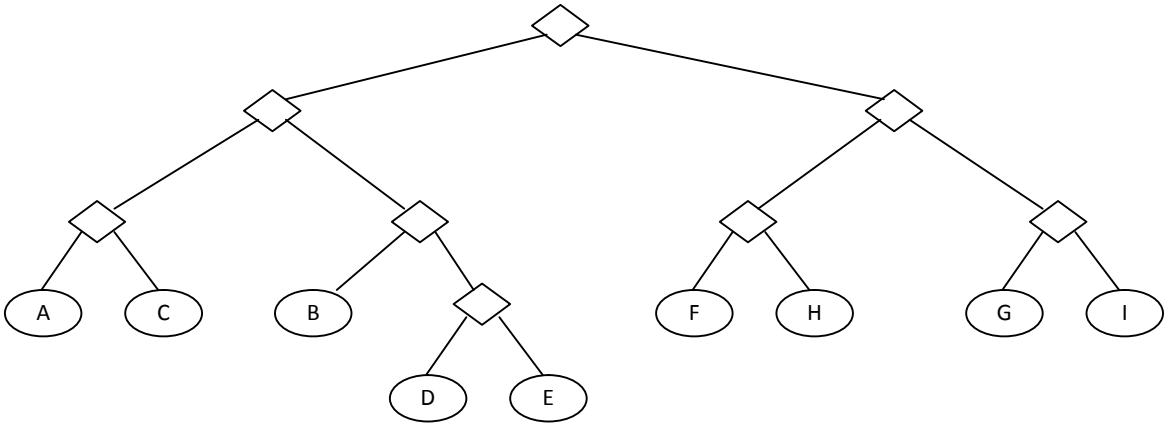


Figure 5.2. Division of a WSN in a k-d tree

Another facet of load sharing is found in [85], where the authors address the problem of congestion at nodes on highly used paths. Not only are these nodes' energy supplies overtaxed, but the limited storage space that they have may not be enough to transiently hold all the packets. The proposal is to group the sensor nodes into small clusters, ideally a highly redundant deployment. When high traffic volume is sensed, the cluster head node starts to divert traffic to neighboring nodes. Because the nodes are in the vicinity, the congestion is localized to the specific area without spreading to other nodes. For critical data, the head node can make the decision to replicate the data several times so as to assure its continuity even in the event of a node failure. One drawback is a potential added delay caused by the added store and forward operations within a cluster.

5.2.2 Storage sharing

In this section, we cover the aspects of sharing the storage space that exists within an entire wireless sensor network. This is already done as part of geolocated data having as aim to facilitate queries. In our case, we assume a network that does not support in-network queries, but relays all data to the sink for offsite processing. Data produced by a node stays on that node until delivered to the sink. However, in the interest of balancing out space needs, we consider the possibility to forward certain data to other nodes as opposed to applying prioritized data reductions.

Until storage sharing, it was rather straight forward to perform a round of data reduction, i.e., select the data unit with the lowest importance and apply the reduction function. This could be done iteratively to bring the memory usage below a certain expected threshold. With an option to package and send data as a method of storage size recovery, several decision factors appear. How much data to send? How to select the most appropriate data to send? How to select the best node to send the data to? There are no definite answers in situations where external inputs dictate how much or how little data needs to be stored. There are however preferred ways to handle storage space sharing.

5.2.3 Energy considerations

When encountering limited data storage space, a node has two options, (i) to reduce the size of the data by applying some primitives, or (ii) to relocate some data. In this section, we consider the factors that affect the decision:

current energy levels

Negotiations to find an appropriate host for data require energy. As the packets involved in negotiations don't contain data, they are fairly short so they don't consume much energy. The energy cost needed for the transfer of the data depends directly on the size of the data transferred. The node needs to save enough energy to eventually transfer the entire contents of the data to the sink node once the sink node becomes available.

expected opportunity to connect to sink

Energy is wasted if a portion of data is relocated only to be transferred to the sink shortly after. In the case of unreliable sinks or sinkless deployments, expected access to sink is hard to compute. It can be done with statistical data, but reliability can vary. If the sink operates on a predefined schedule, it is feasible for this to be taken into consideration. A similar case to a predefined uptime schedule is a mobile sink. With some uncertainty, the sink will travel along a route with high probability of being in specific areas at certain times.

5.2.4 Data considerations

There are characteristics that make data appropriate for relocation. The size is one factor, as we want to have a sizable impact as a result of investing energy in the negotiation part. Other factors relate to the dependency between data units.

The importance calculation function should depend on self values or on time only. In that way, we can preserve the importance calculation function after the data has been moved. In some cases, the importance of a data unit depends on other data units of the same collection. It is conceivable that all the data units with importance interdependence relations be subject to a data transfer as a whole.

The reverse is also true, i.e., the data units to be transferred are used in computing the importance level of other data units. Bundling everything in the data to be relocated is an acceptable solution.

A special case is the situation where yet uncollected data can affect the importance of data units already collected and tagged for relocation. Such data is best not moved as this would imply changed to the importance calculation function.

The parameter K was introduced in Chapter 4 as a compensation factor to affect the importance value for data units that have undergone data reductions. A positive K will increase the importance of post reduction data, while a negative K will decrease it. The

data that is considered for a move needs to have a positive K so as to make it through potential data reductions it will undergo on the receiver node.

5.2.5 Receiver node considerations

The conditions of the receiver node affect how the received data will be handled on that node. The potential recipient of the data needs to have a good situation of its storage space as well as a good outlook for future situations. The recipient also needs to have a good amount of energy left so as not to compromise the long term survival of the data being sent.

The number of active collections and the amount of data they produce is an important factor. If there are many collections going on, the space on the receiving node is quickly filled up. Even if these collections produce low importance data units, a large number of ongoing collections generate lots of data reductions and data relocations. We want to avoid causing such events.

Inactive collections are collections whose start condition is not met. Some of these are predictable as to when collection actually starts, such as collections that only depend on time to be activated. Others depend on data values detected by sensors. Having a large number of inactive collections makes a node less desirable to act as a data recipient. These collections can start unexpectedly and produce data that competes for space with the relocated data.

Finally, the general importance profile of the stored data units are a factor in deciding if a node is a good recipient of relocated data. If most data on the receiver node is low importance, then the node is a good recipient of data. However, the nature of the importance functions on that node is to be taken into consideration. An event could trigger a recomputation that changes the data importance to higher values.

5.2.6 Sending vs. Reducing: the decision

The decision to favor data reduction vs. data relocation is subject to network's owner decisions. In this section, we provide a parameterized approach to handle this task. We specify three functions as follows:

- **SF(i)** specifies the fitness of a node *i* to act as a data source
- **RF(i)** denotes the fitness of a node *i* to act as a data receiver
- **DF({du})** denotes the fitness of a set of data units for a data transfer operation

We also specify a threshold value t such that:

*if (a*SF(n) + b*RF(m))*DF({du}) > t, then node n should initiate a data transfer of the {du} set of data units to node m, where a and b are parameters to give more or less weigh to each of the functions*

SF(i) is defined as a function of the source node's ability and opportunistic interest to spend energy on data movement related operations: negotiations and actual data movement. We define as LD(i) the percent of data stored on node *i* having an importance that is low and decreasing, that is, an importance smaller than a fixed value, decreasing in time, and a negative K. The K does not have to be negative for a fixed number of initial reductions, but it needs to be negative for remainders of the reductions or the reductions are quickly reaching a point where data is entirely dropped. Let *e* be the minimum residual energy required for a node n to send its data contents and *E* the current amount of energy in storage. The value of SF(i) directly related to available energy and to the lack of low and decreasing importance data units. We define SF(i) as follows:

$$SF(i) = k*(1-LD(i)) + j*(E/e) \quad (5.1)$$

where k and j are parameters to adjust the weights of the two factors.

RF(i) is defined as a function to quantify the potential receiver's node fitness to act as a receiver. With respect to energy, RF(i) behaves just like SF(i). With respect to the stored data profile, it behaves in an opposite manner. In addition to these two aspects, RF(i) needs to account for the effect of active collections as well as inactive data collections.

$$RF(i) = k*S*LD(i) + j*(E/e) - k*(Da + p*Di) \quad (5.2)$$

where Da is the storage debit of active collections and the Di is the potential storage debit of inactive collections. The constant p is used to decrease the impact of inactive collections.

DF(i) reflects on the fitness of data to be subjected to transfer. As a limiting factor, we have established that the bundle of data units must be self enclosed as far as importance calculation dependency. We propose the following formula for evaluating a data unit's fitness for transfer:

$$DF(i) = I*(trend(K) - k*|M - s|) \quad (5.3)$$

where M is the ideal data transfer size, s is the size of the current data slated for transfer, k is an adjustment parameter, and trend(K) is a function that evaluates the value trend of K in time and as potentially affected by reductions within the transferred data. The entire result relates to the overall importance I of the data bundle considered for transfer.

With these formulas, we have captured the essence of the decision making process of data transfer vs. data reduction. The approach is heavily parameterized as weighing different factors varies from one business case to another. In light of these proposals, Figure 4.19 undergoes changes:

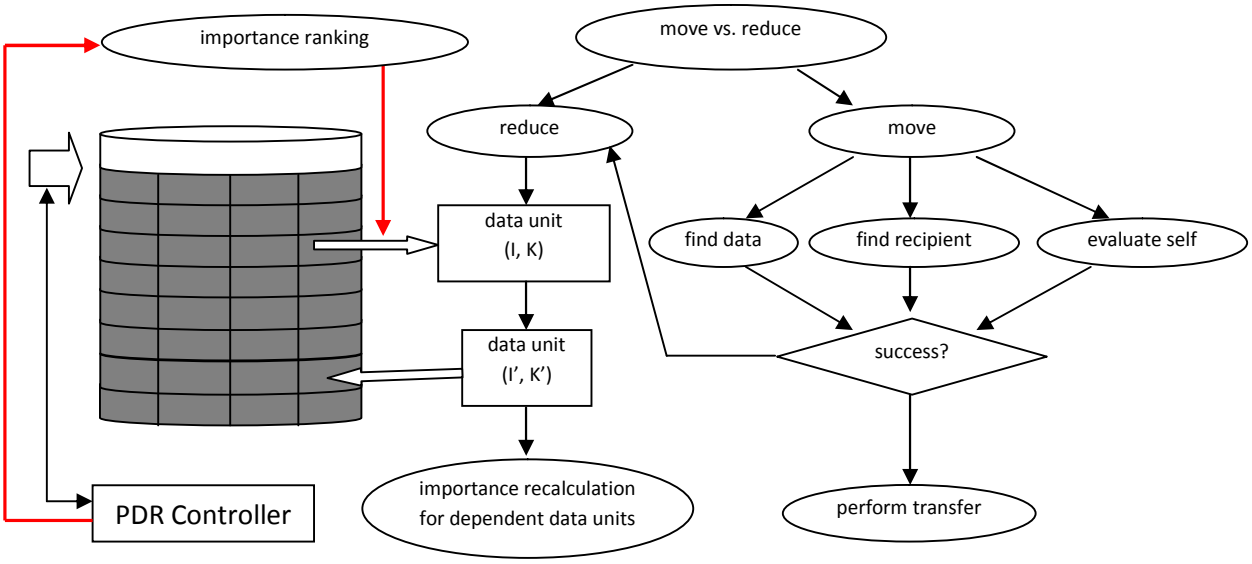


Figure 5.3. Data reduction via move or prioritized reduction

We can assume that process to evaluate the source not for fitness to participate in a data send is straight forward: the value of the SF(i) is computed. Finding a recipient and finding appropriate data to send are more complex as they involve selecting from a pool of possibilities.

Finding data to transfer is a complex process. There can be a large number of data units, and therefore, considering all possibilities can be a lengthy process. We propose a heuristic on how to approach data targeting for relocation with the understanding that an exhaustive process may offer a better solution but at a higher cost. Data units that are targeted for transfer are incrementally added to the set of data for transfer as we seek to fulfill the requirements for self-enclosure for importance function evaluation. We stop as

we reach a data size in the vicinity of M. In certain cases, we reach a dead end and can no longer have hope to find appropriate data to move.

The set of data units to be moved is incrementally increased in steps as follows:

- high importance and positive K data units on which no other data units' importance depends
    - data units' importance dependence limited to self values
    - data units' importance dependence limited to self values and time
- high importance data units with positive K whose importance depends only on other data units already collected (i.e., future data units collected have no impact on the importance of this data)
- recursively add the data units on which the above units depend for importance calculation
- as we reach a value close to M, we stop adding data units to the set of data targeted for a data transfer

We realize that recursively adding data units may be getting out of hand. If the minimum amount of self dependent data units amount to a very large amount of data, we can safely give up the idea of a transfer and opt for a data reduction.

The simplest approach to finding a recipient for data is to query the status of all reachable nodes, evaluate their RF(i) function and select the best option. Depending on the deployment, querying the immediate neighbors only may be artificially limiting the amount of potential recipient candidates for data relocation. Attempting to go the extra distance via additional hops requires the cooperation of other nodes along the path. These nodes need to commit energy reserves towards the data relocation process. Given that we

have already selected the data that will be moved, there is a clear expectation as to what amount of energy would be required from the transient nodes during the data move.

Finding an appropriate receiver for the data follows the following steps:

- decide how many hops away we want the data to potentially go
- broadcast a message to the effect of finding a host, including the number of hops; if more than immediate neighbors are considered, include the data size to be able to obtain commitment along the transfer route
- reachable nodes reply with their current RF(i) computation
- select the best RF(i), compute (a*SF(n) + b*RF(m))*DF({du}) (see section 5.2.6) and if result is satisfactory, proceed with the transfer
- if the computation yields an unsatisfactory result, then data reduction is undertaken instead of transfer

5.2.7 Validation and calibration of heuristics

In the above section, we have proposed a series of heuristics to quantize the fitness of nodes to act as receiver or sender, as well as the fitness of data to be transferred. The proposed formulas are parameterized so they can be tweaked to several circumstances. In this section, we are going to evaluate the trends that the formulas generate under different circumstances depending on the statistical distribution of importance and compensation factor of stored data units.

We recall that the formula to decide the fitness of a node to act as a sender is:

$$SF(i) = k*(1-LD(i)) + j*(E/e) \quad (5.1)$$

By its definition, the fitness function depends on the raw amount (not ratio) of high importance data and level of energy required to relay the entire data load towards the base station. We consider the scenario of different statistical distributions of data importance on a node, and in each case, we vary the amount of energy available. We consider the cutoff for LD(i) to be at the importance level of 0.5. Hence, in the graphs below, the blue bars represent (1-LD(i)).

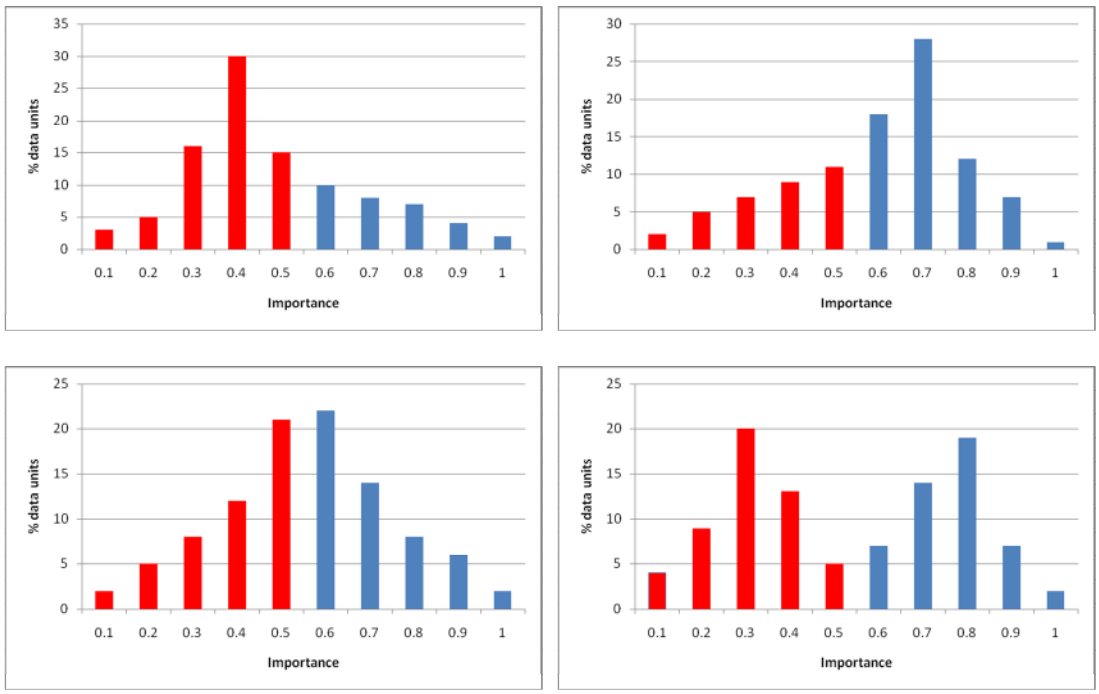


Figure 5.4. Data distribution scenarios for validation of SF(i)

The image above shows the scenarios considered with respect to the distribution of importance values. We have considered skewed, bell curve, and bimodal.

The initial proposal was that the amount of energy available would have a linear impact on the value of SF(i). This works well at points where the available energy levels E are within a few orders of magnitude of *e*, the energy needed for a full data transfer. At higher orders of magnitude (i.e orders of thousands, and above), the linear growth of j*(E/e) poses a problem as it identified a node as a "good node to send data" mostly based on its energy levels. If the nodes deployed fall in such a class, then the problem can be alleviated by using a logarithm to attenuate de higher orders of magnitude:

$$SF(i) = k*(1-LD(i)) + j'*\log(E/e) \quad (5.4)$$

Experiments with different LD(i) values only changes the balance between defining good source nodes via SF(i) and balanced by the ability to find receiver nodes via RF(i) defined in equation 5.2.

Along the same lines, simulations were run to further analyze the proposed formula for RF(i). We notice an important difference between SF(i) and RF(i): the formula for SF(i) depends on a percentage of storage while the RF(i) formula depends on the value of raw storage. The idea is that we don't want to put at a disadvantage nodes that, as per a deployment decision, have less storage space than others.

Analysis of RF(i) brings up a matter similar to SF(i) in that excessive energy levels can drive up the value of the function and artificially flag it as a good receiver node. In reality, this excess of energy may hide data space availability issues on the receiver node. it was found that in reality, aside from using a logarithm to attenuate the energy effect, a

product rather than a sum gives a better assessment for a node's ability to receive data. Therefore, RF(i) as presented in equation 5.2 can be refined as:

$$RF(i) = j*\log(E/e)\ (k*S*LD(i) - l*(Da + p*Di)) \quad (5.5)$$

The evaluation of data fitness for transfer was defined in equation 5.3 as

$$DF(i) = I*(\text{trend}(K) - k*|M - s|) \quad (5.3)$$

Statistical evaluation on this formula with a linear relation between the trend of the compensation factor (trend(K)) and the deviation of the data size from an ideal size (|M-m|) is a good approximation. Since there are already strict conditions on assembling a set of data units for relocation, it would be counterproductive to discount the selected data set in case there are variations from the ideal size. Hence, we opt for a linear dependence on size deviations.

Additional simulations were conducted on the formula assembling the three heuristic items:

$$(a*SF(n) + b*RF(m))*DF(\{du\})$$

We need both a pair of nodes to participate in the data transfer and a data set. If either of these items scores low with our evaluation, we want to prevent the data relocation. Hence, a product is appropriate.

For the purposes of the simulations in this section, we have compared trends in values. Specific parameter values can be assigned for very specific deployment scenarios, which must be validated on a case by case basis via simulation.

5.2.8 Relaxing data dependency for transfer

In the approach described above, we have restricted the data movement to data units whose importance function can be evaluated even after the move. This restricts the set of data units that can be targeted for relocation. While not going into the details of a full solution, we present in this section two approaches that can be used to mitigate the relocation of data units

A first approach is to simply replace the importance function for the relocated units with a version that does not depend on other data units. This basically changes the business case approach and may not be feasible while retaining a reasonable amount of intended resolution.

A more refined approach is to use expected values to replace the parameters inside of the importance function. It is less than desirable and requires some statistical approach in deciding what the expected values are. In the end, this approach may not yield a good compromise either because we may be interested in situations where unexpected values are encountered.

5.3 Redundancy

Up to this point, we have not addressed the issue of data redundancy. Having multiple copies of data in a wireless sensor network is important to balance the high incidence of node failure. In the framework presented in this thesis, having multiple copies of the same data runs counter to idea that we seek to make maximum use of a limited storage space.

5.3.1 Implementing redundancy

The implementation of redundancy is done at the specification of the data instance as shown in section 4.4.2. The additional parameters that we include relate to the number of that we want to store for that specific data instance. Hence, the data instance definition becomes:

Definition of an RDI with redundancy

RDI: (T_{start}, T_{end}, param, recurrence, resolution, compression, copies), where:
T_{start}: start time
T_{end}: stop time
param: the parameter being collected
recurrence: how often the collection is done
resolution: how precise the stored value is
compression: boolean stating if the RDI has been compressed
copies: integer stating on how many nodes the data needs to be stored
e.g., RDI(2009/12/06 16:43:23, ongoing, temperature, 60 seconds, 0.01C, FALSE, 4)

Figure 5.5. RDI definition with redundancy

Defining an NRDI with redundancy

NRDI: (T, param, resolution, compression, copies), where:
T: recording time
param: the parameter being collected
resolution: how precise the stored value is
compression: boolean stating if the NRDI has been compressed
copies: integer stating on how many nodes the data needs to be stored
e.g., NRDI(2009/12/06 16:43:23, temperature, 0.01C, FALSE, 3)

Figure 5.6. NRDI with redudancy

As data is collected one data unit at a time, these data units are stored locally (master copy) and copies are also stored on as many nodes as specified. These redundant nodes are not tied to one specific recurring data instance for example. The node collecting the data can select any other node to store the copies on. The process can be batched as node to node negotiations can be very costly for a small data segment at a time.

5.3.2 Importance calculation

We have seen in Chapter 4 how each data instance has a function that can be used to compute the relative importance of each data unit produced. When we deal with several copies of the same data unit, we need to specify a different importance value for each of the copies. Redundant copies have less importance than master copies. For this reason, in addition to the factors listed in section 4.8.4, we propose an importance decay function

which is applied to the importance calculation of redundant copies. For example, if we have a linear decay, the adjusted value of a data unit importance would be:

$$D(I(du)) = (1/r) * I(du), \text{ where } r \text{ is the redundancy level} \quad (5.6)$$

As nodes collect data and go through data reductions, some of the redundant copies of specific data units may be dropped. This is normal as nodes prioritize data units as a function of importance. However, depending on the environments in which the redundant copies end up, higher level redundancy copies may disappear before lower level ones. In that case, we need to attach a message trigger to update the redundancy level of the copied of the dropped data unit. A limited flood mechanism can be used as it is not expected for data units to migrate very long distance from the point of origin.

Since redundant copies are targeted for transfer, the function used in calculating the redundant copies importance level must be computable on the receiving node. That restricts the computation function in a way similar to the restrictions imposed by load sharing. In consequence, if the data importance calculation function depends on other stored data, then redundant copies need to carry a completely new function, as opposed to a function simply affected by a decay function.

5.3.3 Load sharing under redundancy

The objective of redundancy is to protect the data from eventual sensor node failures. We therefore have several copies stored in the network. As a result of using load

sharing to mitigate storage issues, we may at some point attempt to store more than one copy of a data unit on the same node. This should be prevented as it defeats the purpose of redundancy.

5.4 Conclusion

In this chapter, we have presented our proposal for prioritized data reduction in light of a multi-node deployment. In such a case, it becomes feasible that a node facing storage space shortages look first at offloading some data to neighboring nodes. We have proposed a mechanism by which a node can quickly assess the situation and decide between the option of local data reduction or a date transfer.

The opportunity for redundancy was also evaluated. As wireless sensor networks are often deployed in unreliable and dangerous areas, node failure is high. Data redundancy reduces the possibility of critical data loss caused by node failures. We presented a mechanism to evaluate the importance level of data units that are redundant copies rather than master copies of the collected data.

In the next chapter, we present a use case where the concepts introduced so far are used.

Chapitre 6

Déploiement avec Redondance dans une Région à Haut Risque

Sommaire

Ce chapitre présente un cas d'étude qui utilise les concepts introduits dans les chapitres précédant. Il considère un réseau à six nœuds chargé de la surveillance d'un volcan. Tous les six nœuds sont équipés de capteurs spécialisés pour la concentration de CO2, la température et la concentration des particules.

La collection des données suit un plan préétabli initié par des sommets de température (mesures). Deux nœuds dans la proximité du volcan présentent un très grand risque de perdre les données ; donc par précaution deux copies de données collectées sont gardées dans d'autres nœuds.

La simulation est menée par deux ensembles de paramètres : un ensemble favorisant la réduction des données et l'autre recommandant la relocalisation des données.

En analysant les résultats, nous avons constaté que l'approche par redondance demande des protocoles spécifiques. Les deux nœuds à haut risque ont été crées afin d'avoir leurs données en copie sur d'autres nœuds.

Chapter 6

Use case: Redundant Deployment in High Risk Area

Summary

In this chapter, we present a use case that covers all the presented concepts. The setup is a WSN made up of six nodes in charge of monitoring a volcano environment. All nodes are equipped with sensors for CO2 concentration, temperature, and airborne particle concentration.

Some ongoing data collections are scheduled as well as some triggered by spikes in the temperature measurements. The two nodes closer to the volcano crater are deemed high risk, and therefore, the data they collect needs to have redundant copies on other nodes.

The simulation is run with two different sets of parameters: one set favoring data reduction, and one favoring data relocation. Data relocation is a viable option as the importance computation function of one data instance in most cases does not depend on other data instances.

While analyzing the results, we discover that the redundancy approach behaves in a manner that needs additional protocol specifications. The two high risk nodes were using each other to host redundant copies of collected data.

6.1 Introduction

In this section, we select a use case for the presented approach and analyze the outcome of a simulation. We chose to simulate a deployment where six sensor nodes are placed in proximity of a volcano. The nodes are pre-positioned as follows:

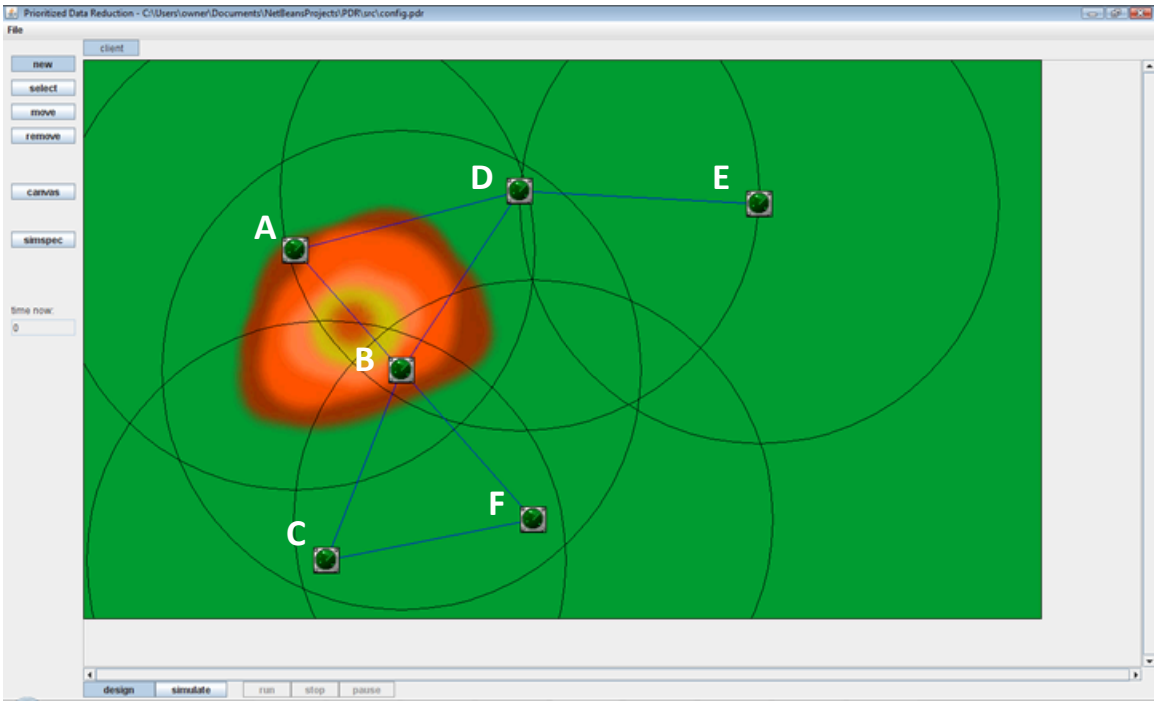


Figure 6.1. Deployment of use case sensor nodes

In the scenario above, the aim is to monitor the volcanic activity both at close range as well as from a longer distance. The nodes are identical as far as hardware configuration. Sensors are equipped with sensors for CO2 concentration, temperature, and airborne particle concentration.

6.2 The business case

The purpose of the data collection is to obtain a baseline level of data collection for the three parameters that a node can collect. In addition to this, several unexpected events can trigger shorter term fine grained data recordings. It is clear that sensor nodes A and B, being in closer proximity to the volcano, are subjected to higher change for a hardware failure. Therefore, redundant copies of the data collected by those nodes should also be stored on other nodes.

6.3 The specifications

For the sake of simplicity, we divide the nodes into two classes of nodes: high risk nodes and low risk nodes. The high risk nodes are those closer to the volcano crater (A and B) while the low risk nodes are further away (C, D, E, and F). We specify data collections and parameterization for each class of nodes.

Low risk nodes

Ongoing collections with no redundancy required:

Table 6.1. Specification of ongoing data collections with no redundancy

name	**start**	**end**	**parameter**	**recurrence**	**resolution**	**compress.**
CO-1	TRUE	FALSE	CO2 %	5 minutes	0.01	FALSE
C-1	TRUE	FALSE	temperature	1 minute	0.01	FALSE
D-1	TRUE	FALSE	airborne ppm	10 minutes	0.001	FALSE

Triggered collections with no redundancy required:

Table 6.2. Specifications of triggered collections with no redundancy

name	start	end	parameter	recurrence	resolution	compress.
CO-2	airborne particles over 1ppm	airborne particles under 0.1ppm	CO2 %	10 seconds	0.01	FALSE
C-2			temperature	5 seconds	0.01	FALSE
D-2			airborne ppm	20 seconds	0.001	FALSE

So what we have described above are constant data monitoring operations in the form of ongoing collections. In addition, while monitoring the airborne particle concentration, if the concentration goes over 1ppm, we monitor all three parameters at a very short interval until the airborne particle concentration drops below 0.1ppm. As these nodes are in low risk areas, at least by comparison to other nodes, we do not require that redundant copies be made.

High risk nodes

Ongoing collections with redundancy required:

Table 6.3. Specifications of ongoing data collections with redundancy

name	start	end	parameter	recurrence	resolution	compress.
CO-1	TRUE	FALSE	CO2 %	1 minutes	0.01	FALSE
C-1	TRUE	FALSE	temperature	10 seconds	0.01	FALSE
D-1	TRUE	FALSE	airborne ppm	1 minutes	0.001	FALSE

Triggered collections with redundancy required:

Table 6.4. Specification of triggered collections with redundancy

name	**start**	**end**	**parameter**	**recurrence**	**resolution**	**compress.**
CO-2	airborne particles over 10ppm	airborne particles under 1ppm	CO2 %	2 seconds	0.01	FALSE
C-2			temperature	1 seconds	0.01	FALSE
D-2			airborne ppm	5 seconds	0.001	FALSE

On the high risk nodes, we have specified a tighter sampling rate for the ongoing data collections. In the case o triggered collections the start and end conditions are also changed as a proximity to the crater has different expected steady state concentration of airborne particles. For the ongoing collections, the data needs an additional copy for redundancy. For the triggered data collections, we need two additional backup copies stored in the network.

For each of the data instances, we have the option to specify a data unit production. This gives the ability to have a more fined grained control regarding potential data reduction during times of high interest. This would apply to cyclical events such as traffic during rush hour, or traffic during the week-end. In our case, we are doing environmental monitoring of a volcano. There is no reason to vary data unit sizes. We decide that for the ongoing data collections, the data is to be divided in one hour long data units. For the triggered collections, we divide the data into five minute long data units.

As a method of data reduction, we favor iterative sparsing. The data units themselves are fairly small. Therefore, we can gather a general picture of the trend in the measured parameters by comparing representative values from each data unit. Having access to the micro-variations that occur within a data unit is deemed only marginally better than having a sparsed data set. When there is a single measurement left in the data unit, we can drop the entire data unit as part of data reduction.

While the sparsing is applied as part of data reduction, we design K, the compensation factor, to age out data that is older than a certain time. For example, we can agree that a default K is 0% for all the data collections. As collections undergo reductions, K grows towards 50% by one percentage point every time the data unit undergoes a reduction.

At this point, we need to construct formulas to compute the importance values for data units. Overall, the business decision is to give more importance to data generated by the high risk nodes. Those nodes are at risk for data loss via node failure and we need to assign higher importance to the data generated by those nodes and for the redundancy copies generated by those nodes.

Low risk nodes

For the ongoing data collections, we assign I as follows:

- 0.85 for the 10 most recent days
- 0.65 for the following 100 days

- 0.55 for the following 1000 days
- 0.40 for any data older than the above
- if we experience a temperature spike above 100C, then
    - all the data importance increases by 0.25, capped by a maximum of 1.0, for all data units 20 days following the temperature spike
    - all the data importance increases by 0.20, capped by a maximum of 1.0, for all data units 30 days prior to the temperature spike

For the triggered collections, we assign I as follows:

- 0.95 for the 5 recent and least recent days
- 0.80 for the days in between

High risk nodes

For the ongoing data collections, we assign I as follows:

- 0.95 for the 10 most recent days
- 0.75 for the following 100 days
- 0.65 for the following 1000 days
- 0.50 for any data older than the above
- if we experience a temperature spike above 200C, then
    - all the data importance increases by 0.25, capped by a maximum of 1.0, for all data units 20 days following the temperature spike

- all the data importance increases by 0.20, capped by a maximum of 1.0, for all data units 30 days prior to the temperature spike

For the triggered collections, we assign I as follows:

- 0.95 for the 5 recent and least recent days
- 0.80 for the days in between

In the case of high risk nodes, we recall that we seek redundant copies of the data collected. We need one redundant copy of the ongoing data as well as two redundant copies for the triggered data. We are going to establish that the importance of the redundant copies will be established as follows:

- we eliminate the special clauses where data importance depends on temperature reading spikes
- we establish that the first redundant copy has an importance reduced to 80% of the computed value
- we establish that the second redundant copy has an importance reduced to 60% of the computed value

6.4 The results

We run two simulations of the above scenario: one that is more inclined to use data reduction, and one that is more inclined to use load sharing. At the end of the simulation, we analyze the data stored in each node (columns) against the data producer

(rows). In Table 6.5, we see the results at the end of a simulation with storage level mitigation weighed towards data movement.

Table 6.5. Breakdown of data storage with preferred relocation

	A	**B**	**C**	**D**	**E**	**F**
A	70%	14%	25%	8%	3%	17%
B	11%	66%	22%	5%	8%	23%
C	7%	8%	38%	3%	2%	6%
D	5%	3%	7%	67%	12%	5%
E	2%	2%	2%	10%	68%	3%
F	5%	7%	6%	7%	7%	46%

To gain better understanding of these numbers, we plot the data relating the data producer and the node that in the end stores the data:

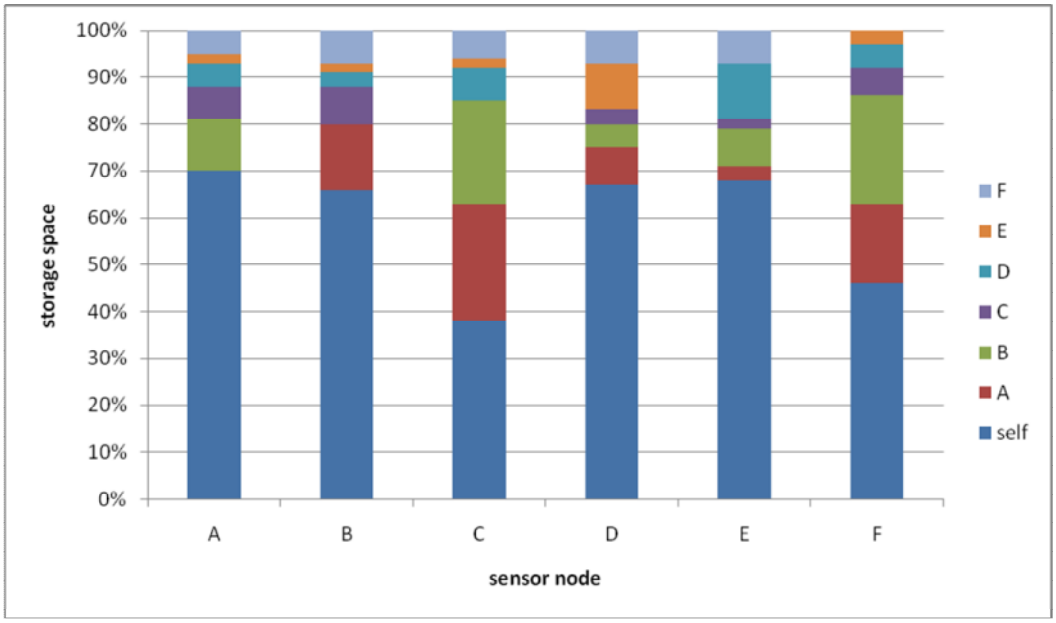


Figure 6.2. Data storage as related to origin of data under relocation favoured conditions

For comparison reasons, Table 6.6 and Figure 6.3 show the data in case data reduction is given preference over data relocation.

Table 6.6. Breakdown of data storage with preferred reduction

	A	**B**	**C**	**D**	**E**	**F**
A	81%	11%	19%	8%	2%	12%
B	8%	73%	16%	4%	6%	13%
C	5%	7%	53%	1%	1%	5%
D	3%	2%	5%	74%	11%	3%
E	1%	2%	2%	8%	77%	2%
F	2%	5%	5%	5%	3%	65%

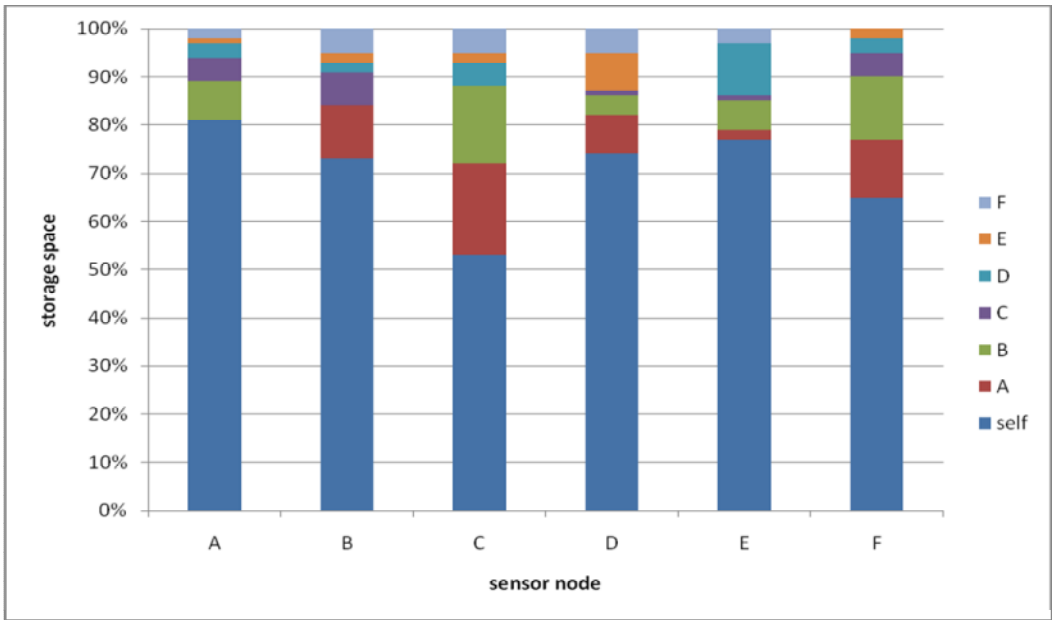


Figure 6.3. Data storage as related to origin of data under reduction favoured condtions

The results above show that the proposed mechanisms perform the task of ensuring longest survival of most important data. By comparing figures 6.2 and 6.3, we can see how the balance can be tilted in favor of data reduction of data relocation.

One item of concern with the above data is the handling of the redundancy data generated by the high risk nodes. The ongoing data collection of the two high risk nodes was required to have a redundant copy saved on a different node. Because of their proximity, nodes A and B ended up using each other to store redundant copies of their data. While this ensures redundancy, the reason for the redundancy was not captured in the mechanism, resulting in two high risk nodes offering redundant space for each other.

We can see that a fair amount of data was relocated even in the case where the decision was tilted towards data reduction. This was facilitated by having importance calculation functions that are very free of data dependencies. In cases with several nodes where data dependency is high, it is harder to find appropriate data to move, and nodes are relegated to data reduction.

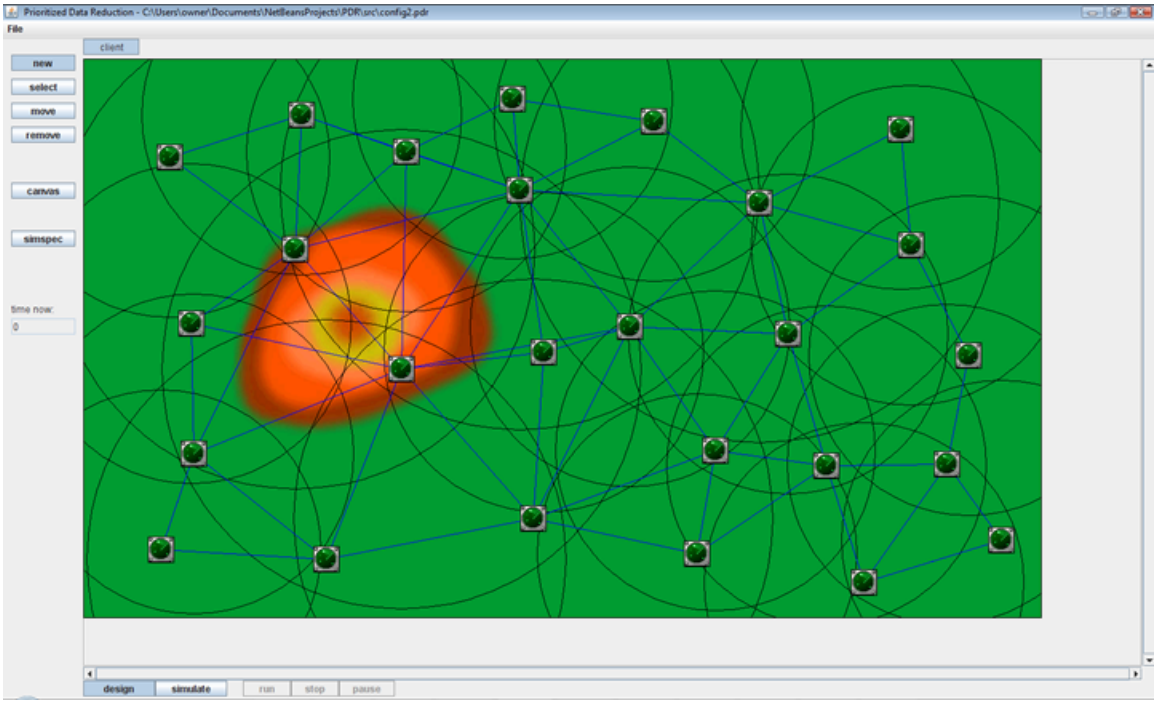


Figure 6.4. Highly connected sensor nodes

The data profile generated by the above deployment will look something like Figure 6.5:

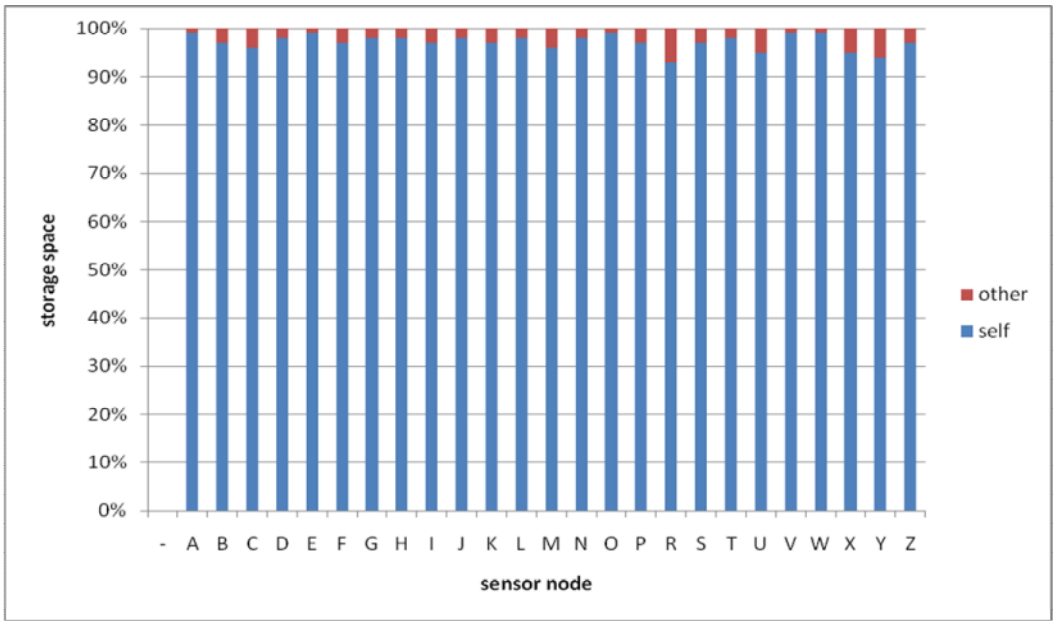


Figure 6.5. The effect of highly dependent importance function

In Figure 6.5, we notice that above 90% of the data stored on a node has been collected by that node. In cases where the importance calculation function needs to frequently access many data units, finding a self enclosed set of data units is not feasible. Lots of energy is spent finding for this type of search and in the end, there is no benefit. It would be a good decision in cases like this to completely turn off data relocation.

6.5 Conclusion

In this chapter, we presented a simple use case of an environmental monitoring deployment. Prioritized data reduction has been used to free storage space so that high importance data can stil be stored. Data load sharing was implemented in order to make

better use of the general storage pool of the network. Redundancy was used to protect the data collected by the nodes in high risk areas. The data collected shows that we can favor data reduction or data relocation as we decide.

One item that needs additional attention is the choice of redundancy nodes. Redundancy is mostly needed for nodes that are in high risk areas, or that by their nature, are proce to faults. At this point, there is no mechanism to specify the danger associated with storing redundant copies on one node versus another. Such a mechanism is needed to prevent redundant copies of data from ending up in high risk areas altogether.

Chapitre 7

Conclusion et Recherches Futures

7.1 Conclusion

Dans cette thèse, nous avons considéré certains problèmes spécifiques aux réseaux de capteurs sans fil. Par leur mode d'opération et leur structure, ces réseaux possèdent des ressources limitées quant au calcul, au stockage et à la transmission. Par conséquent, les protocoles existants, habituellement utilisés dans des réseaux câblés plus fiables, ne peuvent pas être utilisés à cause par exemple, de connexions intermittentes, d'un taux élevé de défaillance des nœuds, etc. Des protocoles spécifiques de transmission et de distribution de données ont été proposés ainsi que des mécanismes pour permettre le groupage dans le cas de transmission de données.

Le problème lié à la gestion de données collectées et à leur stockage, surtout dans un environnement assez peu fiable, a été négligé. Dans les réseaux de capteurs sans fil ayant pour but la collecte intensive de données traitées off-line, une défaillance entre les nœuds émetteurs et la station de base (nœud récepteur) peut compromettre la valeur des informations et mener à une situation dangereuse, en fonction de l'application supposée utiliser les données. La solution immédiate est d'arrêter la collecte de données ou de vider la mémoire pour en recevoir de nouvelles. Cependant, il est possible que les données perdues soit d'une importance majeure et même critiques.

Nos propositions concernent des solutions pour la gestion de données lorsque le nœud émetteur ne peut pas établir une connexion avec la station de base (sink node). Nous avons proposé le concept d'unité de données dans lequel on associe un facteur d'importance à l'aide de primitives et d'une fonction de calcul spécifiquement introduite. Les unités de données ainsi considérées peuvent subir un traitement dans lequel on conserve la valeur de l'information collectée mais avec un volume de mémoire réduit.

Un vue à travers un réseau étendu nous a permit de proposer des mécanismes qui utilisent la mémoire disponible dans le réseau entier pour garder le plus d'information ayant le facteur d'importance le plus élevé. Notons que dans la majeure partie des réseaux, il y a des nœuds qui collectionnent des données avec une grande valeur pour les applications, tandis que d'autres collectionnent des données avec une importance faible. Notre proposition et les mécanismes associés pour la relocalisation de données s'avère très efficace. On a associé à cette option des mécanismes pour évaluer les décisions à prendre entre la réduction des données ou la relocalisation de données.

Un mécanisme supplémentaire a été introduit pour assurer la possibilité de déployer des copies de données lorsqu'elles sont logées dans des nœuds à risque. Dans ces cas, il a des restrictions afin de pouvoir calculer l'importance des données sur les nouveaux sites.

Par le biais de simulations nous avons testé la faisabilité de nos propositions et leurs avantages. Comme la gestion de données est étroitement liée au facteur d'importance, les nœuds continuent à collecter des données et l'espace de stockage disponible se rétrécit. Les paramètres proposés permettent alors d'obtenir de bons

résultats. Cependant, plus de simulations sont nécessaires pour identifier des directions plus précises, tant pour la réduction que pour le partage de données.

7.2 Recherches futures

Les solutions présentées permettent de déployer des nœuds spécialisés. Par exemple, certains nœuds peuvent être équipés de mémoire supplémentaire pour être désignés comme des nœuds cibles pour la relocation de données ou pour la sauvegarde de copies de données. Ces types de ‘mini-data-centers’ peuvent être placés dans des endroits plus sécurisés.

La collection récurrente de données est basée sur des conditions de ‘start’ et ‘stop’. Des mécanismes supplémentaires sont nécessaires pour déterminer à quel moment la collecte de données doit être arrêtée et lorsque plusieurs collectes s’exécutent en parallèle, une décision doit être prise sur les collectes qui doivent être arrêtées.

Un autre aspect à reconsidérer est la manière utilisée pour calculer l’importance d’une unité de données. Dans l’étude présente, nous avons considéré que l’importance des données dans les actions présentes mais il est envisageable que certaines données soient très importantes dans les actions futures.

Dans nos propositions, l’importance des données est calculée localement sur un nœud. Il y a une limitation qui doit être révisée, car il est possible qu’une importance soit dépendante des données qui sont captées ou stockées sur d’autres nœuds. Comme le temps de transmission varie, l’actualisation de l’importance d’une unité de données doit

s’effectuer dans certains intervalles de temps. Comment caractériser ces intervalles et comment identifier l’influence de cette non-synchronisation de calcul restent des problèmes ouverts.

Il sera intéressant d'étudier ce qui se passe lorsque la station de base devient disponible.. Comme les données peuvent être acheminées par des nœuds différents et comme le calcul de l’importance de données doit se faire à travers tous les nœuds impliqués, ceci soulève des questions de précision de calcul et des priorités de transfert sont alors nécessaires

Si la station de base (sink node) est disponible pour une courte période (ou planifiée), le transfert de données n’est pas envisagé, car le temps n’est pas suffisant pour leur transfert et pour le calcul. Dans ce cas, ‘in network querries’ est plus souhaitable et des méthodes de compression de données peuvent s’avérer utiles.

À une échelle plus grande, une fois la connexion avec la station de base établie, la station peut déployer des planifications de collecte pour certains nœuds ou encore introduire des priorités ou des dépendances nouvelles entre les données. Dans ce cas, les aspects de sécurités deviennent plus importants.

Pour conclure, nous considérons que le travail et les solutions proposés concernant la gestion de données dans les réseaux de capteurs sans fil ouvrent des pistes de recherche pour la modélisation et l’optimisation des ressources et la collecte de données.

Chapter 7

Conclusion and Future Work

In this work, we have presented some of the problems inherent to Wireless Sensor Networks. Given their mode of operation, WSNs are limited with respect to the resources they can use. Protocols normally used in reliable wired networks cannot simply be adapted to work in wireless setups with unreliable connections and high node failure rate. New routing protocols have been devised as well as data dissemination protocols. Support for distributed queries has been implemented. Clustering protocols have been proposed in an effort to compact transmissions.

One item that has not received much research interest is the management of the storage space in instances where the network is disconnected from the sink node. In WSNs mandated to conduct intensive data collection for off-site analysis, a disconnect from the sink node can translate into an unmanageable accumulation of data. This can lead to a need to stop the collection until current data can be offloaded. Simple aging out of data can be a simple option, but the assumption that older data is less important than new data can lead to the loss of particular events of interest.

Our proposal addresses the issue of data management while the nodes are unable to transfer data back to the sink node. We proposed the basic building blocks which allow the architect of the WSN to reduce the data load while maintaining the maximum possible amount of most important data. The collected data is broken up in manageable data units.

Several primitives are available that give up some data but still maintain a good representation of the collected data. The primitive operations are applied on data units.

When we consider the entire WSN as a whole, there is opportunity to use the entire available storage to come up with a better solution to storage space limits. Nodes that are actively accumulating high importance data can make use of the storage space on nodes that are not actively collecting data, or that are actively collecting lower importance data. To this effect, we introduced a method for nodes to consider data relocation before considering running a round of data reduction.

To address the issue of WSNs being prone to failure, we proposed a mechanism to address data redundancy within the framework of prioritized data reduction. Restrictions were set on the functions evaluating the importance of redundant copies because the redundant copies need to be immediately sent to other nodes.

The simulated use cases prove that the proposed solutions are feasible. As long as there is energy on the nodes, the nodes continue to collect data while clearing out low importance data to make necessary space. Given that the proposed approach is heavily parameterized, simulations are needed to validate the impact of the parameter values selected.

The proposed solution has been presented as operating on its own. It performs well, but further work is needed to integrate the approach with ongoing protocols in a deployed WSN. As seen in Chapters 2 and 3, many protocols are available. Routing

protocols to move data run continuously, as well as protocols that directly affect the data, such as data aggregation.

The presented solution gives the possibility to deploy specialized data nodes. These nodes can be equipped with additional storage and only server as recipients of data relocation actions and redundancy copies. Such data center nodes can be placed in more secure locations and provide a viable solution to the situation we had in our example where two high risk nodes were using each other to store redundant copies of data.

Data collections are specified with a start and a stop condition. Within those two conditions, data collection is happening. Further work is needed to assess how the importance of the collected data compares with the already stored data, and under what conditions it is acceptable to pause a data collection. The situation gets even more complex with several ongoing data collections. A mechanism is needed to decide which subset of collections is put on hold.

Another aspect that needs more studying is the way we compute importance of data units. At this point, we compare the importance levels of data units without regards to the potential importance that data units can have in the future. That is not always a guarantee as events of interest can increase the importance level to data collected in the past.

Data importance is computed locally on the nodes. The computation function has access to all the data that is already stored there to make a decision on a data unit’s

importance level. There is a limitation as a node may want to define data importance in relation to data being collection on a different node. As inter-node communication is not constant, updating data unit importance depending on data stored on other nodes has to be done at some intervals. More research is needed to specify and characterize such intervals.

Finally, there is the issue of what exactly happens when a sink node becomes within reach of the deployed network. Ideally all the collected data is sent to the sink. There are several issues that appear when this happens. The connection to the sink may not be permanent so an educated ordering must take place for the data transfer. Data can be routed via multiple hops, so some nodes need to manage their own data transfer as well as acting as relay nodes. Another issue that presents itself is with respect to the computation of the importance for data units. Some of these computations depend on the presence of other data. If some data has been sent to the sink, this needs to be accounted for so that the computation of the importance value is correct.

If the access to the sink is very short lived and scheduled, a data transfer is not desirable as it will not complete during the time the sink is available. In these cases, in-network queries are a better choice. Even then, instructions are needed regarding the fate of data that has been queried. Is there still a need for that data to be stored, or can it be discarded?

On a bigger scale, a connection being established to the sink is an opportunity for the sink to assess the resource levels in the network and to push updates to the data

collection schedule. A large range of security issues arise in such a context which need to be further studied.

We consider that this work opens the possibility for more research in the area of storage space optimization in Wireless Sensor Networks.

References

[1] J. Hill, R. Szewczyk, A, Woo, S. Hollar, D. Culler, and K. Pister “System Architecture Directions for Networked Sensors” Ninth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2000, Cambridge, MA, USA, pp.93-104

[2] S. Singh, M. Woo, and C. Raghavendra “Power-aware routing in mobile ad hoc networks” Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, MobiCom 1998, Dallas, TX, USA, pp.181-190

[3] Miguel Garcia, Diana Bri, Sandra Sendra, and Jaime Lloret “Practical Deployments of Wireless Sensor Networks: a Survey” International Journal on Advances in Networks and Services, vol 3, no 1&2, 2010, pp.163-178

[4] T. He, S. Krishnamurthy, J. Stankovic, T. Abdelzaher, L. Luo, T. Yan, R. Stoleru, L. Gu, G. Zhou, J. Hui, and B. Krogh, “VigilNet: An Integrated Sensor Network System for Energy Efficient Surveillance” ACM Transactions on Sensor Networks, vol 2, no 1, February 2006, pp.1-38

[5] G. Virone, A. Wood, L. Selavo, Q. Cao, L. Fang, T. Doan, Z. He, R. Stoleru, S. Lin, and J. A. Stankovic, "ALARM-NET: an Assisted Living-centered and Testbed-oriented Information System Based on a Residential Wireless Sensor Network" Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare, D2H2 2006, Arlington, VA, USA, pp.95-100

[6] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson “Wireless sensor networks for habitat monitoring” ACM International Workshop on Wireless Sensor Networks and Applications, WSNA 2002, Atlanta, GE, USA, pp.88-97

[7] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein “Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet” Tenth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2002, San Jose, CA, USA, pp.96-107

[8] M. Radenkovic and B. Wietrzyk, "Wireless Mobile Ad-hoc Sensor Networks for Very Large Scale Cattle Monitoring" 6th International Workshop on Applications and Services in Wireless Networks, ASWN 2006, Berlin, Germany, pp.47-58

[9] M. Dunbabin, P. Corke, I. Vasilescu, and D. Rus, “Data muling over underwater wireless sensor networks using an autonomous underwater vehicle” IEEE International Conference on Robotics and Automation, ICRA 2006, Orlando, FL, USA, pp.2091-2098

[10] J. Yuh “Design and control of autonomous underwater robots: A survey” Autonomous Robots, vol. 8, no. 1, 2000, pp.7–24, 2000

[11] C. C. Eriksen, T. J. Osse, R. D. Light, T. Wen, T. W. Lehman, and P. L. Sabin “Seaglider: A long-range autonomous underwater vehicle for oceanographic research,” IEEE Journal of Oceanic Engineering, vol. 26, no. 4, October 2001, pp. 424–436

[12] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan "Energy-Efficient Communication Protocol for Wireless Microsensor Networks" 33rd Annual Hawaii International Conference on System Sciences, HICSS 2000, vol. 2, Maui, HI, USA, pp.3005-3014

[13] J. Kim, K.-y. Jang, H. Choo, and W. Kim “Energy Efficient LEACH with TCP for Wireless Sensor Networks” International Conference on Computer Science and Applications, San Francisco, CA, USA, vol. 4706, 2007, pp.275-285

[14] Jin Ding Sivalingam, K. Kashyapa, and R. Lu Jian Chuan “A multi-layered architecture and protocols for large-scale wireless sensor networks” IEEE 58th Vehicular Technology Conference, VTC 2003 Fall, Orlando, FL, USA, pp.1443-1447

[15] J. Ding and K.M. Sivalingam “An improved unified network protocol framework for large-scale wireless sensor networks” Digital Wireless Communications VI, Orlando, FL, USA, Vol. 5440, pp.204-215

[16] Ting Zhu, Ziguo Zhong, Tian He, and Zhi-Li Zhang “Exploring Link Correlation for Efficient Flooding in Wireless Sensor Networks” 7th USENIX Conference on Networked Systems Design and Implementation, NSDI 2010, San Jose, CA, USA, pp.4-4

[17] S. Guo, Y. Gu, B. Jiang, and T. He “Opportunistic Flooding in Low-Duty-Cycle Wireless Sensor Networks with Unreliable Links” 15th Annual International Conference on Mobile Computing and Networking, MobiCom 2009, Beijing, China, pp. 133-144

[18] E. Zanaj, M. Baldi, and F. Chiaraluce “Efficiency of the Gossip Algorithm for Wireless Sensor Networks” 15th International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2007, Split-Dubrovnik, Croatia, pp.1-5

[19] F. Lu, L.-T. Chia, K.-L. Tay, and W.-H. Chong “NBgossip: An Energy-Efficient Gossip Algorithm for Wireless Sensor Networks” Journal of Computer Science and Technology, JCST 2008, vol. 23, no. 3, pp.426-437

[20] D. Braginsky and D. Estrin “Rumor routing algorithm for sensor networks” Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications, WSNA 2002, Atlanta, GA, USA, pp.22-31

[21] Z. Modi, S. Jardosh, P. Ranjan “Optimized Rumor routing algorithm for Wireless Sensor Networks” Fifth IEEE Conference on Wireless Communication and Sensor Networks, WCSN 2009, Allahabad, India, pp.1-6

[22] K. Sohrabi, J. Gao, V. Ailawadhi, and Pottie, G. J., “Protocols for Self-Organization of a Wireless Sensor Network,” IEEE Personal Communications, vol. 7, issue 5, pp.16-27

[23] K. Sohrabi “On Low Power Wireless Sensor Networks,” Ph.D. Dissertation, Department of Electrical Engineering, UCLA, June 2000

[24] J. L. Gao, ``Energy Efficient Routing for Wireless Sensor Networks,” Ph.D. Dissertation, Department of Electrical Engineering, UCLA, June 2000.

[25] C. Intanagonwiwat, R. Govindan, and Estrin D “Directed diffusion: a scalable and robust communication paradigm for sensor networks” 6th Annual International Conference on Mobile Computing and Networking, MobiCom 2000, Boston, MA, USA, pp.56-67

[26] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva “Directed diffusion for wireless sensor networking” IEEE/ACM Transactions on Networking, TON 2003, vol. 11 , issue 1, February 2003, pp.2-16

[27] W. Heinzelman, J. Kulik, and H. Balakrishnan “Adaptive protocols for information dissemination in wireless sensor networks” 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, MobiCom 1999, Seattle, WA, USA, pp.174-185

[28] M. Tubaishat, S. Madria, “Sensor Networks : An Overview”, IEEE Potentials, Volume: 22 Issue 2, April/May 2003, pp20-23

[29] S. Lindsey and C. S. Raghavendra “PEGASIS: Power-Efficient Gathering in Sensor Information Systems” IEEE Aerospace Conference, AERO 2002, Monterey, CA, USA, pp.3-1125-3-1130 vol.3

[30] W. Guo, W. Zhang, and G. Lu, "PEGASIS Protocol in Wireless Sensor Network Based on an Improved Ant Colony Algorithm" The Second International Workshop on Education Technology and Computer Science, ETCS 2010, vol. 3, Wuhan, China, pp.64-67

[31] N. Bulusu, J. Heidemann, and D.Estrin "GPS-less Low Cost Outdoor Localization for Very Small Devices" IEEE Personal Communications Magazine, October 2000, vol 7, issue 5, pp.28-34

[32] D. L. Mills "Internet time synchronization: the Network Time Protocol" IEEE Trans. Communications COM-39, vol. 10 (October 1991), pp.1482-1493

[33] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi "The Flooding Time Synchronization Protocol" 2nd International Conference on Embedded Networked Sensor Systems, SenSys 2004, Baltimore, MD, USA, pp.39-49

[34] A.I. McInnes "Model-checking the Flooding Time Synchronization Protocol" IEEE International Conference on Control and Automation, ICCA 2009, Christchurch, New Zealand, pp.422-429

[35] S. P. Beeby, M. J. Tudor, and N. M. White "Energy harvesting vibration sources for microsystems applications" Measurement Science and Technology, Vol 17, 2006, pp.175-195

[36] B.P. Mann and N.D. Sims "Energy harvesting from the nonlinear oscillations of magnetic levitation" Journal of Sound and Vibration, vol. 319, issues 1-2, 9 January 2009, pp.515-530

[37] D. P. Arnold "Review of Microscale Magnetic Power Generation", IEEE Transactions on Magnetics, vol. 43, issue 11, November 2007, pp.3940-3951

[38] S. Cheng, N. Wang, and D. P. Arnold "Modeling of magnetic vibrational energy harvesting systems using equivalent circuit representations" Journal of Micromechanics and Microengineering, vol. 17, no. 11, Nov. 2007, pp.2328-2835

[39] R. Guigon, J. Chaillout, Thomas Jager, and G. Despesse "Harvesting raindrop energy: experimental study" Smart Materials and Structures, vol. 17, no.1, 2008, pp.15-39

[40] J. Granstrom, J. Feenstra, H. A. Sodano, and K Farinholt "Energy harvesting from a backpack instrumented with piezoelectric shoulder straps" Smart Materials and Structures, vol 16, no. 5, 2007, pp.1810-1820

[41] A. H. Epstein "Millimeter-scale, micro-electro-mechanical systems gas turbine engines" Journal of Engineering for Gas Turbines and Power, vol. 126, issue 2, 2004, p. 205- 226

[42] J. F. Randall "On Ambient Energy Sources for Powering Indoor Electric Devices", Ph D Thesis, EPFL, 2003

[43] D. Steere, A. Baptista, D. McNamee, C. Pu, and J. Walpole "Research challenges in environmental observation and forecasting systems" Sixth Annual International Conference on Mobile Computing and Networking, MobiCom 2000, Boston, MA,USA, pp.292-299

[44] D. M. Rowe "CRC Handbook of Thermoelectronics: Micro and Nano", CRC Press, 2005

[45] T. J. Coutts "Thermophotovoltaic principles, potential, and problems" 10th American Physical Society Topical Conference on Shock Compression of Condensed Matter, AIP 1997, Amherst, MA, USA, pp.217-233.

[46] M. Zenker, A. Heinzel, G. Stollwerck, J. Ferber, and J. Luther "Efficiency and power density potential of combustion-driven thermophotovoltaic systems using GaSb photovoltaic cells" IEEE Transactions on Electron Devices, vol. 48, issue 2, August 2002, pp.367-376

[47] A. Reid and M. Judd "A Novel Self-Powered Condition Monitoring Sensor for Harsh Environments-Feasibility Study" Poster at 15th International Symposium on High Voltage Engineering, ISH 2007, Ljubljana, Slovenia

[48] T. von Bueren, P. D. Mitcheson, T. C. Green, E. E. Yeatman, A. S. Holmes, and G. Troester "Optimization of Inertial Micropower Generators for Human Walking Motion Motion" IEEE Sensors Journal, vol. 6, issue 1, 2006, pp.28-38

[49] J. Yun, S. Patel, M. Reynolds, and G. Abowd, "A quantitative investigation of inertial power harvesting for human-powered devices" 10th international Conference on Ubiquitous Computing, UbiComp 2008, Seoul, South Korea, pp.74-83

[50] A. S. Weddell, N. R. Harris, and N. M. White "Alternative Energy Sources for Sensor Nodes: Rationalized Design for Long-Term Deployment" International Instrumentation and Measurement Technology Conference, Victoria, BC, Canada, IMTC 2008, pp.1370-1375

[51] A.S. Weddell, N.J. Grabham, N.R. Harris, and N.M. White "Flexible Integration of Alternative Energy Sources for Autonomous Sensing" Electronics System-Integration Technology Conference, ESTC 2008, Greenwich, UK, pp.597-600

[52] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak "Compressive Wireless Sensing" Information Processing in Sensor Networks, IPSN 2006, Nashville, TN, USA, pp.134-142

[53] E. Magli, M. Mancin, and L. Merello "Low Complexity Video Compression for Wireless Sensor Networks" International Conference on Multimedia and Expo, ICME 2003, vol. 3, Baltimore, MD, USA, pp.585- 590

[54] M. Rabbat, J. Haupt, A. Singh, and D. Nowak "Decentralized Compression and Predistribution via Randomized Gossiping" International Conference on Information Processing in Sensor Networks, IPSN 2006, Nashville, TN, USA, pp.51-59

[55] H. Chan and A. Perrig. "Security and privacy in sensor networks" IEEE Computer Magazine, pp. 103–105, October 2003

[56] C. Krauss, F. Stumpf, and C. Eckert "Detecting node compromise in hybrid wireless sensor networks using attestation techniques" 4th European Conference on Security and Privacy in Ad-hoc and Sensor Networks, ESAS 2007, Cambridge, UK, pp.203-217

[57] A. Wood and J. Stankovic "Denial of Service in Sensor Networks" IEEE Computer, vol. 35, no. 10, October 2002, pp. 54-62

[58] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler "Spins: security protocols for sensor networks" Wireless Networks Journal, vol. 8, no. 5, 2002, pp.521–534

[59] T. Shu, S. Liu, and M. Krunz "Secure Data Collection in Wireless Sensor Networks Using Randomized Dispersive Routes" IEEE Transactions on Mobile Computing, vol. 9, issue 7, July 2010, pp.941-954

[60] P. M. Cholewinski "Evasive Data Storage in Sensor Networks" Diploma Thesis in Computer Science, Laboratory for Dependable Distributed Systems, RWTH Aachen, 2005

[61] D. Niculescu and B. Nath "Trajectory based forwarding and its applications" Ninth Annual International Conference on Mobile Computing and Networking, MobiCom 2003, San Diego, CA, USA, pp.260-272

[62] M. Busse, T. Haenselmann, and W. Effelsberg "Energy-Efficient Data Dissemination for Wireless Sensor Networks" Fifth Annual IEEE International

Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2007, White Plains, NY, USA, pp.301-306

[63] S. Madden, M. Franklin, J. Hellerstein, and W. Hong "TAG: a tiny aggregation service for ad-hoc sensor networks" ACM Symposium on Operating System Design and Implementation, OSDI 2002, Boston, MA, USA, pp.131-146

[64] B. Krishnamachari, D. Estrin, and S. Wicker "Impact of Data Aggregation in Wireless Sensor Networks" 22nd International Conference on Distributed Computing Systems, ICDCSW 2002, Vienna, Austria, pp.575-578

[65] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann "Impact of Network Density on Data Aggregation in Wireless Sensor Networks" International Conference on Distributed Computing Systems, ICDCS 2002, Vienna, Autria, pp. 457

[66] J. Ledlie, C. Ng, D. A. Holland, K. Muniswamy-Reddy, U. Braun, and M. Seltzer "Provenance-Aware Sensor Data Storage" 21st International Conference on Data Engineering Workshops, ICDE 2005, Tokyo, Japan, pp.1189-1189

[67] T. He, J.A. Stankovic, C. Lu, and T. Abdelzaher. "SPEED: A stateless protocol for real-time communication in sensor networks" 23rd Int. Conf. on Distributed Computing Systems, ICDCS 2003, Providence, Rhode Island, USA, pp.46-55

[68] C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He. "RAP: A real-time communication architecture for large-scale wireless sensor networks" 8th IEEE Real-Time and Embedded Technology and Applications Symposium, RATS 2002, San Jose, CA, USA, pp.55-66.

[69] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J. Stankovic, and T. Abdelzaher "Real-Time Analysis of Tracking Performance in Wireless Sensor Networks" IEEE Real-Time Applications Symposium, RTAS 2006, San Jose, CA, USA, pp.37-48

[70] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and Wei Hong "Tinydb: An acquisitional query processing system for sensor networks" ACM Transactions on Database Systems (TODS)-Special Issue: SIGMOD/PODS 2003, vol. 30, issue 1, pp.122-173

[71] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker "GHT: a geographic hash table for data-centric storage" Wireless Sensor Networks and Applications, WSNA 2002, Atlanta, GA, USA, pp.78-87

[72] P. Bonnet, J. Gehrke, and P. Seshadri "Towards Sensor Database Systems" Mobile Data Management Conference, MDM 2001, Hong Kong, China, pp.3-14

[73] H. Lee, A. Klappenecker, K. Lee, and L. Lin “Energy Efficient Data Management forWireless Sensor Networks with Data Sink Failure” IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, MAHSS 2005, Washington, DC, USA, pp.210-216

[74] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz “ESRT: Event-to-sink reliable transport in wireless sensor networks” Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2003, Annapolis, MD, USA, pp.177-188

[75] J. Deng, Y. Han, W. Heinzelmaan, and P. Varshney “Balanced-energy sleep scheduling scheme for high density cluster-based sensor networks” 4th Workshop on Applications and Services in Wireless Networks, ASWN 2004, Boston, MA, USA, pp.99-108

[76] Y. Xu, J. Heidemann, and D. Estrin “Geography-informed energy conservation for ad hoc routing” Seventh Annual International Conference on Mobile Computing and Networking, MobiCom 2001, Rome, Italy, pp.70-84

[77] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang “PEAS: A robust energy conserving protocol for long-lived sensor networks” Twenty-Third International Conference on Distributed Computing Systems, ICDCS 2003, Providence, RI, USA, pp.28-37

[78] D. Brunelli, L. Benini, C. Moser, and L. Thiele “An Efficient Solar Energy Harvester for wireless sensor Nodes” Design, Automation, and Test in Europe, DATE 2008, Munich, Germany, pp.104-109

[79] J. Ahn and B. Krishnamachari “Fundamental scaling laws for energy-efficient storage and querying in wireless sensor networks” The Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2006, Florence, Italy, pp.334-343

[80] K. Park and R. Elmasri “Query Classification and Storage Evaluation in wireless sensor networks” 22nd International Conference on Data Engineering, ICDE Workshops 2006, Atlanta, GA, USA, pp.35-35

[81] N. Siegmund, M. Rosenmüller, G. Moritz, G. Saake, and D. Timmermann. “Towards Robust Data Storage in wireless sensor networks” Workshop on Database Architectures for the Internet of Things, DAIT 2009, Birmingham, England, pp.335-340

[82] J. Girão, D. Westhoff, E. Mykletun, and T. Araki: “TinyPEDS: Tiny persistent encrypted data storage in asynchronous wireless sensor networks” Journal of Ad Hoc Networks, vol. 5, issue 7, September 2007, pp.1073-1089

[83] E. Mykletun, J. Girao, and D. Westhoff “Public Key Based Cryptoschemes for Data Concealment in wireless sensor networks” IEEE International Conference on Communications, ICC 2006, Istanbul, Turkey, pp.2288-2295

[84] M. I. Khan, W. N. Gansterer, and G. Haring “In-Network Storage Model for Data Persistence under Congestion in wireless sensor network” First International Conference on Complex, Intelligent and Software Intensive Systems, CISIS 2007, Vienna, Austria, pp.221-228

[85] Y. Lai, H. Chen, and Y. Wang “Dynamic balanced storage in wireless sensor networks” 4th International Workshop on Data Management for Sensor Networks, DMSN 2007, Vienna, Austria, DMSN pp.7-12

[86] P. Ratanaworabhan, J. Ke, and M. Burtscher “Fast lossless compression of scientific floating-point data” Data Compression Conference, DCC 2006, Salt Lake City, UT, USA, pp.133-142

[87] C. Dini and P. Lorenz, “Primitive Operations for Prioritized Data Reduction in Wireless Sensor Network Nodes” Fourth International Conference on Systems and Networks Communications, ICSNC 2009, Porto, Portugal, pp.274-280

[88] S. Tilak, W. Heinzelman, and N. Abu-Ghazaleh, “Storage Management Issues for Sensor Networks”, Poster at International Conference on Network Protocols, ICNP 2003, Atlanta, Georgia, 2003

[89] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin, “Data-centric storage in sensornets”, ACM SIGCOMM Computer Communication Review, vol. 33, issue 1, January 2003, pp.137-142

[90] N. Siegmund, M. Rosenmuller, G. Moritz, G. Saake, and D. Timmermann, “Towards Robust Data Storage in Wireless Sensor Networks”, IETE Tech Rev, 2009, vol. 26, issue 5, pp.335-40. Available: http://tr.ietejournals.org/text.asp?2009/26/5/335/55280 [accessed August 2010]

[91] M. Aly, N. Morsillo, and K. Pruhs “Zone sharing: a hot-spots decomposition scheme for data-centric storage in sensor networks” Second International Workshop on Data Management for Sensor Networks, DMSN 2005, Trondheim, Norway, pp.21-26

My Publications

International Journal

1. C. Dini, P. Lorenz, "Prioritized Redundancy of Data Storage in Wireless Sensor Networks", International Journal on Advances in Systems and Measurements", vol. 3, no. 3&4, 2010, ISSN: 1942-261x.

International Conferences

2. C. Dini, P. Lorenz, "Optimizing Parameters of Prioritized Data Reduction in Sensor Networks", Fourth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, UBICOMM'10, October 25 - 30, 2010 - Florence, Italy, pp. 346-350

3. C. Dini, P. Lorenz, "Prioritizing Data Processing in Wireless Sensor Networks", Sixth International Conference on Networking and Services, ICNS'10, March 7-13, 2010, Cancun, Mexico, pp.23-31.

4. C. Dini, P. Lorenz, "Primitive Operations for Prioritized Data Reduction in Wireless Sensor Network Nodes", 4th International Conference on Systems and Networks Communications, ICSNC'09, September 20-25, 2009, Porto, Portugal, pp. 274-281.