

# **Wireless Sensor Networks Maintenance Framework**

**(Wartungsmechanismen für Drahtlose Sensor Netzwerke)**

Vom Fachbereich Informatik der Technischen Universität Darmstadt genehmigte

Dissertation

zur Erlangung des akademischen Grades eines Doktor-Ingenieur (Dr.-Ing.)  
vorgelegt von

M. Sc. Piotr Szczytowski

aus Gliwice, Polen

Referenten:  
Prof. Neeraj Suri, Ph.D.  
Prof. Christian Becker, Ph.D.

Datum der Einreichung: 12. Februar 2013  
Datum der mündlichen Prüfung: 19. April 2013

Darmstadt 2013  
D17



## Summary

The capability of the Wireless Sensor Network (WSN) to perform the information processing and energy saving tasks and their effectiveness largely depends on ensuring stability of network properties. Unpredictable nature of WSN and at the same time strict application requirements call for the incorporation of the maintenance mechanisms. The aim of this thesis is to design a set of maintenance techniques for tackling the most common and important problems impacting WSN. This thesis defines four distinct maintenance problems and presents efficient and robust solutions. The considered problems include: Localized Energy Hole Profiling - mapping of an uneven energy distribution in the network, Topology Oriented Maintenance - finding and remedying topology irregularities, Distributed  $k$ -Connectivity Maintenance - assuring communication reliability by providing low resource cost localized  $k$ -connectivity with latency guarantees and Adaptive Spatial Sampling - dynamic adaptation of spatial sampling resolution in order to match the sampling to the dynamics of monitored phenomena. The presented in this thesis algorithms for tackling the above outlined problems constitute together Maintenance Framework for WSN.

*Localized Energy Hole Profiling:* WSNs display non-uniform energy usage distribution, induced by non-uniform distribution of communication traffic or sensing activities, which manifests itself as *energy holes*. Energy holes are direct source of the network partitioning and sensing voids. The thesis presents new distributed energy profiling algorithms for generalized types of energy holes. The algorithms search for boundary nodes of energy hole and use them as a reference to calculate the energy needs of sensor nodes within the energy hole. These, when aggregated, create angular and radial energy profiles.

*Topology Oriented Maintenance:* Sparse WSN networks even while connected, usually suffer from topology irregularities that negatively impact the network lifetime and responsiveness, i.e., sensor data delivery reliability and latency. The thesis directly targets the problem by proposing algorithms that use the discrepancy between Euclidean and hop distances, to provide in-network and localized strategy that efficiently (i) discovers generic topology irregularities, and (ii) identifies locations for minimal number of new augmented sensor deployments to remedy topology irregularities and sustain the desired operational requirements.

*Distributed  $k$ -Connectivity Maintenance:* A common approach for providing reliability in WSN is to assure global  $k$ -connectivity. This property guarantees that the failure of up to  $k - 1$  sensor nodes does not cause network partitioning. Thesis develops a technique that allows for localized, sustainable maintenance, capable of efficiently restoring the WSN desired  $k$ -connectivity. The approach is based on assuring that each sensor node in its routing tree has at least  $k$  direct or indirect neighbors, which are placed closer to the sink than sensor node itself.

*Adaptive Spatial Sampling:* A prominent application of WSNs is the monitoring of physical phenomena. The monitored phenomena often tend to have unknown spatial distributions, that also change over time resulting in either under- or over-sampling of signals in space. The thesis outlines a Voronoi based adaptive spatial sampling algorithm, which aims at minimizing measured signal variation at neighboring Voronoi sensor nodes. This

approach generates additional new sampling locations in the under-sampling regions to fulfill specified accuracy requirements.

The effectiveness and efficiency of all outlined in this thesis algorithms were tested through a set of extensive simulations.

## Kurzfassung

Die Fähigkeit des Wireless Sensor Networks (WSN), für die Informationen Verarbeitung, Energieeinsparung Aufgaben und ihre Wirksamkeit hängt weitgehend von der Gewährleistung der Stabilität der Netzwerk-Eigenschaften ab. Unberechenbare Natur von WSN, bei gleichzeitiger strengen Anwendung Anforderungen, erfordert die Einbeziehung der Wartungsmechanismen (Maintenance). Das Ziel dieser Arbeit ist es, der Entwurf einer Reihe von Wartungs-Techniken für die Bewältigung der häufigsten und wichtigen Problemen die auf WSN auswirken.

Diese These definiert vier verschiedene Wartungs-Probleme und präsentiert effiziente und robuste Lösungen. Es wurden in dieser These folgende Probleme behandelt: Localized Energy Hole Profiling - das Lokalisieren der ungleichmäßigen Energieverteilung in dem Netzwerk, Topology Oriented Maintenance - das Finden und Beheben von Unregelmäßigkeiten der Topologie, Distributed k-Connectivity Maintenance - Sicherstellung der Zuverlässigkeit der Kommunikation durch Bereitstellung kostengünstigen Ressourcen lokalisierte k-Konnektivität mit Latenz Garantien, und Adaptive Spatial Sampling - dynamische Anpassung der räumlichen Abtastauflösung, um die Dynamik der überwachten Phänomenen zu entsprechen. Die in dieser These präsentierte Algorithmen zur Bewältigung der vorgestellten oben skizzierten Probleme bilden zusammen Wartung Framework für WSN.

Localized Energy Hole Profiling: WSNs zeigten an ungleichmäßige Energieverbrauch Verteilung induziert durch ungleichmäßige Verteilung der Kommunikationsverkehrs oder durch die Messungsaktivitäten, die sich als Energie Löcher manifestierten. Diese Energie Löcher sind direkte Quelle das Netzwerk-Partitionierung und verursachen die Messungshohlräume. Die These präsentiert neue dezentrale Profiling-Algorithmen für generalisierte Arten von Energie Löcher. Die Algorithmen suchen nach Grenzknoten und verwenden diese als eine Referenz, um die benötigte Energie von Sensorknoten im Energieloch zu berechnen. Diese, wenn aggregiert, erstellen winkligen und radialen Energie-Profile.

Topology Oriented Maintenance: dünne WSN Netzwerken leiden unter Topologie Unregelmäßigkeiten, auch wenn verbunden. Das wirkt negativ auf das Netzwerk Lebensdauer und Reaktionsfähigkeit, z.B.: Sensordaten Liefertreue und Latenz übertragen. Die These zielt direkt auf das Problem mit den vorgeschlagenen Algorithmen, die die Diskrepanz zwischen den Euklidischen und den Schritt (Hop) Entfernungen benutzen, um in-Netzwerk lokalisierte Strategien bereitzustellen. Diese Strategien (i) entdecken effizient generischen Topologie Unregelmäßigkeiten und (ii) identifizieren Standorte für minimale Anzahl von neuen Sensor-Installationen zu Behebung von Topologie Unregelmäßigkeiten und Erhaltung der gewünschten betrieblichen Anforderungen.

Distributed k-Connectivity Maintenance: das gemeinsame Konzept für Bereitstellung von Zuverlässigkeit in WSN basiert auf Gewährleistung globaler k-Konnektivität . Diese Eigenschaft garantiert, dass das Scheitern von bis zu  $k - 1$  Sensorknoten keine Netzwerk-Partitionierung verursacht. These entwickelt eine Technik, die lokalisierte, nachhaltige Wartung, und effiziente Wiederherstellung der WSN  $k$ -Konnektivität ermöglicht. Der Ansatz basiert auf dem Prinzip, dass jeder Sensorknoten in seinem Routing-Baum hat

mindestens  $k$  direkten oder indirekten Nachbarn, die sind platziert näher an der Sink als der Sensorknoten an sich.

Adaptive Spatial Sampling: Eine prominente Anwendung des WSNs ist die Überwachung von dem physikalischen Phänomen. Die überwachten Phänomene neigen oft dazu, unbekannte räumliche Verteilungen zu haben. Diese führen im Laufe der Zeit, entweder zum "under-" oder "over-sampling" von Signalen im Raum. Diese These skizziert einen Voronoi basierten, adaptiven räumlichen Sampling-Algorithmus, der auf die Minimierung der gemessenen Signaländerung an benachbarten Voronoi Sensorknoten zielt. Dieser Ansatz generiert zusätzliche neue Mess-Stellen in der "under-sampling" Regionen um spezifizierte Genauigkeit zu erfüllen.

Die Effektivität und Effizienz aller in dieser These beschriebenen Algorithmen wurden getestet durch umfangreiche Simulationen.

# Contents

<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem Statement . . . . .	3
1.1.1 Non-Uniform Energy Distribution in Wireless Sensor Networks . . . . .	3
1.1.2 Topology Maintenance . . . . .	5
1.1.3 Fault Tolerance . . . . .	6
1.1.4 Adaptive Spatial Sampling . . . . .	7
1.2 Thesis Targets and Contributions . . . . .	8
1.2.1 Thesis Research Questions . . . . .	8
1.2.2 Thesis Contributions . . . . .	9
1.2.3 Publications Resulting from the Thesis . . . . .	10
1.3 Thesis Structure . . . . .	11
<b>2 State of the Art and Practice - WSN Maintenance</b>	<b>13</b>
2.1 Coverage Preservation . . . . .	14
2.1.1 Energy optimization . . . . .	16
2.1.2 Summary . . . . .	16
2.2 Connectivity . . . . .	17
2.2.1 k-Connectivity . . . . .	19
2.2.2 Summary . . . . .	22
2.3 Sampling . . . . .	23
2.3.1 Resource optimization in case of over-sampling . . . . .	26
2.3.2 Summary . . . . .	26
2.4 Holes problem in WSN . . . . .	27
2.4.1 Energy Holes . . . . .	27
2.4.2 Topology Holes . . . . .	28
2.4.3 Summary . . . . .	30

2.5	Topology Optimization . . . . .	30
2.5.1	Energy oriented . . . . .	31
2.5.2	Relay nodes . . . . .	31
2.5.3	Summary . . . . .	32
2.6	Chapter Summary . . . . .	33
<b>3</b>	<b>Localized Energy Hole Profiling in Wireless Sensor Networks</b>	<b>35</b>
3.1	Overview . . . . .	35
3.1.1	Contributions . . . . .	35
3.2	System Model . . . . .	36
3.3	LEHP: Localized Energy Hole Profiling . . . . .	37
3.3.1	Overview of LEHP Approach . . . . .	37
3.3.2	Profiling . . . . .	38
3.3.3	Profiling Outcome and Usage . . . . .	43
3.3.4	On the Selection of the Initiator . . . . .	44
3.3.5	Profiling without Position Knowledge . . . . .	45
3.3.6	Reconfiguration . . . . .	46
3.4	Evaluation . . . . .	46
3.4.1	Evaluation Setting . . . . .	46
3.4.2	Profiling Evaluation Metrics . . . . .	47
3.4.3	Energy Efficiency of LEHP . . . . .	47
3.4.4	Simulation Results . . . . .	48
3.5	Summary . . . . .	52
<b>4</b>	<b>Topology Oriented Maintenance in Sparse Wireless Sensor Networks</b>	<b>53</b>
4.1	Overview . . . . .	53
4.1.1	Contributions . . . . .	54
4.2	System Model . . . . .	54
4.3	Problem Formulation and Objectives . . . . .	55
4.4	TOM: Topology Oriented Maintenance . . . . .	58
4.4.1	Overview . . . . .	58
4.4.2	Discovery of Topology Irregularities . . . . .	59
4.4.3	Search for Bridging Nodes . . . . .	60
4.4.4	Reconfiguration . . . . .	65
4.5	Performance Evaluation . . . . .	66
4.5.1	Evaluation Settings . . . . .	66
4.5.2	Evaluation Metrics . . . . .	66
4.5.3	Evaluation Results . . . . .	67
4.6	Summary . . . . .	70

<b>5</b>	<b>Distributed <math>k</math>-Connectivity Maintenance in Wireless Sensor Networks</b>	<b>73</b>
5.1	Introduction . . . . .	73
5.1.1	Contributions . . . . .	73
5.2	System Model . . . . .	74
5.3	Objectives and Requirements . . . . .	75
5.4	$k$ -Connectivity Maintenance . . . . .	77
5.4.1	Guide through our DKM Approach . . . . .	77
5.4.2	Localized Estimation of $k$ -Connectivity . . . . .	78
5.4.3	Resolving Nodes . . . . .	79
5.4.4	Avoidance of Bottleneck (Overlapping) Links . . . . .	82
5.4.5	Optimizing Required Resources . . . . .	82
5.4.6	Self-Sustaining $k$ -Connectivity . . . . .	84
5.4.7	Graph Balancing . . . . .	84
5.5	Performance Evaluation . . . . .	86
5.5.1	Evaluation Settings . . . . .	86
5.5.2	Evaluation Results . . . . .	86
5.6	Summary . . . . .	90
<b>6</b>	<b>Adaptive Spatial Sampling in Wireless Sensor Networks</b>	<b>91</b>
6.1	Introduction . . . . .	91
6.1.1	Contributions . . . . .	91
6.2	System Model . . . . .	92
6.3	Problem Formulation and Objectives . . . . .	92
6.4	The Adaptive Spatial Sampling Approach . . . . .	93
6.4.1	Distributed Voronoi Construction . . . . .	94
6.4.2	Overview of our Approach . . . . .	95
6.4.3	The Proposed ASample Approach . . . . .	95
6.4.4	Coping with Dynamic Phenomena . . . . .	99
6.4.5	Reconfiguration Scenarios . . . . .	100
6.5	Performance Evaluation . . . . .	101
6.5.1	Evaluation Studies . . . . .	101
6.5.2	Evaluation Metrics . . . . .	102
6.5.3	Complexity Analysis . . . . .	102
6.5.4	Evaluation Results . . . . .	103
6.6	Mobility Assisted Adaptive Sampling . . . . .	107
6.6.1	Overview . . . . .	107
6.6.2	gMAP: Efficient Delay Tolerant Monitoring . . . . .	108
6.6.3	Demo Setup . . . . .	109
6.7	Summary . . . . .	112

<b>7</b>	<b>Conclusions and Future Research</b>	<b>113</b>
7.1	Overall Thesis Contributions . . . . .	114
7.1.1	Localized Energy Hole Profiling in Wireless Sensor Networks . . . . .	114
7.1.2	Topology Oriented Maintenance in Sparse Wireless Sensor Networks . . . . .	114
7.1.3	Distributed k-Connectivity Maintenance in Wireless Sensor Networks . . . . .	115
7.1.4	Adaptive Spatial Sampling in Wireless Sensor Networks . . . . .	116
7.2	Lessons Learned . . . . .	117
7.3	Open Ends - Basis for Future Work . . . . .	117
<b>A</b>	<b>Map-Based Modeling and Design of Wireless Sensor Networks</b>	<b>119</b>
A.1	Introduction . . . . .	119
A.2	System Model . . . . .	121
A.3	WSN Simulation Environments . . . . .	121
A.4	The MAP++ Architecture . . . . .	123
A.4.1	Overview of the MAP++ Architecture . . . . .	123
A.4.2	The MAP++ Trace Module . . . . .	124
A.4.3	The MAP++ Tools . . . . .	125
A.5	MAP++ Benefits and Applications . . . . .	126
A.5.1	Overview . . . . .	127
A.5.2	Modeling Stage . . . . .	127
A.5.3	Design Stage . . . . .	129
A.5.4	Debugging and Performance Evaluation Stage . . . . .	130
A.6	MAP++ Evaluation . . . . .	131
A.6.1	Case Study . . . . .	131
A.6.2	Performance Evaluation of MAP++ . . . . .	134
A.7	Conclusions . . . . .	135
	<b>Bibliography</b>	<b>137</b>

# List of Figures

3.1	Angular and radial profiling . . . . .	38
3.2	Operational phases of LEHP . . . . .	39
3.3	Impact of node density . . . . .	49
3.4	Impact of aggregation distance . . . . .	50
3.5	Impact of hole distribution . . . . .	51
3.6	Impact of hole shape . . . . .	51
3.7	Radial and angular energy need distribution . . . . .	52
4.1	Distribution of actual hop distances compared to expected distances. . . . .	56
4.2	Topology irregularities in a sparse deployment example . . . . .	57
4.3	TOM Tunability to repair irregularities of varied severity . . . . .	68
4.4	Impact of the network size . . . . .	69
4.5	Comparison with random deployment strategy . . . . .	70
4.6	Impact on Voronoi diagram calculation . . . . .	71
5.1	Single link elimination . . . . .	76
5.2	Single Link Elimination . . . . .	83
5.3	Virtual Sink Selection . . . . .	85
5.4	Ratio of added resources in function of required $k$ -connectivity . . . . .	87
5.5	Ratio of added resources in function of number of deployed nodes . . . . .	88
5.6	Ratio of added resources in function of communication range . . . . .	89
5.7	Impact of the elimination of bottleneck overlapping links . . . . .	89
5.8	Average local $k$ -connectivity indicator . . . . .	90
6.1	Variants of signal distribution . . . . .	93
6.2	Voronoi neighborhood . . . . .	94
6.3	Impact of communication range . . . . .	104
6.4	Changing accuracy requirement . . . . .	105
6.5	Changing network density . . . . .	105
6.6	Various phenomena models . . . . .	106
6.7	Evolving phenomenon . . . . .	107
6.8	Mobile robot traversal path . . . . .	108

6.9	Structured scenario . . . . .	111
A.1	MAP++ Architecture . . . . .	124
A.2	Voronoi-Based Map Visualization . . . . .	126
A.3	Database Interface . . . . .	128
A.4	Regioning in MAP++ . . . . .	130
A.5	Energy Map for Different Hotspot Energy Distribution Models . .	132
A.6	Accuracy for Different Simulation Scenarios) . . . . .	133
A.7	MAP++ Performance . . . . .	135

# List of Tables

3.1	Tabular representation of angular and radial profile . . . . .	44
-----	--	----



# Chapter 1

## Introduction

Wireless Sensor Networks (WSN)[Laprie, 1992] constitute composite communication and computational systems. A typical WSN consists of multiple battery powered autonomous devices equipped with processing units and communicating wirelessly. Moreover sensor nodes include single or set of sensors for sensing target environmental attributes (e.g. temperature, humidity, objects proximity, etc) and limited storage capacity to temporarily store the acquired data before it can be processed and send any further. The generic goal of these system is the collection of the data by means of sensors, optional in-network data processing, and delivery to data collection point. The commonly accepted name for the collection point is the *Sink*. Sink serves as a mediation entity between fixed infrastructure and WSN. Unlike typical sensor nodes sink is much more powerful both in aspects of energy (mostly fixed power connection) and in terms of computational power and storage. As the wireless communication range of typical sensor node is strictly limited, due to energy considerations (non-rechargeable, low capacity batteries), the process of the data delivery in most cases involves more than a single transmission. Therefore, sensor nodes in order to deliver data to the sink have to establish and maintain routing information. The wireless nature of this autonomous devices allows creation of spontaneous self-organizing networks. Furthermore, the battery power provides much desired, especially for real-world applications, deployment flexibility. This includes the flexibility of the placement as well as a type of delivery (e.g. aerial dropping of nodes).

The characteristics of WSN make it suitable for very wide range of current and potential applications. The basic type of applications is build around generic concept of mapping. The sensor nodes periodically perform sensing operation using installed on board set of sensor and transmit the data to the sink, where data are used to recreate the distribution of the phenomenon of interest. Unfortunately, this straightforward approach of sensing the data and sending it to sink is highly inefficient in context of high energy costs of wireless transmission and energy limited

supply. Therefore, a more sophisticated applications instead of only delivering the data, perform data processing within the network. The distributed application running on the sensor nodes first tries to determine the utility of measurements prior to sending it. Although the computation capabilities of single sensor nodes are very limited they multitude allows for execution of complex algorithms. As a result the network may be able to compress the data (by omitting temporal or spatial samples, or changing their representation) depending on required sampling resolution. Other efficient technique of minimizing number of exchanged messages is submitting only sought for facts instead of raw data. The sensor nodes can cooperate and using distributed localized approach detect single events like e.g. fire, instead of submitting map of temperatures.

Although saving the energy is the most direct aspect influencing the performance of WSN, there exists set of equally important issues involved in operation of the WSN. Lifetime of the network is not only depended on the energy expenditure but also directly impacted by the energy distribution. Failure of few sensor nodes overburdened with traffic may lead to disconnections and network's fragmentation. Patterns of energy distributions are both the function of network topology as well as the distribution of monitored phenomenon. Network topology is equally a function of the sensor nodes distribution and communication range. Low uniformity of sensor nodes distribution or short communication ranges lead to irregular topology and as a result to overextended routing paths and low neighborhood connectivity. Longer paths require higher number of message exchanges and involve in routing higher number of sensor nodes. Low neighborhood connectivity decreases aggregation and compression level of algorithms and hinders detection of spatially correlated events. The dynamics of monitored phenomena render the energy dissipation distribution patterns unpredictable. Moreover, these dynamics impact also the utility of the applications. The changing intensity of monitored phenomena demands adaptation of the sampling resolution in order to allow its proper reconstruction and/or evaluation.

In face of existence of all these enumerated problems, there is an urgent need for technique, or rather set of techniques (tailored to intricacies of given problem) to ensure reliable functioning of WSN. The potential solution should ideally prevent manifestation of the problems (pro-activeness) or at least limit theirs both spatial and temporal occurrence. The postulated techniques should be lightweight, to limit use of resources, and function-wise transparent, so not to impact primary functionality of the network. They also need to be multi-objective, optimized to handle a mixed set of problems at once without creating problems on their own. In order to assure self-sustainability of WSN, the techniques should use only localized knowledge and be capable of reuse of existing resources without outside the network intervention (e.g. deployment of additional resources) to resolve the handled issues. The goal of this thesis is design and development of such tech-

niques, that are later called maintenance techniques and constitute Maintenance Framework.

The remainder of this chapter is organized as follows. First, we highlight the problems being addressed in this thesis. Next, the main ideas driving the research in this thesis are presented. Later, we summarize the thesis goals in terms of research questions followed by the answers in the form of research contributions of this thesis.

## 1.1 Problem Statement

The capability of the network to perform the information processing and energy saving tasks and their effectiveness largely depends on ensuring stability of network properties. Unpredictable nature of WSN and at the same time strict application requirements call for the incorporation of the maintenance mechanisms. Maintenance denotes set of actions taken to assure preservation or restoration of specified operational conditions. The type of the required maintenance actions in context of WSN is often very application specific. The goal of this thesis is the development of a Maintenance Framework consisting of set of maintenance techniques targeting the most important properties of WSN. The core WSN properties identified and studied in this thesis include:

- energy dissipation distribution directly impacting WSN lifetime,
- topology regularity of the network directly impacting performance (route lengths, latencies, energy distribution) of WSN,
- fault tolerance providing survivability and reliability guarantees,
- the adaptive sampling resolution of sensor network translating into data matching application requirements and saving resources.

Following subsections detail the research problems involved and give basic outline of pursued solutions.

### 1.1.1 Non-Uniform Energy Distribution in Wireless Sensor Networks

The energy depletion is the most important factor impacting the lifetime of the WSN. The WSN lifetime estimation is complicated as the energy consumption within the network tends to show non-uniform distribution. That often leads to network partitioning and degradation of sensing and communication coverage.

Typical example of such behavior is sink centered energy hole problem Li and Mohapatra [2005]; Liu et al. [2008]; Olariu and Stojmenoviæ [2006]; Wu and Chen [2006]. In this case, sensor nodes placed in close proximity of the sink deplete their energy at a faster rate as rest of the sensor nodes in the network. This condition has its source in the fact that sensor nodes in sink neighborhood, besides transmitting their own data are also responsible for forwarding rest of the WSN traffic, so the data originating at further placed sensor nodes can arrive at the sink. The closer the sensor node to the sink the more sensor nodes belong to its routing subtree and the greater burden on forwarding the data. As a result, although most of the sensor nodes still retain significant amounts of energy, the loss of sink neighbors disconnects rest of the sensor nodes from sink and as consequence renders the entire network inoperable. This process is well studied and understood. Typical remedies involve adjusting the density of deployed sensor nodes to reflect analytically calculated energy needs in given part of the network. The additional sensor nodes (above the required to provide connectivity or coverage) take over the function of those lost to the energy deletion or switch the states to evenly distribute the transfer load.

Unfortunately, the centered energy hole problem is not the only source of unbalanced energy distribution in WSN. It is necessary very often to deal with much more complex scenarios where the non-uniform energy consumption distribution is not only induced by the topology but also significantly impacted by spatially correlated sensing activities. At the locations where the monitored phenomenon occurs, the sensor nodes need to adapt their sampling and sample reporting rates in order to meet desired accuracy. Higher frequency of reports, with associated higher message traffic, leads to accelerated energy depletion in the region of the network where phenomena is detected. The distribution of the energy usage mimics the distribution of monitored physical phenomena, which usually shows high spatial correlation. Monitoring mostly takes place in the unknown areas with the actual goal of discovering the distribution of the phenomena intensity. This problem could be alleviated by deploying sensor nodes with higher density Liu et al. [2008] in the targeted region. Unfortunately, barring a few trivial cases, communication load and consequently energy consumption patterns cannot be predicted in the pre-deployment stage. Thus, the formation of sensing and communication coverage holes remain serious threats to WSN. In order to provide necessary maintenance, which could react to this unforeseeable energy distribution pattern it is crucial to precisely abstract the anticipated energy needs by proactive energy hole profiling.

Current techniques allow only estimation of the holes periphery, which does not provide insight into the energy hole dynamics. The energy hole growth rate and the increased energy usage causing the emergence of hole are neglected. Only a static picture of the energy hole (its size and shape) is provided but not its internal

distribution required to estimate its future energy needs. Typical remedies for holes are: (a) to opportunistically reposition movable sensor nodes, or (b) place new sensor nodes to cover the hole. The actual amount of needed energy is often not considered. Therefore, if energy added is significantly higher than the energy needed, resources are wasted; if it is lower then the hole may reoccur negating the overall efforts.

### 1.1.2 Topology Maintenance

The wireless communication employed in WSN and battery power allow high flexibility for deployment. While the simple construction of sensor nodes reduces their unit cost, this unfortunately is counterweighted by the fact that a typical deployment requires multiple sensor nodes to provide communication coverage or effective monitoring. It is especially evident for uncontrolled deployments Bettstetter [2002], where the network has to be over-supplied with sensor nodes to assure connectivity (despite power depletion, connectivity losses and failures, etc). This rapidly increases the WSN deployment costs. Hence, a natural trend is to target the deployment of sparse networks while preserving the deployment flexibility. A related incentive to limit the density of the deployed sensor nodes are network capacity benefits Gupta and Kumar [2000] from reduced congestion, interference and collisions. Unfortunately, a sparse deployment also comes with set of serious disadvantages. Although the network may be connected, even the optimal spanning tree over such sparse network often results in a highly irregular topology.

The direct consequence of irregular topology is the existence of topological holes, critical Jorgiæ et al. [2004] / bottleneck Gou and Yoo [2010] nodes, and the non-uniform deployment of sensor nodes. They can be determined by measuring the discrepancy between Euclidean induced distance Vural and Ekici [2005, 2010] and the topology induced hop distance to the sink. This discrepancy arises only when there exists no *straight* route between the affected sensor node and the sink. Lengthy routes translate in higher latencies and higher/unbalanced energy consumption among the sensor nodes. As a result the WSN can suffer from unbalanced energy usage even under optimal energy conservation schemas, which usually leads to potential premature disconnections and partitioning. In addition, there exists a related major source of topology irregularities, namely the network failures. Sensor nodes often fail as they operate with finite energy capacities and in harsh environments. Usually in sparse networks, node failures directly impact the topology irregularities and negatively influence the network lifetime and responsiveness.

The overextended routing paths, critical and bottleneck nodes, and unbalanced energy usage are all symptoms of an underlying irregular topology. Unfortu-

nately, most maintenance strategies often overlook this aspect of the topology (ir)regularity. The typical maintenance solutions concentrate on detecting and partially remedying a subset of these symptoms, e.g., restoring connectivity or coverage while neglecting the actual source of root causes. Therefore, it is desirable to avoid such irregularity induced problems by developing efficient techniques to maintain a regular topology. As the network regularity requirements are mostly application specific, the maintenance strategy should offer tunability in setting bounds on the extent of irregularities to tolerate. In order to provide efficient and contextualized maintenance and decentralized solutions, the techniques should utilize local information. It can be easily shown that the relevance of an irregularity depends on the position of the sink in the network. Therefore, the proposed approach has to identify relevant irregularities and remedies only the *relevant* ones.

### 1.1.3 Fault Tolerance

The low reliability of wireless communication and the bounded energy of sensor nodes frequently lead to failures and consequently decrease the resilience of WSN. Therefore, for a wide range of applications, there is an urgent need to carefully design the fault tolerance mechanism. A required condition for meaningful WSN operations is the maintenance of network connectivity. An established approach is to provide the so-called  $k$ -connectivity property. The  $k$ -connectivity property guarantees that the removal of up to  $k - 1$  sensor nodes will not disconnect any of the sensor nodes in the network. The main efficiency metric of this approach is the number of the resources employed for this purpose. A popular approach for providing  $k$ -connectivity is to stock selected sensor nodes constituting the backbone of the network with up to  $k - 1$  additional sensor nodes Almasaeid and Kamal [2009]. Alternatively, missing links (required for  $k$ -connectivity and missing in current topology) are reconnected using  $k$ -stocked intermediate sensor nodes Bredin et al. [2005]; Pu et al. [2009]. The selection of proper sensor nodes for stocking depends on the used algorithm. That approach although effective and theoretically sound unfortunately does not consider the case when  $k$  co-located sensor nodes fail not due to energy discharge but as a result of external physical force, i.e., their failure is not result of internal state. In such situations although network is  $k$ -connected, single event can has disastrous consequences and still lead to partitioning.

In order to prevent such situations, it is necessary for a  $k$ -connectivity algorithm to guarantee that for each sensor nodes among its  $k$  alternative paths, none of the these paths share the same sensor node location. Another important requirement for the  $k$ -connectivity algorithms (which is generally neglected) is to assure that the use of alternative paths, in case of failures, does not substantially

degrade the performance of the network. A typical example is the failure of single sensor nodes (critical Jorgiæ et al. [2004] / bottleneck Gou and Yoo [2010] nodes) which, while not partitioning the network, significantly extend the length of the routes resulting in higher latencies, frequent congestion and higher energy overhead. Additionally, although the network remains connected on the topology level, traffic rerouting may be necessary causing temporary communication outages until new routes are established.

#### 1.1.4 Adaptive Spatial Sampling

The monitoring of physical phenomena constitutes a key application of WSNs. One of the key metrics to assess the quality of monitoring is the accuracy of the conducted measurements. The accuracy depends on several aspects, which can be categorized as *network* and *phenomenon* related. The main network related aspects are the spatial distribution of sensor nodes and sensors accuracy, which directly translate into spatial resolution of reconstructed signal, and communication reliability having significant impact on measurements fidelity. The central, phenomenon-related aspects, include the temporal and spatial distributions/dynamics of the phenomena of interest. The network related aspects can, in most cases, be efficiently handled by an appropriate state-of-the-art design and maintenance of the WSN. However, the phenomenon related aspects are provided as ranges and hard to forecast, which complicates the design and maintenance of the WSN. A worst-case-driven uniform deployment is expensive, and may waste valuable resources such as bandwidth and energy, while it may be still possible that the density of phenomena is higher than the granularity of the deployment. Consequently, an efficient and flexible strategy is required to maintain the desired monitoring accuracy depending on phenomenon-related and network-related factors. For specified accuracy requirements, such a strategy consists of achieving efficient reconfiguration and maintenance options in order to react to the varying physical phenomena. The network can be reconfigured by adjusting networks resources on the basis of gathered profiles Guestrin et al. [2005]; Krause et al. [2006]; Popa, D.O. and Mysorewala, M.F. and Lewis, F.L. [2006]. As the accuracy of spatial monitoring strongly depends on the sensor nodes spatial distribution, the appropriate generation of spatial samples through suppressing some Arici and Altunbasak [2004]; Gedik et al. [2007]; Willett et al. [2004] or adding some Butler and Rus [2003]; Zhang and Sukhat [2005] is possible. There are two possible approaches for such spatial sampling adaptation. The first approach aims at optimizing the use of the resources already present in the network (self-reconfiguration). The second approach exploits the possibility of supplementing the network with additional sensor nodes (maintenance). The scope/extent of WSN reconfiguration techniques depends on the available resources. In practi-

cal terms the optimization of resource utilization entails repositioning of the mobile nodes.

The challenge this desired adaptive spatial sampling poses is to locally identify both over- and under-sampled regions. In the regions of under-sampling, new sampling locations must be chosen, to bring the WSN within the accuracy requirements, while minimizing required resources. The redundant nodes from the over-sampled regions can be relocated to these new sensing locations. If the available resources are limited, then new sampling locations should be selected so as to minimize the unavoidable inaccuracy. Also combinations of two given approaches are possible. First, the network can try to balance itself by means of repositioning the mobile nodes, then it can conclude maintenance by supplementing only missing number of sensor nodes to reach required accuracy.

As the monitored phenomenon evolves, so must the WSN in order to sustain the fidelity of its measurements. Failing to do so results in (a) waste of resources in over-sampled regions and (b) lack of accuracy in under-sampled regions. Consequently, there is a need for a holistic approach that can identify both types of regions to drive adaptive sampling, and perform it in a distributed manner for meaningful adaptive reactivity in evolving WSNs.

## 1.2 Thesis Targets and Contributions

### 1.2.1 Thesis Research Questions

The research questions driving the research presented in this thesis consider wide range of maintenance techniques for WSN.

#### Research Questions

**Research Question 1 (RQ1)** *How to profile the energy hole distribution efficiently?*

Chapter 3 explains the origin and unpredictable nature of energy holes in WSN. Moreover, a generalized technique is described for efficient mapping the future energy needs of the energy holes, allowing devising preventive actions.

**Research Question 2 (RQ2):** *How to detect, measure and handle topology irregularities in WSN?*

Chapter 4 raises and answers this research question, while discussing the impact of topology properties on the performance of WSN.

**Research Question 3 (RQ3):** *How to assure  $k$ -connectedness, that in presence of failures would also maintain WSN performance?*

The ability to accurately answer this question is a crucial condition for providing efficient and low resource demanding failure tolerance mechanism in WSN.

**Research Question 4 (RQ4):** *How to devise sampling locations that correspond to the dynamics of monitored phenomenon?*

Chapter 6 discusses the problem of providing adaptive sampling in presence of unknown and dynamic phenomena. Chapter also offers answer to problem of determining, which sampling locations are redundant and therefore resources can be saved by setting them in idle state.

## 1.2.2 Thesis Contributions

The research presented in this thesis makes several important contributions for the OS and testing research community. Listed below, the main contributions also enumerate the research questions they help answering.

**Contribution 1 (C1) – Localized Energy Hole Profiling:** An efficient, balanced and accurate profiling technique of the energy needs within the hole. The technique is proactive allowing for prevention and proactive maintenance, and applies for generalized energy holes with different sizes, shapes and energy spatial distributions. (RQ1)

**Contribution 3 (C2) – Topology Oriented Maintenance in Sparse WSNs:** A technique that efficiently detects topology irregularities and provides maintenance options for a sustainable regular topology. It allows a measure of tunability by setting bounds on the tolerable level of topology irregularity. Its execution is localized, thus, allowing for efficient and sustainable WSN self-healing. (RQ2)

**Contribution 4 (C3) – Distributed  $k$ -Connectivity Maintenance:** A fault tolerance strategy that efficiently restores and preserves the  $k$ -connectivity property. It provides  $k$ -connectivity along with guaranteeing that the length of particular routes may become extended only up to  $k - 1$  hops, and is distributed and localization free. (RQ3)

**Contribution 2 (C4) – Adaptive Spatial Sampling:** A distributed technique for identifying under- and over-sampled regions. It utilizes current measurement and SNs placements, (a) to insert new sampling locations or (b)

to remove redundant spatial samples, in order to exactly meet the accuracy requirements. The proposed approach is holistic as it is valid for both over-sampling and under-sampling profiling. Accordingly, it simplifies network-level decisions on moving nodes from the over-sampled regions to the under-sampled ones and for tuning the network sampling resolution according to the changes in the monitored phenomena. (RQ4)

**Contribution 5 (C5) – Map-based Support for WSN Simulation:** A map-based simulation framework that offers extended topology generation, including setting of spatially correlated initial conditions and batch script generation for simplifying modeling and scenarios generation. It also allows for trace data visualization support that identifies spatial dependencies for design and debugging and offers export to and processing of trace data with support of relational databases for performance evaluation.

### 1.2.3 Publications Resulting from the Thesis

The work reported in this thesis is supported by several international publications:

- **Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *LEHP: Localized Energy Hole Profiling in Wireless Sensor Networks*, in Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Riccione, pp. 100–106, 2010
- **Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *ASample: Adaptive Spatial Sampling in Wireless Sensor Networks*, in Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), Newport Beach, pp. 35–42, 2010
- **Piotr Szczytowski**, Faisal Karim Shaikh, Vinay Sachidananda, Abdelmajid Khelil, Neeraj Suri, *Mobility Assisted Adaptive Sampling in Wireless Sensor Networks*, in Proceedings of the International Conference on Networked Sensing Systems (INSS), Demo session, Kassel, 2010
- **Piotr Szczytowski**, Abdelmajid Khelil, Azad Ali and Neeraj Suri, *TOM: Topology Oriented Maintenance in Sparse Wireless Sensor Networks*, in Proceedings of the IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), Salt Lake City, 2011
- **Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *DKM: Distributed k-Connectivity Maintenance in Wireless Sensor Networks*, in Proceedings

of the Annual Conference on Wireless On-demand Network Systems and Services (WONS), Courmayeur, 2012

- **Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *Map-based modeling and design of Wireless Sensor Networks with OMNeT++*, in Proceedings of International Symposium on Performance Evaluation of Computer & Telecommunication Systems (SPECTS), Istanbul, pp. 162–169, 2009

## 1.3 Thesis Structure

The structure of the following chapters follows the structure of the research questions described earlier:

**Chapter 1** presents the background of the problems driving this research, introduces the research problems and the contributions of this thesis.

**Chapter 2** introduces the terminology used throughout the thesis and surveys the state of the art and practice in the field of Wireless Sensor Network Maintenance.

**Chapter 3** describes distributed energy profiling algorithms for generalized types of energy holes in WSN.

**Chapter 4** describes an in-network and localized strategy that efficiently discovers generic topology irregularities, and identifies locations for minimal number of new augmented sensor deployments to remedy topology irregularities.

**Chapter 5** describes k-connectivity technique that provides avoidance of network partitioning caused by the failure of co-located sensor nodes, and preserves WSN responsiveness through preventing the routes from extending in hop distance as result of sensor node failures.

**Chapter 6** describes a Voronoi based adaptive spatial sampling solution responsible for identifying the regions of over- or under-sampling and suggesting the appropriate countermeasures.

**Chapter 7** finally concludes the thesis, re-evaluating the value of the conceptual and experimental contributions. A discussion on the applicability of the thesis results to field of network maintenance is provided, alongside with an outline of the future research directions.



## **Chapter 2**

# **State of the Art and Practice - WSN Maintenance**

The goal of this chapter is to provide the context important for the research presented in this thesis. This chapter identifies and discusses in detail the state-of-the-art maintenance and related techniques employed in Wireless Sensor Networks. Maintenance in general terms can be defined as set of actions taken to assure preservation or restoration of specified operational conditions. Therefore, there exists wide range of the maintenance techniques are categorized in classes depending on the aspect of the network that they are targeting and the approach that these techniques employ. Each paragraph of this chapter describes an important maintenance class, presents the most related works and summarizes the techniques by comparing their properties.

## 2.1 Coverage Preservation

The goal of coverage preservation class of algorithms is to assure that at all times each point within the area of interest is placed within the sensing range of at least single sensor node. The sensing range is defined as a maximal distance at which sensor node can detect changes in the monitored phenomenon. There exists also stronger variant of this requirement called  $k$ -coverage. In this case each point within area of interest has to be placed within sensing range of at least  $k$  sensor nodes. The aim of discussed algorithms is to try to repair an existing sensing coverage holes by relocating sensor nodes so that sensing ranges of sensor nodes completely cover the area of interest. Depending on the algorithm different additional aspects are considered, e.g. minimization of movement distance of sensor nodes.

One of the common techniques to detect the coverage loss is based on using a virtual overlay grid over the deployment area Wang et al. [2005]. Size of the cells of the grid is in general function of the sensing radius of sensor node. Sensor nodes belonging to given cell elect coordinating node, which locally collects the sensor nodes positions and uses this information to detect if the cell is covered. If coverage is lacking the coordinator node can estimate number of required nodes for closing the gap. In opposite case, when the cell is covered then the coordinator node tries to check if any of the sensor nodes is redundant, meaning that its removal does not impair the cell coverage. Furthermore authors propose to perform process of matchmaking the resources, to find which redundant sensor nodes should be relocated to which under-provisioned cell. The matchmaking process is performed using grid quorums, where nodes communicate their interest or resource availability sending the information to sensor nodes sharing the same row or column of the grid. The decision which sensor nodes should move additionally depends on the energy levels of the sensor nodes. In order to furthermore balance the energy the move can be performed as series of cascaded moves, where each sensor node moves only for a distance of single cell freeing a sensor node in neighboring cell, which then continues the movement.

Sensor nodes can also use the neighbor information in order to estimate local sensing coverage map by calculating intersections of sensing ranges of neighboring nodes Ganeriwal et al. [2004]. Furthermore sensor node may also estimate the shape and size of area that will become uncovered in case of their failure. When they are close to completely discharging their batteries, they can inform other nodes about imminent danger of coverage loss. The sensor nodes receiving the distress message can estimate if their relocation would also result in local coverage loss. They send this information including the size of potentially uncovered area to dying node. The dying nodes decides, based on the potential coverage loss, if the relocation is beneficial and then it chooses for relocation the sensor

node which movement has smallest impact on coverage.

Alternative solution uses extended neighborhood information of 2 to n hops (when in direct neighborhood there are not enough nodes to cover the coverage hole). When sensor node dies and coverage hole is detected, remaining sensor nodes evaluate their possibility to move towards the coverage hole to restore the coverage. Decision is made depending on whether movement will lead to creation of a new hole.

Wang et al. [2004] presents algorithm based on bidding mechanism, where mobile nodes provide their bidding prices (depended on the coverage hole size they actually cover) to cover larger and larger holes in iterative manner. The process of bidding is localized to limited number of hops in relation to current placement of the mobile node. In order to avoid multiple and in most cases not optimal movement patterns, the movement itself is replaced by message sending and establishing proxy nodes (static nodes in neighborhood of proposed deployment) acting as virtual mobile node. Only when bidding process comes to conclusion (no new movement is necessary), proxy node informs the mobile node which it represents, that the final position is found and actual movement should take place.

The coverage can also be achieved using a technique based on potential fields Heo and Varshney [2005]. Each sensor node has a potential in relation to each other sensor node and the potential depends on the distance between the nodes. If the distance is too small and results in overlapping of sensing radiuses then the force between nodes causes repulsion. When distance between nodes is large enough then it may result in attraction force. Second proposed technique explores clustering method, where cluster head node is elected and given task of optimizing movements of rest of the sensor nodes in cluster while conserving energy and assuring coverage. Last strategy calculates the Voronoi diagram for the network and moves the sensor nodes to the centroid of their Voronoi cells. These techniques brings additional benefit of providing uniform distribution of sensor nodes.

Another approach postulates semi-controlled incremental deployment to achieve full coverage Filipe et al. [2004]. After initial random deployment the algorithm iteratively searches for the largest empty circle whose center is in the convex hull of a set of sensor nodes. In order to increase the efficiency of the approach, the algorithm uses information about energy levels of deployed sensor nodes and their dissipation models. The sensor nodes projected to die are not ignored in search for the non-coverage circles. After determining the sizes of circles additional targeted deployment takes place.

Sahoo et al. [2007] proposes scheme where sensor nodes monitor state of their neighbors and upon neighbors failure they try to move in the direction of failed neighbor to repair the lost coverage. Upon failure of a node X, a neighboring node B calculates for each pair of its neighbors a position  $P_i$  that is a closer intersection of the neighbors coverage radiuses. The node B can move in direction of failed

node B not further so that the distance between B and  $P_i$  remains smaller than the coverage radius. Node B checks, which among possible movements minimizes the movement distance. To further improve the energy efficiency of the algorithm it is proposed to move first the nodes with higher remaining energy level.

### 2.1.1 Energy optimization

A separate subclass of coverage preservation algorithms assumes that WSN contains oversupply of resources and aims to optimize the resource usage (e.g. duty cycling) as to conserve the sensor nodes energy, while complying with the main objective of coverage preservation.

In Tian and Georganas [2005] authors use dependency between communication and sensing range to design localized scheduling algorithm that preserves the connectivity and assures that the coverage does not degrades. Scheduling algorithm performs duty cycling of the sensor nodes for saving the resources of the WSN.

Jiang and Sung [2009] describes a distributed, localization free algorithm for providing coverage, while minimizing number of active sensor nodes in the network. It is assumed that sensor nodes are capable of sending two types of messages using two different power settings that are function of sensing range. Depending on the type of messages that the sensor node can receive from its neighbor it can estimate the distance between them. If two sensor nodes receive the messages sent at lower power setting then they conclude that they are close to each other and one of them can enter inactive state.

For certain class of applications assuring simple coverage may not be sufficient. Wang et al. [2003] considers the problem of k-coverage. Sensor nodes communicate with neighboring nodes and compare whether particular area is k-covered. If the local area is not yet k-covered then a sensor nodes decides to remain active. Otherwise, if the required coverage is already provided the sensor nodes enters sleep state in order to conserve energy. The checking of the coverage degree is done periodically in order to possibly repair the coverage.

### 2.1.2 Summary

The common theme of the coverage preservation/restoration algorithms is to use the preferably localized (but in some cases also centralized) location information to calculate the intersection of the sensor nodes sensing ranges to determine if any fragment of monitored area is out of the sensing range. What differentiates these approaches is the way they handle the selection of the locations that should be provided with new resources and in case of mobile networks, also the decision, which resource can be safely moved.

The exception to this rule constitute algorithms that use the potential forces principle, where the distance between the mobile nodes translates directly into the repulsive or attraction force that causes the nodes to move. Setting proper force-distance functions assures that the nodes will be appropriately placed as to guarantee the coverage.

The major approach to reduce the energy usage while maintaining the (k-)coverage is based on the assumption that there already exists oversupply of the sensor nodes to provide full coverage. The sensor nodes locally (e.g. within local cluster) decide about the order of the sleep/idle cycle for the sensor nodes as to assure only redundant nodes at given time are turned off. The proper schedule should result in even energy distribution and full coverage.

## 2.2 Connectivity

The connectivity is one of the most important properties of WSNs. In most of WSN applications, sensor nodes collecting the data ultimately have to communicate some portion of the (possibly processed) data to the sink. Additionally the local connectivity of the sensor nodes can have critical impact on the reliability and quality of the obtained data. Therefore, the connectivity maintenance is crucial for reliable operation of the WSN. The goal of connectivity oriented maintenance techniques is to prevent the network partitioning or enable the restoration of the connectivity in the network. These maintenance techniques include traffic rerouting in order to save the energy, active repositioning of mobile sensor nodes, pro-active re-/deployment of sensor nodes, message ferrying and communication range adjustment.

Shih et al. [2007] presents a proactive algorithm aiming at prevention of the WSN partitioning. The algorithm classifies neighbors of each sensor node depending on their distance to sink relative to distance of the considered sensor node. Those neighbor nodes placed closer to sink are called upstream nodes. Each sensor node is given task of monitoring the state of its upstream neighbors. If upstream neighbors fail the sensor node moves in direction of the sink to look for a new upstream node to maintain the connectivity. The movement calculation considers also sensing range in order to avoid creation of coverage holes.

In the disastrous scenarios the WSN may become partitioned. Introducing additional mobile nodes allows to reconnect the partitions Dini et al. [2008]. The disconnected network may be classified as safe partition or isolated partition. Upon detecting the partitioning, the sink floods network with new epoch ID. Sensors node that acquire the new ID belong to the safe partition that also includes sink. The mobile node can use this information to navigate toward the partitioning edge. The edge is detected when sensor nodes connection degree drops below certain

level. The mobile node moves further deploying behind sensor nodes. The movement continues until the mobile nodes is able to communicate with sensor nodes with ID corresponding to the epoch prior to partitioning.

Lorenz and Dini [2005] describes two alternative approaches to tackle the problem of partitioning in the network. The first approach is based on extending the transmission range of the sensor nodes until they are able to reestablish the connection with the remaining part of the network. Separated cluster selects its cluster head using as the selection criteria the shortest path to the sink from before separation. The selected sensor node step by step extends its transmission range sending request for reconnection. The second approach involves mobility on the part of disconnected cluster head, which starts ferrying the messages between two parts of the network.

Yu et al. [2007a] presents two algorithms for reconnecting sub-networks. In both cases global and centralized knowledge is assumed. First algorithm models sub-networks as vertices and searches for the minimum spanning tree connecting them. Then for vertices with the degree of  $k \geq 2$  optimize the connection by constructing triangle selecting nodes from three different convex hulls of sub-networks to reconnect. Next, the Fermat point of so constructed triangle is computed, and its location is used for deploying the additional relay node. Second approach is based on divide-and-conquer approach. Algorithm, splits the network in smaller squares and reconnects the network on smaller scale, later reintegrating them in whole network.

Also Koskinen et al. [2006] presents network reconnection algorithm based on the finding the minimum spanning tree. After discovering the spanning tree, algorithm looks for the edges in the tree, which length is greater than the communication range. In that case it places sufficient number of sensor nodes in between to reconnect the nodes at both ends of the edge. The authors furthermore study the problem in context of limited resources and propose two additional reconnecting algorithms. Second algorithm proposes to perform Voronoi tessellation and to place the relay nodes at the locations of the vertices of Voronoi polygon. Only vertices are considered, which are result of placement of sensor nodes belonging to different network partitions. The third algorithm simplifies the approach and searches only for such triplets of sensor nodes, which are not placed further apart as double distance of connection range and each node belongs to different partition. The relay nodes are placed at point equally distant from the corners of a so created triangle.

Chang and Jan [2006] also aims at finding suitable locations for relay nodes placement to keep the network graph connected. The proposed approach utilizes Delaunay triangulation. The algorithm connects sensor nodes at the vertex of triangle when each of nodes belongs to a different sub-graph. The reconnection takes place by placing a relay node at center of circumcircle only if radius of circle

is smaller than communication range. Rest of the sensor nodes connects directly to the closest node.

Zeng et al. [2010] proposes two stage approach for finding minimal connected dominating set (MCDS) and keeping the nodes belonging to MCDS active, while rest of the nodes remains in sleep mode in order to conserve energy. First a maximal independent set is found. Independent set denotes a set of vertices of graph so that there exist no edge between the vertices belonging to the graph. A minimal independent set is also a dominating set of the graph but it may be disconnected. The task of the second step of the algorithm is to find minimal number of nodes to make previously achieved dominating set connected. The construction of connected dominating set utilizes minimum spanning tree construction algorithm.

### 2.2.1 k-Connectivity

The k-connectivity is a stronger form of connectivity requirement. The k-connectivity property guarantees that the removal of up to  $k - 1$  sensor nodes will not disconnect any of the sensor nodes in the network. The goal of the k-connectivity algorithms is to add/reposition minimal/optimal number of sensor nodes to comply with the k-connectivity objective.

The Bredin et al. [2005] presents centralized approach for providing k-connectivity. Algorithm assigns between each pair of sensor nodes weight, equal to number of sensor nodes to be placed required to connect the nodes in straight line. Algorithms continues by searching an approximately minimum-weight k-vertex-connected subgraph of weighted complete graph. Afterwards, the algorithm places at each edge of this subgraph, which weight is greater than 0, clusters of k sensor nodes at locations required to connect the nodes. Additionally at each end of the edge, algorithm places additional  $k - 1$  sensor nodes. Optimized and more practical approach is also advocated. In this case algorithm iteratively, in growing order of weight, connects the edges by placing k sensor nodes at each location required to reconnect them and  $k - 1$  at the end point of the edge. After each step algorithm verifies if the graph is k-connected. When graph becomes connected the algorithm searches for resource optimizations and tries to remove redundant edges. In the second phase algorithm iteratively in decreasing order of edge weight tries to remove the connecting the edge relay nodes. If after the removal, the graph remains k-connected then relay nodes are permanently removed.

The approach presented in Pu et al. [2009] builds on the Bredin et al. [2005]. In this case authors introduce a notion of partial k-connectivity. In order to save the resources k-connectivity is provided only for initially deployed sensor nodes. The algorithm, while adding additional sensor nodes, does not guarantee that these too will achieve k-connectivity. Therefore, the sensor nodes used to provide missing links does not need to be k-stacked.

Almasaeid and Kamal [2009] aims to optimize the amount of resource required to reconnect the network and establish the  $k$ -connectivity. In the reconnection phase, the algorithm exploits the fact that there exist a significant number of intersections between communications ranges of sensor nodes in a disconnected network. Algorithm iteratively places additional nodes at intersection locations, which connect maximal number of nodes from different subgraphs. After reconnecting the network, algorithm constructs a minimal spanning tree and at each non-leaf node places additional  $k-1$  sensor nodes.

Authors of Bai et al. [2008b] and Bai et al. [2008a] aim at designing a geometrical deployment pattern that guarantees 4 and 6 connectivity and full coverage, while using minimal number of sensor nodes.

Dai and Wu [2005] presents approach for constructing  $k$ -connected  $k$ -dominating set ( $k$ CkDS). Algorithm starts with checking for each node its coverage condition. Coverage condition determines if a node is non-backbone node of  $k$ CkDS. Sensor node checks if each of its neighbor nodes pairs can communicate independently of its own existence and if any of alternative paths consists of higher priority nodes, where priority may correspond to node ID. After removing all of the non-backbone nodes, remaining nodes constitute connected dominating set (CDS). A  $k$ -CDS is defined as a CDS where each node not belonging to CDS has at least  $k$  neighbors that are members of CDS. Authors propose two probabilistic approaches to find the  $k$ -CDS by deciding the status of the node (if it belongs to the set or not) depending on the probability  $p_k$ , which depends on desired  $k$ . Also a deterministic approach is proposed where node becomes non-backbone node if for each pair of its neighbors there exists  $k$  disjoint alternative paths with ID higher than this of considered node.

Wu and Li [2008] delivers description of an algorithm for construction of the  $k$ -connected  $m$ -dominating set, which members are used for routing information over the network. First the 1-CDS is constructed using the algorithm CDS-BD-D, based on principle of nodes coloring. All the black nodes at the end of the algorithm constitute 1-CDS. In next step, the algorithm adds iteratively the white nodes (non CDS nodes) with the highest weight. The weight depends on the connectivity degree of the node. The algorithm continues until the CDS becomes  $k$ -connected. Next algorithm tries removal of each node. If after removal CDS remains  $k$ -connected then the node is removed permanently.

Atay and Bayazit [2008] states the problem of providing increased reliability differently from previous works. Authors consider problem of repairing mobile WSN and assuring  $k$ -redundancy of the sensor nodes.  $k$ -redundancy of a sensor node is defined as the minimum number of sensor node removals required to disconnect any two neighbors of that node.  $k$ -redundancy is important for the robustness of the network because as the redundancy of sensor nodes increases, the routes between neighboring nodes increases. Sensor nodes move to replace

the failing sensor nodes as to restore  $k$ -redundancy property and minimize the traveled distance.

Jorgic et al. [2007] discusses problem of localized detection of  $k$ -connectivity and proposes three algorithms. First algorithm checks if all of the nodes placed at most  $p$  hops from given sensor nodes have connectivity degree of  $k$ . The second algorithm additionally checks if  $p$  hop neighborhood is  $k$ -connected. In third algorithm the nodes check if they are critical nodes and propagate this information  $p$  hops away. If none of the neighbors in  $p$  hop distance declares itself critical, then  $p$  hop neighborhood verifies if it is 2-connected. If it is 2-connected then algorithm progress trying to find out if there are nodes that are not 3-connected.

### Topology Control

Additional class of connectivity focused maintenance strategies constitute algorithms based on the concept of topology control. Those algorithms employ their functionality of adjusting transmission power to establish connected topology as well as to optimize power usage. The power of the transmitter directly impacts the communication range and therefore the topological neighborhood of sensor nodes. The goal is to find individual power settings for each sensor node to render connected topology.

The problem of assignment of power for wireless devices to guarantee  $k$ -fault tolerance is known to be NP-hard. Hajiaghayi et al. [2003] presents set of heuristics that achieve  $O(k)$  approximation.

Bahramgiri et al. [2006] presents an algorithm for providing  $k$ -fault tolerance by adjusting the power settings of the sensor nodes. The solution is based on the assumption that the  $k$ -fault tolerance is possible at maximal power setting. First, each sensor node broadcasts Hello messages on each power setting and awaits acknowledgments from the neighboring nodes that were able to receive the message at given power setting. Then the sensor node stores the information about location of the neighboring nodes and power setting at, which they are reachable. Each node extends its power setting until the maximal angle distance between neighboring nodes is lower than  $\alpha < \frac{2\pi}{3}$ . Next, authors show the proof that this principle can be used to provide also  $k$ -fault tolerance as long as  $\alpha < \frac{2\pi}{3k}$ . Furthermore, the algorithm is extended to support 3 dimensional networks. Each nodes has to increase its transmission power until there is no 3D-cone of degree larger than  $\alpha$ .

Li and Hou [2004] presents centralized and decentralized algorithm for construction of global spanning subgraph. The centralized approach orders all possible edges in the graph by their weight (distance between the sensor nodes that the edge connects), then in ascending order algorithm evaluates if the sensor nodes connected by the edge are  $k$ -connected. If they are not connected, then the edge

under consideration is added to the constructed subgraph, otherwise if edges are already  $k$ -connected, then the algorithm verifies if all sensor nodes belong to the same  $k$ -connected component. If that is the case, then the algorithm ends. In the localized version of the algorithm, each sensor node executes the centralized algorithm but only for its visible neighborhood and based on the outcome chooses the list of proper neighbors for final topology. Last step involves adjustment to list of neighbors to ensure that all links are bi-directional.

Li et al. [2003] provides probabilistic approach to providing  $k$ -fault tolerance in the network. Authors show the analytical dependency between the transmission range of the network and desired  $k$ -fault tolerance level. Extending the transmission range to a required degree can ensure with high probability that network is  $k$ -connected. Furthermore, a distributed algorithm based on Yao graph is presented to provide  $k$ -fault tolerance. Each sensor node defines  $p$  any equal-separated rays originating at its location, thus defining  $p$  equal cones, where  $p > 6$ . Then in each cone, sensor node chooses the  $k + 1$  closest nodes in that cone, if there is any, and adds directed links from  $u$  to these nodes. The authors provide proof that such structure sustains  $k$  node faults.

Saha et al. [2007] proposes a distributed algorithm for providing  $k$ -connectivity in the network. The sensor nodes collect the location and the information about maximum power setting from their surrounding nodes. Next, each sensor node selects power adjustment for the nodes in its close vicinity so that all the nodes in that vicinity have  $k$ -disjoint paths. Authors show proof that fulfillment of this condition is sufficient to provide  $k$ -connectivity. The concept is also extended for the network where the nodes are movable.

### 2.2.2 Summary

The ultimate goal of each of the presented algorithm is to reestablish the connectivity or prevent occurrence of partitioning. In most cases the solution leads to finding the sensor nodes placement, where new additional nodes should be deployed, or some of the mobile nodes (if present in the network) to be repositioned. Preferable solution should be based on a distributed algorithm, especially in case of mobile network, so the WSN could make autonomically decision and lower the communication cost. However in some case, the centralized solution can also render benefits, despite its possibly higher cost, by providing optimized deployment location that requires less resources for deployment.

In case of  $k$ -connectivity algorithms there exist two distinctive approaches. The first approach is based on the assumption that the network already poses enough resources that allow for the  $k$ -connectivity. The task of the algorithm in this case is to find (under given constraints) the minimal set of sensor nodes that is  $k$ -connected forming the topology backbone, and all of the remaining sen-

sensor nodes are connected directly to at least one of the nodes belonging to the backbone. The second approach assumes that the network is missing necessary resources to provide  $k$ -connectivity or even the network initially is disconnected. The task is to establish new or/and reinforce existing connectivity graph edges with stocking possibly minimal number of additional sensor nodes that new topology results in  $k$ -connected graph. In both cases the presented approaches use at their core the similar strategy, e.g. their search for the minimal connected dominating set (first approach) of existing topology graph or minimal spanning tree (second approach) of the graph where edges possibly exists between every sensor node pair.

The  $k$ -connectivity maintenance can be also exercised by the topology control mechanism. In this case the transmission power of the sensor nodes is locally adjusted to fulfill requirement of some other  $k$ -connectivity technique, i.e.: maximal angle distance between neighboring nodes, local connectivity degree or existence of disjoint paths.

## 2.3 Sampling

The goal of large class of WSN applications is the monitoring of the distribution of signal within area of interest or the detection of events. The efficiency of both of these task depends on the quality of sampling. The quality of sampling directly translates in the task of choosing sampling locations that most effectively reduce the entropy (variance) of the monitored signal. The sampling locations can then be occupied by static sensor nodes or traversed and sampled by mobile nodes. The major optimization criteria include minimization of required resources (sampling locations) or reducing the area of interest traversal path length.

Krause et al. [2006] proposes two (or more) stage deployment schema. During the initial deployment, sensor nodes collect the coarse data about the phenomenon distribution and the initial connectivity. Afterwards, the collected data are used for modeling the phenomenon using Gaussian processes. A covariance matrix is created, to find out the most informative spots for placing additional sensor nodes. The algorithm can progress iteratively through additional stages to improve the sampling. The algorithm requires centralized knowledge of the sensor nodes position and measurement values. These costly requirements, limit the scalability of the proposed solution for large networks and dynamic environments.

The spatial phenomena under monitoring are often modeled as Gaussian Processes (GPs). For WSN where the number of sensor is limited choosing sensor locations is a fundamental task. A common strategy is to iteratively place sensors at the points of highest entropy (variance) in the GP model. After deployment of each sensor the entropy of the model is recalculated and new sensor nodes are

placed at current maximum of the entropy for given iteration step. The disadvantage of this approach is the inefficiency that the algorithm does not consider the fact that placing the sensor node close to the border of the deployment area may waste significant portion of sensing coverage. Guestrin et al. [2005] proposes a mutual information criteria to avoid exactly such situations. Furthermore, it is proved that finding the configuration that maximizes mutual information is NP-complete.

Approach of iterative placement of sensors at the points of highest entropy Popa, D.O. and Mysorewala, M.F. and Lewis, F.L. [2006] can also be utilized for WSN deployment, where mobile sensors have a task of mapping the monitored phenomenon. The mobile sensors use the data collected when visiting sensing locations to establish the Kalman-Filter covariance matrix. Then the covariance matrix is used to derive the most informative points to visit in the next iteration step.

Zhang and Sukhatme [2007] considers more sophisticated scenario involving mobile robot and multiple statically deployed sensor nodes. The algorithm accepts the measurements made by the static nodes as inputs and computes a path for the mobile robot. The path minimizes the integrated mean square error of the reconstructed field subject to the constraint that the robot has limited energy. The proposed adaptive sampling algorithm is based on local linear regression and is guaranteed to be optimal in the sense of minimizing the integrated mean square error of the field reconstruction.

Rahimi et al. [2004], Rahimi et al. [2005] present mapping strategy for monitoring physical phenomena, where the system model assumes that the mobile nodes hang between the trees and are capable of vertical and horizontal movement. The sampling space is divided in the sampling points called pixels. The task of algorithm is to minimize number of sampled pixels while maintaining sampling accuracy. Authors developed a Nested Stratified Sampling method for NIMS as a "divide and conquer" algorithm that defines a sampling "tree" where individual strata occupy leaves on this tree. The algorithm uses first low number of strata's and evaluates the variance within them. If variance is above given accuracy, the strata are further stratified.

Popa et al. [2004] considers a scenario, which includes the set of static sensors providing network coverage and substantially smaller set of mobile sensors equipped with enhanced sensing capability to provide more advanced analysis of previously detected phenomena. Static sensor nodes have task of detecting user defined events and then request presence of mobile sensor nodes to increase the sampling resolution of the event. The proposed algorithm handles the problem of assigning the mobile nodes to observe the detected events. If number of events is lower than number of mobile nodes then algorithm assigns pairs of events and mobile nodes that the expenditure of energy between the mobile nodes is balanced. In

case when the number of detected events is higher than number of mobile nodes, then events are first clustered so that number of clusters equals number of mobile nodes. The central and distributed approach are proposed. In distributed approach network is divided using grid into cells and cells in rows and columns exchange information about availability of mobile nodes and request for events inspection.

Approach presented in Butler and Rus [2003] considers system model involving a network, consisting only of mobile nodes. The task of the mobile nodes is to mimic the distribution of the events in the network. The basic algorithm uses the virtual force approach where mobile nodes are attracted by the event and move to it in turns. The distance traveled in each turn depends on the current distance from the event. The mobile node can only get closer to the event but cannot go pass it. The efficiency of algorithm is improved by creating the history of events using cumulative distribution function (CDF) and using this knowledge to better plan the movement. In order to maintain the coverage three strategies are proposed. Mobile nodes make the list of neighbors ordered by the angle they create with the mobile node and if the gap of size  $\pi$  is detected, then the mobile node stops. The two other strategies use concept of Voronoi polygons. First strategy assumes global knowledge and uses prediction of other mobile nodes movement. The prediction is based on the fact that algorithm knows all original position of mobile nodes and all of them are using the same movement algorithm. Optimization involves only using the information on the original direct neighbors assuming that they remain in close distance.

Huguenin and Rendas [2007] discusses the case of fleet of mobile robots where clustering is used to collect the local measurements and calculate the model of a local phenomenon. The model is used to determine the new sampling locations. The mobile nodes move to the assigned positions and the process is repeated. As the decision is taken locally it is likely not optimal and requires an iterative multi-step movement.

Similar approach is presented in Zhang and Sukhat [2005]. Proposed algorithms have a task of adjusting the mobile nodes distribution to the distribution of a scalar field. The network is clustered using LEACH protocol and each cluster head collects measurement data from its children. Cluster head evaluates the data, reconstructs local distribution of the phenomena and assigns to the mobile nodes new positions.

Sukhatme and Sukhatme [2004] proposes a solution for the problem of the one-dimensional adaptive spatial sampling for water temperature. The goal of mobile sensor nodes placed on a line is to find the depth at which water temperature changes rapidly. Each mobile node is given a section to measure the variance of measurements at its ends. If the variance is above a specified threshold, the section is divided and the sub-sections are further evaluated.

### 2.3.1 Resource optimization in case of over-sampling

A separate subclass of adaptive sampling algorithms assumes that WSN already contains more than adequate number of sampling locations. The task is therefore reversed. The algorithms try to identify the redundant sensor locations and to put sensor nodes that correspond to these locations in idle mode, while maintaining required sampling fidelity.

The basic approach to minimize resources required to provide requested sampling resolution Willett et al. [2004] is to collect the data from a small subset of sensors nodes at the sink. Then, this information is used to estimate the environment conditions of the monitored area. On the basis of so created model, the sink selects the regions where additional sensor nodes should be activated.

In the more sophisticated approach, the decision about sleeping/activating sensor nodes is distributed over the network Arici and Altunbasak [2004]. In particular, sensor data is periodically collected at powerful macronodes. The macronodes fit a low-order model to compute a prediction area for each sensor node. This prediction area is used to measure the redundancy level of the sensor nodes covered by the prediction area. The redundant sensor nodes are put into passive mode by macronodes while keeping the distortion of the measurement low.

The main goal of algorithm proposed in Gedik et al. [2007] is too minimize number of resources required to provide the requested sampling resolution. First, sensing-driven cluster construction is used to create clusters within the network such that sensor nodes with similar sensor readings are assigned to the same clusters. Second, correlation-based sampler selection and model derivation are used to determine the sampler nodes and to calculate the parameters of the probabilistic models that capture the spatial and temporal correlations among the sensor readings. In the last step the sensor nodes use the prediction models of the neighboring sensor nodes and if they are successful, the modeled nodes are put in the sleep mode to save energy.

### 2.3.2 Summary

The presented sampling optimization algorithms have a task of proposing a new sampling locations, either for deployment of additional sensor nodes or for navigating mobile nodes, so that the measurement taken at new locations at most reduce the entropy of the monitored signal. The common approach involves iterative estimation of the current distribution of the monitored signal using Gaussian distribution and then selecting the location of highest entropy. The new sensor node is deployed in this location and the approach is repeated until desired level of accuracy is achieved. In case of mobile network, the mobile node navigates to so discovered point and after performing sensing repeats the process of se-

lecting new sensing location. Main drawback of such approaches is the global information requirement. Alternative distributed solutions instead of searching for location of the highest entropy try adapt the distribution of the sampling points to monitored signal distribution using information within for this purpose formed clusters.

There exists also other class of sampling optimization algorithms, which are not focused on bringing the sampling accuracy to requested level, but assume that it is already provided and their goal instead is conservation of the energy. The common approach tries to create the estimation of the sampled signal. Then either the sink centrally or the sensor nodes locally (e.g. forming cluster) devise a scheduling plan for duty cycling the sensor nodes as to maintain the sampling resolution, while maximizing energy savings.

## 2.4 Holes problem in WSN

Holes problem is to some degree similar to connectivity problem. In case of holes problem the objective of algorithm is to take into consideration a high level impact of distribution of sensor nodes properties and processes taking part in the network. The existence of a hole does not necessarily indicates connectivity problems but may be a cause for serious problems for different classes of applications.

The survey Ahmed et al. [2005] defines several classes of holes, e.g., coverage holes, routing holes, jamming holes and sink/black/worm-holes. The coverage (sensing) holes are concerned with loss of coverage, routing (communication) holes describe problems with the network topology, which cause the routing take place along much longer paths than would be necessary without presence of the hole. The sink/black/worm-holes are virtual type of holes (e.g. induced by security attacks) and are not directly relevant to maintenance problem. The jamming holes can be seen as special case of coverage holes, where the hole profiling is inherently impossible.

### 2.4.1 Energy Holes

Energy holes represent a special case of hole problem. An energy hole can be seen as a spatially correlated region of the network where energy drain is higher than average and therefore the sensor nodes within the region show much higher energy usage and as consequence much shorter lifetime.

The most common source of the energy holes is uneven distribution of traffic in the wireless sensor network. Particular case is the traffic pattern imposed by the central placement of the sink Wu and Chen [2006]. The sensor nodes closer to the sink, besides transmitting locally generated traffic, must also forward the traffic

originating at further parts of the network. Therefore, their energy is deployed much faster creating energy hole around the sink. In order to alleviate the problem, sensor nodes need to be deployed with correspondingly higher concentration of around to the sink. The proper distribution can be derived using analytical approach, which can calculate expected traffic between sensor nodes. The adjusted deployment needs to be supplemented with the proper algorithm, which can balance the usage of additional (as compared to typical deployment) nodes.

Li and Mohapatra [2005] proposes alternative solution. First, the paper delivers analytical description of the energy hole building up around centrally located sink. This is imposed by many to one communication pattern where sensor nodes placed in the "rings" closer to sink have to forward the data generated in further located rings. It is showed that increased density of the sensor nodes does not necessarily solves the problem. Instead of increasing density of sensor nodes around the sink, authors postulate hierarchal deployment schema and traffic compression.

Olariu and Stojmenoviæ [2006] also analyzes the energy hole problem around the sink. As solution authors propose approach that takes advantage of capability of sensor nodes to adjust their transmission power. Sensor nodes modify their transmission range depending on their distance to sink. Sensor nodes further from the network are expected to increase the transmission power as to reduce the number of hops to the sink, with goal of balancing the energy usage.

Liu et al. [2008] also provides analytical description of the energy hole problem around centrally placed sink. Authors propose changing mixture of previous strategies. First strategy adjusts the density of the sensor nodes deployment depending on their distance from the sink. The redundant sensor nodes remain in sleep state to preserve the energy, and later replace those of sensor nodes, which already depleted theirs battery. Second strategy increases the transmission power of the sensor nodes occupying locations further away from the sink.

Also Maleki and Pedram [2005] considers the problem of the energy hole around sink. The authors propose to tackle the problem at the deployment stage. The sensor nodes should be placed in such a manner that the energy density depends on the sensor nodes placement in the network. The algorithm allows for analytic calculation of the required density to fulfill stated lifetime and Quality of Monitoring requirements. Generally, the higher density of the energy (sensor nodes) should be concentrated in neighborhood of centrally placed sink.

### **2.4.2 Topology Holes**

Topology holes are none routeable regions of the WSN. In the analogy to coverage holes they can be defined as regions, within the deployment area that are not covered by the communication range of any of the deployed sensor nodes. They discovery is very important for providing efficient routing within the WSNs.

The algorithm presented in Yu et al. [2007b] provides a geographic routing algorithm, which uses the topological information to route around topology holes. For that purpose the algorithm upon detecting a boundary of topology hole starts its modeling. At the sensor node, at which routing stopped, the algorithm starts hole boundary detection sending messages along the boundary of the hole using the well-known right hand rule. The boundary nodes forward the message (adding its own location) until it is received by initiating node. The data collected in the message are used to model ellipse enclosing all of the boundary node. Each sensor node within the ellipse, while routing, is treated as non-existent to assure that the geographic routing algorithm will not enter this zone. Such routing reduces energy depletion in border nodes and reduces amount of collisions.

In the Fang et al. [2006] authors use Delaunay triangulation to detect the hole boundaries. Furthermore, the routing geographic routing algorithm uses this information to continue forwarding message, when it arrives at sensor nodes, which has no immediate neighbors placed closer to the routing destination. The routing algorithm, forwards the message along the boundary until it can continue using geographic information.

The algorithm for detecting the boundary nodes presented in Deogun et al. [2005] makes an assumption that every sensor node deployed in the network has connectivity degree of at least three. Algorithm then iterates through all possible triplets of the neighbors and calculates whether each sensor node is placed within the triangle created by the triplet using the Heron equation. If the given sensor nodes is not placed in any such triplet, then it is a boundary node.

Much more sophisticated boundary detection technique (applicable for detecting inner and outward network boundaries) is presented in Wang et al. [2006b]. The proposed algorithm starts by flooding the network in order to create shortest path tree. Then the algorithm detects nodes, which form a cut (nodes at which the tree merges after passing by the obstacle and share the common ancestor, which is placed  $k$ -nodes away. Algorithm joins the branches of multiple cuts and as a result a single cycle encompassing all holes remains in the network. Then using flooding and finding local hop maxima from the cycle, the boundaries are refined. Removed branches are restored and locally refined.

In Srinivasan et al. [2008] authors propose use of mobile nodes to discover the region contour within the monitored area. In order to optimize the algorithm mobile sensors use the gradient information to select the right direction for approaching the edge of the region. Algorithm also uses this information to spread mobile nodes at right time so they start form different fragments of the contour. After reaching the edge the mobile nodes move around the contour until they detect that given fragment was already scanned by other mobile node.

### 2.4.3 Summary

In case of energy holes, they are the result of uneven energy depletion in WSN. The generally prescribed solution is to deploy the additional sensor nodes with density that corresponds to the energy dissipation density pattern. The mostly considered case of the energy hole occurrence is the situation when sensor nodes closer to sink deplete additional energy, as they also have to forward the messages from further parts of the network. Authors of numerous solutions show that the required re-/deployment density can be analytically determined knowing the communication range of the sensor nodes.

The common approach for handling the topological holes is for the algorithm to detect the location, shape and size of the hole and provide this information to allow safe by-passing of the affected regions of the network. The algorithms for the topology hole shape discovery include Delaunay triangulation, looking for cycles, and use of gradient information about the monitored phenomenon.

## 2.5 Topology Optimization

The topology optimization techniques represent set of stricter requirements on the topology than previously discussed connectivity requirement.

Jorgiæ et al. [2004] introduces localized definitions of critical nodes and critical links, using topological or localization information. A node is critical if the subgraph of  $k$ -hop neighbors of sensor node (without the sensor node itself) is disconnected. Three definitions of critical links are proposed, based on verifying common  $k$ -hop neighbors, loop length, and critical status of link endpoints, respectively.

Gou and Yoo [2010] proposes algorithm for finding bottleneck nodes in distributed manner. Each sensor node checks if each of its neighbors can connect to at least one other neighbor, if not then the given sensor node becomes candidate node as its only intermediary for at least one of its neighbors. In order to verify its status it sends the message to isolated node but by addressing only rest of the sensor nodes, if message arrives then there exists alternative path.

Ghosh and Boyd [2006] makes step forward, not only identifying topology problems, but also offers remedies. The presented algorithm proposes deployment of relay nodes for improving the connectivity of the network based on maximization of Fiedler value of Laplacian matrix of graph connectivity. Algorithm shows how, by maximizing the Fiedler value, to reconnect the partitioned network but the centralized knowledge about the topology of separated parts has to be provided.

### 2.5.1 Energy oriented

Wang and Ding [2008] targets optimization of the network topology, but also takes under the consideration network energy dissipation patterns. System model assumes clustered network and the monitored phenomenon with the non-uniform probability function. Centralized algorithm uses these assumptions to build energy usage model of the network and simulates energy distribution. Based on the simulation results algorithm predicts which clusters will drain the energy at most. Sensor nodes are redeployed at clusters where they will at most extend life-time of the network.

Li and Cassandras [2005] proposes iterative self-deployment scheme, where new sensor nodes are added in order to optimize the direct distances between neighboring sensor nodes. Such placements of additional sensor nodes aim at reducing power necessary for transmissions between neighboring nodes. First bottleneck nodes defined as sensor node with highest sparsness factor are found. Next, all possible new positions for placement of new sensor nodes are considered and finally a sensor node is placed in the location that is at most minimizing the sparsness.

### 2.5.2 Relay nodes

The topology optimization techniques can also be classified based on the technique that they employ. There exists a family of techniques utilizing relay nodes for optimizing topology characteristics. Relay nodes are additional nodes introduced into WSN for aiding the communication in the network, in most cases they are not able to perform sensing. The relay nodes can have the same or superior communication capabilities.

Ibrahim et al. [2007] proposes the algorithm for deployment of relay nodes for improving the connectivity of the network. Similarly to the idea presented in Ghosh and Boyd [2006], the proposed approach is based on maximization of Fiedler value of Laplacian matrix of graph connectivity.

Han et al. [2010] proposes a centralized algorithm for reconnecting nodes and assuring k-connectivity. Algorithm adds the relay nodes from lowest weight (number of nodes needed to connect selected nodes) until network gets k-connected, then it tries removing nodes that do not breach k-connectivity. The algorithm is similar to Bredin et al. [2005] and Pu et al. [2009] but extends the concept to heterogeneous networks taking advantage of extended range of relay nodes.

Misra et al. [2008] discusses a problem of placement of the relay nodes, where they possible deployment is constrained to preselected locations. Algorithm first constructs the graph that includes all the locations of sensor nodes, base station(s)

and potential relay nodes deployment depending on the communication range of sensor and relay nodes. Next, for each sensor node the algorithm calculates positive weight that equals to the number of relay nodes locations that the sensor node is incident with. Finally, algorithm computes a low weight tree subgraph that spans all the sensor nodes and base station(s). The locations of the relay nodes placements that are also included in the spanning tree are selected for the deployment.

In Srinivas et al. [2009] authors present the problem of maintaining the backbone of the WSN that consist of set of mobile relay nodes. The proposed algorithm splits the network horizontally into the strips of the width equal to the fraction of the diameter of the communication disk. Next, within each strip the algorithm places sequentially the mobile relay nodes so that they cover all the sensor nodes. The placement starts from the leftmost uncovered sensor node. The relay node is placed so that the uncovered sensor node is placed on the edge of the communication range of the relay node. Next, the algorithm looks for a new leftmost uncovered node and repeats until all sensor nodes in each strip are covered. If the sensor nodes are mobile or get failing, then the backbone node can move in 1-D plane to merge their coverage areas or move apart to follow the sensor nodes.

Lloyd and Xue [2007] presents solutions to two relay placement problems. First problem is defined as finding the placement of relay nodes so that the WSN is connected using sensor and relay nodes. Authors propose to find the minimum spanning tree of the graph, where the weight of the edges corresponds to the distance between the nodes connected by the edge. The relay nodes are deployed on the edges of the spanning tree, which distance is greater than the communication range, as to reconnect the sensor nodes. The second problem requires that WSN is connected using as a network backbone using exclusively relay nodes. In this case the algorithm first solves the minimum geometric disk cover problem for the deployed sensor nodes. Then for the set of locations  $C$  obtained in the first step, the algorithm constructs a set of locations  $D$ , such that for each  $c_i \in C$ , there is exactly single  $d_j \in D$ , where distance between  $c_i$  and  $d_j$  is smaller or equal to the communication range and for each  $d_j \in D$ , there is exactly single  $c_i \in C$ , where distance between  $d_j$  and  $c_i$  is smaller or equal to communication range. Next the first algorithm is applied to the set  $D$ , while communication range between members of set  $D$  is assumed to be that of relay nodes. The relay nodes should be placed on the locations belonging both to the sets  $C$  and  $D$ , and output of the first algorithm.

### 2.5.3 Summary

The focus of topology optimization algorithms is to detect topological features of the graph that could severely impact the reliability of the network. The main such

feature is existence of critical/bottleneck nodes. Their detection can be localized by analyzing nearest neighborhood connectivity, or centralized by analyzing the connectivity matrix of the graph. In case of energy oriented approaches, the proposed solutions take also into the account the impact that the energy distribution in the network has on the network's reliability. Although topologically network is well connected, still if the energy of the sensor nodes is accounted it may lead to discovery of bottlenecks in energy domain resulting in faster than average energy depletion. The alternative solution for optimizing the network topology is deployment of the relay nodes, whose sole task is to aid the message transfer within network, as to lower the energy burden on the sensor nodes that results from forwarding messages. In most cases the relay nodes are capable of transmitting the data on greater distances and are equipped with larger energy source.

## **2.6 Chapter Summary**

This chapter presented the overview of major maintenance problems in WSN and the state of the art maintenance techniques. The presented maintenance techniques were categorized in the classes depending on the functionality and employed approach. The most representative works for each class were described giving general insights in the nature of the problem and details of the proposed solution. Each of the presented problem classes poses serious treat for the reliability and functionality of the WSN and most of presented solutions still require certain compromises. Therefore there exists urgent need for further sophisticated and more comprehensive approaches. Next chapter outlines the techniques that offer substantial improvements to the classes of problems presented in this chapter.



# Chapter 3

## Localized Energy Hole Profiling in Wireless Sensor Networks

### 3.1 Overview

Wireless Sensor Networks display non-uniform energy usage distribution. This is mainly induced by the sink centric traffic or by non-uniform distribution of sensing activities and manifests as *energy holes* throughout the WSN. Energy holes can threaten the availability of the WSN by network partitioning and sensing voids. They are hard to predict, and consequently, proper function of the network requires systematic maintenance. Unfortunately, existing approaches do not systematically profile holes and focus only on very specific type of holes. This chapter presents new distributed energy profiling algorithms for generalized types of energy holes. The algorithms search for boundary nodes and use them as a reference to calculate the energy needs of sensor nodes within the energy hole. These, when aggregated, create angular and radial energy profiles.

#### 3.1.1 Contributions

Localized Energy Hole Profiling (LEHP):

- is based on an efficient, balanced and accurate profiling of the energy needs within the hole;
- uses an in-network aggregation strategy that provides for an angular and radial profiling resulting in a compact profile;
- is proactive allowing for prevention and proactive maintenance, and may be easily combined with existing maintenance strategies;

- applies for generalized energy holes with different sizes, shapes and energy spatial distributions.
- optionally does not require position knowledge from sensor nodes.

Extensive simulations show that the algorithms, when used for WSN maintenance, significantly help to extend the lifetime of the network.

The remainder of the chapter is structured as follows. Section 3.2 describes the system model. Section 3.3 details the LEHP profiling algorithm, the chapter's main contribution. The evaluation the efficiency and accuracy of the algorithm is presented in Section 3.4.

## 3.2 System Model

Similar to models employed in literature concerned energy holes problem Li and Mohapatra [2005]; Liu et al. [2008]; Olariu and Stojmenoviæ [2006]; Wu and Chen [2006], this work assumes a WSN to consist of a large number of static resource constrained sensor nodes and a sink. Their communication range  $r$  is limited, and two neighboring sensor nodes can communicate only if the Euclidean distance between them is smaller than  $r$ . The sensor nodes are battery powered and dominant energy consumption source is wireless communication (amount of energy used is function of number of transmitted messages). The only considered cause of sensor nodes failure is their energy depletion. The distribution of the sensor nodes in the deployment area follows the uniform random distribution. The sensor nodes know their position using on board GPS receivers or alternative measure of localization He et al. [2005]. An alternative approach also exist to handle profiling without means of localization though at the cost of accuracy.

The sensor nodes objective is defined as monitoring of the physical phenomena localized within the deployment area. Sensor nodes can increase their sampling rate, operate longer in active mode, and transmit more data reflecting the dynamics of the phenomena being monitored. This implies that sensor nodes placed in the region of high activity of phenomena consume energy at a higher rate. We assume the spatial correlation of physical phenomena and consider the following three spatial distributions: Exponential, Pareto and Normal. Accordingly, sensor nodes perceive the phenomenon and generate a message with the following probability  $p = f(d)$ , where  $d$  is distance to the center of the phenomena (Hotspot model Zhao et al. [2002]). The non-uniform distribution of physical phenomena manifests as a non-uniform energy depletion among sensor nodes and leads to the emergence of energy holes. We assume a local extremum (energy minimum) close to the epicenter of the hole. This local minimum is surrounded by irregular rings of energy levels (Figure 3.1, darker shades denote lower energy levels).

### 3.3 LEHP: Localized Energy Hole Profiling

In order to avoid or delay the drops in sensing and communication coverage, the energy holes should be systematically profiled by collecting information about the hole's shape and energy distribution. The collected profile can be used to perform systematic maintenance tasks. This section presents novel algorithms for the energy hole profiling.

#### 3.3.1 Overview of LEHP Approach

The main objective of LEHP is to create an accurate profile of the necessary energy to be injected into a given WSN energy hole in order to maximize the network lifetime. The goal of presented approach/contribution is to determine the energy needed to distribute over the energy hole to prevent hole reoccurrence during the lifetime of the rest of the network. LEHP consists of two phases. First, an initiator node  $I$  from the interior of energy hole starts a discovery phase to identify the reference energy level on the energy hole's border and to build a spanning tree rooted at  $I$ . Second, through efficient in-network and tree-based aggregation, the profile of the energy needs within the hole is collected at  $I$ . The sensor nodes located closest to the epicenter of monitored phenomena are usually the first to reach a given energy threshold  $E_{TH}$  (Alg. 1 lines: 2-5). The value of  $E_{TH}$  can be set arbitrarily but more desirable is to determine it based on the energy depletion rate of sensor nodes as well as the required time to perform, derived from the obtained profile, pro-active maintenance. The first sensor node to approach the threshold triggers profiling activity. Details on selection of initiator node are given in later subsection.

In order to provide highly accurate profiling, an aggregation technique was developed that models the energy hole as sectors and concentric rings around the local extremum. Accordingly, the energy levels are aggregated at quantified distances ( $h_i$ , for  $0 \leq i \leq q$ ) (Fig 3.1). As energy redeployment requires not only the distribution of needed energy, but also the values representing its amount, therefore, it cannot limit the reports to simple contours of rings as in Solis and Obraczka [2005]. Also, as the energy hole often shows irregular shapes (shading represents the energy distribution), relying only on distance aggregation may prove inaccurate. In order to alleviate this problem the reports are aggregated separately at different angles ( $\alpha_j$ , for  $0 \leq j \leq m$ , where  $m$  is equal to connectivity degree) as shown in Fig 3.1. The radial positions ( $\alpha_i$ ) are defined by the positions of 1-hop neighbors of  $I$  as they are the roots of  $I$  subtrees. The width of the arcs ( $\widehat{\beta_{j-1,j}\beta_{j,j+1}}$ ) corresponds to the minimal and maximal angles of sensor nodes belonging to the subtree. This approach allows for accurate identification of the energy needs for different positions in the energy hole.

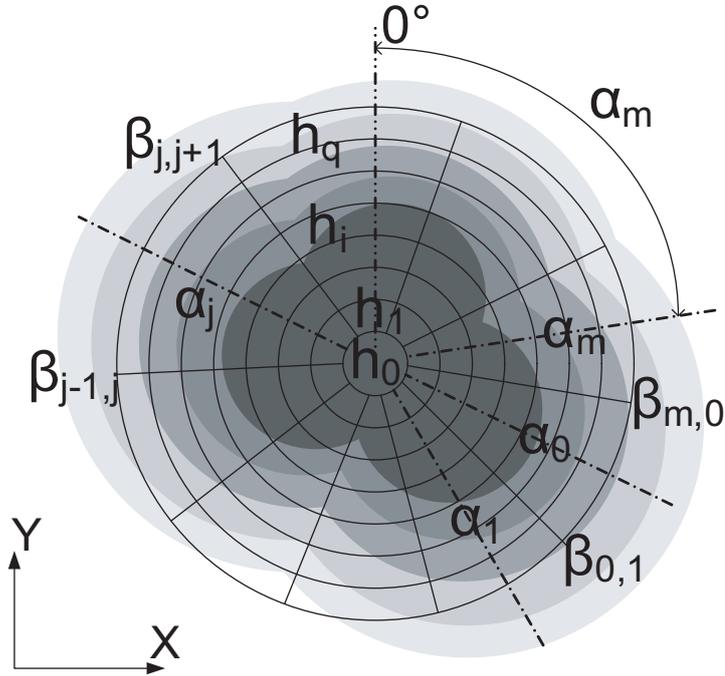


Figure 3.1: Angular and radial profiling

### 3.3.2 Profiling

In the following section details the two operational phases as well as some optimizations of LEHP.

#### Phase 1 of LEHP: Detecting Energy Hole Border and Establishing the Aggregation Tree

This phase is inspired by route discovery Mahfoudh and Minet [2008] and introduces new optimizations to increase the accuracy of angular profiling. The initiator  $I$ , located close to the energy hole epicenter, propagates a request (including its position and a unique profiling ID) for the spanning tree construction by traversing the areas of different energy levels (Figure 3.2(a)) and counting the number of hops traveled. As detailed in Alg. 1, the sensor nodes upon receiving the request message check whether the entry in the profiling-specific routing table for profiling ID exists (Alg. 1 line: 9). If not, a new entry is created storing the ID of the parent node (the address of immediate sender of the request message), the hop distance to  $I$  (that the message traveled so far) and the position of  $I$ . In case that the entry already exists, its content is evaluated against current message. If the hop distance traveled by the message is greater than contained in the entry,

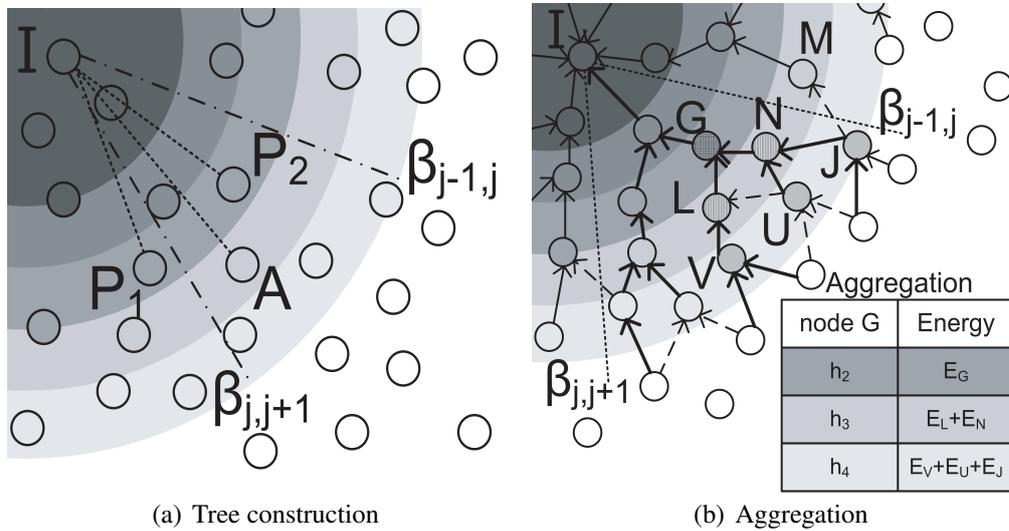


Figure 3.2: Operational phases of LEHP

then the message is discarded (i.e., a route shorter was already propagated) and not forwarded (Alg. 1 lines: 11 and 16). When the hop distance of the message is smaller, the message information replaces information stored in the entry and the message is forwarded (Alg. 1 lines: 14 and 22). If the hop distances indicated by the message and table entry are equal, further evaluation takes place as detailed in the following.

**Algorithm 1** Hole Discovery & Aggregation Tree Construction

---

```

1: On Energy Change
2: if Energy <  $E_{TH}$  then
3:   Node becomes Initiator;
4:   send(REQ, MAC_BROADCAST);
5: end if
6:
7: On Receiving a Profiling Request REQ
8: var static routingTable;
9: if existsEntry(REQ.profileID) then
10:  entry = extractEntryFromTable(REQ.profileID);
11:  if entry.Hop > REQ.Hop then
12:    replaceEntry(REQ.profileID, REQ);
13:  else if entry.Hop = REQ.Hop && entry.angle > REQ.angle then
14:    replaceEntry(REQ.profileID, REQ);
15:  else
16:    return;
17:  end if
18: else
19:  addEntry(REQ);
20: end if
21: REQ.Hop++;
22: send(REQ, MAC_BROADCAST);

```

---

In order to support the desired angular profiling, a sensor node should select its parent node in the same sector, i.e., belonging to a proper arc  $\widehat{\beta_{j-1,j}\beta_{j,j+1}}$  as depicted in Figure 3.1 and Fig 3.2. A Node  $A$  (Figure 3.2(a)) selects the closest parent ( $P_i$ ) in the angular distance, to minimize the angle  $\angle AIP_i$ . Assuming that the current parent of the Node  $A$  is  $P_1$  and the currently processed message was sent by  $P_2$ , then if the  $\angle AIP_1$  is larger than  $\angle AIP_2$  the entry is replaced (Alg. 1 lines: 13-14). This indirectly increases the chance that the sensor node will belong to the branch of subtree representing the arc to which its child nodes geometrically belong. Node  $A$  does not forward a request message as the hop distance does not change.

In order to limit message flooding within the energy hole area, two approaches are proposed. The first approach is tailored to the energy hole shape. The message should propagate until the variance of energy values of sensor nodes that message traverses drops under a certain threshold ( $TH_{break}$ ). For that purpose each message carries the energy values of the two last traversed sensor nodes. When both conditions in Eq. (4.5) and Eq. (4.6) are satisfied then the break condition is met.

$$\frac{E_{hop-1}}{E_{hop-2}} < TH_{break} \quad (3.1)$$

$$\frac{E_{current}}{E_{hop-1}} < TH_{break} \quad (3.2)$$

It is necessary to compare the energy from at least two preceding hops to avoid a situation where a sensor node selects a sensor placed next to it as a parent, which, as consequence of its closeness, has similar energy value and therefore, variance can easily fall below the defined threshold.

In the second approach, the maximal number of hops for the message to travel or the maximal distance from  $I$  is set to a maximal value, called Time-To-Live (TTL). When relaying only on TTL value, the boundary of the energy hole can be approximated from the final profile. It should be noted that some of sensor nodes marked as border nodes using TTL will not be the real border nodes of the energy hole. This approach is used to limit the profiling to energy holes of size that can be handled by maintenance (e.g. under maintenance cost criteria). The TTL value can be set at pre-deployment stage. *Border nodes* (BN) are defined as those sensor nodes that, upon receiving a broadcast message, detect one of the break conditions.

### Phase 2 of LEHP: Aggregation-based Profiling

A naive profiling approach would have each sensor node send its individual measurement and position to  $I$ . Though this approach provides high accuracy profiling, it also causes high traffic in particular on the sensor nodes closer to  $I$  which are more threatened by battery crash. Furthermore,  $I$  would have to process the entire aggregation on raw data, which can become complex for large energy holes. Accordingly, the following in-network profiling strategy was developed to realize the second phase of LEHP. The profiling technique needs to aggregate the data while retaining information about its distribution.

The approach builds on tree-based aggregation Madden et al. [2002] and uses the constructed spanning tree to implement a new efficient technique for accurate in-network profiling of the needed energy. The energy levels of border nodes serve as a reference energy to quantify the energy needed for the energy hole. Border nodes initiate the aggregation phase by sending their lifetime reference value towards  $I$  along the spanning tree. Border nodes estimate their lifetime reference using a simple linear regressive model (Eq. (6.1)), where  $E_{initial}$  is the initial energy of the sensor node,  $T_{elapsed}$  the time passed since deployment and  $E_{current}$  the current energy level of the sensor node. If needed, this model can be replaced by a more accurate one Chiasserini and Garetto [2004].

**Algorithm 2** Aggregation-based Profiling

---

```

1: On receiving a Reply Message REP
2: var static P_REP;
3: const T;
4: evaluationTime = now() + T;
5: if REP.DestID == this.ID then
6:   for i = 0; i < maxNumberOfEntries; i++ do
7:     P_REP.Energy[i] += REP.Energy[i];
8:     P_REP.Count [i] += REP.Count[i];
9:   end for
10: else if entry.Hop > REP.HopToTravel then
11:   return;
12: end if
13: P_REP.Ref = recalculateReference(P_REP.Ref, REP.Ref);
14:
15: On Data Evaluation Event Handler
16: entry = extractEntryFromTable(REP.profileID);
17: energyNeeded = estimateEnergyNeeded(P_REP.Ref);
18: P_REP.Energy[entry.Hop] += energyNeeded;
19: P_REP.Count[entry.Hop]++;
20: send(P_REP, entry.ParentID);

```

---

As outlined in Alg. 2, sensor nodes collect and process the energy information received from their child nodes. First, the received partial profiles are grouped based on their hop distance to  $I$ . Next, aggregation is applied separately to each group. Aggregation within LEHP sums up the energy need for each sensor node within a group (Alg. 2 lines: 6-9). Summation was chosen as it is essential to provide the total value of energy to deliver. After receiving messages from the child nodes, the sensor node estimates its energy needs given its locally estimated lifetime and energy consumption rate (Eq. (4.2), Alg. 2 line: 17), and the received reference lifetime of the border nodes. The goal of estimation is to calculate how much additional energy sensor node requires in order to last as long as the border nodes which lifetime reference it is using.

$$Lifetime_{REF} = \left( \frac{E_{initial}}{E_{initial} - E_{current}} - 1 \right) * T_{elapsed} \quad (3.3)$$

$$E_{needed} = Lifetime_{REF} * \frac{E_{initial} - E_{current}}{T_{elapsed}} - E_{current} \quad (3.4)$$

The result of the aggregation for each group is the sum of needed energy as well as the number of sensor nodes contributing to the sum. The messages also contain the maximal and minimal angles at which traversed sensor nodes are located.

Taking the advantage of the broadcast nature of wireless communication, the message traveling towards  $I$  is processed not only by the addressed sensor nodes but also by other sensor nodes belonging to the spanning tree and placed at the same or smaller hop distance from  $I$  than the addressed sensor node. These sensor nodes do not aggregate the energy needs but extract the lifetime reference value. They calculate the average of all children intercepted lifetime reference values and use this value to estimate their own energy needs using Eq. (4.2). This approach allows for better approximation of the lifetime reference value, as the reference is calculated from more sources. Figure 3.2 shows a few examples (dashed links). This is especially useful for those sensor nodes that do not have any child node. Node  $M$  does not serve as a parent to any of the sensor nodes, therefore, the interception of the communication between  $J$  and  $N$  is useful to receive a more accurate lifetime reference. Without this optimization  $M$  will use its own lifetime as a reference.

An alternative to hop distance is to use Euclidean distances. Varying the width of rings allows for controlling the aggregation accuracy and also the size of messages transmitted. The smaller the width values, the higher is the accuracy but at the cost of a larger message size.

### 3.3.3 Profiling Outcome and Usage

At the end of the profiling process, the initiator  $I$  possesses a collection of reports from different arcs ( $\widehat{arc_j \beta_{j-1,j} \beta_{j,j+1}}$ ) of the energy hole. Before using the information,  $I$  attempts to fuse the radial information. The neighboring arcs  $arc_{j-1}, arc_j$  (Table 3.1) are inspected regarding the energy density required at certain distances. If the variance is not greater than predefined threshold value  $arc_{var_{th}}$  then aggregation takes place, as described in the Phase 2. The memory required to store profile is equal to number of angles times the number of distances.

Proper maintenance should not only restore the network coverage, but also assure deployment of enough resources for the endangered region (energy hole and area potentially to be consumed by energy hole growth) to last as long as the rest of the network, by taking into account its estimated energy needs. The number of maintenance actions should be limited to the minimum. Optimally maintenance should be preformed only once in the manner that prevents the reoccurrence of the profiled energy hole, delivering in advance the right amount of needed energy and not just placing new sensor nodes to replace the failed ones. In order to achieve this goal it is necessary to know the distribution of energy needs within the energy hole as well as in its surroundings and energy holes expected size and shape basing on energy holes dynamics. Using this knowledge it is possible to deploy adequate

	$distance_0$ ( $ring_0$ )	...	$distance_i$ ( $ring_i$ )	...	$distance_q$ ( $ring_q$ )
$arc_0$	$sum_{0,0}$ $cnt_{0,0}$	...	$sum_{i,0}$ $cnt_{i,0}$	...	$sum_{q,0}$ $cnt_{q,0}$
...	...	...	...	...	...
$arc_j$	$sum_{0,j}$ $cnt_{0,j}$	...	$sum_{i,j}$ $cnt_{i,j}$	...	$sum_{q,j}$ $cnt_{q,j}$
...	...	...	...	...	...
$arc_m$	$sum_{0,m}$ $cnt_{0,m}$	...	$sum_{i,m}$ $cnt_{i,m}$	...	$sum_{q,m}$ $cnt_{q,m}$

Table 3.1: Tabular representation of angular and radial profile

amount of energy, neither under-stocking (reoccurrence of the energy hole) or over-stocking (wasting resources, increasing costs).

For the idealized redeployment for every sensor node  $n_i$  within the profiled hole, the angle  $\delta_i$  is calculated at which sensor nodes is positioned in relation to  $I$ . Then, this is used angle to select arc  $arc_j$  or arcs of energy angular profile (Table 3.1) to which a sensor node belongs. From the selected arc the value of energy at the  $distance_i$  is divided by the amount of reporting sensor nodes at this distance.

The obtained profile has as simple form of two matrices and can be used by sensor nodes, or other assist nodes, or on the sink for reconfiguration or maintenance planning. The profile could be sent to the sink using the routing protocol.

In the case of the energy hole center displacement, when the first sensor node to approach threshold is not positioned in the center of the energy hole, the algorithm remains effective, but loses on its efficiency. The local spanning tree length in different directions is not balanced, causing transmission of longer messages over longer distances, which is not offset by shorter messages from other directions. The distribution of energy still can be reconstructed from the non-monotonic changes in the energy densities at hop distances. The same remarks apply in case when the energy hole has non circular shape. The actual shape of the energy hole can be deduced from the varying densities of required energy in different directions from the initiator node.

### 3.3.4 On the Selection of the Initiator

In order to select  $I$  as close to the energy hole epicenter as possible and to prohibit singular (faulty) sensor nodes to initiate profiling, it is important for the candidate sensor node to query the energy values of its 1-hop neighbors before initiating the

profiling. The candidate becomes  $I$  only if the average of these values is below a certain threshold ( $E_{TH}$ ). It is also possible that two or more initiators simultaneously start profiling within the same energy hole. In this case the simple solution is for sensor nodes to join the spanning tree rooted at  $I$  with the lowest profileID. In order to prevent repetition of profiling by another sensor node reaching  $E_{TH}$ , every sensor node once joining a spanning tree delays the start of a new profiling for the time required to undertake maintenance actions. It is evident that a higher connectivity degree of  $I$  increases the number of branches of the aggregation tree and consequently also the radial resolution of the energy needs profile. Therefore, it is meaningful to require a minimal value of connectivity degree for a sensor node to become  $I$ .

### 3.3.5 Profiling without Position Knowledge

Alg. 1 assumes that sensor nodes know their position. In case of lack of this information, it is still possible to adapt the algorithm though at the cost of partial decrease in accuracy as it is shown in the evaluation section (Figure 3.7).

Without the position knowledge, during the first phase of profiling, sensor nodes now cannot ascertain the parent with the angle closest to their relative angle to the initiator node. Therefore, the parent selection decision should be based on the signal strength of a potential parent, assuming that the stronger signal indicates closeness of the sender. The closer the parent node, the lower the chances of extending the arc width.

In the second phase for aggregation, only the hop distance information can be used. The initiator node requires additional data from the neighboring nodes about their neighbors to properly order the angular information. This approach is based on technique of approximating the distance between sensor nodes by comparing number of shared neighbors as presented in Wang et al. [2007]. As a conclusion, it is also possible to deduct the proper angular ordering of submitted information, by comparing the sets of neighbors. As it is required from the initiator to poses a minimum connectivity degree, therefore, the neighboring nodes share sets of neighbors. These sets are compared. Sensor nodes that share most of the same neighbors are each other neighbors. The collected energy information contains an approximated distribution of energy needed, but the location and orientation of the energy hole is not provided. In order to partly alleviate this problem the sensor nodes forwarding the report to sink should also build a return path, so when the maintenance is automated by using robot, the robot could use this routing information for navigating towards the energy hole.

### 3.3.6 Reconfiguration

The sink upon receiving the energy hole energy requirements profile informs the WSN operator about the necessity of performing maintenance activity. The topic of redeployment is not in the scope of this chapter, therefore only some basic proposals for handling it are presented in this section. The profile provides the operator information about the energy inadequacy within the energy hole. Information about reference energy allows the estimation how much the lifetime of the network can be extended when delivering the energy. The cost of the maintenance can be confronted with the that potential gain and decision can be made about actual maintenance procedure. The redeployment energy can be delivered by deployment of new fully charged nodes. In case of locations where the energy needs exceed capacity of single node, the solution could be the deployment of several sensor nodes and waking them up from sleep one after another as proposed in Santi and Simon [2004]. The delivery method can take various forms. The simplest one is the manual delivery of the sensor nodes by the operator and placing them according to the provided distribution. The automated delivery could involve using a robot or an unmanned vehicle (UXV) equipped with stock of sensor nodes for deployment. In this case the energy profile would be required only for estimation of energy needs, the distribution information could be used only for initial navigation. When the robot approaches the energy hole it can verify directly the distribution profile by queering the sensor nodes inside the energy hole about their energy needs assuming that the sensor nodes retain the information received during execution of the profiling algorithm. Energy requirement profile delivers the information at which locations and how many nodes should be deployed assuming certain displacement possibility of the method. This form of the deployment is part of further research, with goal to extend the proposals included in Leoncini et al. [2005].

## 3.4 Evaluation

In order to evaluate the performance of LEHP, first the simulation settings and performance metrics are defined.

### 3.4.1 Evaluation Setting

The simulation was performed using the OMNeT++ simulator and MAP++ framework. As the LEHP profile accuracy depends on the average connectivity degree of the network, the network topology is set up to evaluate its impact. Based on Bettstetter [2002] the following network parameters were chosen to vary the

connectivity degree. All sensor nodes use the same fixed communication range  $r = 25\text{m}$ . The sensor nodes are deployed uniformly random over an area of size  $250\text{m} \times 250\text{m}$ . The results are provided by different connectivity degrees by deploying 250, 500 and 750 sensor nodes. All sensor nodes, independent of their placement, are assumed to consume energy at a constant rate (period of 25 sec) for their normal operation. Additionally, every 1.5 sec, the sensor nodes conduct sensing according to the hotspot model.

Only a single energy hole in the network is considered, but its size and shape vary. The simulation results are shown for aggregation distances of  $r/3$ ,  $r/2$  and  $r$  and fixed hop distance. The phenomena was modeled using Exponential, Pareto and Normal spatial distributions as discussed in Section 3.2. If not otherwise stated, a period of 25 sec is assumed for operational energy usage, Exponential distribution model of phenomena, deployment of 500 nodes, aggregation of messages using the Euclidean distance of  $r/2$ .

### 3.4.2 Profiling Evaluation Metrics

An ideal maintenance of the energy hole is simulated for purpose of evaluating of the proposed LEHP approach. The created profile is used (Table 3.1) to retrieve the energy needed for each sensor node. The needed energy is added to the sensor nodes' current energy. Next, simulation continues and the lifetime of the repaired network is measured. At the end of simulation the lifetime of the energy hole is measured in relation to the lifetime of the rest of the network. The optimal redeployment should lead to possibly simultaneous energy dissipation induced failures of the sensor nodes inside and outside the energy hole. The premature failure of the sensor nodes in the energy hole signifies undervaluation of the energy needs, the delayed failure is indication of overstocking with energy. The ratio of alive sensor nodes in the energy hole to the initial number of sensor nodes over relative time is used as a metric to quantify the efficiency of LEHP. Relative time is expressed as a percentage of the lifetime of the network without energy injection. The lifetime is defined as the time of first failure of sensor node outside the energy hole. Consequently, if some sensor nodes within the energy hole as the result of energy injection are overstocked with energy, it is possible for them to remain alive at time greater than the network lifetime. An important performance metric is the energy cost of LEHP.

### 3.4.3 Energy Efficiency of LEHP

While describing the energy cost of a profiling algorithm the computational expenditure is disregarded as its imprint is insignificant in relation to the dominant communication energy costs measured as the number of messages sent per sensor

node. The energy efficiency evaluation also gives an upper bound for messages length.

During the spanning tree construction phase,  $I$  starts limited flooding/broadcast that propagates through the energy hole. Each sensor node receiving the broadcast message evaluates the length of the path to  $I$  and only if it is shorter than the one currently stored in the routing table, the message is propagated further. In the optimal case each sensor node within the energy hole should forward the message only once. Because of the varied propagation delays along different possible paths, sensor nodes do not always receive the first message on the optimal path hence multiple forwards result. In order to account for this additional transmission, the number of forwards per sensor node was measured, for the assumed evaluation settings. The results show that on average 74.53% of sensor nodes performed only a single forward, 24.27% two forwards and only 1.20% three forwards. The average is then 1.26 forwards per sensor node for the spanning tree construction, where the length of the transmitted message is constant.

The evaluation of the second phase is straightforward. Every sensor node within the energy hole aggregates received messages from its children and sends only a single message to its parent. The length of aggregation messages varies, growing on the way towards  $I$ . It contains a fixed length header for routing and a varying number of entries for each quantified distance for which aggregation is performed. The maximum length of the message payload cannot exceed  $L$  (Eq. 4.4), where  $q_{size}$  is the aggregation quant size (fraction of  $r$ ).

$$L = \max(TTL_{max}, \lceil \frac{r}{q_{size}} \rceil) \times \text{sizeof}(\text{data\_struct}) \quad (3.5)$$

Summarizing, the amount of energy required per sensor node for performing the profiling is very limited and amounts to forwarding 2.26 messages on average. In addition, it is important to note that the profiling is performed only rarely as profile based maintenance should prevent energy hole reoccurrence.

Additional comparative simulations were conducted that involved data transport without the aggregation. The results show that the average number of transmission per sensor nodes amounts to 10.5 forwards as each message must be transmitted independent from other messages.

### 3.4.4 Simulation Results

All of the presented plots share the same axes. Axis  $x$  represents relative time expressed in percents. Axis  $y$  shows percentage of sensor nodes within the energy hole that have enough energy for operation over the time. The ideal curve (perfect accuracy of energy needs profiling) should be a straight line at value of 100% alive sensor nodes reaching the relative time of 100% and then instantaneously

falling to 0%. Deviations from the ideal curve imply a degree of inaccuracy. The curve falling below 100% alive sensor nodes before reaching 100% relative time indicates level of under-stocking of the network with the energy within the energy hole. The value above 0% alive sensor nodes at relative time over 100% indicates level of overstocking of the network within the energy hole.

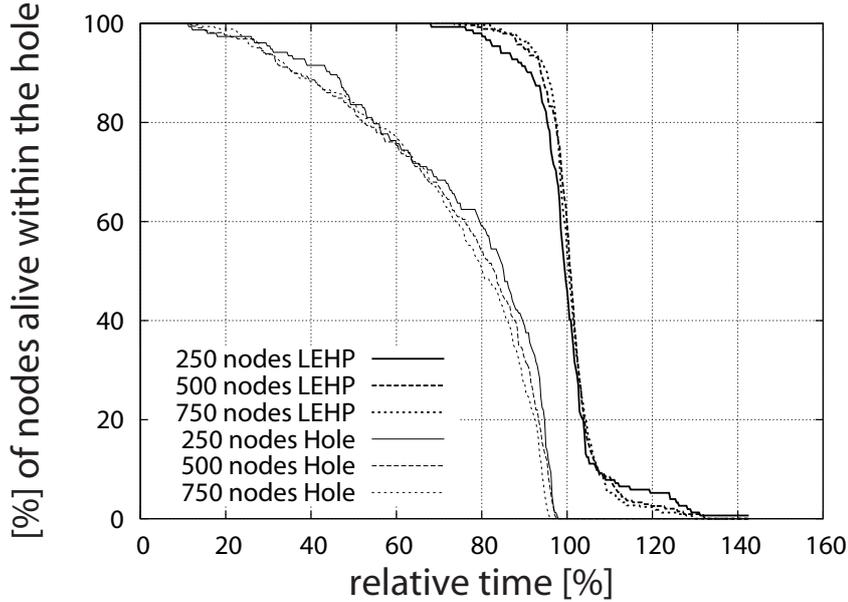


Figure 3.3: Impact of node density

Figure 3.3 depicts the impact of density of the deployment on the accuracy of LEHP. The labels "x nodes LEHP" and "x nodes Hole" represent simulation with and without LEHP respectively. As expected, the network deployed with 250 sensor nodes shows slightly lower accuracy. This occurs as lower average connectivity degree degrades angular resolution. Although the network remains connected, sparse connectivity leads to the assignment of sensor nodes to an arc which is less accurate than in denser deployments. The deployment of over 500 sensor nodes assures high connectivity degree, therefore, higher angular resolution is reached to result in more accurate profiling. Without energy injection, the first sensor nodes (closest to the center of hotspot) start to fail already at time of 15% of network lifetime and the rest gradually follow. *The accuracy of LEHP is highly independent from the density of the network.*

The impact of the aggregation step (Alg. 2) is presented in Figure 3.4. As expected, the higher the granularity, the better the accuracy is achieved. Both  $r/3$  and  $r/2$  (step size  $r/3$  and  $r/2$ ) perform very well. The smaller area over which aggregation takes place the greater chances that lower variance in energy need

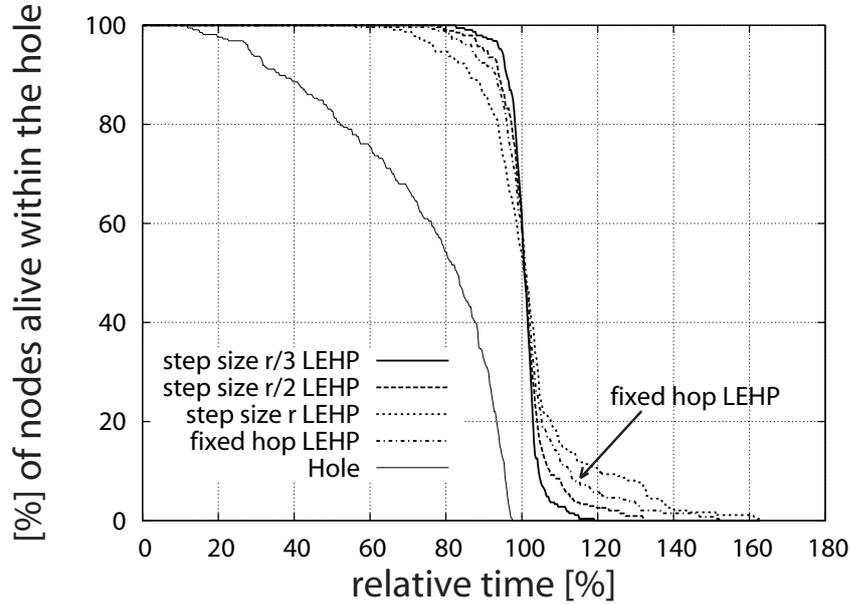


Figure 3.4: Impact of aggregation distance

levels, therefore lower averaging error. It is to observe that applying hop distance metric (fixed hop) outperforms Euclidean distance approach if  $step = r$ . When using the hop distance, the distance corresponds to shorter Euclidean distance than  $r$  as it would be expected by multiplying the range of communication by number of hops. The curve *Hole* represents the failing of the sensor nodes without energy injection. *Euclidean based aggregation performs better for ring widths smaller than communication range, otherwise it is more efficient to use hop based quantification.*

The best performance is achieved for the Exponential distribution model (Figure 3.5). Here, the highest energy depletion is concentrated near the epicenter of the energy hole, what assures high accuracy of profiling for this most dynamic region. For Normal and Pareto distributions the energy is more evenly distributed, degradation of network starts later but is much steeper afterwards (Pareto Hole and Normal Hole curves), therefore, the implications of lower accuracy at greater distances are more evident. *The LEHP offers better accuracy for the distributions which show smaller variance.*

In order to study the impact of irregular shape on the performance of LEHP, additional simulations were conducted on energy holes as illustrated in Fig 1. These irregular energy holes result from the overlap of three symmetric energy holes. The accuracy of LEHP decreases but remains adequate (Figure 3.6). The main cause of decreased accuracy is the selection of the initiator. The selection of proper location for initiator is hard in flat regions of phenomenon distribution. The

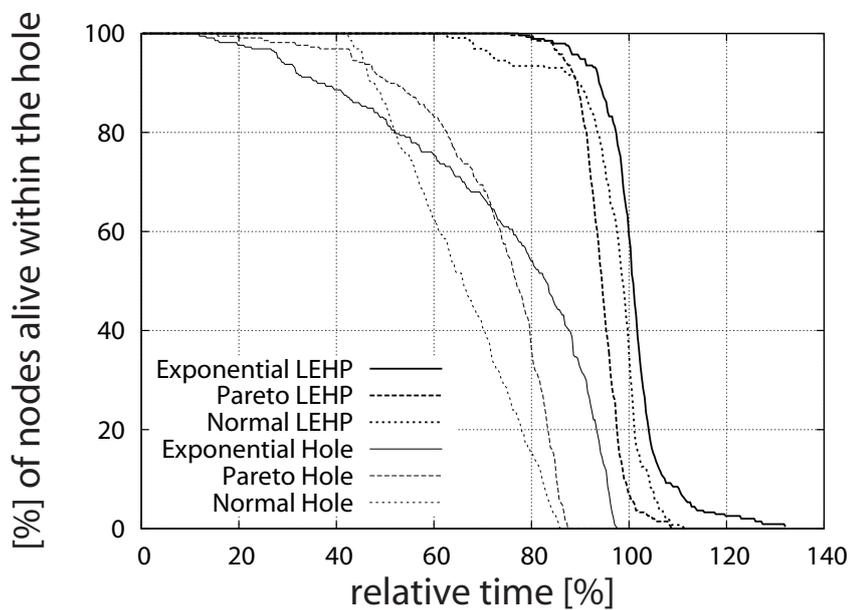


Figure 3.5: Impact of hole distribution

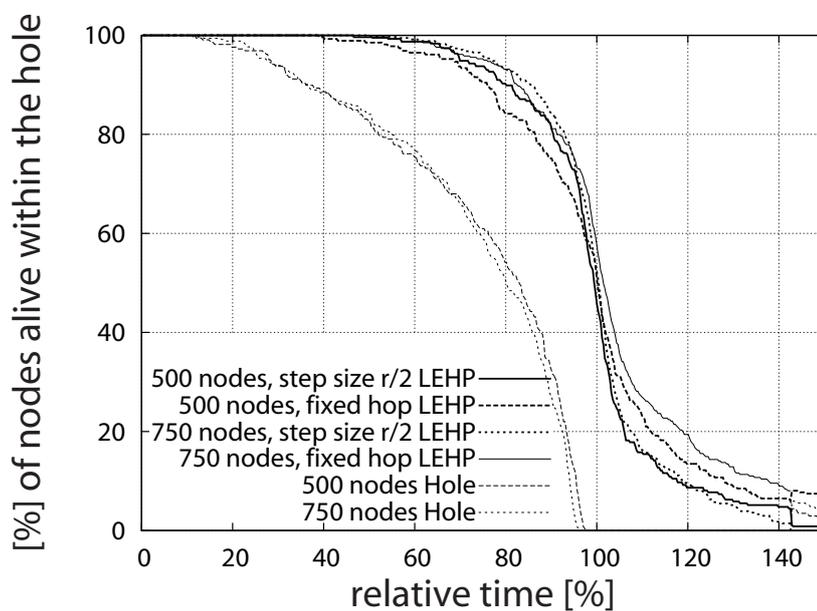


Figure 3.6: Impact of hole shape

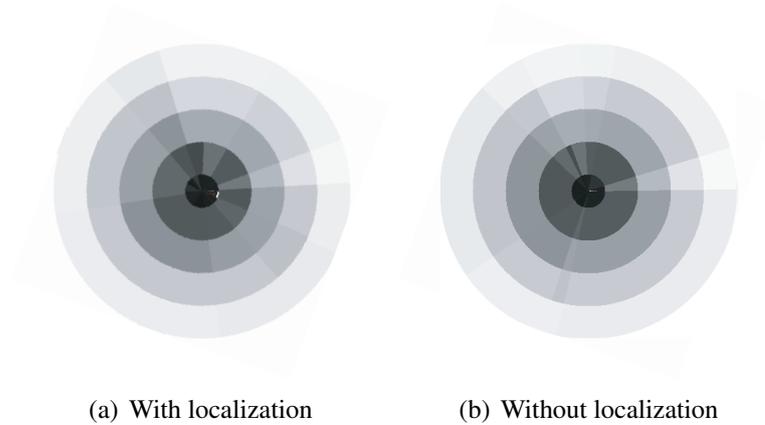


Figure 3.7: Radial and angular energy need distribution

accuracy of profiling decreases on the part of the energy hole border that is furthest placed from the initiator. Despite these obstacles the LEHP still manages to keep most of the sensor nodes alive at time the original network is already partitioned. *The LEHP suffers from irregularity of energy holes, but remains robust.*

Figure 3.7(a) visualizes the profiles of two simulation results of angular energy need profiles. The rings represent the quantified distances from the initiator and the arcs divide the profile in directions from which data were gathered. The shades of grey represent density of energy needed. The darker the marking, the higher energy density is required. The highest density is located at the center of the hotspot and gradually decreases in direction of the hole border. The different shades of grey of arcs, shows how the density of required energy changes in different directions. Figure 3.7 highlights a slight accuracy difference for LEHP with and without position information.

### 3.5 Summary

This chapter provides the first necessary steps for making the energy based maintenance a viable option for WSN functionality conservation. The developed energy hole profiling algorithm, LEHP, is shown to be an efficient strategy for valuable early warning of likelihood of energy holes. LEHP allows the accurate positioning and size estimation of developing energy hole, which usually leads to sensing and communication coverage loss. The efficiently computed and accurate angular and radial information about the energy needs, provides means for effective maintenance actions, which optimally are long term. This assertion is substantiated by the extensive simulation results.

# Chapter 4

## Topology Oriented Maintenance in Sparse Wireless Sensor Networks

### 4.1 Overview

The physical number of sensor nodes constitutes a major cost factor for WSN deployments. Hence, a natural goal is to minimize the number of sensor nodes to be deployed, while still maintaining the desired properties of the WSN. However, sparse networks even while connected, usually suffer from topology irregularities that negatively impact the network lifetime and responsiveness, i.e., sensor data delivery reliability and latency. In addition, sensor node failures easily complicate/enforce/aggravate these irregularities. Valuable efforts have been conducted to discover topology specific anomalies such as coverage holes or critical/bottleneck nodes. Unfortunately, these efforts suffer from at least one of the following drawbacks: (a) They are centralized and consequently inefficient in large-scale networks, (b) they are tailored to one class of anomalies, or (c) do not propose how to remedy the identified anomaly. This chapter focuses on sparse WSN which usually show varied topology irregularities and propose an in-network and localized strategy that efficiently (i) discovers generic topology irregularities, and (ii) identifies locations for minimal number of new augmented sensor deployments to remedy topology irregularities and sustain the desired operational requirements.

Overall, the benefits of repairing the topology irregularities can be classified into two broad classes, namely network- and functionality-centric improvements. The network operation is enhanced through shorter routing paths, reduced latencies, and balanced and lower energy overhead. The WSN functionalities are optimized by maintaining a regular topology leading to uniform sampling, more accurate information extraction and hence achieving a higher Quality of Informa-

tion (QoI) level for the WSN objectives. Therefore, the proposed contribution can make potentially a significant impact on the network and application design in WSN as well as their deployment.

The TOM qualities are validated with an extensive set of evaluation studies. They show TOM effectiveness at reducing the discrepancy between hop and Euclidean distances, balancing the energy usage and shortening average hop distance to the sink. The evaluation studies also demonstrate that TOM effectively and efficiently maintains an overall regular topology by enhancing an initial sparse WSN. Thus, TOM outperforms the costly approaches that require a certain level of redundancy in node deployment.

### 4.1.1 Contributions

Topology Oriented Maintenance (TOM):

- efficiently detects topology irregularities and provides maintenance options for a sustainable regular topology,
- is tunable and allows setting bounds on the tolerable level of topology irregularity,
- is localized, thus, allowing for efficient and sustainable WSN self-healing, and
- provides means for a balanced and minimal initial WSN that fulfils the requirements.

The chapter is structured as follows. Following the system model in Section 4.2, Section 4.3 presents a precise formulation of the proposed approach objectives and requirements. Section 4.4 details the proposed TOM technique as the chapters's main contribution. The evaluation is presented in Section 4.5.

## 4.2 System Model

Conforming to contemporary WSN literature concerned with topology problematic, the system model assumes a WSN consisting of  $n$  resource constrained sensor nodes and a sink. The sensor nodes have finite battery energy and usually possess limited processing and storage capabilities. The communication range  $r$  is limited and fixed for a given deployment. Two neighboring sensor nodes can communicate directly only if their Euclidean distance is smaller than  $r$ . This communication dominates the imprint on the energy depletion of the sensor nodes. The initial sensor deployment is sparse, but the resulting network is connected

or at least the *giant component* Li et al. [2003] is connected to the sink. In the latter case the relevance of remaining unconnected sensor nodes is de-emphasized as their connection using uncontrolled deployment significantly increases the cost Santi [2005]. The sensor nodes know their position using on-board GPS receivers or alternative GPS-free techniques of localization He et al. [2005]. The proposed approach consider cases where (a) all sensor nodes are static, or (b) a mix of static and nodes of controlled mobility. The applied model implicitly considers sensor node failures and duty cycling as this is equivalent to relocating a sensor node. The network lifetime is defined as the time elapsed from the deployment till the first partitioning of the network. The sensor nodes know their hop distance to the sink, e.g., based on a shortest path tree routing. The sensor nodes are also aware of the presence of their 1-hop neighboring sensor nodes, including their position and hop distance to the sink.

### 4.3 Problem Formulation and Objectives

The existence of topological holes, critical Jorgiæ et al. [2004] / bottleneck Gou and Yoo [2010] nodes, and the non-uniform deployment of sensor nodes manifests itself and can be determined through the discrepancy between Euclidean induced distance Vural and Ekici [2005, 2010] and the topology induced hop distance to the sink. This discrepancy arises only when there exists no *straight* route between the affected sensor node and the sink. Lengthy routes translate in higher latencies and higher/unbalanced energy consumption among the sensor nodes. As a result the WSN can suffer from unbalanced energy usage even under optimal energy conservation schemas, which usually leads to potential premature disconnections and partitioning. In addition, there exists a related major source of topology irregularities, namely the network failures. Sensor nodes often fail as they operate with finite energy capacities and in harsh environments. Usually in sparse networks, sensor node failures directly impact the topology irregularities and negatively influence the network lifetime and responsiveness.

In order to show the severity of topology irregularities, the Figure 4.1 illustrates the distribution of the expected hop distance (deduced from the Euclidean distance of sensor nodes to the sink -  $Y$  axis) as a function of actual hop distance (derived from the shortest path routing tree -  $X$  axis). The size of the circles corresponds to the number of sensor nodes having the same characteristics and illustrates the occurrence frequency. The distribution was obtained from 1000 random topologies generated for deployments of 230 (sparse WSN) and 350 (dense WSN) sensor nodes over the same area (exact simulation settings are detailed in Section 4.5). For each sensor node, in the optimal scenario, the topology induced hop distance should be equal to the Euclidean induced hop distance and consequently

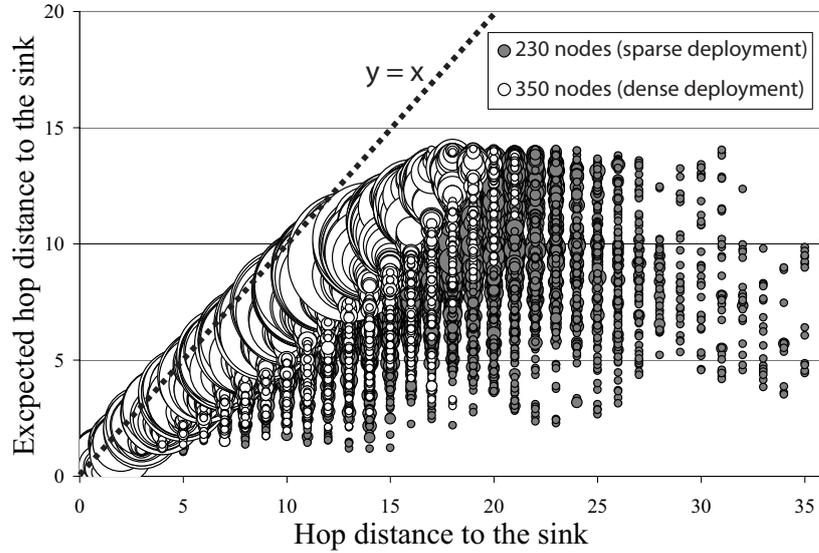


Figure 4.1: Distribution of actual hop distances compared to expected distances.

all points should lie on the  $y = x$  line. The case of dense network deployments shows that the topology generally follows a uniform distribution. Only a few sensor nodes have topology induced hop distances substantially different from the expected Euclidean induced hop distances. This property is much desired for high responsiveness, optimized load balancing and hence prolonged lifetime. In case of a sparse deployment, the topology induced hop distance can substantially differ from the optimum, with great concentration of circles below the indicated  $y = x$  line. Also the topology induced distances significantly exceed the maximal hop distance for the dense network scenario (15 hops). A topology, where the correlation between topology and Euclidean induced hop distances is low, is termed as an *irregular topology*. The observed irregularities in sparse deployments need to be alleviated using an efficient and comprehensive maintenance.

In WSNs it is crucial to provide balanced resource usage to assure longest possible network lifetime. It is especially evident for a sparse network where the low connectivity degree usually leads to the creation of lengthy routing paths. As shown in Figure 4.2 the routing path of *Node A* (to *Sink 1*) vastly differs in length from the routing path of *Node B* (to *Sink 1*) although both are placed at comparatively equal Euclidean distances to the sink. A longer route translates in higher energy cost of sending data to the sink. Additionally, the lack of straight path to the sink means that the route has to be shared with many more sensor nodes (*Region U* in Figure 4.2) than in the case of a regular topology. As a consequence sensor nodes, such as *Node C*, are under an additional burden and serve as focus points for routing protocols. Their energy becomes discharged much faster and

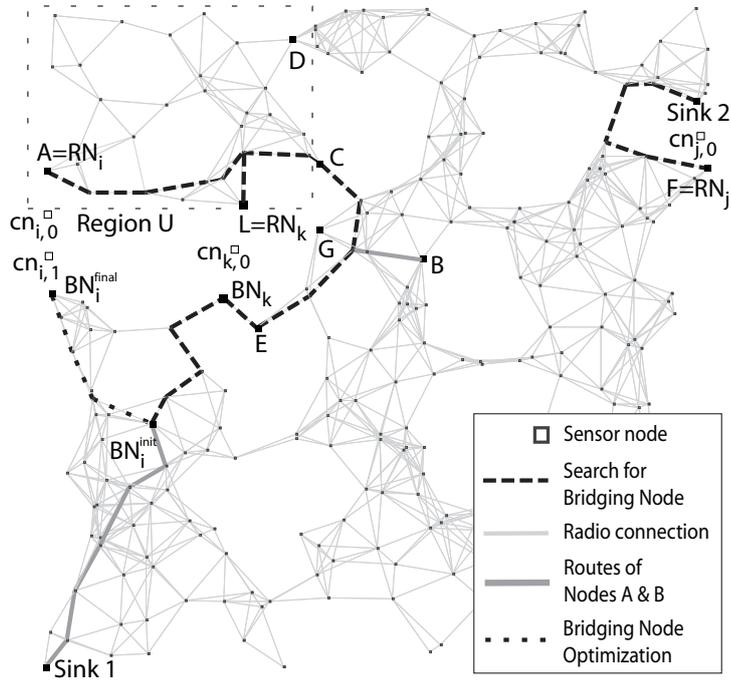


Figure 4.2: Topology irregularities in a sparse deployment example

may lead to network partitioning.

The goal of this chapter is to provide a distributed technique for finding possibly few re-/deployment locations, where the placement of new sensor nodes or the repositioning of mobile nodes will restore a balanced energy usage and data traffic. While proposed approach directly targets only the discrepancy between Euclidean and hop distances, the reduction of that metric improves many QoI aspects of WSN such as lifetime, latency and data accuracy. That's way it is important to provide regular topology, otherwise applications waste a significant amount of resources for transmitting the data at higher latencies and achieving lower reliability.

The provided solution should be tunable in regard to the extent of irregularities it can tolerate. Also the relevance of the irregularity should be taken into consideration. For instance, the position and orientation of irregularity in relation to the sink should be considered. While *Region U* is affected by the irregularity in relation to *Sink 1*, it does not exhibit adverse effects in relation to *Sink 2*. An opposite situation takes place in case of *Node F*. In relation to *Sink 1* it is only marginally affected by the irregularity, but in case of *Sink 2* the effects of the irregularity are evident (extended routing path to *Sink 2*). Moreover, this technique should limit the amount of resources required for maintenance. For this purpose it is crucial to provide feedback regarding the possible gain from a certain maintenance option

(extent to which the topology is improved, e.g., reduction in hop distances to the sink) and its cost (resources required to complete maintenance). The proper evaluation of the gain requires inclusion of the localized information, e.g., the actual sensor data volume. This feedback value could be treated as a normalizing weight. In case of limited resources TOM should allow to allocate them according to the best tradeoff between potential gain and costs.

## 4.4 TOM: Topology Oriented Maintenance

After an overview of the proposed approach, the section details techniques for discovering the sensor nodes that are most affected by topology irregularities. Subsequently, it describes the search process for finding a suitable part of the network where new connections could be established to repair the irregularity. Finally, section describes optimizations for this search process and techniques for local evaluation of possible reconfiguration gains.

### 4.4.1 Overview

The primary step for providing an appropriate maintenance of the topology is the identification of those sensor nodes that suffer from an irregularity in the topology. As mentioned in the introduction, the hop distance discrepancy is an appropriate indicator of *relevant* irregularities *independent* from their cause. Accordingly, the sensor nodes that suffer at most from irregularities are: (a) Sensor nodes whose hop distance shows a discrepancy that exceeds a certain threshold, and (b) sensor nodes that have no hop discrepancy, however, they have to relay the traffic of those sensor nodes that suffer from discrepancy. The sensor nodes that show the highest discrepancy in their hop distance are referred to as Reconnecting Nodes (*RN*) (e.g.,  $RN_i$  in Figure 4.2). Their selection is influenced by tunable parameter set to determine acceptable extent of irregularities. Therefore, they are the first to be considered for bridging to the closest part of the network where a regular topology may be maintained. Further network/node properties should also be taken into consideration while selecting RNs, e.g., residual node energy. There are disadvantages of using an energy constrained sensor node as an RN, as this sensor node would forward the data from sensor nodes in its neighborhood, significantly increasing energy depletion rate.

The selection of the sensor nodes, to which RN could be reconnected is the second step of topology maintenance. The algorithm starts with the RNs and locally traverses the topology in the direction of sink searching for the sensor nodes, which are located in the part of the network where regularity is maintained. The sensor nodes are selected under the condition that the newly established connec-

tion (between RN and the selected sensor node) reduces the distance discrepancy to the acceptable level (after maintenance the indicator of irregularity will fall below a specific threshold), determined by the network operator (tunable parameter). Such a sensor node is referred as a Bridging Node (*BN*) (e.g.,  $BN_i$  in Figure 4.2). Also in this case it is valuable to consider the energy level of the BN candidate. After performing topology reconfiguration BN will have to forward messages from the reconnected area. Therefore, it is essential that BN has enough energy to fulfill its role. Furthermore, the set of sensor nodes placements proposed by TOM to connect RN and BN is referred as Connecting Nodes  $CN_i = \{cn_{i,0}, \dots, cn_{i,j}\}$ .

Not all proposed reconfigurations are equally important and beneficial to the topology regularity. Some of the reconfigurations may be redundant. For example, maintenance may apply to the region of low phenomenon activity, and as consequence the added sensor nodes would transport only low volume of data, which may not justify the maintenance cost. It is especially important to provide the feedback based on localized data for the scenarios where resources for providing maintenance are limited and need to be prioritized. The goal of the last step is to determine the importance of each given reconnection option by weighting its impact on the topology. The weighting may include single or a set of factors (e.g., hop distance reduction, data transport volume, etc).

#### 4.4.2 Discovery of Topology Irregularities

It should be noted that the repair of an irregularity that is closer to the sink may resolve the regularity perceived by sensor nodes that are farther from the sink. Therefore, the discovery process should trigger depending on sensor node distance from the sink. The farther from the sink the later should the discovery process start.

Therefore, it is important to identify the sensor nodes, whose reconnection to the closest part of the network with regular topology will bring highest gain in balancing topology distances. To this end, the algorithm looks for sensor nodes whose discrepancy between Euclidean and hop distance is locally maximal. In order to estimate the extent of the Euclidean and hop distance discrepancy, the algorithm calculates the sensor node discrepancy indicator  $f_i$  (Eq. (6.1)).

$$f_i = r - \frac{\delta_{sink,i}}{hop_i \cdot \theta} \quad (4.1)$$

$hop_i$  is the hop distance of Node  $i$  from the sink,  $\theta$  ( $0 < \theta < 1$ ) is the tunable parameter that expresses the fraction of the communication range  $r$  that one hop covers on average in a single transmission, and  $\delta_{sink,i}$  is the Euclidean distance of Node  $i$  from the sink.  $f_i$  effectively shows the magnitude of the detour of the route from Node  $i$  to the sink under the stated  $\theta$  parameter for communication

range utilization. Therefore,  $f_i$  represents an indicator for topology irregularities and it is utilized to measure how much communication range is lost on average by each hop within a given route. The higher the value of  $f_i$  the larger the discrepancy between Euclidean and topology induced distances, and the longer the route detour. At first, Eq. (6.1) would lead to the following counterintuitive perception:  $f_i$  increases with  $r$ . However, it should be noted that the connectivity increases with  $r$ . Consequently the number of hops ( $hop_i$ ) decreases with  $r$ , thus, offsetting the increase in  $r$ . A threshold discrepancy value  $f_{TH}$  is defined so that for a given Node  $i$  if  $f_i > f_{TH}$  then that Node  $i$  is located in a network region experiencing a topology irregularity.

After fixing the indicator of topology irregularities it is necessary to find the local maximum of  $f_i$ . The sensor node with the maximum value is the one that should be reconnected to a closer regular part of the network. At this point it should be also assured that the selected sensor node is capable of assuming its role of RN, e.g., it has sufficient energy level. Therefore, RN is chosen only amongst the sensor nodes whose  $f_i > f_{TH}$  and whose energy level  $E(i)$  is higher than a given threshold  $E_{TH}$ . Each sensor node calculates its  $f_i$  value as well as those of its neighbors (using the collected neighborhood information) and calculates their overall maximum. If the maximum value  $f_i$  exceeds  $f_{TH}$  and the energy level is above  $E_{TH}$  then the value is transmitted to all 1-hop neighbors. Only a sensor node that receives from all its 1-hop neighbors a value which is (a) lower or equal to its own, (b) higher than  $f_{TH}$  and (c) has a residual energy level above  $E_{TH}$ , is designated as RN.

This discovery process is based solely on local information that is updated upon changes in topology. The necessary data for this phase of TOM can be collected using piggy-backing. The event triggered for the re-evaluation of the  $f_i$  value takes place only upon changes of routing paths, establishing new routing paths to a new sink (e.g., in a multi-user scenario), or upon sending the  $f_i$  value by one of the neighbors.

### 4.4.3 Search for Bridging Nodes

After the completion of the discovery phase, an RN can start the second phase of *Search for Bridging Node* (Algorithm 3). In this phase, the current topology is locally explored to find the BN. The search occurs in two steps: (a) *Initial search*, where the topology is explored to find an initial candidate for assuming BN role, and (b) *Bridging Node Optimization*, where further sensor nodes are considered to shorten the distance between the bridging and reconnecting nodes.

### Initial Search

At first only a simplified version of the search algorithm is presented that does not consider the updates to the topology by other RNs. The strategy is to explore the path by circling the topology irregularity which is the source of the discrepancy.

$$\alpha_{i,j} = \arctan\left(\frac{i.y - j.y}{i.x - j.x}\right) \quad (4.2)$$

$$\beta_{i,j,k} = \alpha_{i,k} - \alpha_{i,j} \quad (4.3)$$

The main idea is to circumvent the irregularity using a strategy inspired by geographic forwarding. The search also includes optimizations to skip some border nodes that otherwise would extend the search path (*Node G* in Figure 4.2). The search tries, as possible, to maintain a constant heading in order to reduce the hop distance for locating the targeted regular network region.

---

#### Algorithm 3 Search for Bridging Nodes

---

```

1: function SearchPath(curNode, prvNode) : BridgingNode
2: var candCW, candCCW, candNode;
3: // search for closest (counter)clockwise angular neighbor
4: for all sn in Neighbors(curNode) do
5:   if  $\beta(\text{curNode}, \text{sn}, \text{prvNode}) < \beta(\text{curNode}, \text{candCW}, \text{prvNode})$  then
6:     candCW = sn;
7:   end if
8:   if  $\beta(\text{sn}, \text{curNode}, \text{prvNode}) < \beta(\text{candCCW}, \text{curNode}, \text{prvNode})$  then
9:     candCCW = sn;
10:  end if
11: end for
12: if candCW.hop < candCCW.hop then // find neighbor closer to sink
13:   candNode = candCW;
14: else if candCW.hop > candCCW.hop then
15:   candNode = candCCW;
16: else if  $\delta_{\text{candCW}, \text{sink}} < \delta_{\text{candCCW}, \text{sink}}$  then
17:   candNode = candCW;
18: else
19:   candNode = candCCW;
20: end if
21: if  $\delta_{RN, \text{curNode}} + \text{hop}_{\text{curNode}} \cdot R \cdot \theta < \frac{\delta_{\text{sink}, RN}}{\theta}$  and  $E(\text{curNode}) > E_{TH}$  then
22:   return candNode;
23: end if
24: return SearchPath(candNode, curNode); // hand-off to next candidate

```

---

An RN first establishes its angle ( $\alpha_{RN,V}$ ) in the relation to the previous visited Sensor Node  $V$  (initially  $V = \text{sink}$ ) using Eq. (4.2). Then, searching across its neighboring sensor nodes, it looks for two sensor nodes whose angle distances

are minimal in clockwise  $\beta_{RN,m,V}$  and counterclockwise  $\beta_{m,RN,V}$  directions. The precision of Eq. (4.2 - 4.3) depends on the ratio between the localization error and the distance between a pair of sensor nodes. The precision of the radial position of the closest neighbors can tolerate errors with the only impact it has being a longer search path. One of the two (one if RN has only a single neighbor) closest radial neighbors is the possible next sensor node to explore while searching for the BN. The search process follows the path of the sensor node whose hop distance to the sink is smaller. If the hop distances of both candidates are equal, the next sensor node to explore will be chosen as the one placed closer to the sink w.r.t. to the Euclidean distance. In an unlikely event, as confirmed by simulations, if the search process revisits a sensor node, then opposite choice regarding the selection of next sensor node is made. The search continues until (a) Inequality (4.4) is valid, where  $\delta_{i,j}$  represents the Euclidean distance between two nodes  $i$  and  $j$ , and (b) the energy level of the considered sensor node is larger than  $E_{TH}$ . Each sensor node on the search path sends a single message. The number of hops required for performing initial search in worst case does not exceed the hop distance from sink, as the sink is the ultimate bridging node.

$$\delta_{RN,i} + hop_i \cdot R \cdot \theta < \frac{\delta_{sink,RN}}{\theta} \quad (4.4)$$

### Virtual Search

When an RN searches for a proper BN, it may happen that some other RN has already executed the search process and created alternative routing paths. It would be beneficial to use this information when searching for BN. Instead of connecting the isolated part of the network directly to a region with regular topology, it can be connected to the part of network where the connectivity balance has already been restored. Therefore, the reconnecting links become shorter and require less sensor nodes. Using the prior paths also accelerates the search process. Upon finding updated part of the network, the search process uses already established paths instead of exploring all of the border nodes.

For this goal, the `Neighbors()` function (Alg. 3 L. 4) was modified as follows. Instead of considering all neighbors as potential next steps in the search algorithm, the function selects only their subset. If some neighbors already have alternative routes to an RN, the function checks if in the worst case scenario connecting directly to that RN would allow to fulfill the stated requirements. In such cases, the `Neighbors()` function chooses only from such neighboring sensor nodes that have such an alternative route. If some neighbors have two alternative routes to different RNs then the neighbor which yields a lower value of Eq. (4.5) is selected and no the other neighbors are considered anymore. In case the paths

via the neighboring sensor nodes with alternative paths are selected, only the virtual hop distance is needed, i.e., distance to selected RN plus RN's distance to the sink. This virtual hop distance is then further used for optimizing the selection of BN.

---

**Algorithm 4** Optimized Selection of Bridging Nodes
 

---

```

1: function OptimizeBridge(curNode) : BridgingNode
2: var candNode = nil;
3: var minDist =  $\delta_{curNode, RN}$ ;
4: for all sn in Neighbors(curNode) do // look for best BN candidate
5:   if  $E(i) < E_{TH}$  then
6:     continue;
7:   end if
8:   curDist =  $\delta_{curNode, RN}$ ;
9:   if  $\delta_{RN, sn} + hop_{sn} \cdot R \cdot \theta < \frac{\delta_{sink, RN}}{\theta}$  and curDist < minDist then
10:    minDist = curDist;
11:    candNode = sn;
12:   end if
13: end for
14: if candNode != nil then // if candidate found, try searching further
15:   return OptimizeBridge(cand);
16: end if
17: return curNode; // hand-off to next candidate

```

---

**Optimization of the Selection of Bridging Nodes**

After executing Algorithm 3 the initial Bridging Node ( $BN^{init}$  in Figure 4.2) is found. Although the  $BN^{init}$  complies with the requirements to restore the route balance, there could still be a better candidate (closer to RN) to act as a BN. The optimization is to find the sensor node that is possibly closest to RN (w.r.t. the Euclidean distance) and still fulfills the requirements of BN ( $f_i < f_{TH}$  and  $E(i) > E_{TH}$ ). Algorithm 4 checks among all neighbors of the current BN if any of them also fulfills the conditions of acting as a BN. If such sensor nodes are found, then the one with the minimal distance to RN is selected. Algorithm 4 then re-initiates with this selected BN value. If no better candidate for acting as BN is found then the current sensor node becomes the final BN ( $BN^{final}$  in Figure 4.2).

**Local View Update**

The RN, when initiating the search process, can use its new virtual hop distance to the sink (Eq. (4.5)) that it will have after connecting to the BN. This new virtual value needs to be propagated to all the sensor nodes in RN neighborhood as the hop distance of these sensor nodes will also get reduced. This propagation process resembles a classic spanning tree construction algorithm with a few optimizations

to reduce the message traffic. The RN starts a local broadcast sending a message containing the expected hop distance to the sink. Each sensor node receiving the message checks whether the new path proposed by RN is shorter than the current one. If this holds, the new path is added to the routing table pointing to the RN as root. Each sensor node also decides whether to further broadcast the message. If any of the neighbors can benefit from the new routing path, then the message is forwarded, otherwise it is suppressed. The sensor nodes receiving the message but not benefiting from the new path also suppress the broadcast message. The broadcast message can also be suppressed when it traverses more than a given limit of hops  $hop_{TH}$ . That can be interpreted as the scope of the update is large enough to render this repair a priority. Each sensor node sends only a single message if the new path can influence routing of any of its to be children.

$$vhop_i = \lceil \frac{\delta_{i,sink}}{r \cdot \theta} \rceil - 1 \quad (4.5)$$

Depending on the extent of the irregularity the initial reconnection may be insufficient to completely recover from irregularity consequences. After performing local view update sensor nodes are aware of their new virtual distance to the sink. Still some of the updated  $f_i$  (e.g.,  $f_i$  of *Node L* in Figure 4.2) using the new virtual distance may remain above  $f_{TH}$ . Such sensor nodes initiate a new discovery and search procedures that should lead to further minimizing the impact of topology irregularity. The TOM execution progresses iteratively as long as one sensor nodes still suffers from the topology irregularity with regard to the parameter  $\theta$ .

### Weighting Update

After the Local View Update process finishes, the impact of newly proposed topology adjustment needs to be weighted. Weighting is performed to provide feedback on the importance and urgency of the update to the WSN operator or mobile nodes within the network to make the decision upon executing maintenance. In order to normalize the weight of new route, an additional set of factors may be incorporated (e.g., reduction in hop distances, volume of transmitted data, phenomenon activity level). Initially, presented approach considers the reduction in hop distances for all sensor nodes benefiting from the topology adjustment. For this purpose, each of the sensor nodes that improved its path sends a message containing the number of hops reduced in its path.

Over the *Local View Update* phase, each sensor node can ascertain whether any of its neighbors benefits from the route update. Hence, it waits for their response message about the obtained reduction. It may happen that a sensor node did not receive a response because the other sensor nodes sent their response along a shorter route. Such a sensor node can observe these responses and if necessary

it starts itself the response process. Each intermediary sensor node aggregates the data collected from its children. When it receives data from all its children or sniffs that the children already has responded to other nodes, then it forwards the aggregated data to its own parent. At the end of the weighting process the RN receives a single number describing the weight of the gain from bridging the gap in the network.

### **Bridging**

When both  $RN$  and  $BN$  have been determined, the number and positions for the connecting nodes  $CN_i$  still have to be decided. A cost effective approach is to connect  $RN$  and  $BN$  in a straight line, placing the connecting nodes at equi-distant positions. The number ( $h$ ) of connecting nodes needed depends on the communication range and the coefficient  $\lambda$  ( $0 < \lambda < 1$ ) describing the fraction of usable range of the communication range.  $\lambda$  generally takes values close to 1 meaning that the added sensor nodes utilize almost the full communication range as the deployment is assumed to take place in a controlled manner. If the placement is semi-automated and depends on the estimated localization, the  $\lambda$  value may be lowered to reflect that condition.  $h$  is calculated using Eq. (4.6);

$$h = \lceil \frac{\delta_{RN,BN}}{r \cdot \lambda} \rceil - 1 \quad (4.6)$$

After placing the connecting nodes, the BN should either trigger update mechanism of underlying routing protocol or initiate local broadcast to update the topology with new routing path utilizing added connecting nodes. The update/broadcast will immediately stop at sensor nodes that do not benefit from added sensor nodes and only propagate to those which do.

#### **4.4.4 Reconfiguration**

The following two main reconfiguration classes are considered: (Class 1) support for initial deployment of sparse network, (Class 2) active maintenance of the network with use of mobile nodes.

Class 1 represents the static network. In order to keep the deployment cost of the network low, the network is sparsely deployed, but dense enough that it is connected with high probability. Topology of such networks is highly irregular regarding hop distances between sensor nodes and the sink. The main idea behind this scenario class is to deploy the network with low-costs and then let the network provide the feedback how could it be optimized using limited resources while keeping overall low cost of deployment. Network provides deployment locations

with their weight for improving the network properties. Then the sensor nodes are deployed in controlled manner: manually or semi-automated.

Class 2 involves a WSN, where in addition to static sensor nodes, mobile nodes (e.g., robots) are present (constituting  $x\%$  of sensor nodes). It is common for WSN that due to harsh environment, restricted resources (battery power) or fragile components that the sensor nodes fail. Failure of the sensor nodes, especially the critical/bootleneck ones, can lead to worsening of network properties and in extreme cases to disconnections of parts of network. In that context proposed TOM algorithm can provide for WSN self-healing property. The network can in the distributed manner detect the topology irregularities that resulted from failures. Next, by moving the mobile robots, proper functioning of the network can be restored. In cases when the demand for interventions outnumbers the number of mobile nodes, bidding algorithm Wang et al. [2004] can be used, where bids depend on the given weight of the possible improvement to topology regularity.

## 4.5 Performance Evaluation

This section describes the evaluation settings and the metrics chosen for the evaluation.

### 4.5.1 Evaluation Settings

The evaluation scenario considers a WSN network consisting of 270 sensor nodes, with communication range  $r = 3m$ , deployed over the area of  $30m \times 30m$ . The value of  $\theta$  gradually changes from very relaxed regularity  $\theta = 0.65$  to very strict one  $\theta = 0.95$ . Also the number of sensor nodes varies from 200 nodes (sparse scenario) to 300 nodes (dense scenario) in order to measure the utility of TOM under different network densities. The TOM approach was validated and its efficiency measured using stand-alone implementation.

### 4.5.2 Evaluation Metrics

TOM addresses the problem of network relating the topology irregularity to the affected WSN properties. The objective purpose of the presented metrics is to show the efficiency of the proposed TOM approach in improving these properties.

A mean square error (MSE) based metric was chosen in order to quantify the topology regularity improvement for the observed discrepancy between the hop and Euclidean distances. The metric is calculated as follows. The topology induced hop distance is subtracted from the actual Euclidean distance result value is squared. Finally all square values, for all deployed sensor nodes, are summed. The

metric measures the percentage reduction in MSE value before vs. after executing TOM (*Distance*). This percentage reduction is compared against the reduction achieved using the strategy based on random deployment of the same number of sensor nodes as added by TOM (*Rnd Distance*).

The energy balancing properties of TOM are quantified in similar manner. The energy balancing metric calculates the square value of the difference between the initial energy supply and remaining energy after performing an arbitrary number of data collection rounds and then sums the value for all deployed sensor nodes. As for previous metric also the energy balancing metric measures percentage reduction between original deployment and this after applying TOM (*Energy*).

The energy saving and latency reduction properties of TOM are shown by comparing the average hop distance to sink (using shortest path tree routing) before and after executing TOM (*Hop Avg*). Each saved message transmission translates into lower energy expenditure. A shorter route also means a lower latency.

The goal of the last two metrics is to quantify the achievements of TOM for applications that use the Voronoi abstraction Sharifzadeh and Shahabi [2004]; Szczytowski et al. [2010]; Zhou et al. [2009]. For that purpose, the sensor nodes compute the Voronoi diagram based on 2-hop neighborhood knowledge. The Voronoi computation is executed before and after completing the maintenance as proposed by TOM sensor nodes. In the former case, added sensor nodes are omitted while calculating the Voronoi diagram. They are only utilized for communication between original sensor nodes. The first metric (*False Lines*) is directed at measuring the proper identification of Voronoi neighbors Szczytowski et al. [2010]. The square value of misclassified Voronoi neighbors is summed up for all sensor nodes and the result is divided by the number of sensor nodes. The first metric measures the percentage reduction in these values before and after executing TOM. The second metric is an MSE based metric (*Surface MSE*), which shows how much the areas of the Voronoi diagrams differ from the optimal one Zhou et al. [2009]. From the properties of the Voronoi diagram implicitly follows, that the area size of polygons obtained using complete topology information is minimal. Therefore, as first step the metric calculates the area of each Voronoi diagram for each sensor node and subtract the area size of optimal Voronoi diagram. Then the square values are summed and divide by the total number of sensor nodes. Similar to the first metric, the second metric measures the percentage reduction in considered values before and after executing TOM.

### 4.5.3 Evaluation Results

The results are divided in two classes. The first class represents the impact of TOM on network-centric properties of the WSN. The second class shows the functionality-centric benefits of maintenance using TOM.

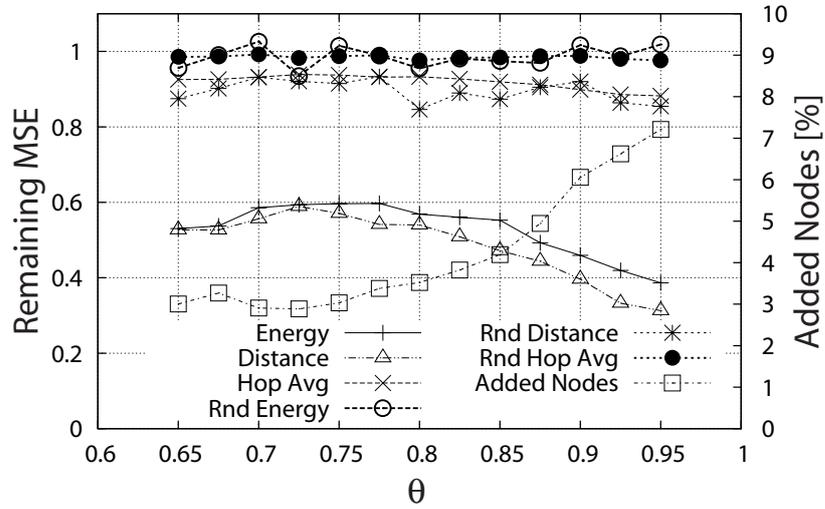


Figure 4.3: TOM Tunability to repair irregularities of varied severity

The first evaluated property is the tunability of TOM for controlling the extent of topology irregularities. Figure 4.3 shows that even for fairly relaxed requirements ( $0.65 < \theta < 0.75$ ) TOM reduces the MSE of *Energy* and *Distance* metrics to 60% of its original value, balancing the energy usage among the sensor nodes and hop distances to the sink. The reduction progresses further with the increasing  $\theta$  reaching in the end as low as 30% compared to the case without maintenance. These improvements of network properties come at the cost of increased resources demand, which over  $\theta = 0.85$  steeply increases. For most cases to maintain the network only approximately 3% - 4% of original number of deployed sensor nodes are required. That translates for the given setup to roughly 8 - 11 sensor nodes. Only in the extreme case of  $\theta = 0.95$  the need for resources reaches 7% corresponding to 19 sensor nodes. The average hop distance also decreases with increased  $\theta$  but not so rapidly like other metrics. Nonetheless, it equals to at least 10% decreases in hop distance. By average distance of 11 hops this corresponds to a drop of 1 hop from the route on average. The accumulated effect for all sensor nodes translates in the reduction of 270 transfer hops. The maintenance based on random strategies (*Rnd Distance*, *Rnd Energy*, *Rnd Hop Avg*) shows no improvement of the properties of the network.

The performance of TOM is also closely related to the network size (number of sensor nodes deployed). In order to illustrate this relation, the previous deployment settings are kept and the value of  $\theta$  is fixed to 0.85. Then, the network size changes gradually from 200 sensor nodes, generating very sparse but still connected networks, up to 300 deployed sensor nodes, representing relatively dense deployments. As concluded easily from Figure 4.4 the possibility to lower

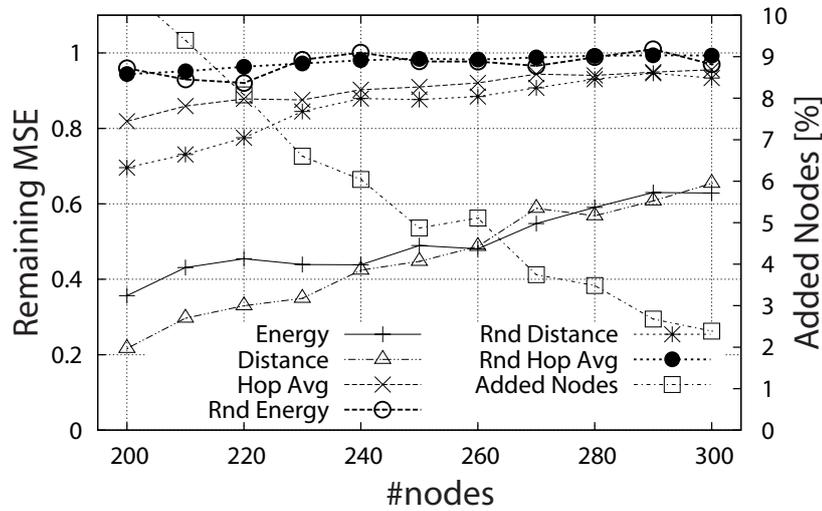


Figure 4.4: Impact of the network size

the MSE value of metrics decreases (*Energy*, *Distance*, *Hop Avg*) with the increase in the density of the network. The demand for resources follows these trends in reverse (*Added Nodes*), meaning that at higher node densities, less resources are needed to maintain a regular topology. The obtained results confirm the expectations. In a denser network the occurrences of irregularities are rare and therefore also the need for their reduction subsides. Even if irregularities appear their magnitude is limited and as a consequence they do not require large number of resources to tackle them. In contrast, the maintenance executed using the random strategy (*Rnd Energy*, *Rnd Distance*, *Rnd Hop Avg*) and deploying equal number of sensor nodes only marginally improves the topology of the network and remains mostly independent of the network density.

Next plot illustrates the deployment cost savings provided by TOM. For this purpose, additional sensor nodes are randomly deployed until the simulation reaches the same properties as achieved using TOM for a specified  $\theta$ . Then, the number of randomly added sensor nodes is calculated in comparison with the original deployment. Figure 4.5 clearly indicates that for any of the presented metrics, providing the same results as TOM by random dropping of sensor nodes is very expensive. For  $\theta$  ranging from 0.65 to 0.85 the demand for randomly deployed sensor nodes varies from 30% to 40% of the initial number of sensor nodes. For  $\theta$  larger than 0.85 the demand drastically increases from 70% for *#Rnd Distance* up to 80% for *#Rnd Hop Avg* to keep up with TOM. For the *#Rnd Energy* situation is a bit better, but still it requires over 40% and up to 50% of additional sensor nodes. For comparison, also the plots of the TOM approach were added. In this case, for the whole range of  $\theta$  and each applied metric, the demand for new sensor

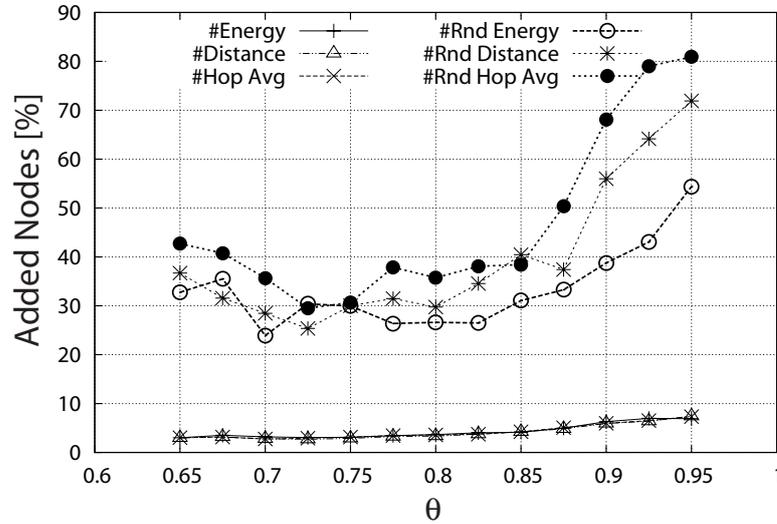


Figure 4.5: Comparison with random deployment strategy

nodes never exceeds 10%.

In order to show the functionality-centric aspects of maintenance, the evaluation scenario considers the impact of implementing the TOM strategy on the performance of the Voronoi diagram construction with 2-hop neighborhood knowledge. Figure 6.2 depicts the achieved performance for the applied metrics. The *False Lines* metric shows that TOM is capable of reducing its value close to 20% for lower  $\theta$  and over 25% for higher values of  $\theta$ . The higher number of sensor nodes placed to increase regularity results also in better communication in the sensor nodes neighborhood, allowing for better identification of the Voronoi neighbors. Random deployment also leads to reduction in the value of this metric but it significantly underperforms TOM. The same holds for the *Surface MSE* metric which shows how the Voronoi area size is approximated. For the values of  $\theta$  up to 0.75 the reduction of *Surface MSE* metric mimics that of *False lines* metric. For higher values of  $\theta$  the reduction progresses at faster pace and reaches up to 35% lower value of the *Surface MSE* metric. Also in this case TOM outperforms the random deployment (*Rnd Surface MSE*), even with larger margin.

## 4.6 Summary

As presented in this chapter, TOM is an efficient distributed strategy for topology optimization driven maintenance. Using localized information, TOM is capable of detecting the relevant and varied topology irregularities in a deployed WSN. In addition, TOM searches for the suitable placements of additional or redeployable

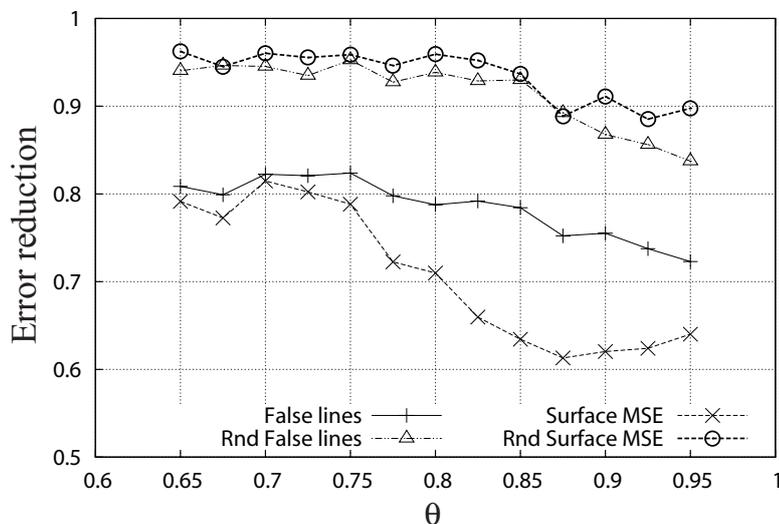


Figure 4.6: Impact on Voronoi diagram calculation

or mobile nodes that allow the optimization of the topology regularity and its maintenance. The immediate benefits from applying the proposed maintenance strategy are balanced energy usage, shorter routes, longer network lifetime and lower latency for message transport. Also the application functionality enhancing properties of TOM were shown by example of applications using the Voronoi abstraction to benefit from regular topology by providing better accuracy.



# Chapter 5

## Distributed $k$ -Connectivity Maintenance in Wireless Sensor Networks

### 5.1 Introduction

The reliability of Wireless Sensor Networks (WSN) is detrimentally impacted by unreliable wireless communication and the finite energy of sensor nodes. An advocated approach for assuring fault tolerant WSN is providing global  $k$ -connectivity. This property guarantees that the failure of up to  $k - 1$  sensor nodes does not cause network partitioning.  $k$ -connectivity is a well studied property of WSN including the aspects of topology control,  $k$ -connected dominating set construction, controlled deployment, relay nodes placement, and detection of level of  $k$ -connectivity. This chapter targets the repair/maintenance aspect of  $k$ -connectivity. The goal is to allow the network to provide localized, sustainable maintenance, which is capable of efficiently restoring/maintaining the WSN desired  $k$ -connectivity. The chapter present a fully distributed technique that is competitively resource efficient to state-of-the-art approaches. Unlike existing techniques, the presented approach also provides the necessary efficient mechanisms to avoid network partitioning and the longer routing paths caused by node failures. Both analysis and simulations show the effectiveness and efficiency of the presented solution to maintain high responsiveness.

#### 5.1.1 Contributions

Distributed  $k$ -Connectivity Maintenance (DKM):

- efficiently restores and preserves the  $k$ -connectivity property,

- eliminates potential WSN disconnections by assuring at least two disjoint paths between sensor nodes,
- provides k-connectivity along with guaranteeing that the length of particular routes may become extended only up to  $k - 1$  hops,
- is distributed and localization free.

The chapter is structured as follows. Following the system model in Section 5.2, Section 5.3 presents a precise formulation of our objectives and requirements. Section 5.4 details the proposed Distributed k-Connectivity Maintenance (DKM) technique as the chapter's main contribution. The evaluation is presented in Section 5.5.

## 5.2 System Model

Conforming to contemporary WSN models, the chapter assumes a WSN consisting of  $n$  resource constrained sensor nodes and a sink. The sink is a powerful sensor node connected to infrastructure with unlimited energy supply. Hence, it can be considered as a sensor node virtually stacked with an infinite number of backup sensor nodes. Without loss of generality, the sink is assumed to be an arbitrary sensor node that is stocked with at least  $k - 1$  additional backup sensor nodes. The sensor nodes have finite battery energy and usually possess limited processing and storage capabilities. The communication range  $R$  is limited and fixed for a given deployment. Two neighboring sensor nodes can communicate directly - establish a link - only if their Euclidean distance is smaller than  $R$  (it is noteworthy that this implies that we consider symmetric/undirected links). This communication dominates the imprint on the energy depletion of the sensor nodes. The network is modeled as a graph  $G = (V, E)$ ,  $V$  is the set of Vertices (sensor nodes),  $E$  is the set of Edges (existing links between sensor nodes for the specified  $R$ ). Sensor nodes are not required to know their positions, but available location information (e.g., as in He et al. [2005]) can positively influence the efficiency of the proposed algorithms. The presented approach considers cases where (a) all nodes are static, or (b) a mix of static and nodes of controlled mobility. Additionally, it is assumed that sensor nodes know their hop distance to the sink, e.g., based on a shortest path tree routing.  $N_i$  refers to the set of sensor nodes that are  $i$  hops from the sink ( $i \in \{0, 1, \dots, n\}$ ).  $N_0$  is the sink (or its equivalent virtual co-located  $k$  sensor nodes). Sensor nodes are aware of their 1-hop neighboring sensor nodes, including their position and hop distance to the sink. The message losses are not considered as these are typically handled at the message transport protocol level.

## 5.3 Objectives and Requirements

This section presents the background concepts that represent the basis of presented  $k$ -connectivity maintenance technique.

**Definition 1.** A Sensor Node  $L$  is called a closer neighbor of Sensor Node  $F$  if (a)  $L$  is a neighbor of  $F$  and (b)  $L$  has a smaller hop distance to the sink than  $F$ .

**Lemma 1.** If each sensor node has at least  $k$  closer neighbors then each sensor node has at least  $k$  independent paths to the sink.

*Proof.* Consider Sensor Node  $F \in N_i$ , i.e., placed  $i$ -hops from the sink.  $F$  has at least  $k$  neighbor nodes  $\in N_{i-1}$  that are 1-hop closer to the sink than  $F$  itself. That means that  $N_i$  can choose from  $k$  possible neighbors to send the data to the sink, so at least  $k$  sensor nodes have to be removed to disconnect  $F$  from all of  $N_{i-2}$  sensor nodes. For instance, each sensor node  $\in N_2$  has at least  $k$  paths to the sink. Therefore, by induction if  $F$  has  $k$  paths to the  $N_{i-1}$  then it also has  $k$  paths to the sink.  $\square$

**Lemma 2.** If each sensor node in a WSN has  $k$  independent paths to the sink, then there exist at least  $k$  paths between each pair of sensor nodes and as the result the WSN is  $k$ -connected.

*Proof.* If each sensor node has  $k$  independent paths to the sink, then (as undirected graph/network is considered) reverse is also true, meaning that there exist  $k$  independent paths from the sink to each of the sensor nodes. If each sensor node would send data to each other sensor node over the sink then there exist at least  $k$  independent paths that can be used for this purpose. As the sink is considered as an infinitely stocked node, or at least  $k$ -stocked node, then the WSN is  $k$ -connected.  $\square$

**Definition 2.** If Sensor Node  $F$  has an equidistant neighbor  $H$  (with same hop distance to the sink) and  $H$  has a closer neighbor  $K$  that is not a neighbor of  $F$  then  $K$  is called an indirect closer neighbor and  $H$  is called a semi-closer neighbor. Each equidistant sensor node can provide at most one indirect closer neighbor and only if that indirect closer neighbor is not already provided by other equidistant sensor node. The set of all indirect closer neighbors of sensor node  $F$  is referred by  $ICN(F)$  and the set of all closer neighbors  $CN(F)$ .

**Definition 3.** Support Nodes are any closer or indirectly closer neighbors. The set of Support Nodes of Node  $F$  is represented as  $SNS(F) = ICN(F) \cup CN(F)$ .

**Definition 4.** The set of sensor nodes that depend on routing support (towards the sink) of Sensor Node  $F$  are termed the Dependent Nodes Set of  $F$  ( $DNS(F)$ ).

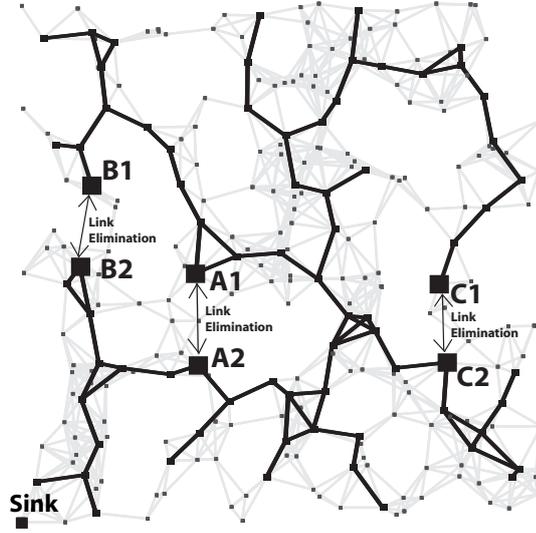


Figure 5.1: Single link elimination

**Theorem 1.** *If each sensor node has  $k$  or more support nodes ( $|SNS| \geq k$ ), then the WSN is  $k$ -connected.*

*Proof.* If a sensor node  $F$  has  $j$  (less than  $k$ ) closer neighbors and at least  $k - j$  semi-closer neighbors each of them providing one indirect closer neighbor, then  $|SNS| \geq k$ . For  $|SNS| \geq k$  there has to be at least  $k$  nodes  $\in N_{i-1}$  to be removed in order to disconnect  $F$  from nodes  $\in N_{i-2}$ , i.e., from the sink. Following the proof of Lemma 1 and concluding from proof of Lemma 2 if each sensor node  $\in N_i$  has a  $|SNS| \geq k$ , then the WSN is  $k$ -connected.  $\square$

In order to generalize the  $k$ -connectivity maintenance of our solution,  $r_F$  is defined as the effective redundancy factor of Node  $F$ . For classical  $k$ -connectivity approaches,  $r_F$  is equal to the number of sensor nodes stored at the location of  $F$ . In a more generic approach  $r_F$  may represent a variety of resource redundancy types such as additional batteries, redundant hardware components on the same node, etc. Accordingly, the values of  $r_F$  are not necessarily considered integer but real values.

**Approach Requirements:** This chapter uses the proposed definitions, lemmas and Theorem 1 to describe the design of an efficient  $k$ -maintenance technique. The main requirement is to achieve this goal with minimal supplemental resources measured by the additional required  $r_F$  to maintain  $k$ -maintenance. The second requirement is to maintain shorter routes from sensor nodes to the sink in order to avoid degrading the WSN responsiveness. In order to better illustrate the problem, Figure 5.1 presents a WSN with marked Minimum Connected Dominated Set (MCDS). The  $k$ -connectivity in this case is provided when each of the sensor

nodes belonging to MCDS is stocked with  $k - 1$  additional sensor nodes. As can be easily observed, although  $k$ -connectivity is assured, a failure of even less than  $k$  nodes from among those connecting pairs (A1; A2), (B1; B2) or (C1; C2) may result in overly extended routing paths from A1, B1 or C1 to the sink.

## 5.4 k-Connectivity Maintenance

After an overview of the proposed approach, this section details a technique for localized verification of the necessary conditions to provide  $k$ -connectivity. Subsequently, the section describes a method on how to deploy a minimal number of resources in order to restore the  $k$ -connectivity property. Additionally, an optimization mechanism is presented, which aims to eliminate co-located links. The section ends with discussion about the self-sustainability aspect of the installed  $k$ -connectivity.

### 5.4.1 Guide through our DKM Approach

The key idea behind our approach is based on Theorem 1. If each sensor node has at least  $k$  support nodes (i.e.,  $|SNS| \geq k$ ) where  $SNS(F)$  not necessarily assumes an integer value, as it is the sum of redundancy factors  $r$  of its members. Then each sensor node has at least  $k$  alternative/disjoint paths to reach the sink, leading to that the WSN is  $k$ -connected. Accordingly, sensor nodes require only localized knowledge to verify if the condition ( $|SNS| \geq k$ ) is met. Every sensor node, for which  $|SNS| < k$  should check if all of its supporting nodes themselves fulfil this condition. If at least one member of  $SNS$  does not meet the condition, then the sensor node waits until all of its  $SNS$  members themselves have at least  $k$  support nodes. At that point each sensor node can locally try to resolve the situation using a min-max approach. A sensor node  $D$  sends the preliminary request for adding  $k - z$  additional resources ( $r_D$ ) to all of their  $SNS(D)$  members, where  $z = |SNS(D)|$ . The  $SNS(D)$  members collect preliminary requests from all of their children and calculate the minimum. The minimum is sent as response to all the children that sent a request. Each child selects from all the responses maximal value and sends the commit request to the parent that submitted the maximum response. After each such iteration at least one, but generally most of the sensor nodes will get enough additional support nodes to meet the condition of having  $k$  alternative paths.

## 5.4.2 Localized Estimation of k-Connectivity

The first step in DKM is for each Sensor Node  $X$  to calculate the  $|SNS(X)|$ . If  $|SNS(X)| \geq k$  then  $X$  is called a Resolved Node (RN) otherwise Unresolved Node (UN). *Algorithm 5* describes a distributed method for calculating  $|SNS(X)|$ .  $X$  checks for each of its neighbors, the routing induced hop distance to the sink. If the neighbor is placed farther from the sink than  $X$  itself, then such a sensor node is ignored. Each neighbor that is placed closer to the sink than  $X$  increases  $|SNS(X)|$  by the resource redundancy factor  $r_{neighbor}$  of the considered neighbor's location. The neighbor identifier (ID) is also added to a temporary set called *closerNeigh* to exclude considering this node as indirect closer neighbor.

If a neighboring Sensor Node  $Y$  is equidistant to  $X$  then the neighbors of  $Y$  are further investigated. If  $Y$  possesses neighbors closer than  $X$  to the sink and not yet included in *closer-Neigh* then these neighbors are added to *closerNeigh* and they contribute to  $|SNS(X)|$  by the minimum of  $r_Y$  and the sum of the marked closer  $Y$ 's resource redundancy factors.

After *Algorithm 5* sensor nodes can identify if they are resolved (i.e., the resources of closer neighbors are higher than  $k$ ) or not. Unresolved nodes need to execute the resolving algorithm. The message exchange cost of executing *Algorithm 5* per sensor node depends on allowed message size but, in general, is limited to two messages. In order to execute the algorithm, it is enough for each sensor node to inform its neighbors about its ID, hop distance to sink and list of IDs of its neighbors along with their resources whose hop distance to the sink is smaller than its own. This process necessarily happens in two steps, each requiring sending a single message. First, each sensor node sends only its ID and hop distance. This broadcast message is received by all direct neighbors in a single transmission. Each sensor node receives the messages from its neighbors and creates the list of neighbors whose hop distance to the sink is smaller than its own. Next, each sensor node sends a second message containing the constructed list. After receiving this second message each sensor node has sufficient information to perform *Algorithm 5*.

**Algorithm 5** Local Connectivity Resolution at each node

---

```

1: function estimateLocalConnectivity(curNode) : float
2: var float closerResources = 0;
3: var set closerNeigh =  $\emptyset$ ;
4: for all sn in curNode.Neighbors do
5:   if sn.hop < curNode.hop then
6:     closerNeigh  $\leftarrow$  closerNeigh  $\cup$  sn; //add to SNS
7:     closerResources += sn.resources;
8:   else if sn.hop == curNode.hop then
9:     var float indirectResources = 0;
10:    for all snn in sn.Neighbors do
11:      if snn.hop < curNode.hop and snn  $\notin$  closerNeigh then
12:        closerNeigh  $\leftarrow$  closerNeigh  $\cup$  snn; //add to SNS
13:        indirectResources += min(sn.resources, snn.resources);
14:        if indirectResources  $\geq$  sn.resources then
15:          break; //indirect neighbors cannot provide more resources than
            direct ones
16:        end if
17:      end if
18:    end for
19:    closerNeigh  $\leftarrow$  closerNeigh  $\cup$  sn; //add to SNS
20:    closerResources += min(indirectResources, sn.resources);
21:  end if
22: end for
23: return closerNeighCount;

```

---

**5.4.3 Resolving Nodes**

A sensor node, say  $F$ , initiates the resolving process when the Local Connectivity Resolution Algorithm (Algorithm 5) returns a value lower than the desired  $k$ . This means that  $F$  needs to inform current members of  $SNS(F)$  about the required resources to itself become a resolved node. The presented resolving process utilizes voting. First, the sensor nodes whose complete SNS is resolved, such as  $F$ , send the demand for the resources as a vote request (vote representing the amount of missing resources  $r_F$ ). The SNS nodes gather the votes and sum them. Simultaneously, they additionally calculate the minimal value of votes (the minimal resources required to resolve at least one node). The SNS nodes send response to their DNS (from Definition 4,  $F \in DNS$ ). Each DNS node (e.g.,  $F$ ) selects from the responses (only from closer neighbors) the sender, which responded with highest accumulated votes. The highest accumulated vote indicates that this sender is

the one that is shared by most of sensor nodes and therefore stocking it with additional resources will be profitable for increasing local connectivity of unresolved nodes. This sender receives a request for stocking additional resources equal to the minimal vote that it received during voting. This measure guarantees that, at each step, at least one sensor node becomes resolved to progress the process and it minimizes chances of overstocking the sensor nodes.

The resolving algorithms (Algorithm 6 & 7) are detailed using two threads running at each sensor node (the actual implementation does not require threaded execution, this abstraction is used just for a better explanation of the algorithm). Each sensor node can be a member of a DNS or a SNS of some other sensor nodes. Algorithm 6 describes the sensor node behavior while acting as a member of a DNS and Algorithm 7 the situation when the sensor node is a member of a SNS.

---

**Algorithm 6** Resolve Node: As member of DNS

---

```

1: function resolveNodeThread(curNode, k)
2: while not checkIfSNSResolved(curNode, k) do
3:   waitForNeighborNotification(); //wait until SNS is resolved, does not in-
   volve sending any messages
4: end while
5: while estimateLocalConnectivity(curNode) < k do
6:   var vote;
7:   vote.value = k - estimateLocalConnectivity(curNode);
8:   vote.sender = curNode;
9:   for all sn in closerNeigh do
10:    sendVote(sn, vote); //ask for resources
11:   end for
12:   var maxResponse;
13:   for all response in collectResponses() do
14:     if response.sender.hop < curNode.hop and
       response.value > maxResponse then
15:       maxResponse = response; //choose best bidder
16:     end if
17:   end for
18:   sendSelection(maxResponse.sender);
19: end while
20: notifyNeighbors("curNode resolved");

```

---

Before F can start resolving its status, it checks whether all of the members of its SNS are already resolved (Algorithm 6, Line: 2). Two situations are possible: (a) All members of SNS(F) are resolved as a result of already available

resources, (b) some sensor nodes belonging to  $SNS(F)$  are not yet resolved and  $F$  has to wait until they resolve their status. In Situation (a), the algorithm may execute in parallel in several regions of the WSN. Not resolved nodes notify their neighbors when their status changes to resolved (Algorithm 6, Line: 20), so the sensor nodes whose  $SNS(F)$  is not completely resolved may easily detect changes (Algorithm 6, Line: 3). When all  $SNS(F)$  are resolved  $F$  may start sending its votes.  $F$  calculates its vote by subtracting the value of its local connectivity from the desired  $k$  (Algorithm 6, L. 7). The vote is sent only to closer neighbors as only those nodes will be considered for adding resources.  $SNS(F)$  members collect the votes (Algorithm 7, Line: 3) and calculate their sum and value of the minimal vote (Algorithm 7, Lines: 6 - 7). After collecting all votes  $SNS(F)$  nodes send the response to all of the voters (including  $F$ ) (Algorithm 7, Lines: 10 - 12).  $F$  collects the responds and selects the responder with the highest sum of votes (Algorithm 6, Lines: 13 - 17). Then,  $F$  sends to this responder a request for adding additional resources.  $SNS(F)$  members wait for *maxDelay* time for selections from their DNS members. Once, they receive them, they increase their  $r$  for the value of the minimal vote received in the voting step (Algorithm 7, Lines: 13 - 15). After  $SNS(F)$  nodes add resources,  $F$  re-evaluates its condition (Algorithm 6, Lines: 5).

The message cost of each iteration (of the resolving node's algorithm) requires from DNS nodes the sending of two messages (vote and selection messages) and from SNS nodes only a single response message. The number of iterations is limited and depends on the discrepancy between the current and desired connectivity levels. In case of  $k$ -connectivity, at each iteration the discrepancy is reduced by at least 1. Additionally, the algorithm chooses for stocking those nodes whose stock increases the benefits for possibly many unresolved nodes. Pragmatically,  $k$  is generally a low number and the maintenance action is required sporadically, the number of message transmissions required for resolving nodes is small and limited.

---

**Algorithm 7** Resolve Node: As member of SNS

---

```

1: function voterDeamonThread(curNode, k)
2: while true do
3:   var votes = collectVotes()
4:   var response;
5:   for all vote in votes do
6:     response.value += vote.value;
7:     response.minVote = min(response.minVote, vote.val);
8:   end for
9:   response.sender = curNode;
10:  for all voter in voters do
11:    sendResponse(vote.sender, response) //actually only single message is
    sent as all recipients are direct neighbors
12:  end for//wait maxDelay for selections
13:  if collectSelections(maxDelay) !=  $\emptyset$  then
14:    curNode.stock += response.minVote;
15:  end if
16: end while

```

---

#### 5.4.4 Avoidance of Bottleneck (Overlapping) Links

When the SNS of a sensor node (e.g.,  $Q$  in Figure 5.2) consists of a single node ( $V$ ) then it is necessary to add an additional sensor node ( $Z$ ) (extend the SNS) that is not co-located/stocked with  $V$ . If  $V$  was stocked with additional  $k - 1$  sensor nodes, then although  $k$ -connectivity would be provided, a single failure related to location (e.g., sensor nodes burned, trampled under larger objects, stolen, etc.) could cause immediate network disconnection. For this purpose,  $Z$  should allow to forward data to the sink independent form the existing single link (over  $V$ ).  $Z$  itself should become  $k$ -connected. In order to ensure  $k$ -connectivity of  $Z$ , it is enough to select a new location that is intersection of enough of  $SNS(V) = \{S, P\}$  such that their summed up stocks are at least equal to  $k$ . From the possible location, the one that is placed farthest from  $V$  but still in  $Q$  range should be selected.

#### 5.4.5 Optimizing Required Resources

A sensor node, say  $X$ , may be over-provisioned following Algorithm 7, i.e., if  $r_X$  is larger than required by all sensor nodes  $\in DNS(X)$  for maintaining  $k$ -connectivity. Subsequently, the proposed technique identifies resource over-provisioning in order to ensure minimal cost maintenance. The calculation of over-provisioning is conducted as follows:

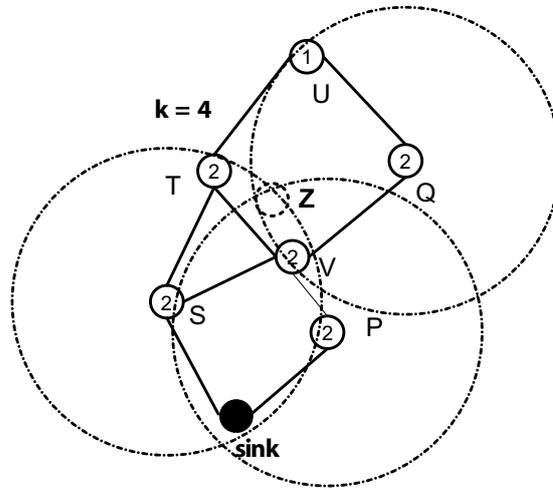


Figure 5.2: Single Link Elimination

1. Each sensor node sends to its SNS nodes the value indicating the difference between its local connectivity and desired  $k$  ( $overK$ ).
2. Each SNS node collects the  $overK$  values and calculates their minimum value ( $overK_{min}$ ).  $overK_{min}$  indicates the amount of over-provisioned resources whose removal would not cause a drop in local connectivity in any of DNS nodes of SNS below  $k$ . Each SNS node responds by sending to its corresponding DNSs a value equal to calculated  $overK_{min}$  or number of its own stoked resources less one.
3. DNS nodes receiving a response from their SNS select the sensor node ( $SNS_B$ ) that sent the highest  $overK_{min}$ . If two sensor nodes send equal  $overK_{min}$  then the node identifier (ID) breaks the tie. Each DNS node sends the ID of its  $SNS_B$  back to its SNS.
4. If all received IDs are equal to the ID of receiving node than this sensor node can remove the proposed number of stocked nodes.

In case of limited resources, the sensor nodes after calculating missing resources could execute a bidding process Dini et al. [2008], where the weight of the bid corresponds to the closeness to sink (to prevent partitioning close to sink, where it could disconnect much larger part of the network) or to the  $|DNS|$  of bidding sensor node.

### 5.4.6 Self-Sustaining $k$ -Connectivity

If the scenario of supplying the WSN with additional resources (e.g., batteries, sensor nodes, etc) is not possible but the network contains mobile elements, the mobility can be used for topology maintenance. The following strategy is proposed to rearrange sensor nodes in order to sustain the  $k$ -connectivity in face of failures. For this purpose, the WSN needs to identify sensor nodes whose repositioning does not impair the  $k$ -connectivity property. Each of the sensor nodes that do not belong to SNS of any other node are called leaf nodes and can safely move to the locations selected by DKM. Each removal of leaf nodes creates potentially new leaf nodes. In Figure 5.2, Sensor Node U is a leaf node as it does not belong to any SNS. When U gets removed, then as consequence sensor nodes T and Q become leaf nodes, as they were members only of SNS(U). The leaf sensor nodes can be iteratively removed and used to repair other parts of the network until remaining topology becomes  $k$ -connected. It is evident that as this process progresses the deployment area accordingly shrinks. Therefore, the user should balance the priorities between sensing coverage and fault tolerance. If sensing coverage is given higher priority then there should be assigned sensor nodes whose removal is not permitted. This however may impair the self-sustainability capacity of the algorithm as it may run out of resources.

### 5.4.7 Graph Balancing

A further aspect to consider is the scenario is where the end-to-end traffic is not bound to the sink but is also desired between sensor nodes. DKM also works in such scenarios. For this purpose it reverts to the original concept presented in Almasaeid and Kamal [2009], with a slight modification. Instead of creating MST, the preferred method is to create MCDS as this results in selecting fewer sensor nodes, which needs to be stocked but keeps the crucial property of MST for assuring  $k$ -connectivity. Figure 5.3 shows an example of WSN with marked sub optimal MCDS using the algorithm described in Butenko et al. [2003]. This algorithm was chosen for further consideration, as it allows distributed execution, nonetheless also further techniques may equally be used to select MCDS nodes. Having established the MCDS, DKM in this scenario refrains from stocking  $k - 1$  sensor nodes on the location of MCDS nodes as proposed in Almasaeid and Kamal [2009]. Instead, a virtual sink is selected. A virtual sink is a sensor node belonging to MCDS, which assumes role of the sink under the condition of having stocked at least  $k$  sensor nodes. The position of the virtual sink should be placed possibly close to the geographic center of the deployment area, so that the average hop distance to this sink is kept minimal (in uniform networks), and so that in case of failures, when the data has to be rerouted through the virtual sink the extended

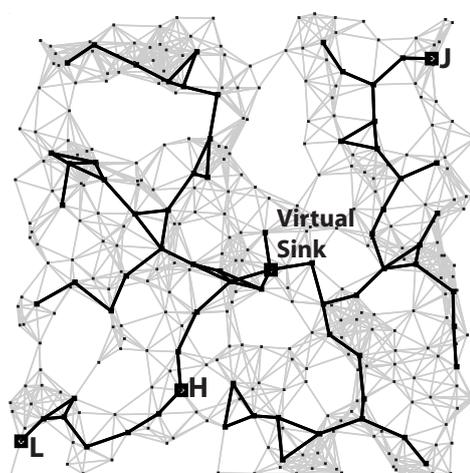


Figure 5.3: Virtual Sink Selection

routes will be kept possibly short. After finding the proper virtual sink DKM proceeds as in the real scenario.

The following strategy is proposed to establish the position of the virtual sink. The sensor node with the highest ID (Figure 5.3 sensor node H) is allowed to create the routing spanning tree among the members of the MCDS set. As a result the members of the MCDS set establish their hop distance to H. The sensor node with the highest hop distance (Figure 5.3 Sensor Node J) in uniform network is placed at the border of the deployment area. Now J is allowed also to start routing spanning tree construction (over MCDS) so that the sensor nodes can establish their hop distance to J. The sensor node with the highest hop distance to J is marked as L (Figure 5.3). L also starts spanning tree construction among MCDS nodes. After this step each sensor node has established the hop distance to both opposing border sensor nodes J and L. In order to find the center of the MCDS, algorithm looks for the sensor node, whose value  $(hop(J) - hop(L))^2$  is the lowest. This sensor node is selected to function as a Virtual Sink. Although the spanning tree construction algorithm executes three times, the number of the messages is kept low as in construction of the tree participate only the MCDS nodes. Therefore, for the whole process each MCDS node sends only three messages. It is noteworthy that as the simulations have showed, the selection of sink location at the center of the deployment area does not increase or decrease the demand for the resources. Establishing the virtual sink at a central location only serves the purpose of decreasing (only in case of failures) hop distances in case of sensor node to sensor node communication scenarios. For the construction cost of MCDS are discussed in Butenko et al. [2003].

## 5.5 Performance Evaluation

### 5.5.1 Evaluation Settings

The evaluation scenario considered a WSN network consisting of sensor nodes, with a communication range of  $R = 3m$ , deployed over the area of  $30m \times 30m$ . The number of deployed sensor nodes gradually changed from 230 (sparse scenario - low initial  $k$ -connectivity) to 750 (dense scenario - high initial  $k$ -connectivity) in order to measure the utility of DKM under different network densities and initial conditions. For each of these settings, also the desired  $k$  varied from 2 to 10. These values were chosen to cover a wide range of real deployments and future application requirements. An additional study was performed to measure the effectiveness of DKM under varying initial conditions induced by changing communication range. The number of deployed sensor nodes was kept at 350 sensor nodes and communication ranged was varied from 2.5m to 10m. Also in this case the desired  $k$ -connectivity was varied from 2 to 10. All sets of simulation were executed for two situations. In Situation 1, with no localization available the single link elimination step was skipped. The Situation 2, additionally considered handling the existence of bottleneck overlapping links and resource over-provisioning. The DKM efficiency was measured using stand-alone implementation.

DKM was compared to the solution postulated in Almasaeid and Kamal [2009] based on MST/MCDS approach. The comparison with Bredin et al. [2005] can be found in Almasaeid and Kamal [2009]. The general conclusion from the comparison follows that the approach in Bredin et al. [2005] requires substantially more sensor nodes to assure the  $k$ -connectivity. The amount of resources required to be added in order to assure desired  $k$ -connectivity was chosen as the driving metric for measuring the effectiveness of DKM. For this purpose, after the each execution of DKM the percentage increase in number of sensor nodes in reference to the originally deployed network was calculated. The same operation was also performed for the MST/MCDS based algorithm.

### 5.5.2 Evaluation Results

Figure 5.4 presents the demand for resources using the DKM and MST/MCSD algorithms in function of the desired  $k$ -connectivity. For the sparse topology (230 sensor nodes) the trends of both algorithms show linear characteristics separated by a margin starting at approximately 30% sensor nodes for  $k = 3$  and dropping to around 10% sensor nodes for  $k = 10$ . In case of denser initial deployments (500 and 750 sensor nodes), DKM changes to an exponential trend but with small enough exponent to stay well below the linear trend of MST/MCSD based solu-

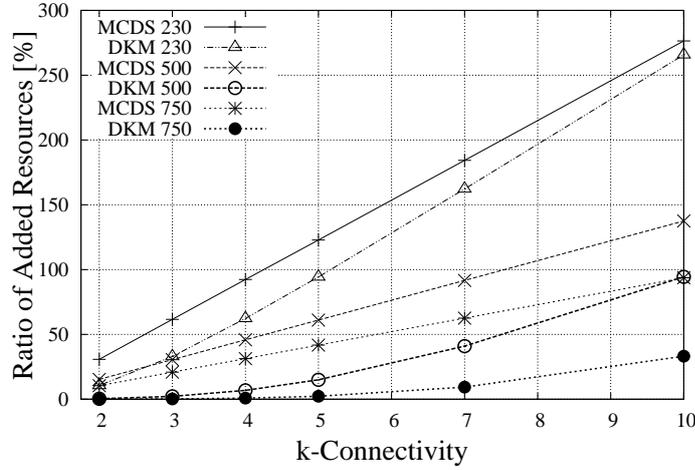


Figure 5.4: Ratio of added resources in function of required  $k$ -connectivity

tion for the whole spectrum of tested  $k$ -connectivity. For higher densities, DKM requires around 50% less of initial sensor nodes to add. This is due to the fact that an increase in density does not translate to reducing the number of members of the MST/MCDS sets as compared to increasing the initial connectivity of sensor nodes. DKM exploits this fact and tries to reuse the already deployed sensor nodes in order to reduce the demand for additional resources.

Figure 5.5 presents the demand for resources for fixed value of  $k$  and varying initial conditions. For each value of  $n$  DKM outperforms the reference algorithm. The discrepancy in performance for benefit of DKM rises with the rising number of initially deployed sensor nodes  $n$ . The initial lower discrepancy can be explained from the reasoning that in initially sparse networks the DKM operates similar to the MCDS/MST algorithm. Similarity in the discrepancy arises because in sparse networks the spanning routing tree, which DKM uses, has only few alternative parents for each sensor node. As the density of network rises with higher number of deployed sensor nodes DKM benefits from the availability of multitude of supporting nodes in the routing tree.

Another important property of WSN is the wireless communication range  $R$ . Figure 5.6 presents the ratio of added resources in function of  $R$  for a fixed number of deployed sensor nodes  $n = 350$ . For low values of  $R$  the network is sparse DKM outperforms MCDS/MST based solutions. The performance advantage of DKM rises with  $R$  but only up to approximately  $R = 4m$  and then slowly converges close to 0%. For high value of  $R$  (10m) in the considered scenario maximal hop distance is very low and that explains the outperformance of DKM, but also MCDS/MST involves only few additional nodes, as sensor nodes have a high con-

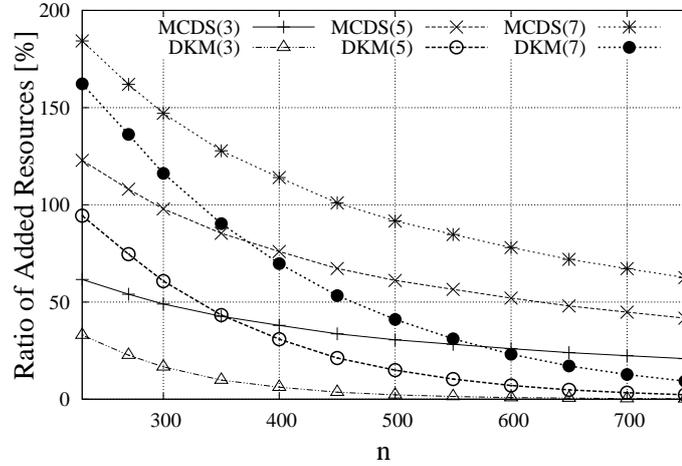


Figure 5.5: Ratio of added resources in function of number of deployed nodes

nectivity degree.

Now, the evaluation presents the additional cost associated with avoidance of bottleneck (overlapping) links. Figure 5.7 shows trend lines for DKM with and without avoidance of bottleneck (overlapping) links for two density scenarios. In case of lower initial density (230 sensor nodes) there exists noticeable difference in performance of both approaches. Adding additional sensor nodes for purpose of avoidance of bottleneck (overlapping) links increases demand for resources and is performed much more often in sparse networks. For sparse networks there is higher probability for sensor nodes of having initially only single supporting node. For higher density scenario (500 sensor nodes) the difference between both approaches is unnoticeable, as initially single links appear only rear. For reference, also MST/MCDS plots were provided, which show clearly that DKM outperforms MST/MCDS strategies independent from applying the single link elimination strategy.

Finally, the evaluation considers the established average local  $k$ -connectivity indicator using Algorithm 5. For each originally deployed sensor node,  $|SNS|$  was calculated before and after executing DKM. Afterwards, the average value for both cases is calculated. Figure 5.8 shows the average local  $k$ -connectivity indicator in function of originally deployed sensor nodes for  $k \in \{3, 5, 7\}$ . The basic trend for the original topology of WSN (no changes) is consistently linear and shows inherent localized  $k$ -connectivity indicator for a network of a given density. If  $k = 3$ , the trend only initially differs from base trend and for network size  $n = 400$  and denser it nearly merges. For the low values of  $k$ , the inherent  $k$ -connectivity is sufficient to provide the requested fault tolerance. Therefore,

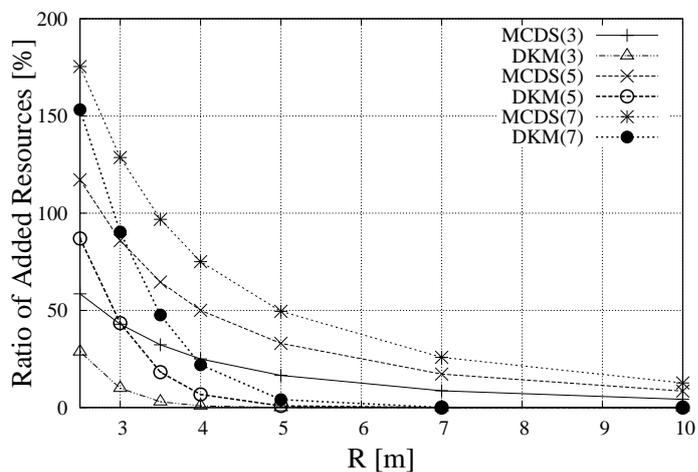


Figure 5.6: Ratio of added resources in function of communication range

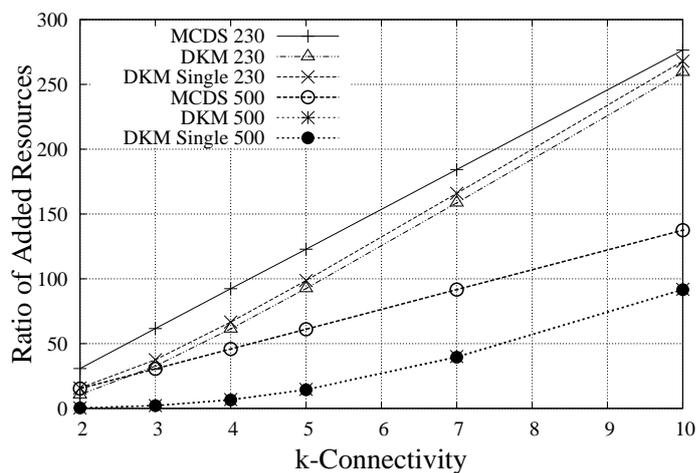


Figure 5.7: Impact of the elimination of bottleneck overlapping links

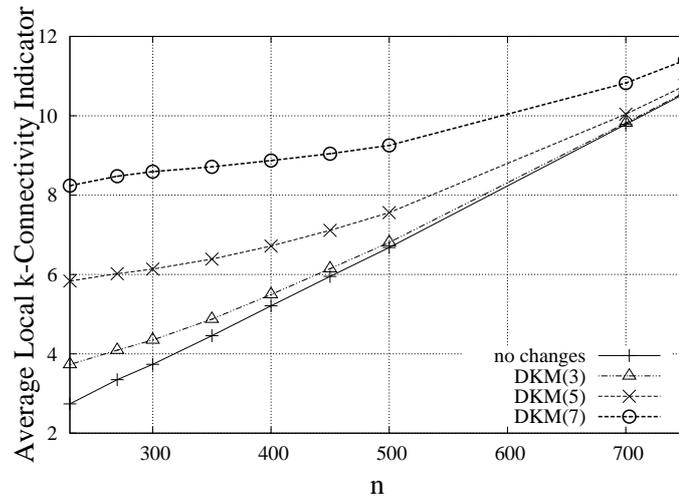


Figure 5.8: Average local  $k$ -connectivity indicator

there is a limited need for additional resources so that the local  $k$ -connectivity does not increase. It is different for larger  $k \in \{5, 7\}$  where especially in initially sparse topologies there is a low inherent fault tolerance. It is obvious that the average local  $k$ -connectivity indicator has to increase to at least the value of  $k$ , what explains the initial discrepancy from the base trend. But as can be observed from lines (DKM(5) and DKM(7)) the achieved local  $k$ -connectivity indicator is only marginally higher than the requested  $k$ , confirming the efficiency of DKM.

## 5.6 Summary

This chapter has presented DKM, an efficient distributed  $k$ -connectivity maintenance technique. While using only local views, DKM is capable of restoring global  $k$ -connectivity property. The presented solution, while not optimal, offers substantially better resource utilization performance. The algorithm allows also the detection of redundant sensor nodes, which can be used for performing maintenance. Mobile sensor networks can use this principle for providing self-sustainability.

# Chapter 6

## Adaptive Spatial Sampling in Wireless Sensor Networks

### 6.1 Introduction

A prominent application of Wireless Sensor Networks is the monitoring of physical phenomena. The value of the monitored attributes naturally depends on the accuracy of the spatial sampling achieved by the deployed sensors. The monitored phenomena often tend to have unknown spatial distributions at pre-deployment stage, which also change over time. This can detrimentally affect the overall achievable accuracy of monitoring. Consequently, reaching an optimal (accuracy driven) static sensor node deployment is generally not possible, resulting in either under- or over-sampling of signals in space. The goal of this chapter is to provide for adaptive spatial sampling. The key challenges consist in identifying the regions of over- or under-sampling and in suggesting the appropriate countermeasures. This chapter proposes a Voronoi based adaptive spatial sampling (ASample) solution. This approach removes unnecessary samples from regions of over-sampling and generates additional new sampling locations in the under-sampling regions to fulfill specified accuracy requirements. Simulation results show that ASample significantly and efficiently reduces the mean square error of the achieved measurement accuracy.

#### 6.1.1 Contributions

Adaptive Spatial Sampling (ASample):

- efficiently proposes additional sampling locations to meet required accuracy,
- identifies redundant sensor nodes in regards to required accuracy,

- allows for adapting the network sampling resolution to changes in monitored phenomena.

The chapter is structured as follows. Following the system model in Section 6.2, the problem formulation is presented in Section 6.3. Section 6.4 details the proposed adaptive spatial sampling (ASample) technique as the chapter's main contribution. The evaluation of ASample is presented in Section 6.5.

## 6.2 System Model

Conforming to the established WSN models, the employed system model assumes a WSN consisting of a large number  $n$  of resource constrained sensor nodes and a sink. The sensor nodes have finite capacity battery power. The communication range  $r$  is limited, and two neighboring sensor nodes can communicate only if the Euclidean distance between them is smaller than  $r$ . The communication dominates the imprint on the energy depletion of the sensor nodes. The only assumption made regarding the sensor nodes distribution in the network is that the full area is covered by these nodes and the resulting WSN is connected. The sensor nodes know their position using on-board GPS receivers or alternative GPS-free techniques of localization He et al. [2005]. Three distinct mobility classes are considered where (a) all nodes are static, (b) all nodes can move, or (c) a mix of static and mobile nodes. The proposed approach implicitly tolerates sensor node failures as this is equivalent to moving a node away from its position or putting it in sleep mode. Message loss will be explicitly considered while evaluating our approach. The algorithm investigates arbitrary continuous two dimensional signals in the sensor field.

## 6.3 Problem Formulation and Objectives

For environmental monitoring applications it is crucial to reconstruct the spatial distribution of the signal with an acceptable accuracy. This task may be transformed into specifying a desired maximally allowed value difference between any two neighboring sampling points  $Acc_{TH}$ , which is referred to by accuracy requirement. Therefore, it is important to bound the value difference between the measurements of any sensor nodes and its closest neighbors (in each direction on the plane) and consequently to reduce the possible variations of the signal.

Consider three sensor nodes A, B and C, located in each others communication range (Figure 6.1). Node A, B and C measure values  $S_A$ ,  $S_B$  and  $S_C$ , respectively. The Figure 1 distinguishes five, among infinitely many, interesting possibilities for the signal shape between A, B and C (curves 1 - 5 in Figure 6.1).

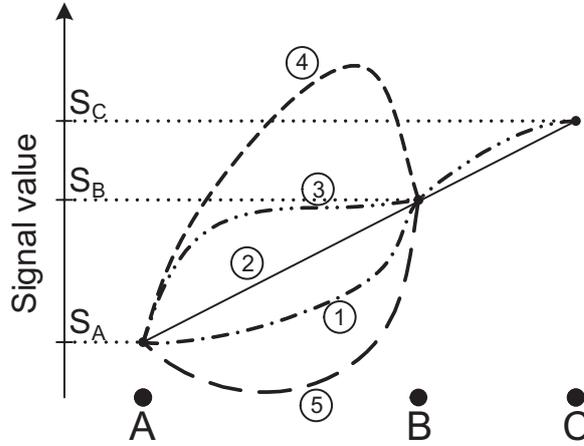


Figure 6.1: Variants of signal distribution

If  $S_B - S_A > Acc_{TH}$ , then the accuracy requirement is violated by all five signals. This can be easily detected if sensor nodes share their value with their closest neighbors. It is now crucial to select supplementary sampling locations where adding/relocating the sensor nodes makes WSN adhere to the requirements of monitoring accuracy. The algorithm should minimize the number of resources needed for the reconfiguration.

If  $S_C - S_A < Acc_{TH}$  the sample of Node B is redundant. Therefore, our holistic technique should help identify such redundant sensor nodes, which can be used for the re-deployment in under-sampled regions.

If  $S_B - S_A < Acc_{TH}$ , then the accuracy requirements may still be violated by signals such as Signals 4 and 5. This situation can only be detected through additional measurements, which implies a more dense pre-deployment of sensor nodes. The detection of such local minima or maxima is not the focus of this contribution. However, proposed approach is applicable if such local extrema can be detected.

Overall, our techniques react to the changes in the monitored area as the result of dynamics of the phenomenon.

## 6.4 The Adaptive Spatial Sampling Approach

This section first describes the necessary preliminaries concerning the Voronoi diagrams as they form the basis for our approach for efficiently modeling of the sensor nodes direct neighborhood. Subsequently, it presents an overview of proposed approach, and then detail profiling and reconfiguration techniques in order to provide for adaptive spatial sampling.



communication cost to construct it. Not all of the positions of 2-hop neighbors may be relevant for calculating the Voronoi diagram of a sensor node. In order to limit the amount of information transmitted, the sensor nodes may pre-compute, which of their neighbors are relevant for submission as the 2-hop neighbors, e.g. Node  $VG_5$  (Figure 6.2) may omit transmitting the position of Node H to Node C. Node  $VG_5$  knowing the positions of  $VG_4$  and  $VG_6$  may conclude that Node H cannot be a Voronoi neighbor of Node C.

### 6.4.2 Overview of our Approach

The first step towards an adaptive spatial sampling is for the sensor nodes to locally check the fulfillment of the accuracy requirements. It is important to perform this in an efficient distributed manner as each redundant communication message reduces the battery lifetime of the involved sensor nodes. This detection is based on the knowledge of sensor nodes Voronoi diagram, which efficiently models each sensor node closest neighborhood. Each sensor node evaluates whether any of its Voronoi neighbors sensor readings violates the required accuracy. In such case the sensor node decides to place a virtual node (VN) at the Voronoi edge in order to separate the Voronoi neighbors whose values variance violates the required accuracy. The value of the VN is linearly interpolated and the Voronoi diagram is rebuilt including the new VN. As the approach is heuristic, the nodes are initially added as VNs, as some of them may be redundant. The redundant VNs can be removed so the sensor node proposes only the necessary sampling locations. The process iterates until the accuracy requirements are met by adding the VNs. For the situation of an evolving phenomenon, the sensor nodes update their data to maintain accuracy.

### 6.4.3 The Proposed ASample Approach

The description of the approach continues with development of two algorithms - Algorithm 1 for *adding virtual nodes*, and Algorithm 2 for *removing redundant nodes* that collectively constitute the proposed ASample technique.

#### Adding Virtual Nodes to Under-sampled Regions

After the Voronoi construction phase is completed, the sensor nodes hold the list of their Voronoi neighbors. The sensor nodes evaluate the list in search of sensor nodes whose measurements differ from their own measurement by more than the requested accuracy  $Acc_{TH}$  (Algorithm 8, lines: 6-7). In case of detecting such disparity, both sensor nodes agree on adding the VN. The placement of the VN is decided as follows. If the Voronoi edge separating neighboring Voronoi cells is

a line segment (Figure 6.2 - Voronoi edge of nodes C and  $VG_5$ ), then the VN is placed in middle of this segment. The goal is to remove the Voronoi edge between the Voronoi neighbors violating the required accuracy. Placing the VN in the middle of the Voronoi edge maximizes the extent to which added VN separates the neighboring Voronoi neighbors (Alg. 8, line: 8). Otherwise, if the Voronoi edge is a half-line (Figure 6.2 - Voronoi edge of nodes H and R) then a VN is added at the beginning of the half-line, as it is not possible to calculate the mid-point of a half-line. Both sensor nodes sharing a common Voronoi edge will conclude the same decision of adding the VN. As newly added node is initially a virtual node, then at least one of these both sensor nodes must impersonate it. Such a node is referred as a *Proxy node*. The second node is referred to as a *Support node*. It has to be decided which sensor node will assume the Proxy or Support role. The simple solution is to choose as the Proxy node the sensor node which has a lower measurement value. As both nodes know each other values, they agree on the roles to assume.

Due to the heuristic nature of the Voronoi construction, the Proxy and Support nodes may result in different views on a common Voronoi edge. In this case, the Proxy node sends to the Support node its proposition of the VN placement, with its view on the Voronoi edge. If the views differ (Figure 6.2 - Node F view is segment  $VE_{F1}VE_{F2}$ , Node G view is segment  $VE_{G1}VE_{G2}$ ), then a new view is chosen so that it consist of a section of line segments shared by both sensor nodes. In this case that corresponds to the view of segment  $VE_{F1}VE_{G2}$  as the edge.

Adding a single VN between two Voronoi neighbors may not be sufficient to fulfill the accuracy requirement. This situation may arise from two reasons. The first results from the fact that the discrepancy between measured value may be greater than the double value of the accuracy threshold. In that case, placing the VN optimally between the Proxy and Support nodes, such it assumes a measurement value exactly between the values of Voronoi neighbors, still cannot resolve the inaccuracy problem. The second situation arises when the added VN still does not geometrically fully separates the Voronoi cells of the Proxy and Support nodes that detected the breach of accuracy requirements.

**Algorithm 8** Adding Virtual Nodes

---

```

1: var VoroNeighs=CreateVoronoi(self, 2_Hop_Neigh);
2: while (!CheckAccuracyOK());
3:
4: function CheckAccuracyOK() : boolean
5: var VirtualNodes;
6: for all VN in VoroNeighs do
7:   if abs(v(self) - v(VN)) > AccTH then
8:     VirtualNodes.Add(middle(self .pos, VN.pos));
9:   end if
10: end for
11: if VirtualNodes.Count > 0 then
12:   VoroNeighs=CreateVoronoi(self, VoroNeighs ∪ VirtualNodes);
13:   return FALSE;
14: end if
15: return TRUE;

```

---

Consequently, the algorithm continues to iteratively add VNs until the accuracy requirements are met (Alg. 8, line: 2). After adding the VNs, the Proxy and Support nodes include the positions of the added VNs to recalculate the Voronoi cells (Alg. 8, line: 12). The complexity of recalculating the Voronoi cells may be reduced by considering only the positions of sensor nodes that (at the previous step of iteration) were already Voronoi neighbors. An additional VN can only reduce the area of Voronoi cell. In order to recalculate the inaccuracy, the VNs also require to have some measurement value assigned. As the VN is added at equal distance between the Proxy and Support node, for this purpose their mean value is used.

It is also evident that the Voronoi neighbors of the Proxy and Support nodes should be informed about added VNs. Both Proxy and Support nodes calculate, which neighboring node's Voronoi neighborhood will be affected by adding VN and only these sensor nodes are notified. In order to further limit the amount of transmitted information, the Proxy and Support nodes inform only their set of closest common Voronoi neighbors. The notification sent to selected Voronoi neighbor nodes includes the position and the interpolated value of VN.

### Removing Redundant Nodes from Over-sampled Regions

As the described solution is based on a heuristic, it may add more VNs than required. In order to minimize the use of resources and the reconfiguration costs, it is desirable to find the redundant VNs. In the case of a network composed of only mobile nodes, it is important to find sensor nodes which could move freely while not violating the accuracy requirements of the WSN. The proposed algorithm allows each node to determine its redundancy status locally.

The candidate Node  $C$  to become redundant, iterates through the list of its Voronoi neighbors ( $VG_i$ ) (Alg. 9, line: 2). From Figure 6.2 it is evident that by removing Node  $C$ , each of the  $VG_i$  nodes Voronoi polygon will grow in size and gain new Voronoi neighbors. Concluding from the property of the Voronoi diagram, the new edges of extended polygons will be created with the current neighbors of  $C$ . Therefore, for each Voronoi neighbor,  $C$  calculates local Voronoi polygons using the positions of the rest of the Voronoi neighbors (Alg. 9, line: 4). Next, for each Voronoi neighbor it checks whether any of its new neighbors sensing measurement diverges more than the required accuracy (Alg. 9, line: 5). If it is the case then Node  $C$  cannot be removed (Alg. 9, line: 6). In case that the removal of Node  $C$  does not cause local inaccuracy, further evaluation takes place. Each sensor node informs its Voronoi neighbors about the possibility of its removal.

It may happen that, apart from Node  $C$ , some of its Voronoi neighbors may conclude that they are also potentially redundant nodes. At that point it is necessary to decide which sensor node ultimately will be designated redundant. We propose here a criteria based on the estimated size of Voronoi cell of each candidate node. We justify this approach on the basis that the smaller the area of the Voronoi diagram (sensor node has to have many closely placed neighbors), the less representative is the sample and hence smaller is its impact on the overall measurement accuracy. Therefore, each candidate node sends, along with the notification about its potential redundancy, the information about its Voronoi cell area. After sending the notification, each sensor node waits a pre-defined time  $t_{wait}$  for the response from the neighboring nodes in the case any of them declares itself redundant. If any sensor node responds or the received responses carry larger area value then the sensor node becomes redundant status. It communicates to its Voronoi neighbors the new status that it will leave the network and should not anymore be considered for calculating the Voronoi diagram. The sensor nodes receiving this information recalculate their Voronoi polygons. If the sensor node was the candidate node but got suppressed by a neighbor with a smaller Voronoi cell area, then it recalculates its chances to become redundant node. The process progresses as long as any of the sensor nodes are eligible for becoming a redundant node.

**Algorithm 9** Removing Redundant Nodes

---

```

1: function CheckRedundancy() : boolean
2: for all VN in VoroNeighs do
3:   VN_NewVoroNeighs=CreateVoronoi(vn,VoroNeighs\self\VN);
4:   for all VNh in vn_NewVoroNeighs do
5:     if abs(v(VN) - v(VNh)) > AccTH then
6:       return FALSE;
7:     end if
8:   end for
9: end for
10: return TRUE;

```

---

The relocation of sensor nodes, taking only the accuracy constraints under consideration, may lead to the break-up in the network connectivity. In order to eliminate this threat, each sensor node also checks whether its removal will disrupt the connectivity among its Voronoi neighbors. As each sensor node already holds the list of Voronoi neighbors and their locations, such connectivity check is easy to execute. In Figure 6.1 Node C can be safely removed as all other Voronoi neighbors  $VG_i$  can communicate.

It should also be noted that in some boundary cases the repetitive removal of the sensor nodes may lead to the removal of the whole region of the network. This happens when the sensor node to be removed is in an area of local maximum or minimum of the measured value. In order to avoid/prohibit this situation we do not allow such sensor nodes to be removed and as consequence they yield to other possible redundant candidates even if their Voronoi cell covers smaller area.

#### 6.4.4 Coping with Dynamic Phenomena

The fulfillment of the accuracy requirements depends directly on the monitored phenomenon. Consequently, it is evident that as the phenomenon is changing, the algorithm requires a scheme to keep the data updated. A simple approach could be used to allow keeping the data consistent and minimize the sensor nodes communication effort.

The accuracy requirements may be violated only when the measurement difference between two Voronoi neighbors ( $C, VG_i$ ) exceeds  $Acc_{TH}$ . The constant notifications upon changes in the measurement value (eg. measurement oscillation) would incur large communication costs which may lead to an accelerated energy depletion of sensor nodes. The sensor nodes know each other's measurement values when for the first time they collect the measurement and execute the evaluation. In order to limit the amount of exchanged data, sensor nodes only need to inform about the substantial changes in the measurement values. On the other hand, our scheme should avoid the situation when the update does not take

place despite the fact that accuracy  $Acc_{TH}$  was already reached. Assume that  $v(C) < v(VG_i)$  and the current difference in the measurement values equals  $\delta_v = |v(C) - v(VG_i)|$ . Node  $C$  sends the update to  $VG_i$  when it decreases for  $\gamma = 0.5 \times (Acc_{TH} - \delta_v)$ . That is the highest possible value of  $\gamma$  for which we can be sure that even if the  $v(VG_i)$  increases for the value of  $\gamma$  it would not require a reconfiguration. This prevents the case where the threshold  $Acc_{TH}$  is exceeded without having  $C$  or  $VG_i$  send a notification to each other. In most of cases the  $\delta_v$  will just narrow with each update causing further updates. In order to prevent this effect,  $\gamma_{min}$  can be defined. Reaching the value of  $\gamma_{min}$  would already trigger a process of adding a new VN.

It is also possible that  $v(C)$  will increase while  $v(VG_i)$  decreases. At first this process will reduce the  $\delta_v$  but after reducing it to zero  $\delta_v$  will start growing. In this case  $\gamma$  should be adjusted to  $\gamma = 0.5 \times (Acc_{TH} + \delta_v)$ .

When searching for the redundant sensor nodes, it is not necessary to use active data update as described. The active redundant sensor nodes will not violate accuracy requirements. Therefore, the update can be relaxed to a desired frequency.

### 6.4.5 Reconfiguration Scenarios

The results produced by the ASample technique consist of a set of points selected for additional sampling. Along with the sampling locations, the trajectory connecting the Proxy and the Support nodes where a sampling location was added in between, can also be provided. At the given sampling point, the interpolated value may be different from the real one. In order to find the right position the mobile node may move along the trajectory and stop upon reaching the position where the measured value of the phenomenon equals the one which was calculated by algorithm for the VN.

The reconfiguration requires dissemination of gathered information, which can be discussed only in context of usage scenarios. At present foresee three classes (Class 1 - 3) of usage scenarios are foreseen where ASample can come into play.

Class 1 assumes no mobility in the network. For this purpose the sensor nodes that detected violation of accuracy requirements should send the gathered information to the sink using existing in the network routing protocol.

Class 2 involves a WSN, where apart from static sensor nodes, mobile sensor nodes (more powerful nodes e.g.: robots) are present and navigate within the network in order to collect the measurements. The mobile sensor nodes equipped with their own sensors can use the proposed sampling locations to optimize path planning and increase sampling resolution. Mobile sensor nodes alternatively could also pick-up redundant static sensor nodes and place them in the proposed

locations. The static sensor nodes do not need to route the locations information through the network, but only to piggy-back it when transmitting sensor measurements. In this scenario the ASample algorithm can be executed on the mobile robots, instead of the resource limited sensor nodes. The robot collecting the measurements already holds the necessary information to calculate each sensor node Voronoi diagram and to find new sampling locations.

Class 3 involves network where a subset or all of sensor nodes can move. That kind of WSN can adapt its spatial resolution in self organizing manner without involvement of outside operator. Such a scheme requires from the redundant nodes to announce their availability and from sensor nodes detecting under-sampling to request new sampling locations. For this purpose adapted version of the Wang et al. [2006a] can be used. Sensor nodes use virtual quorum to announce and request resources.

Additional problem may arise when the number of redundant nodes is insufficient to provide for all requested sampling locations. The problem may be resolved using adapted version of the bidding algorithm proposed in Wang et al. [2004]. Using this algorithm the sensor nodes requesting new sampling locations place their bids. The value of the bid corresponds to the size of the area of the VNs on behalf of which the sensor node acts as a proxy.

## 6.5 Performance Evaluation

This section describes the evaluation settings and the metrics chosen for the evaluation. The algorithms complexity is evaluated concentrating on their implementation feasibility and the communication costs its execution incurs on the WSN. Furthermore evaluation results are presented along with discussion of their implications.

### 6.5.1 Evaluation Studies

The evaluations settings are based on the reconfiguration scenario classes introduced in Section 6.4.5. Five experiments were designed for this purpose. The first three experiments correspond to the static WSN scenario where no form of mobility is provided. These experiments foresee supplementary deployment of additional sensor nodes in the locations proposed by the ASample algorithm. The first experiment (*Add all*) deploys as many new sensor nodes as ASample suggests. It is to evaluate the efficiency of ASample when sufficient resources are available. The second and third experiments (*Add 10%* and *Add 20%*) add only additional 10% or 20% new sensor nodes (compared to the total number of sensor nodes). If the computed number of the required sampling locations is higher than

the available additional sensor nodes, then the sampling locations with the larger Voronoi cell area are deployed first. The fourth experiment (*Move only*) assumes a network where all sensor nodes are mobile. In this experiment, only the redundant nodes are moved to proposed locations. This evaluates the self-sustainability of the WSN. The fifth experiment (*Move & add*) allows for both moving of the redundant sensor nodes and adding the additional sensor nodes if still some sampling location are not occupied. This scenario tries to limit the amount of resources necessary to restore the network fidelity. The scenario uses a simulated model of physical phenomena where the phenomenon intensity  $p = f(d)$ , depends on the distance  $d$  to the center of the phenomenon (Hotspot model Zhao et al. [2002]). In the evaluation scenario, the phenomena is modeled using several distributions: exponential, pareto, normal and linear. Their parameters are adjusted that in the network area they assume values in the range between 0 and the peak value of 100 units in the center of phenomenon. The evaluation also studies the adaptability of ASample to the changes in the phenomenon activity. For this purpose the peak value of the phenomenon was changed between 30 and 100 units. Throughout all experiments (if not stated otherwise) the deployment area is fixed to the size of  $250 \times 250$ m, the radio range to 25m and the number of sensor nodes to 300. The parameters were chosen so as to generate a connected WSN Bettstetter [2002]. The ASample efficiency was simulated using stand-alone implementation.

### 6.5.2 Evaluation Metrics

A mean square error (MSE) based metric was used, in order to quantify the accuracy enhancement of the phenomenon monitoring. The following methodology was used that to calculate the metric. A virtual grid was put over the network deployment area. At the coordinates of each vertex of the grid, the real value induced by the simulated phenomenon was measured and subtracted from it was the value measured by the closest sensor node. This decision was based on the assumption that the sensor node closest to the sampling point gives the best estimate of the value. Two MSE values were calculated for each simulation and proposed algorithm. The first calculation is performed as described above, the second (MSE30) applies only for the points where the value of the measured phenomenon is higher than 30% of its peak value. The reasoning behind the second value is to evaluate the considered algorithm in the areas of the phenomenon activity.

### 6.5.3 Complexity Analysis

The complexity analysis considers both the computational and the message complexity involved in the execution of ASample. The computation overhead is incurred primarily by the calculation of the Voronoi polygon on basis of the set

of neighbors locations. For each location a bisectional line between the location and sensor node is calculated - a relatively simple algebraic operation. For each line then the possible intersection is checked against the segments and the semi-segments constituting till now calculated Voronoi polygon. In the worst case scenario there are  $\frac{u^2}{2}$  operations, where  $u$  denotes the number of neighbors of sensor node. On average there will be  $u \times VG_{cnt}$  operations, where  $VG_{cnt}$  is the final number of Voronoi neighbors. It should be noted that for most of sensor nodes (not detecting adherence to the accuracy requirements) the calculation of Voronoi diagram takes place only at the initialization of the network.

Regarding the message complexity for the construction of the Voronoi diagram, each sensor node locally broadcasts twice 1-hop message. During the first broadcast the sensor node announces its ID. During the second one, it sends the list of neighbors which may be relevant for constructing its neighbors Voronoi diagrams. If the list of neighbors is too long it may be split into smaller ones containing the neighborhood information concerning only particular nodes and then directly addressed to proper nodes. Sensor nodes repeat this step only if a new sensor node is added or removed in their neighborhood.

When agreeing on new sampling locations, only up to 2 messages need to be exchanged between Proxy and Support nodes. Additional messages are sent only when the measured values change such at the notification of neighbors is requested to check for accuracy requirements.

#### 6.5.4 Evaluation Results

Figure 6.3 presents the results for all five described experiments (Sec. 6.5.1) and additionally three random deployments (the deployment of additionally 10%, 20% and the number of sensor nodes added in the first experiment) for the reference. The results are presented for varied communication ranges, which have a direct impact on the accuracy of the created Voronoi diagram.

The best performing is the first experiment (*Add all*) as it is only adding sensor nodes and is not limited by their number. The reference experiment *Add all random*, which adds the same number of sensor nodes but randomly, delivers the results, that in most of cases offer 20% less MSE reduction. Increasing the number of sampling points must lower the MSE value. As results show ASample utilizes the additional sensor nodes more efficiently than the random approach. That clearly proves the utility of algorithm. Pairs of experiments *Add 10%*, *Add random 10%* and *Add 10%*, *Add random 10%* show how efficiently ASample utilizes limited resources. Each pair shows that the reconfiguration executed using ASample preforms better by providing lower MSE values. In case of the first pair the results are approximately 10% and in case of the second pair up to 15% better than random deployment.

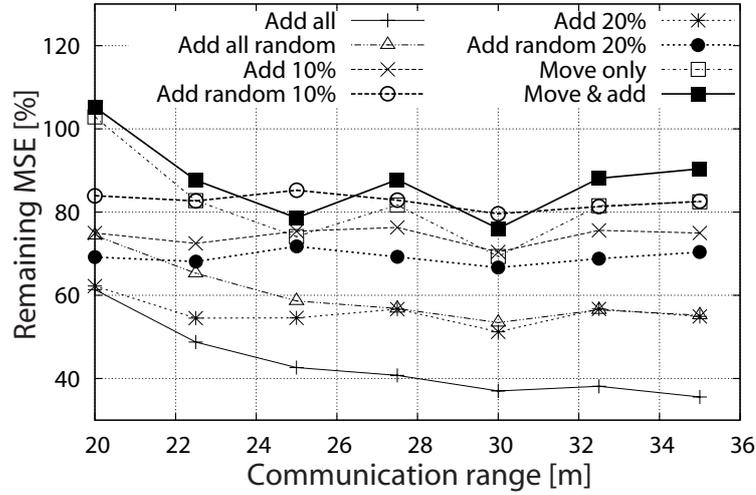


Figure 6.3: Impact of communication range

The strategies aimed at reducing the resource costs of reconfiguration *Move only* and *Move & add* also reduce MSE value but this is less significant as the number of sampling points is lower.

Figure 6.4 shows ASample reaction to different accuracy requirement settings. As expected lower bound on difference in the values between neighboring sensor nodes results in lower MSE values. It comes at the cost of increased resource usage. Adding the same number of sensor nodes but randomly leads to higher MSE values. Using the strategy of moving redundant sensor nodes significantly reduces the need for the new resources. When the accuracy bound is set to a large value, ASample puts up to 40% of the existing nodes in the redundant mode. This results in higher MSE values, which are consequence of lower accuracy requirements. The last two experiments show the impact of message losses on the efficiency of ASample. *Add all (msg loss)* and *# Add all (msg loss)* plots show the MSE value reduction and number of sensor nodes added in environment where 20% of messages were lost. As a result, the accuracy of constructed Voronoi diagrams suffers and hence some violations of accuracy requirements are not detected. Consequently, less VN are deployed thus providing lesser reduction in MSE value compared to the lossless message scenario.

Figure 6.5 shows the results of ASample for different sensor nodes densities. The achieved reduction in the MSE value remains constant. It is much smaller than the one presented using full Voronoi diagram, because of the missing information. For higher densities ASample builds a nearly full Voronoi diagram but as the network is already saturated with sensor nodes the margin for improving networks MSE is limited. Random based deployments show much smaller reduction

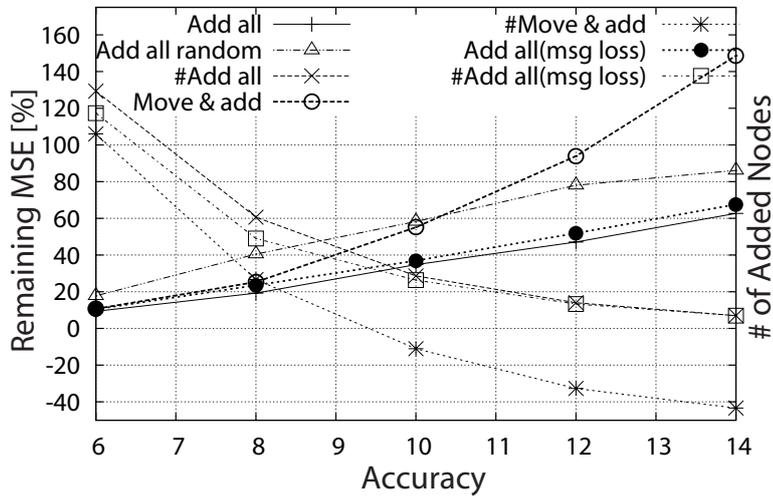


Figure 6.4: Changing accuracy requirement

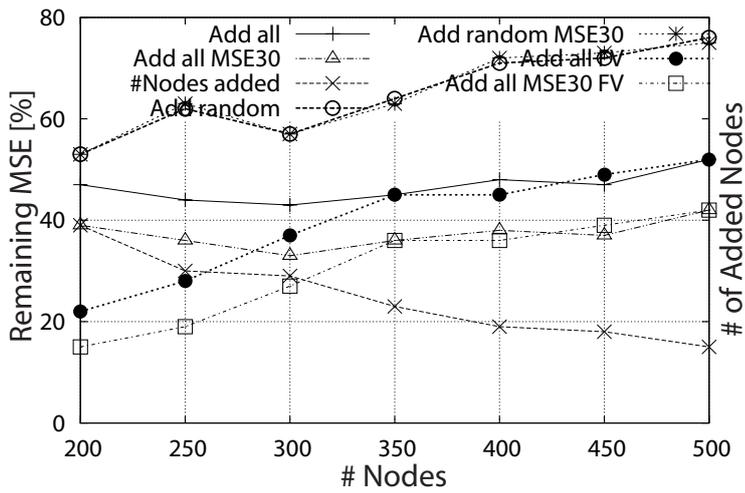


Figure 6.5: Changing network density

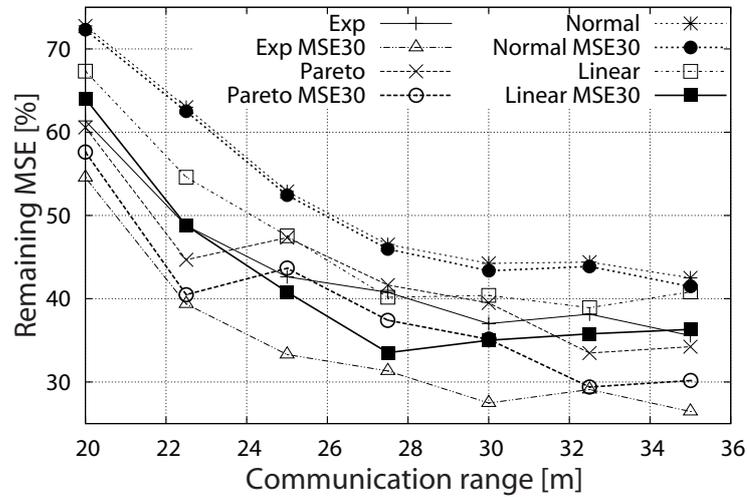


Figure 6.6: Various phenomena models

of the MSE. As the density increases there is less demand for adding additional sensor nodes as the possible improvement in MSE value saturates.

Figure A.5 illustrates four pairs of experiments that show how ASample handles different types of phenomena. Each pair represents a different phenomenon distribution model (exponential, normal, pareto and linear) and reduction of the MSE values for the whole deployment area and the area where activity of the phenomenon reached at least 30% of the peak value. The curves of each pair are very similar, they differ only by the fixed offset. The MSE values for the activity area are better as the majority of sensor nodes are added there. The exponential distribution shows the most significant difference and the reduction in the MSE values. This is because it represents the environment where the changes in the value show the largest volatility. The exponential distribution demands the highest number of sensor nodes to add. On the other extreme, for the normal distribution the difference between the MSE values for the whole area and the activity region are negligible as the phenomenon changes gradually. Still, the case of normal distribution shows that ASample is able to significantly reduce the MSE value of the network.

Figure 6.7 shows how ASample adjusts to the changes in the phenomenon. The *Phenomenon*-plot shows how the intensity of the phenomenon changes over the course of the simulation. The two curves *Original MSE* and *Original MSE 30* present how the absolute values of MSE change with the phenomenon. It can be seen that for the low intensities of phenomenon the original deployment may be sufficient, but as the phenomenon develops and its volatility increases the accuracy of measurements suffers. The two plots *Add all* and *Add all MSE30* show the MSE

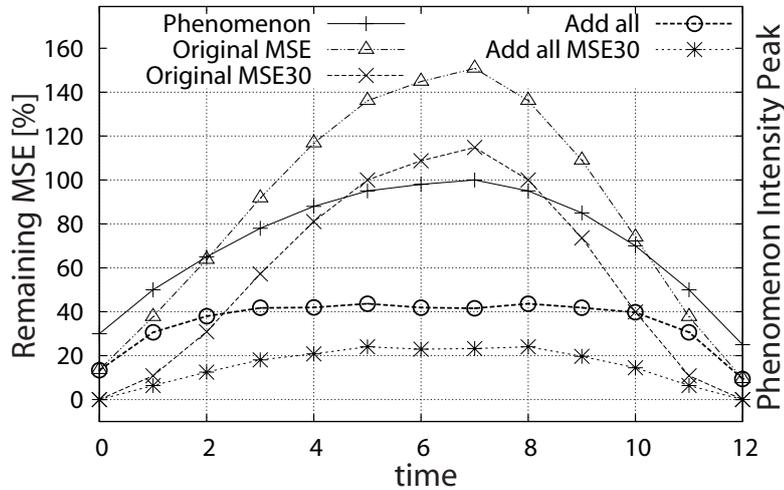


Figure 6.7: Evolving phenomenon

value with the additional sampling points proposed by ASample. Initially, the MSE values slightly rise but then saturate quickly and remain on a steady level. It demonstrates the capability of the algorithm to maintain the required accuracy.

## 6.6 Mobility Assisted Adaptive Sampling

The goal of mobility assisted adaptive sampling demo is to show the practical applicability of ASample. The demo is based on the Class 2 reconfiguration scenario. The demo scenario assumes WSN consisting of set of fixed sensor nodes and a single mobile robot. The mobile robot can navigate across the network and collect sensor data directly avoiding the multi-hop routing. The mobile robot route includes also the sink location, in order to deliver collected data for further processing. The robot is equipped with a set of sensors to take samples at the arbitrary locations.

### 6.6.1 Overview

Using ASample, sensor nodes can locally detect the regions of over- or under-sampling and identify redundant or additional sampling locations for the desired accuracy. A mobile robot can exploit this information from sensor nodes to optimize its traversal path and increase the sampling resolution in under-sampled regions. In over-sampled regions the traversal path can be shortened to conserve the mobile node's energy. The shorter traversal path also means faster data collection.

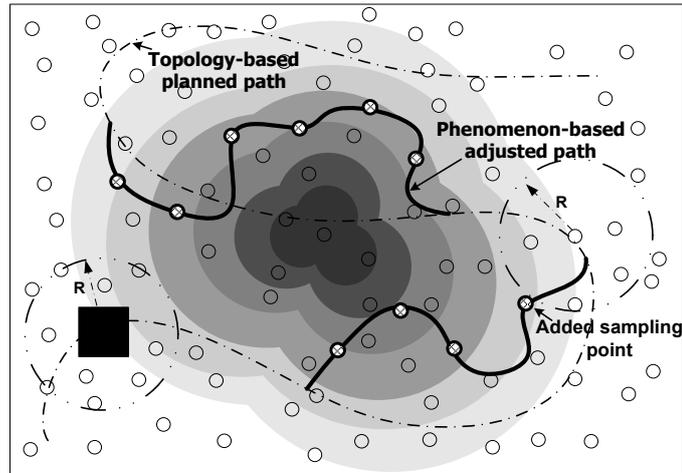


Figure 6.8: Mobile robot traversal path

Figure 6.8 depicts a common WSN scenario that we use to define our demo scenario in Paragraph 6.6.3. A single mobile robot (indicated by the black box) was used for the sake of simplicity. A real implementation can consider multiple robots Khelil et al. [2010] or some primary/secondary arrangements. The mobile robot starts the movement from the sink using a pre-planned path. While collecting the samples from static sensor nodes, the mobile robot receives proposal for new sampling positions (in case of under-sampled regions). The mobile robot recomputes the current traversal path using new sampling positions. As a result, the mobile robot moves to the suggested locations to perform the required supplemental sampling. As an extension, the mobile robot can also use collected data and newly taken samples to verify whether under-sampling problem was resolved or there is need for finding further sampling positions.

The presented demo is designed on the basis of (a) gMAP Khelil et al. [2009] algorithm and (b) ASample algorithm. gMAP is a delay-tolerant and mobility-assisted data collection algorithm and is responsible for a path planning for the mobile robot. The following subsection provides detailed descriptions of gMap algorithm.

### 6.6.2 gMAP: Efficient Delay Tolerant Monitoring

The gMAP approach comprises algorithms for (a) collecting data in a mobility-assisted way, and (b) map construction at the sink. The demo scenario uses the temperature maps as an example, which are referred to as tMAP. The mobile robot traverses the sensor field and collects the temperature data from all sensor nodes. At specific locations, the mobile robot sends a short beacon, to which sensor nodes reply with their temperature value. In order to plan the movement of the mobile

robot, the employed approach uses a stepwise decomposition of the problem into different less complex subproblems: (1) Finding suitable breakpoints, (2) reducing overlaps in the communication range, and (3) planing a shortest-path-tour. Subsequently, the sub-solutions are integrated into a single path planning algorithm. In the context of demo the gMap algorithm is looking for a set of points, considerably fewer than the number of sensor nodes, where the mobile robot stops and communicates with the sensor nodes within its communication range. First, the algorithm gets candidates for a set of mobile robot positions by solving a nonlinear problem and removing unnecessary points. Then, it iterates solving an approximative mixed-integer linear program to finally get the optimal sequence Traveling-Salesman-Problem solution. Figure 6.9 depicts the result of proposed path planning. A detailed description of our topology-based path planning algorithm is presented in Khelil et al. [2010].

### 6.6.3 Demo Setup

This subsection outlines the demo deployment issues, discusses hardware choices and implementation details of the proposed algorithms.

#### Deployment Setup

The demo scenario consists of the deployment of  $n = 10$  sensor nodes in a rectangular area. Sensor nodes form a grid of 2 lines with 5 sensor nodes each (Figure 6.9). The distance between two neighboring sensor nodes equals to 1 m.

The mobile robot starts from one end of the grid and continues its movement till it reaches the border of the sensor field. Then the mobile robot changes the direction to back and moves straight forward. This movement pattern will continue till the mobile robot encounters all sensor nodes. A stop-and-wait strategy was adapted in order to reliably collect the temperature samples (to construct tMAP). The mobile robot moves to the designated point, where it pauses for a random time period between 0 to  $T_{max}$  sufficient to collect sensor data from sensor nodes in its radio range. The mobile robot first performs a snapshot by sending a REQ message to all sensor nodes in its transmission area using a MAC broadcast. A sensor node replies by sending a message containing its node-ID and temperature value ( $\alpha_t$ ). In order to reduce collisions, sensor nodes schedule their reply for a random time  $t_\gamma$  between 0 and a  $T_{max}$ . The optimal result of the collection operation is a set of  $n$  elements with the following tuple: {node-ID,  $\alpha_t$ }.

During the demo, the temperature distribution is synthetically changed in order to create an under-sampled region, i.e., the difference between the value of two sensor nodes is greater than  $Acc_{TH}$ . This is realized by placing a heat source on one sensor node. The sensor node close to the heat source detects this condition

and calculates additional sampling positions. Next it sends them to the mobile robot for use over the next round of temperature data collection. The mobile robot processes the new locations and adjusts its path in order to visit the proposed sampling points and to take samples using on board sensors.

Additionally, a visualization application was developed to run on a laptop to act as a sink. The application presents, on the map, the proposed locations to compare them with the locations that the mobile robot reached. It also shows the new Voronoi diagram with the new sampling points along with the collected values. The cells of the Voronoi diagram are shaded with the intensity of grey corresponding to the measured value.

### Hardware

Following paragraphs briefly detail the technical specifications of the used hardware, i.e., sensor nodes and mobile robot.

**Sensor Nodes** The demo scenario uses TelosB and MICAz sensor nodes. TelosB nodes function as static sensor nodes. A single MICAz sensor node is used to control the robot via a serial interface. Sensor nodes communicate with each other over IEEE 802.15.4 radio with communication range  $R$  ( $\sim 2\text{m}$  - artificially reduced for the purpose of this demo). The power of radio is set in such a way that only the neighbor nodes can communicate with each other to have a multihop scenario in a limited deployment area. The battery powered sensor nodes are equipped with a suite of basic sensors, i.e., temperature, light and humidity.

**Mobile Robot** The mobility in the WSN is provided by a simple COTS robot from Fischertechnik. The robot contains two engines that separately propel two wheels. The engines are controlled by an integrated controller. The serial interface is used for establishing a connection to the mounted MICAz sensor node. This sensor node controls the movement of the robot by issuing commands to the robot's engines over the serial interface. The movement of the robot is currently limited to vertical and horizontal axes of the coordinate system. The mounted sensor node is equipped with the same set of sensors, such that it is capable of taking the same type of samples as static sensor nodes. The demo utilizes only the temperature sensor for taking temperature samples to construct tMAP.

### Robot Localization

The mobile robot uses a simplified localization strategy for the purpose of demo. In this approach, the mobile robot is in the possession of the deployment map. The deployment map includes positions of all the deployed sensor nodes in a virtual

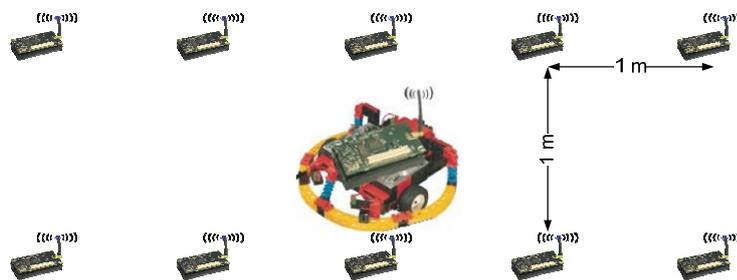


Figure 6.9: Structured scenario

coordinates system. The topology of deployment in this scenario has form of grid, where sensors nodes are placed in the corners of the virtual cells spread over the deployment area (Figure 6.9). More sophisticated localization schemes such as that presented in Somov et al. [2008] can be utilized in complex deployments. The mobile robot moves only in parallel to the axes of the grid.

Before executing the movement mobile robot tries to localize itself in the network. For this purpose it broadcasts an ECHO message to sensor nodes. Each sensor node that receives the ECHO message responds with the message of its own containing the unique ID of the sensor node. The mobile robot collects the responses along with the received signal strength indication (RSSI) values. The trial takes several repetitions (three to five) in order to eliminate influence of statistical inaccuracy. The robot compares number of received messages and RSSI value of each responding sensor nodes. The robot concludes its position at the location of the sensor node which responded the most times with the strongest RSSI value. At initialization, the mobile robot is also unaware of its orientation. The mobile robot after a successful localization moves a single distance unit (size of the cell) in its current direction and stops. Then, it compares its current location with the previous one. On basis of the local virtual map, the mobile robot concludes its current orientation.

### Demo results

There are several observations made while designing and implementing the described demo. The most important experience drawn for the presented demo is the validation that both gMAP and ASample techniques can be implemented on existing realistic WSN platforms. The scope of the demo was constrained by the hardware we have used for the mobility as well as by the considered basic localization algorithm. The demo shows that low-cost COTS robots are well suited for introducing mobility into the network. Unfortunately, it comes at the cost of implementing auxiliary techniques for interfacing it with sensor nodes and proper controlling of the motion. The used localization algorithm limits the current appli-

cability of the demo to grid topologies. Additionally, the occasional false positives were encountered for the mobile robot localization, since the mobile robot concludes wrong position due to similar RSSI values from different sensor nodes.

## 6.7 Summary

This chapter presented ASample, an efficient distributed adaptive spatial sampling technique. Using only local views, ASample is capable of detecting the regions of under- and over-sampling. In case of the under-sampled regions it proposes a set of new sampling positions and for the over-sampled regions it discovers redundant sensors. Both operations are executed in a unified manner. Such an holistic approach allows for self-sustainability of WSN. As simulations show, the proposed approach effectively and exactly maintains the required accuracy of the measurement and allows fulfillment of the stated accuracy requirements. The practical applicability of proposed ASample algorithm was verified by demo deployment.

## **Chapter 7**

# **Conclusions and Future Research**

This thesis developed concepts and efficient methods for maintaining the reliability and functionality of WSN. This chapter concludes the thesis by summarizing the main contributions made, and discussing them from their applicability perspective.

The work presented in this thesis opens up new interesting research directions. Therefore, this chapter also discusses the key issues, and presents ideas for further improvements in the maintenance techniques introduced in this thesis.

## 7.1 Overall Thesis Contributions

The main goal of the thesis was to develop new and improve existing maintenance techniques for WSN. The research effort was driven in particular by the problems of energy distribution in the network, sampling accuracy and reliability of the network. Accordingly, this section discusses the key contributions made by the research presented in this thesis. Driven by the research questions listed in Section 1.2.1 and grouped by topic, the thesis contributions are surveyed and their relevance is discussed.

### 7.1.1 Localized Energy Hole Profiling in Wireless Sensor Networks

Localized Energy Hole Profiling provides important improvements in the field of the energy profiling and energy management in the Wireless Sensor Networks, in comparison to the state of the art solutions. The algorithm delivers information not only on accurate positioning and size estimation of developing energy hole, but also estimates future energy usage, which unbalanced in time, usually leads to sensing and communication coverage loss. This allows for proactive maintenance of the sensor network and long term mitigation of disruptions in the operation of the network. The LEHP profiling strategy is applicable independent of the character and source of the energy hole and optionally can be used without position knowledge from sensor nodes.

Overall, the development of the Localized Energy Hole Profiling algorithm for Wireless Sensor Networks energy maintenance represents the contribution **C1** of the thesis, as defined in the introductory chapter of this thesis (Section 1.2.2).

#### Resultant publication

- **Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *LEHP: Localized Energy Hole Profiling in Wireless Sensor Networks*, in Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Riccione, pp. 100–106, 2010

### 7.1.2 Topology Oriented Maintenance in Sparse Wireless Sensor Networks

Topology Oriented Maintenance takes on very important problem of unbalanced sensor nodes distribution in (sparse) Wireless Sensor Networks, which results in topology irregularities that produce serious negative impact on key properties of

the sensor network. To the most prominent properties of WSN that suffer from topology irregularities belong increased network latency, uneven energy distribution, occurrence of bottlenecks and decreased sampling resolution of the network. The presented in Chapter 4 algorithm provides solutions that mitigate to high degree enumerated problems. TOM localizes the network irregularities and proposes new deployment locations, where placement of sensor nodes alleviates the problems and restores topology regularity. Algorithms execution is localized, and while using low number of mobile nodes, allows for efficient and sustainable WSN self-healing. TOM also allows to lower cost of the sensor network deployment by minimizing number of required sensor nodes by being able to repair irregularities of sparse networks.

Concluding, the TOM algorithm provides an efficient maintenance technique for alleviating topology irregularities in the sensor network and represent the thesis contributions **C2**.

### **Resultant publications**

- **Piotr Szczytowski**, Abdelmajid Khelil, Azad Ali and Neeraj Suri, *TOM: Topology Oriented Maintenance in Sparse Wireless Sensor Networks*, in Proceedings of the IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), Salt Lake City, 2011

### **7.1.3 Distributed k-Connectivity Maintenance in Wireless Sensor Networks**

Reliability of the Wireless Sensor Networks is another very relevant maintenance problem. The very nature of WSNs renders the problem of assuring reliability very difficult. An important aspect of the reliability of the network is its tolerance to failures of single or groupings sensor nodes. In extreme case, the failure of the single sensor node may be the cause for disconnection of a large part of the network. TOM algorithm already takes partly on this problem, but cannot give reliability guarantees. In contrast, the k-connectivity property guarantees that even failure of the k-1 will not disconnect the network. Unfortunately, the state of the art solutions concentrate exclusively just on providing this property, while trying minimizing number of messages exchanged or sensor nodes deployed. This contribution discussed in this thesis not only provides algorithm for placements of the sensor nodes to achieve k-connectivity property, but also targets additional important aspects of the problem. In particular DKM algorithm delivers guarantees about the impact of sensor nodes failures on latency (minimizes the increase of

hop distance due to failures) and at the same time provides disjoint paths (localized failure of stacked sensor nodes does not disable  $k$ -connectivity property). Moreover the presented solution allows distributed execution and is able to detect the sensor nodes redundant to maintaining  $k$ -connectivity, so that they can be relocated to repair other regions of the network. That allows with use of mobile nodes for self-repairing, self-sustainable WSN.

The presented in this thesis DKM algorithm represents the contribution **C3** of this thesis (see Section 1.2.2 for detailed descriptions thereof).

#### **Resultant publications**

- **Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *DKM: Distributed  $k$ -Connectivity Maintenance in Wireless Sensor Networks*, in Proceedings of the Annual Conference on Wireless On-demand Network Systems and Services (WONS), Courmayeur, 2012

### **7.1.4 Adaptive Spatial Sampling in Wireless Sensor Networks**

The sampling constitutes one of the most common applications of the Wireless Sensor Network. The quality of the sampling directly depends on the spatial sampling resolution. The higher the spatial resolution the higher the quality of the collated data. Unfortunately, the increase in data quality comes at steep cost in resources required. The main task of the presented in the thesis ASample algorithm is to assure adequate sampling resolution corresponding to the dynamics of the monitored phenomenon. ASample is a distributed algorithm using minimal communications resources to identify regions of under-sampling. In order to resolve detected problem, ASample provides new locations for placement of sensor nodes so that their deployment assures required (user controlled) sampling resolutions. Moreover, ASample can also detect regions of over-sampling and select sensor nodes that may remain idle to conserve energy. In case of mobile network the selected sensor nodes can be relocated to resolve the under-sampling problem. Such a scenario was demonstrated using mobile robot in the demo trial.

The ASample algorithm constitutes the contribution **C4** presented in Chapter 6.

#### **Resultant publications**

- **Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *ASample: Adaptive Spatial Sampling in Wireless Sensor Networks*, in Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), Newport Beach, pp. 35–42, 2010

- **Piotr Szczytowski**, Faisal Karim Shaikh, Vinay Sachidananda, Abdelmajid Khelil, Neeraj Suri, *Mobility Assisted Adaptive Sampling in Wireless Sensor Networks*, in Proceedings of the International Conference on Networked Sensing Systems (INSS), Demo session, Kassel, 2010

## 7.2 Lessons Learned

The work presented in this thesis has developed a basis for providing effective maintenance of Wireless Sensor Networks.

The task of maintaining proper functionality and reliability of the Wireless Sensor Networks provides is very challenging. Set of considered maintenance problems largely depends on the deployment contexts of WSN. The WSN depending on their application render different set of requirements. The task of the maintenance algorithms is to assure that the requirements posed by WSN during its assumed complete lifetime will be fulfilled, or the lifetime under so stated requirements will be possibly maximally extended. The WSN some requirements like high reliability, low latency network communication (coming at cost of dense deployment) may stay in direct opposition to challenge of energy conversation in resource limited network.

Therefore, it is important observation that while designing a maintenance strategy, the design effort cannot concentrate only on the single considered aspect. Any maintenance strategy with any objective has to account also for the impact on rest of the network properties. Optimizing algorithm for multiple objectives is a demanding task and can be in most cases only accomplished taking into account the context of the deployed application. Providing a single generic maintenance framework is for this reason unrealistic. This thesis therefore targeted handling the most important maintenance objectives for the most common applications. Still the techniques to large extend account for mixture of requirement. All of the presented techniques aim at reducing resource required for accomplishing the task, additionally consider aspect of self-sustainability of the network by means of self-repair and where possible try to improve second-tier network properties (e.g. DKM algorithm, besides assuring  $k$ -connectivity, also provides guarantees regarding latency measured by hop distance).

## 7.3 Open Ends - Basis for Future Work

While the work presented in this thesis addressed the research questions driving it towards making the discussed contributions, it also opened new and interesting research perspectives along its way. In the following, some of the most promising

ones are briefly described.

### **Pro-active Reconfiguration**

The LEHP can be used to design efficient techniques for pro-active reconfiguration. In case of renewed deployment information gathered by LEHP can be used at design step of new deployment to plan the sensor nodes distribution. The discussed examples demonstrated that LEHP can be used for handling the maintenance based on the data collected by algorithm.

### **Application Driven Topology Optimization**

There is potential to extend the TOM approach to consider functionality centric and application based objectives while performing topology optimizations. In particular, maintenance should assure not only topology regularity in the direction to the sink but also in the areas of interest desired by the application.

### **Increasing Resource Reuse for k-Connectivity Maintenance**

Currently, DKM considers only 2-hop neighborhood connectivity information for identifying closer and semi-closer neighbors. Increasing the size of immediate neighborhood information to further distances could potentially allow for the identification of more connectivity redundancies. These potential savings would be achieved at the cost of a higher algorithm complexity and communication overhead. Accordingly, it is crucial to carefully investigate the gain-cost ratio.

### **Adaptive Spatial Sampling for Three Dimensional WSN Deployments**

The Voronoi abstraction used for techniques presented in Chapter 6 is a powerful abstraction, which can be used beyond the presented problem in context of two-dimensional network deployment. In the ongoing work, it will be important to investigate how with a limited effort the ASample techniques can be extended for application in the three-dimensional network deployments. Hereby, the neighboring sensor nodes instead of sharing common edges, share common planes. Therefore, the additional sampling points can be found using the planes' center of gravity instead of a middle point of Voronoi edge.

# Appendix A

## Map-Based Modeling and Design of Wireless Sensor Networks

### A.1 Introduction

As simulation is a frequently used approach to test and validate approaches, simulation environments need to be able to support the various WSN design schemes. Though the research trend in WSN is to address regions instead of single sensor nodes, existing WSN simulation environments still do not support modeling and design on this abstraction level.

Sensor nodes inherent redundancy and spatial nature of the monitored physical phenomena results in spatially correlated sensor readings and indirectly in network properties, like energy distributions. The natural abstraction to capture the inherent spatial correlation of sensor nodes states is the map paradigm. A *Map* is an aggregated view on the spatial distribution of a certain attribute at a certain time. The literature describes two main classes of maps, i.e., the choropleths Robinson et al. [1995] and the isomaps Liu and Li [2007], Solis and Obraczka [2005], Meng et al. [2006]. The map construction groups spatially correlated sensor nodes with similar attribute's values into *regions*. We define a region by its border (a set of spatial points) and an aggregate (e.g., average) of the attribute's values obtained from all sensor nodes located in the region's area. A map is then the collection of all regions of the WSN.

The map paradigm builds on the region principle and therefore, provides excellent modeling primitives for WSNs. Global maps are created for the sake of network monitoring (e.g., residual energy map Zhao et al. [2002]) or of event detection (e.g., oxygen map Li et al. [2007]; Xue et al. [2006]). A promising direction consists in using maps to optimize protocols Goussevskaia et al. [2005a,b], detect Ren et al. [2008] or track Sarkar et al. [2008] event boundaries. These pa-

pers highlight the map-based methodology as a powerful and highly promising abstraction level. The Khelil, Abdelmajid and Shaikh, Faisal Karim and Ayari, Brahim and Suri, Neeraj [2008] proposes the maps to be the natural step towards a holistic *Map-based World Model* and *Map-based WSN design*.

## Contributions

In most of network simulators, both the design of topologies as well as storing and representation of the trace data have textual format. Expression of the spatial dependencies in that form is both complex and limited. WSN usually involve large scale simulations, where manual preparation of the multitude of topologies and simulation scenarios is beyond simple textual representation. The common solution to this problem is using text generation scripts which create topologies following the simulator syntax. Unfortunately, techniques to verify and validate the created topology are limited. A direct and effective solution is visual validation.

The established OMNeT++ omn simulation environment introduces the notion of the interactivity by offering a visual editor for its topology format. However, topology is only one feature of scenario generation. In WSN other important features such as temperature's spatial and temporal distribution present a fundamental part of scenario generation. The lack of support for visualization is even more clear in case of trace data. Currently, trace data is collected in a text file used for statistical analysis. Usually, analysis is done at sensor node or event level. This is tedious work and does not follow the region abstraction level as discussed earlier. Visualization rendered as a map is natural approach for the discovery of spatial dependencies. Of course the map representation cannot fully replace the traditional statistical approaches, which are required for understanding the details of investigated protocols. However, it offers considerable support for high-level designing and debugging. The processing of data across series of simulations requires extraction of relevant information from complex traces with mixed content. The efficient way of handling the large amount of data is to export it to relational database and querying using database engine.

This chapter provides a description of a map-based MAP++ framework dew that delivers following contributions, highlighted through a case study:

- Extended topology generation, including setting of spatially correlated initial conditions and batch script generation for simplifying modeling and scenarios generation
- Trace data visualization support that identifies spatial dependencies for design and debugging

- Export to and processing of trace data with support of relational databases for performance evaluation

The remainder of the chapter is structured as follows. Section A.2 describes the assumed system model. Related work is presented in Section A.3. Section A.4 details the architecture and implementation of MAP++. In Section A.5 illustrations of application scenarios are presented. The framework performance and usability is analyzed in Section A.6 and the chapter is concluded in Section A.7.

## A.2 System Model

The assumed system model considers a two dimensional WSN consisting of stationary, resource-constrained sensor nodes with limited processing, storage and battery-capacity. Each sensor node is capable of short range wireless communication with a fixed transmission range. The focus is put on large scale deployments involving hundreds or thousands of sensor nodes. Sensor nodes measure the physical signal at given sampling rate (e.g. temperature, humidity, as well as its own battery status). There also exists one or more base stations (called sinks), which are designated resource-rich nodes serving as a gateways between the WSN and its operator users. The proposed framework also allow for scenarios, where sensor nodes are equipped with different set of sensors, monitoring different phenomena.

## A.3 WSN Simulation Environments

The increasing interest in WSN research resulted in development of a variety of simulation environments. TOSSIM Levis et al. [2003] being a simulator for the common TinyOS platform tin serves as a very good example of such environment. Its attractive feature is that code developed for the simulator can be directly deployed on the physical sensor nodes running TinyOS. The simulation is very accurate on the node architecture level. Unfortunately, these architecture details (e.g. module wirings) as well as the details of communication layer consume much of simulation processing resources while providing only limited impact on global properties of WSN. Consequently, they also put constraint on network scale, rendering the simulator inapplicable for large scale simulations.

Also some of existing network simulators are becoming adapted to support simulation of WSN. One of the examples is the established NS-2 simulator McCanne and Floyd. The MannaSim Framework man provides such an extension by delivering the topography generation tool, modules simulating an antenna, and radio propagation models. However, till now the capabilities of NS-2 simulator in the area of WSN are very limited due to its limited scalability Xue et al. [2007].

Projects like Ahmed, S. and Bilal, M. and Farooq, U. and Fazl-e-Hadi, N-W.F.P [2007]; Chin, Gilbert Y. and Branch, Joel W. and Brevdo, Eugene and Zhu, Lijuan and Szymanski, Boleslaw [2004]; Estrin et al. [2000]; Krop et al. [2007]; Lessmann et al. [2008]; Levis et al. [2003]; Sobeih, Ahmed and Chen, Wei-Peng and Hou, Jennifer C. and Kung, Lu-Chuan and Li, Ning and Lim, Hyuk and Tyan, Hung-ying and Zhang, Honghai [2006] deliver rich libraries of components and algorithm templates leveraging different aspects of WSN and some of them offer various forms of visualization. All of them are targeted for displaying the messages flows and debugging rather than rendering the attributes distribution in regards to map paradigm.

OMNeT++ omn also follows the approach of adapting its features to use for simulation of wireless networks. OMNeT++ is a component-based, modular, and event discrete simulator. It is based on generic and flexible open-architecture, provides extensive GUI interface and is platform independent. These characteristics make it especially suitable for highly efficient network simulations. Owing to its modularity and as consequence extensibility it receives a growing recognition in the WSN community resulting in the implementation of mobile and sensor frameworks for this simulator. Example of such effort is the Mobility Framework for OMNeT++ mob, which is intended to support wireless and mobile simulations within OMNeT++. It provides features like sensor node mobility, dynamic connection management and a wireless channel model. A similar approach is presented in MiXiM Köpke et al. [2008], which is prepared as a simulation framework with a concise modeling chain for mobile and wireless networks. While not yet implemented, it promises to deliver models for mobile environments, sensor nodes and objects, radio propagation models for multiple signal dimensions, physical layer models for modulation, coding and diversity receivers, library of MAC protocols and localization algorithms. Also Pham, Hai N. and Pediaditakis, Dimosthenis and Boulis, Athanassios [2007] builds up on top of OMNeT++ delivering an advanced radio/channel model allowing multiple transmission power levels, a physical process model, and a resource monitoring MAC protocol.

Authors of *EYES WSN Simulation Framework* EYE and *NesCT* nes projects investigated the possibility to use TOSSIM simulations in OMNeT++ and provide necessary extension for handling wireless networks. The simulation profits from rich library of components written for TinyOS, increased efficiency, and extensive GUI provided by OMNeT++. EYES also uses maps, but only as mean of defining the failing probabilities of deployed sensor nodes or modeling the radio propagation by marking the obstacles. However, they are not the attribute distribution maps. Dreiholz and Rathgeb [2008] presents the tool-chain for providing the parametrization, distributed execution and result-postprocessing and debugging. Similar to MAP++ approach, Lessmann and Heimfarth [2008] provides offline visualization of traces. The difference is that Lessmann and Heimfarth [2008]

works on the node-level basis, not utilizing the fact of spatial correlation of sensor nodes attributes. The tool is only limited for offline replaying of simulations, and provides no support for scenario generation and evaluation.

The multitude of mentioned works regarding WSN confirms the rising interest in developing the simulation environment for wireless networks. Unfortunately, no project within the WSN community provides a holistic support for map design and modeling in simulation environments. Approaches in other communities like Space Time Toolkit vast provide advanced capabilities for integrating spatially and temporally-disparate data within 2D or 3D display domain. Unfortunately, lack of access to source code renders the integration with WSN simulations systematically difficult.

## A.4 The MAP++ Architecture

This section describes the main elements of the MAP++ framework architecture, their interdependencies and interactions with the OMNeT++ simulator.

### A.4.1 Overview of the MAP++ Architecture

The presented problem and system model drive the MAP++ design. Figure A.1 explains the MAP++ framework architecture and interactions between the framework and the OMNeT++ simulator. Simulation studies usually require a large number of scenarios to evaluate a vast parameters domain. *Scenario Generator* is responsible for generation of the topologies designed by the user and supplemented by simulation scripts for batch execution using varied parameters. The framework requires flexible and robust way of tracing the data. The generation of trace should be decoupled from the implementation of the simulation modules. Therefore, a separate module *Trace Module* was designed, which is independent from the rest of simulation and delivers well formatted *Trace*. In order to simplify the process of configuring and embedding the *Trace Module* into the simulation, the *Trace Module Configurator* was developed. It provides the user interface, which allows seamless setting of *Trace Module* parameters. The OMNeT++ simulator executes defined scenarios and embedded trace modules produce *Traces*. Traces are stored in XML format, which makes them suitable for automated parsing and subsequently usable for producing maps. The core functionality of representing the network using maps is realized through the *Visualization & Regioning* module. Visualization renders the map reflecting the trace data and allows basic map operations. *Regioning* allows further map operations by providing the grouping of the nodes depending on their class membership, where a class is defined as a value range of the selected map attribute. More accurate and statistical evalua-

tion of data is provided through the *SQL Database Interface* module. It allows the import of XML formatted *Trace* data into the relational database and the querying data using database engine. The query creation is simplified by integration of database module with the visualization module, where users may visually create queries.

From the implementation viewpoint, the framework consists of two distinct components, i.e., the Trace Module and the MAP++ Tools.

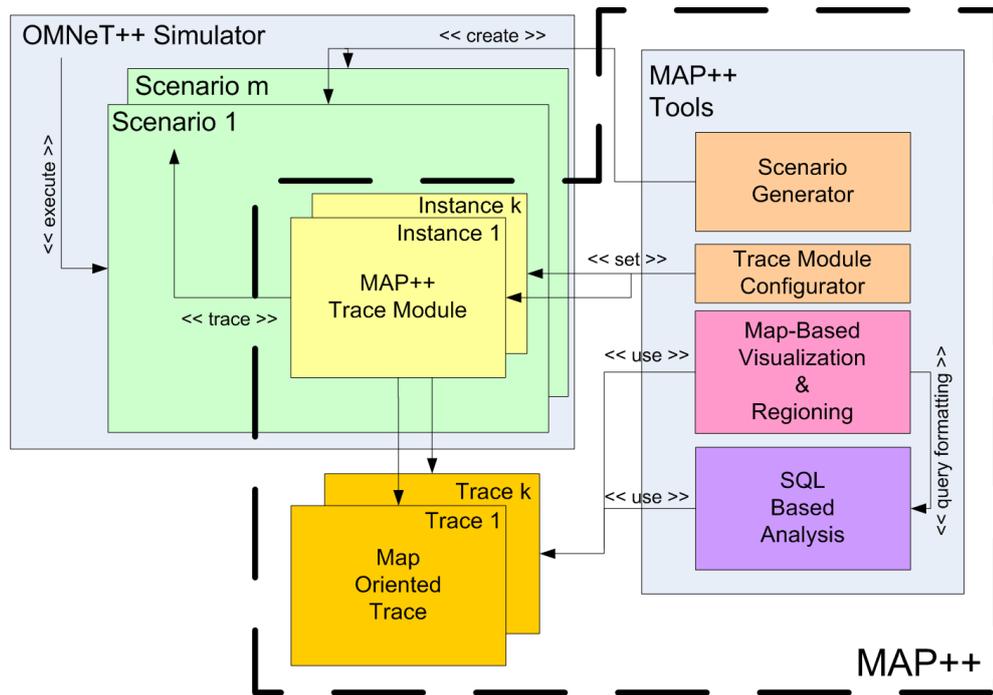


Figure A.1: MAP++ Architecture

#### A.4.2 The MAP++ Trace Module

The Trace Module is embedded into the simulation environment as an instance of a simple module (object class implementation unit) defined within the topology file. The module is loosely coupled with the simulation environment, imposing only limited amount of additional work on simulator users. Its task is to periodically (period can be arbitrary set by the user) query all simulation modules of defined classes and store their selected published attributes. The Trace Module was developed with two objectives in mind. The first objective is for the Trace Module to be independent from the implementation of the simulation modules.

The modules developer should not be required to integrate the necessary traces into modules source code. The module should only publish the attributes required to track its state. This approach allows to use the MAP++ framework even with already existing simulations without need of making simulation code changes.

The second objective is to maintain consistent and reusable trace format. The traces format has to be consistent for proper interpretation for visualization by the MAP++ framework and their proper processing. The employed XML format, fully meets these requirements. The format's self description characteristic allows for automated import into relational database and processing. An additional benefit is the simplified conversion using stylesheet transformation to the desired format for use with a wide range of tools.

In case of WSN scenarios, with multiple map attributes of interest, for each of the module classes, a separate instance of the Trace Module can be initialized, allowing parallel tracking of different sets of network perspectives. For example if network contains two different classes of modules each equipped with different sensor (e.g. temperature and humidity sensor nodes) then two instances of Trace Module are need, as each Trace Module may trace only one module class. In case that single module class provides two or more sensors, then only single Trace Module is necessary for tracking.

### A.4.3 The MAP++ Tools

The second component constituting the MAP++ framework is a suite of tools for generation of simulation scenarios, configuration of traces, visualization and analysis of simulation results.

Scenarios consist of topologies and batch scripts. The static topologies are created by randomly assigning the location for defined number of sensor nodes within defined deployment area. The links are established between the sensor nodes whose distance is shorter than the transmission range. Topologies and sensor nodes deployed in topology publish set of attributes, which are varied using batch scripts. Scenarios are stored in XML format and transformed by the provided stylesheet to comply with simulator file format. The use of stylesheet transformation allows to adjust to the possible future changes in the file format and provides means for reusing the scenarios in different simulators for possible comparative studies.

The visualization is created by loading the topology definition and creating its two dimensional representation. The area occupied by nodes is fragmented into Voronoi Aurenhammer [1991] polygons, bringing the additional benefit of reflecting the sensor nodes density (Figure A.2). The individual polygons are filled with the shades of grey corresponding to the value of the selected attribute.

The results analysis is provided using the relational database. The XML formatted trace allows automatized import of the results. Pre-parsing of the data allows the automation of the queries generation. The integration with the visualization, provides means for spatially correlated queries, where queries are applied only to the nodes within a selected region.

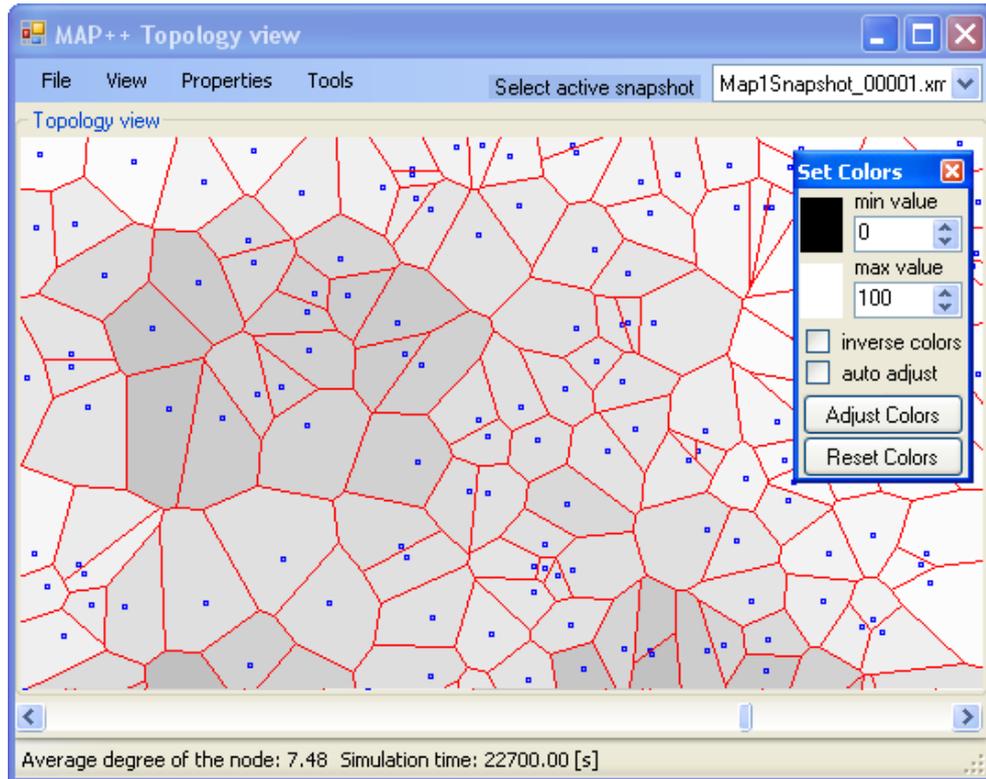


Figure A.2: Voronoi-Based Map Visualization

## A.5 MAP++ Benefits and Applications

In this section, first the main functionalities of the framework are presented at different stages of simulation process. Next, the main usage scenarios are presented. They which include modeling, design, debugging and performance evaluation. An OMNeT++ implementation of the eScan algorithm Zhao et al. [2002] is used in order to better illustrate the framework's capabilities. eScan is an efficient energy map construction algorithm based on polygon aggregation. A detailed case study using this algorithm is described in Section A.6.1.

### A.5.1 Overview

Three distinct stages of the simulation process can be identified, namely: modeling, execution and results analysis. In the following the MAP++ benefits are presented at each stage.

In the modeling stage MAP++ supports the definition of the modules, generation of the topology consisting of defined modules and creation of scenario sets. The definition of a new module consists of the implementation of a new module class and defining set of class attributes. The implementation of attributes on the developers side is limited to updating their value inside the module source code. The topology generation results in an automatized creation of the topology files. Scenario sets correspond to producing the initialization files with sections describing varying values of published attributes. The initial conditions (initial values of published attributes) can be visualized and effectively defined (spatially correlated initial values) using the visualization tool.

Regarding the execution stage of the simulation, MAP++ framework currently offers only assistance in the form of the Trace Module. However, the framework modular implementation allows for extending the usability of the Trace Module in future. Conceptually, the Trace Module could evaluate the maps at run-time. That capability would allow definition of conditions to which a set of actions could be assigned. For example on detecting a certain event the frequency of taking snapshots could be altered or snapshots apart from being periodical become event triggered. Other possibility of extending the framework functionality at this stage would be the run-time data visualization and real-time streaming of traces into a database.

The MAP++ support for result analysis consists of the visualization and data processing using relational database. The visualization is projected as two dimensional map, colored using shades of grey corresponding to the value of the selected attribute at a given time instant of the sensor nodes occupying the area. Basic operations like creation of differential maps are supported. Scrolling along time axis displays the progress of simulated phenomenons. The simulations result in the creation of a large set of data. There is a need for extracting significant data from a large set of less relevant data. MAP++ framework provides a solution for this problem by allowing the import of stored trace data into a relational database (Figure A.3). The use of structured queries allows an easy extraction of data of interest.

### A.5.2 Modeling Stage

The modeling capabilities of the framework are illustrated through the example of energy consumption modeling. As a result of the WSN inherent sensor nodes

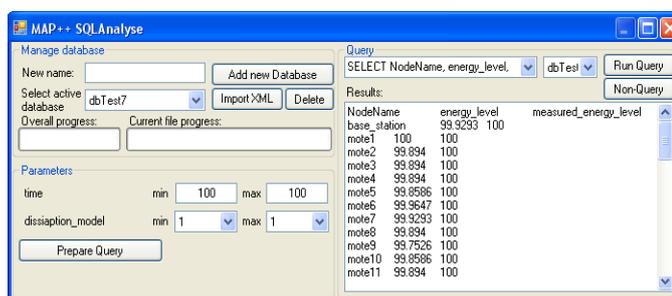


Figure A.3: Database Interface

redundancy, the energy depletion tends to show high spatial correlation. The sensor nodes placed in close proximity of the sink deplete their energy at a much faster rate Li and Mohapatra [2005]; Wu and Chen [2006] as, besides their own operations, they forward most of the WSN traffic. Another case of spatially correlated energy depletion is the spatial nature of most of the monitored physical phenomena. In the regions where the phenomena shows high activity, the reporting frequency has to be adjusted to meet the required accuracy of monitoring, resulting in an increased energy usage ratio.

Modeling starts with generating the topology corresponding to the assumed system model. MAP++ framework allows the setting of the communication range, size of the sensor field, number of sensor nodes and topology type (grid or uniform random). The creation of the scenarios containing mixed types of sensor nodes is supported by iteratively adding new module classes to existing topologies.

The topology generation is followed by accurate modeling of the phenomena to be monitored by the WSN. As physical phenomena usually show spatial correlations, the natural approach to model that is the generation of the corresponding maps. The iterative generation of maps for different scenarios (e.g. various distribution models) followed by their comparison enables precise modeling. MAP++ partially automates this process. Varying the values of the topology properties allows batch generation of scenarios. The visualized trace data simplifies investigation of their impact.

The form for designing batch execution currently supports three types of properties: numerical, boolean and string. In case of numerical values the user defines the minimal, maximal values and the step (value by which the parameter should be incremented) for generating values range. For the string properties, the creation of a list of values is possible. Boolean values are limited to select between one of two constant values, or their variation. The *Add batch* function automatically generates all spectrum of scenarios given the defined value ranges, while the *Add Single* function enables the generation of single scenarios.

### A.5.3 Design Stage

The most significant contribution of MAP++ framework is at the design phase. The visualization of maps leads to a better understanding of design choices. The designer can use framework tools to test tentative solutions and their preliminary performance. In the exemplary case of energy monitoring, the visualization may show the regions susceptible to partitioning, their shape and expected time of occurrence. This knowledge results in the development of valuable countermeasures for design or deployment. For example, the threatened regions of the network can be supplied with higher energy reserves, e.g., through denser deployment of nodes. Considering also the visualization of the physical phenomena such as temperature allows for an interactive design of algorithms. Such a visualization allows for a coarse-grained understanding of both problems and developed solutions. For example visualizing the energy distribution of the network along the network topology allows the identification of energy holes and some of their properties.

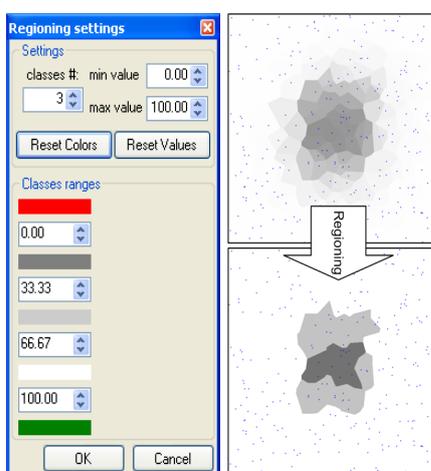
A static snapshot does not provide enough insights into the dynamics of the network. Therefore, the MAP++ visualization tools support smooth scrolling through the trace data along the time axis. This procedure unveils the causality of the events during the lifetime of the network. An example could be the investigation of the adaptability of a routing protocol to sensor nodes failures. For this purpose the designer may create a connectivity map, which visualizes the hop distance of each node to the sink. The visualization of this map (small differences in grey scale values show neighboring sensor nodes) reveals when and how the routing adjusts to changes in the network. Observing the network at the same time in different perspective (e.g.: energy and connectivity map) points the cause of rerouting.

The impact of varying parameters becomes visible when comparing the different scenarios at same time indexes. Visualization allows loading several traces at once and switching between their views, remaining at the same time index.

Testing the solutions and countermeasures for the identified problems requires the generation of additional scenarios. These scenarios should not only vary the simulator's parameters but also the spatial distribution of initial conditions. Manual text edition of the topology is a demanding and error prone task. Additionally, taking the spatial correlation of the initial values into consideration makes the process of scenario definition even more complex. Using the graphical interface and network visualization at design time, it is easy to identify and select neighboring sensor nodes and assign to them attribute values in collective manner. In case of energy monitoring the user can easily create the desired energy distribution simulating higher energy densities in more threatened regions.

MAP++ also allows for map-based tracing. Such a trace gives the set of re-

gions constituting the map for a certain time interval and lists the regions splitting and merging operations during that time. In order to set region information a *regioning* technique was implemented. The regioning algorithm logically groups neighboring sensor nodes belonging to the same value class (defined values range). Figure A.4(a) shows the value classes definition form. The designer can set the number of the classes and range of the values defining the class membership. For visualization purposes, every class is assigned a color, which is used to represent it on the map (Figure A.4(b)).



(a) Regioning Settings (b) Regioning

Figure A.4: Regioning in MAP++

#### A.5.4 Debugging and Performance Evaluation Stage

An important activity in the implementation of an algorithm is debugging. The map perspective simplifies the task of localization of a spatially correlated errors.

For example the energy distribution map can be used to identify routing algorithm implementation errors. Energy map showing occurrence of energy holes at unexpected locations, may indicate such routing implementation errors. The connectivity map offers additional information about the nature of the problem, when showing irregular distances to the sink.

The data aggregation errors can be identified by comparison of the ideal map with the one created by the aggregation algorithm. Comparison is directly supported by MAP++ and is achieved by creating the differential map of both views. The values of two selected attributes of each sensor node (e.g. real and aggregated

value) are subtracted from each other and the resulting value is used for coloring the area occupied by the corresponding sensor node.

Besides visualization, the trace data can be evaluated using the MAP++ database interface. The query tool searches active database for a set of changing parameters and creates a list of their distinct values. The user may choose the range of the values from the list in order to simplify the creation of database query. The selected values are automatically encoded as a query string (*Prepare Query* button, Figure A.3), so even user with limited knowledge of SQL may use the database interface.

The query tool is also integrated with the visualization. In order to find the snapshot of interest, the user may use the visualization for navigating to the desired time index. Additionally using the visualization interface it is possible to select only the nodes that should be addressed in the query. The query tool automatically generates the query that uses current time index and references only the selected nodes. An example is the evaluation of the localized accuracy of eScan as discussed in Section A.6.1.

## A.6 MAP++ Evaluation

First, MAP++ framework powerful utilities are demonstrated through a case study. Next, the framework is evaluated with respect to its impact on the scalability of OMNeT++.

### A.6.1 Case Study

The utility of the MAP++ framework is demonstrated using the example of *eScan* Zhao et al. [2002]. In the following first a brief description of eScan algorithm is given. Next, the MAP++ visualizations are used to select energy consumption model. Finally, the performance of eScan is investigated along with results.

The eScan energy map construction approach is based on polygon aggregation. The sink disseminates the interest/query for a map to all network nodes. The query is disseminated using flooding creating a spanning tree rooted at the sink. This tree is used to aggregate the attribute values while being reported. A leaf node sends its raw value to its parent node. An internal node (parent) gathers the input of all its children, aggregates it with its value and forwards it to its parent node. The aggregation consists in grouping sensor readings that meet a certain criteria (being geographically adjacent and in the same value range). The degree of aggregation is defined by the *tolerance* parameter  $T$ , corresponding to the pre-central difference between potential attribute values to be merged. The outcome of the aggregation is a list of (spatial) regions. A region is a polygon that is defined

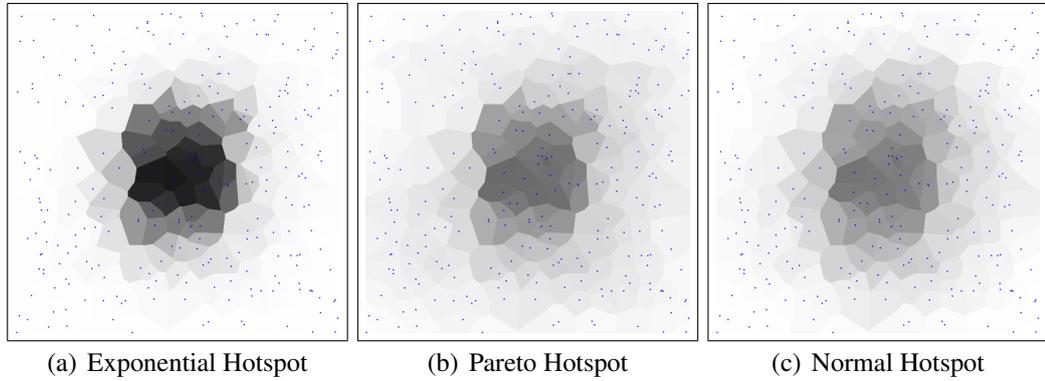


Figure A.5: Energy Map for Different Hotspot Energy Distribution Models

by the line spanning its border sensor nodes. At the sink the aggregation results in a complete map. Sensor nodes reply with their current values immediately on query and later only with necessary updates (*update interval* parameter defined as the precentral value change, after which the sensor nodes transmit an update).

eScan introduces three hotspot energy distribution models: exponential, pareto and normal, which are easily visualized using MAP++ in Figures A.5(a), (b), (c) respectively. In this models each sensor node  $n$  has a probability  $p = f(d)$ , where  $d$  is distance to the closest hotspot, that  $n$  and its neighboring sensor nodes perform sensing, which consumes fixed amount of energy. For simplification the case study arbitrarily concentrates only on one of distribution models, namely the exponential distribution. In order to closely recreate the eScan original scenario, derived from its source code, a network of 270 sensor nodes is considered, spread uniformly over the square area of size 30m x 30m. Sensor nodes have a communication range of 3m. The hotspot epicenter is located in the geographical center of the network (Figure A.5).

The batch scenario generator is used to vary the values of the tolerance and update interval parameters. A total of 6 scenarios for tolerance parameter values of 5%, 10% and 25% and update interval parameter values of 0.1% and 1% were generated. Additionally a relative differential map (modified version of the differential map discussed in Section A.5) is used, as the best way of illustrating the accuracy of the eScan algorithm. It is defined as a map of percentage difference between measured and actual value of energy. Coloring the map requires definition of an appropriate color schema. The white color is assigned to the lowest obtained value 0 (no relative error). The black color is assigned to the maximum, defined as the maximum discrepancy for all scenarios (in this case 30% relative error). Values between minimal and maximal values are colored in shades of gray.

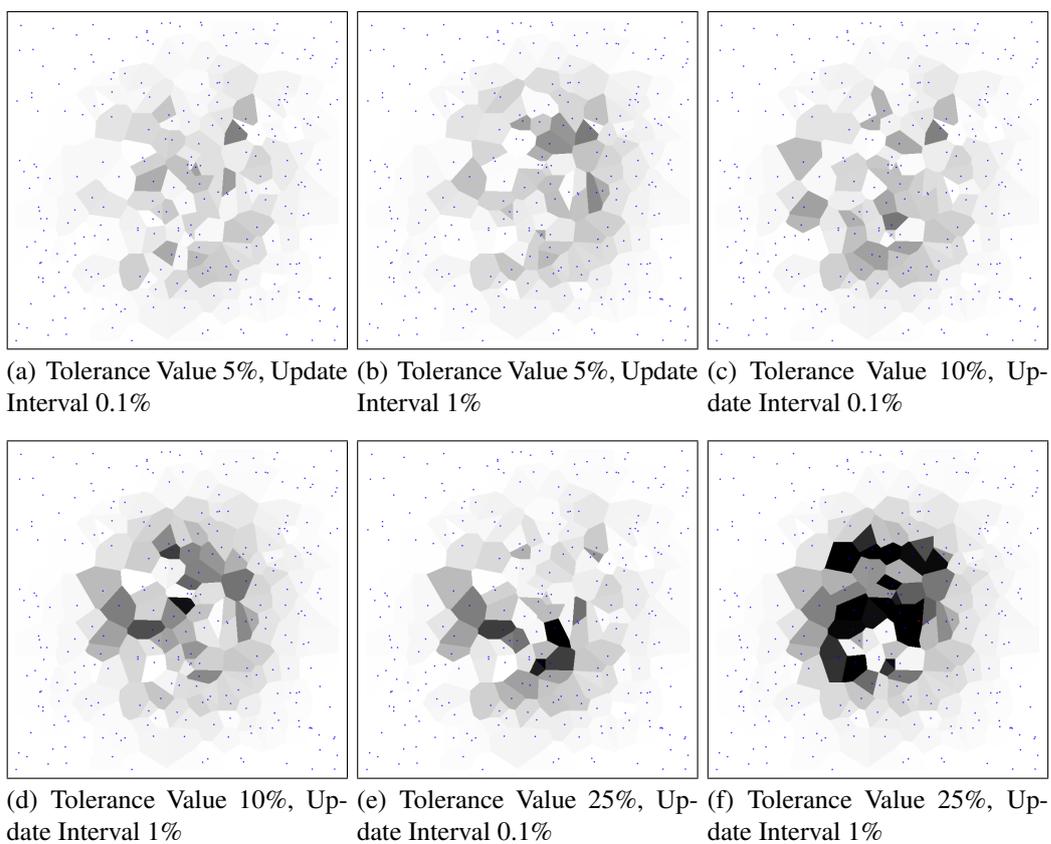


Figure A.6: Accuracy for Different Simulation Scenarios)

The same color schema is used for each visualization.

The comparison of Figure A.6(a) and Figure A.6(b) shows no significant difference regarding the accuracy of the algorithm as well the localization of error concentration. The hotspot occupied area (which extent was simulated in first step in Figure A.5(a)) is filled only with light shades of grey. Therefore, the use of higher update interval could be advised to reduce number of messages transmitted. The results presented in Figure A.6(c) show that despite higher tolerance value the accuracy of algorithm is acceptable, the concentration of light shades of grey is comparable with two previous snapshots. Figure A.6(d) shows that the tolerance is set too high to compensate the inaccuracy introduced by increased update level and significant differences between both sub-figures are manifested by occurrence of very dark spots in Figure A.6(d). The last set of figures (Figure A.6(e) and Figure A.6(f)) shows that at 25% of tolerance value the eScan map accuracy significantly suffers, and the negative effect is amplified by an increase of update interval value (Figure A.6(f)).

When analyzing the visualized results the important added value of proposed approach becomes evident. In original results the accuracy is averaged for the entire network. In case of the map visualization the regions that contribute mostly to the error can be easily localized. In case of eScan the conclusion is obvious that the center of hotspots show highest dynamics of the changes and consequently highest relative and absolute errors.

## A.6.2 Performance Evaluation of MAP++

With respect to performance evaluation of MAP++ the most important factor is the additional simulation time overhead generated by the Trace Module. This determines the usability and the overall scalability of MAP++ and its OMNeT++ environment. The ratio between simulation time with and without MAP++ Trace Module is used as a metric to quantify this overhead. The impact of the number of sensor nodes and the snapshots period is considered.

Figure A.7(a) presents the relative execution time overhead for varying number of sensor nodes (between 250 and 750) keeping the snapshot period at 200s of simulation time. It can be easily concluded that the time overhead remains fairly constant and equal to approximately 5%.

Figure A.7(b) depicts the impact of changing snapshot period from 50s to 800s of simulated time, for a fixed number of sensor nodes (500) on the execution time. The choice of the snapshot period is strongly dependent on the evaluated algorithm. The range that have been chosen was based on the properties of evaluated eScan and adjusted to its dynamics. In case of very short periods values (50s), the snapshot activities dominate the execution time (snapshots are taken more often than simulation events related to the eScan take place) resulting in a higher time

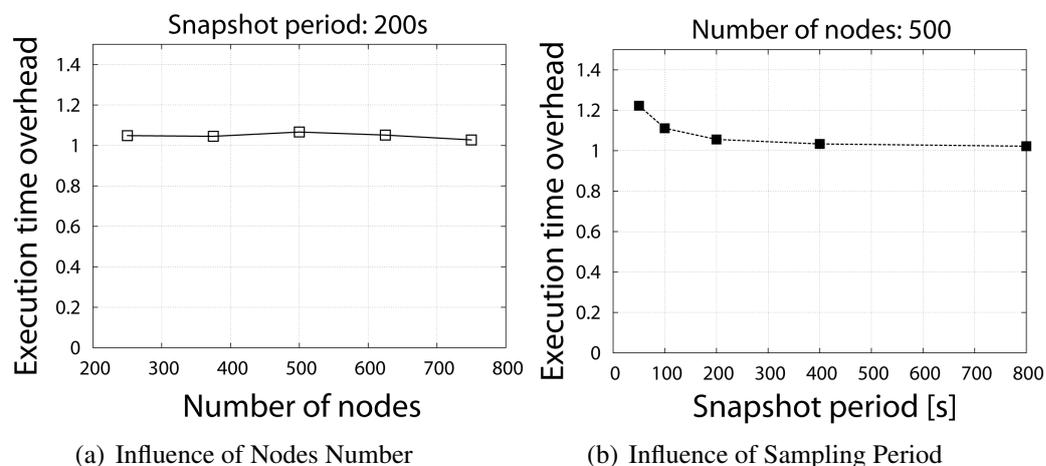


Figure A.7: MAP++ Performance

overhead.

Another issue for the performance of MAP++ is the file size of generated trace data. MAP++ generates files of acceptable sizes. For example, a simulation consisting of 500 snapshots of 750 sensor nodes, corresponds to a 20 MB trace file.

## A.7 Conclusions

In the field of the WSN research, the maps are the intuitive abstraction of the network. They expose the spatial nature of the network attributes and allow addressing the regions instead of the single sensor nodes. Layering the set of the maps discloses the dependencies existing in the network. MAP++ framework constitutes a comprehensive set of tools providing considerable support for the map abstraction. It supports all main steps of design effort for the network. Beginning with the user definition of the sensor nodes, through generation of the topologies and scenarios based on variation of simulation parameters. Continuing with the visualization of the results both in spatial and time domain, interactive comparison of the maps, and ending with the SQL powered results analysis.



# Bibliography

EYES WSN Simulation Framework. <http://wwwes.cs.utwente.nl/ewsnsim/>.

DEWSNet Dependable Embedded Wireless Sensor Networks. <http://www.deeds.informatik.tu-darmstadt.de/dewsnet/>.

Mannasim Framework. <http://www.mannasim.dcc.ufmg.br/>.

Mobility Framework for OMNeT++. <http://mobility-fw.sourceforge.net/>.

NesCT: A language translator. <http://nesct.sourceforge.net/>.

OMNeT++ Community Site. <http://www.omnetpp.org/>.

TinyOS. <http://www.tinyos.net/>.

Nadeem Ahmed, Salil S. Kanhere, and Sanjay Jha. The holes problem in wireless sensor networks: a survey. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9:4–18, April 2005. ISSN 1559-1662.

Ahmed, S. and Bilal, M. and Farooq, U. and Fazl-e-Hadi, N-W.F.P. Performance Analysis of various routing strategies in Mobile Ad hoc Network using Qual-Net simulator. In *Proceedings of the International Conference on Emerging Technologies*, pages 62–67, 2007.

Hisham M. Almasaeid and Ahmed E. Kamal. On the minimum k-connectivity repair in wireless sensor networks. In *Proceedings of the 2009 IEEE international conference on Communications, ICC'09*, pages 195–199, Piscataway, NJ, USA, 2009. IEEE Press. ISBN 978-1-4244-3434-3.

Waleed Alsalih, Kamrul Islam, Yurai Núñez Rodríguez, and Henry Xiao. Distributed voronoi diagram computation in wireless sensor networks. In *Proceedings of the Twentieth Annual Symposium on Parallelism in Algorithms and*

- Architectures*, SPAA '08, pages 364–364, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-973-9.
- T. Arici and Y. Altunbasak. Adaptive sensing for environment monitoring using wireless sensor networks. In *Proceedings of the Wireless Communications and Networking Conference*, pages 2347–2352, 2004.
- Nuzhet Atay and O. Burçhan Bayazit. Mobile wireless sensor network connectivity repair with k-redundancy. In *Workshop on the Algorithmic Foundations of Robotics*, pages 35–49, 2008.
- Franz Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23:345–405, September 1991. ISSN 0360-0300.
- Mohsen Bahramgiri, Mohammadtaghi Hajiaghayi, and Vahab S. Mirrokni. Fault-tolerant and 3-dimensional distributed topology control algorithms in wireless multi-hop networks. *Wireless Networks*, 12:179–188, March 2006. ISSN 1022-0038.
- Xiaole Bai, Dong Xuan, Ziqiu Yun, Ten H. Lai, and Weijia Jia. Complete optimal deployment patterns for full-coverage and k-connectivity ( $k \geq 6$ ) wireless sensor networks. In *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '08, pages 401–410, New York, NY, USA, 2008a. ACM. ISBN 978-1-60558-073-9.
- Xiaole Bai, Ziqiu Yun, Dong Xuan, Ten-hwang Lai, and Weijia Jia. Deploying four-connectivity and full-coverage wireless sensor networks. In *Proceedings of IEEE INFOCOM*, pages 296–300, 2008b. doi: 10.1109/INFOCOM.2008.68.
- Christian Bettstetter. On the minimum node degree and connectivity of a wireless multihop network. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '02, pages 80–91, New York, NY, USA, 2002. ACM. ISBN 1-58113-501-7.
- Jonathan L. Bredin, Erik D. Demaine, MohammadTaghi Hajiaghayi, and Daniela Rus. Deploying sensor networks with guaranteed capacity and fault tolerance. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '05, pages 309–319, New York, NY, USA, 2005. ACM. ISBN 1-59593-004-3.
- Sergiy Butenko, Xiuzhen Cheng, Carlos A. S Oliveira, and P. M. Pardalos. *Recent Developments in Co-operative Control and Optimization*. 2003. ISBN 9781402076442.

- Zack Butler and Daniela Rus. Event-based motion control for mobile-sensor networks. *IEEE Pervasive Computing*, 2:34–42, October 2003. ISSN 1536-1268.
- Jyh-Huei Chang and Rong-Hong Jan. An efficient relay sensors placing algorithm for connectivity in wireless sensor networks. In *Proceedings of the International Conference on Embedded and Ubiquitous Computing*, pages 874–883, 2006.
- C.-F. Chiasserini and M. Garetto. Modeling the performance of wireless sensor networks. In *Proceedings of IEEE INFOCOM*, 2004.
- Chin, Gilbert Y. and Branch, Joel W. and Brevdo, Eugene and Zhu, Lijuan and Szymanski, Boleslaw. SENSE: A Sensor Network Simulator. *Advances in Pervasive Computing and Networking*, pages 249–267, 2004.
- Fei Dai and Jie Wu. On constructing k-connected k-dominating set in wireless networks. In *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Papers - Volume 01*, IPDPS '05, pages 81.1–, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2312-9.
- Jitender S. Deogun, Saket Das, Haitham S. Hamza, and Steve Goddard. An algorithm for boundary discovery in wireless sensor networks. In *Proceedings of HiPC'2005*, pages 343–352, 2005.
- Gianluca Dini, Marco Pelagatti, and Ida Maria Savino. An algorithm for reconnecting wireless sensor network partitions. In *Proceedings of the 5th European conference on Wireless sensor networks*, EWSN'08, pages 253–267, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 3-540-77689-3, 978-3-540-77689-5.
- Thomas Dreibholz and Erwin P. Rathgeb. A powerful tool-chain for setup, distributed processing, analysis and debugging of omnet++ simulations. In *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, Simutools '08, pages 74:1–74:8, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). ISBN 978-963-9799-20-2.
- Deborah Estrin, Mark Handley, John Heidemann, Steven McCanne, Ya Xu, and Haobo Yu. Network visualization with nam, the vint network animator. *Computer*, 33:63–68, November 2000. ISSN 0018-9162.

- Qing Fang, Jie Gao, and Leonidas J. Guibas. Locating and bypassing holes in sensor networks. *Mobile Networks and Applications*, 11:187–200, April 2006. ISSN 1383-469X.
- Luiz Filipe, M. Vieira, Marcos Augusto, M. Vieira, Linnyer Beatrys Ruiz, Antonio A. F. Loureiro, Diógenes Cecílio Silva, and Antônio Otávio Fern. Efficient incremental sensor network deployment algorithm. In *Proceedings of IEEE Wireless Communications and Networking Conference*, pages 78–92, 2004.
- S. Ganeriwal, A. Kansal, and M.B. Srivastava. Self aware actuation for fault repair in sensor networks. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 5244–5249, 2004.
- Bu&#728;gra Gedik, Ling Liu, and Philip S. Yu. Asap: An adaptive sampling approach to data collection in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 18:1766–1783, December 2007. ISSN 1045-9219.
- Arpita Ghosh and Stephen Boyd. Growing well-connected graphs. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 6605–6611. Ieee, 2006.
- Haosong Gou and Younghwan Yoo. Distributed bottleneck node detection in wireless sensor network. In *Proceedings of the 10th IEEE International Conference on Computer and Information Technology*, CIT '10, pages 218–224, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4108-2.
- Olga Goussevskaia, M. D. V. Machado, Raquel A. F. Mini, Antonio Alfredo Ferreira Loureiro, Geraldo Robson Mateus, and José Marcos S. Nogueira. Data dissemination based on the energy map. *IEEE Communications Magazine*, 43: 134–143, 2005a.
- Olga Goussevskaia, Raquel A. F. Mini, Cristiano G. Rezende, Antonio Alfredo Ferreira Loureiro, Geraldo Robson Mateus, and José Marcos S. Nogueira. Data dissemination in autonomic wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 23:2305–2319, 2005b. doi: 10.1109/JSAC.2005.857209.
- Carlos Guestrin, Andreas Krause, and Ajit Paul Singh. Near-optimal sensor placements in gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pages 265–272, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5.
- Piyush Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions On Information Theory*, 46(2):388–404, 2000.

- MohammadTaghi Hajiaghayi, Nicole Immorlica, and Vahab S. Mirrokni. Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, MobiCom '03, pages 300–312, New York, NY, USA, 2003. ACM. ISBN 1-58113-753-2.
- Xiaofeng Han, Xiang Cao, Errol L. Lloyd, and Chien-Chung Shen. Fault-tolerant relay node placement in heterogeneous wireless sensor networks. *IEEE Transactions on Mobile Computing*, 9:643–656, May 2010. ISSN 1536-1233.
- Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek F. Abdelzaher. Range-free localization and its impact on large scale sensor networks. *ACM Transactions on Embedded Computing Systems*, 4:877–906, November 2005. ISSN 1539-9087.
- Nojeong Heo and P. K. Varshney. Energy-efficient deployment of Intelligent Mobile sensor networks. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 35(1):78–92, 2005.
- Kévin Huguenin and M. João Rendas. Distributed adaptive sampling using bounded-errors. In *Proceedings of the 1st international conference on Robot communication and coordination*, RoboComm '07, pages 54:1–54:6, Piscataway, NJ, USA, 2007. IEEE Press. ISBN 978-963-9799-08-0.
- Ahmed S. Ibrahim, Karim G. Seddik, and K. J. Ray Liu. Improving connectivity via relays deployment in wireless sensor networks. In *Proceedings of GLOBE-COM'07*, pages 1159–1163, 2007.
- Jehn-Ruey Jiang and Tzu-Ming Sung. Energy-efficient coverage and connectivity maintenance for wireless sensor networks. *Journal of Networks*, 4:403–410, 2009. doi: 10.4304/jnw.4.6.403-410.
- Milenko Jorgić, Ivan Stojmenović, Michaël Hauspie, and David Simplotryl. Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks. In *Proceedings the Third Annual IFIP Mediterranean Ad Hoc Networking Workshop, Med-Hoc-Net*, pages 360–371, 2004.
- Milenko Jorgic, Nishith Goel, Kalai Kalaichelvan, Amiya Nayak, and Ivan Stojmenovic. Localized detection of k-connectivity in wireless ad hoc, actuator and sensor networks. In *Proceedings of the 16th International Conference on Computer Communications and Networks*, pages 33–38, 2007.

- Abdelmajid Khelil, Faisal Karim Shaikh, Azad Ali, and Neeraj Suri. gMAP: an efficient construction of global maps for mobility-assisted wireless sensor networks. In *Proc. of WONS*, pages 189–196, 2009.
- Abdelmajid Khelil, Faisal Karim Shaikh, Azad Ali, and Neeraj Suri. *Delay Tolerant Networks: Protocols and Applications*, chapter Delay-Tolerant Monitoring of Mobility-Assisted Wireless Sensor Networks. Auerbach Publications, Taylor & Francis Group, 2010.
- Khelil, Abdelmajid and Shaikh, Faisal Karim and Ayari, Brahim and Suri, Neeraj. MWM: a map-based world model for wireless sensor networks. In *Proceedings of the 2nd International Conference on Autonomic Computing and Communication Systems*, Autonomics '08, pages 5:1–5:10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). ISBN 978-963-9799-34-9.
- A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. Klein Haneveld, T. E. V. Parker, O. W. Visser, H. S. Lichte, and S. Valentin. Simulating wireless and mobile networks in omnet++ the mixim vision. In *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, Simutools '08, pages 71:1–71:8, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). ISBN 978-963-9799-20-2.
- Henri Koskinen, Jouni Karvo, and Olli Apilo. Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks. In *IFIP*, pages 169–178, 2006.
- Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. In *Proceedings of the 5th international conference on Information processing in sensor networks*, IPSN '06, pages 2–10, New York, NY, USA, 2006. ACM. ISBN 1-59593-334-4.
- Tronje Krop, Michael Bredel, Matthias Hollick, and Ralf Steinmetz. Jist/mobnet: combined simulation, emulation, and real-world testbed for ad hoc networks. In *Proceedings of the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*, WinTECH '07, pages 27–34, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-738-4.
- Jean-Claude Laprie, editor. *Dependability: Basic Concepts and Terminology*. Springer Verlag, 1992.

- M. Leoncini, G. Resta, and Paolo Santi. Analysis of a wireless sensor dropping problem in wide-area environmental monitoring. In *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks*, pages 239–245, April 2005.
- Johannes Lessmann and Tales Heimfarth. Flexible offline-visualization for mobile wireless networks. In *Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, pages 404–409, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3114-4.
- Johannes Lessmann, Tales Heimfarth, and Peter Janacik. Shox: An easy to use simulation platform for wireless networks. In *Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, pages 410–415, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3114-4.
- Philip Levis, Nelson Lee, Matt Welsh, and David Culler. Tossim: accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 126–137, New York, NY, USA, 2003. ACM. ISBN 1-58113-707-9.
- Jian Li and Prasant Mohapatra. An analytical model for the energy hole problem in many-to-one sensor networks. In *Proceedings of Vehicular Technology Conference (VTC)*, pages 2721–2725, 2005.
- Mo Li, Yunhao Liu, and Lei Chen. Non-threshold based event detection for 3d environment monitoring in sensor networks. In *Proceedings of the 27th International Conference on Distributed Computing Systems*, ICDCS '07, pages 9–, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2837-3.
- Ning Li and Jennifer C. Hou. Flss: a fault-tolerant topology control algorithm for wireless networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, MobiCom '04, pages 275–286, New York, NY, USA, 2004. ACM. ISBN 1-58113-868-7.
- Wei Li and Christos G. Cassandras. A minimum-power wireless sensor network self-deployment scheme. *IEEE Wireless Communications and Networking Conference 2005*, 3:1897–1902, 2005.
- Xiang-Yang Li, Peng-Jun Wan, Yu Wang, and Chih-Wei Yi. Fault tolerant deployment and topology control in wireless networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '03, pages 117–128, New York, NY, USA, 2003. ACM. ISBN 1-58113-684-6.

- An-Feng Liu, Xian-You Wu, and Wei-Hua Gui. Research on energy hole problem for wireless sensor networks based on alternation between dormancy and work. In *Proceedings of the 2008 The 9th International Conference for Young Computer Scientists*, pages 475–480, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3398-8.
- Yunhao Liu and Mo Li. Iso-map: Energy-efficient contour mapping in wireless sensor networks. In *Proceedings of the 27th International Conference on Distributed Computing Systems*, volume 22, pages 36–36. Ieee, 2007.
- Errol L. Lloyd and Guoliang Xue. Relay node placement in wireless sensor networks. *IEEE Transactions on Computers*, 56:134–138, January 2007. ISSN 0018-9340.
- Pascal Lorenz and Petre Dini, editors. *A Study of Reconnecting the Partitioned Wireless Sensor Networks*, volume 3420 of *Lecture Notes in Computer Science*, 2005. Springer. ISBN 3-540-25339-4.
- Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. *ACM SIGOPS Operating Systems Review*, 36:131–146, December 2002. ISSN 0163-5980.
- Saucene Mahfoudh and Pascale Minet. Survey of energy efficient strategies in wireless ad hoc and sensor networks. In *Proceedings of the Seventh International Conference on Networking*, pages 1–7, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3106-9.
- Morteza Maleki and Massoud Pedram. Qom and lifetime-constrained random deployment of sensor networks for minimum energy consumption. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press. ISBN 0-7803-9202-7.
- S. McCanne and S. Floyd. NS Network Simulator. <http://www.isi.edu/nsnam/ns/>.
- Xiaoqiao Meng, Thyaga Nandagopal, Li Li, and Songwu Lu. Contour maps: monitoring and diagnosis in sensor networks. *Computer Networks*, 50:2820–2838, October 2006. ISSN 1389-1286.
- Satyajayant Misra, Seung Don Hong, Guoliang Xue, and Jian Tang. Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirements. In *Proceedings of IEEE INFOCOM*, pages 281–285, 2008. doi: 10.1109/INFOCOM.2008.65.

- Stephan Olariu and Ivan Stojmenoviæ. Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting. In *Proceedings of IEEE INFOCOM*, pages 1–12. Society Press, 2006.
- Pham, Hai N. and Pediaditakis, Dimosthenis and Boulis, Athanassios. From Simulation to Real Deployments in WSN and Back. In *Proceedings of the 8th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 1–6, 2007.
- D.O. Popa, A.C. Sanderson, R.J. Komerska, S.S. Mupparapu, D.R. Blidberg, and S.G. Chappel. Adaptive sampling algorithms for multiple autonomous underwater vehicles. In *Proceedings of IEEE International Conference on Autonomous Underwater Vehicles*, pages 108–118, 2004.
- Popa, D.O. and Mysorewala, M.F. and Lewis, F.L. EKF-based Adaptive Sampling with Mobile Robotic Sensor Nodes. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, pages 2451–2456, 2006.
- Juhua Pu, Zhang Xiong, and Xiaofeng Lu. Fault-tolerant deployment with k-connectivity and partial k-connectivity in sensor networks. *Wireless Communications & Mobile Computing*, 9:909–919, July 2009. ISSN 1530-8669.
- M. Rahimi, R. Pon, W.J. Kaiser, G.S. Sukhatme, D. Estrin, and M. Srivastava. Adaptive sampling for estimating a scalar field using a robotic boat and a sensor network. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 3537–3544, 2004.
- M. Rahimi, R. Pon, W.J. Kaiser, G.S. Sukhatme, and D. Estrin. Adaptive sampling for environmental field estimation using robotic sensors. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3692–3698, 2005.
- Kui Ren, Kai Zeng, and Wenjing Lou. Secure and fault-tolerant event boundary detection in wireless sensor networks. *IEEE Transactions on Wireless Communications*, 7:354–363, 2008. doi: 10.1109/TWC.2008.060550.
- Arthur H. Robinson, Joel L. Morrison, Phillip C. Muehrcke, A. Jon Kimerling, and Stephen C. Guptill. *Elements of Cartography*. John Wiley & Sons, New York, 1995. 6th Edition.
- Indranil Saha, Lokesh Kumar Sambasivan, Ranjeet Kumar Patro, and Subhas Kumar Ghosh. Distributed fault-tolerant topology control in static and mobile

- wireless sensor networks. In *Proceedings of the 2nd International Conference on Communication Systems Software and Middleware*, pages 1–8, 2007.
- Prasan Kumar Sahoo, Jang-ping Sheu, and Wei-shin Lin. Dynamic coverage and connectivity maintenance algorithms for wireless sensor networks. In *Communication System softWARE and MiddlewaRE*, 2007. doi: 10.1109/COMSWA.2007.382574.
- Paolo Santi. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys*, 37:164–194, June 2005. ISSN 0360-0300.
- Paolo Santi and Janos Simon. Silence is golden with high probability: Maintaining a connected backbone in wireless sensor networks. In *EWSN'04*, pages 106–121, 2004.
- Rik Sarkar, Xianjin Zhu, Jie Gao, Leonidas J. Guibas, and Joseph S. B. Mitchell. Iso-contour queries and gradient routing with guaranteed delivery in sensor networks. In *Proceedings of IEEE INFOCOM*, pages 960–967, 2008.
- Mehdi Sharifzadeh and Cyrus Shahabi. Supporting spatial aggregation in sensor network databases. In *Proceedings of the 12th annual ACM international workshop on Geographic information systems, GIS '04*, pages 166–175, New York, NY, USA, 2004. ACM. ISBN 1-58113-979-9.
- Kuei-Ping Shih, Hung-Chang Chen, Jing-Kuen Tsai, and Chun-Chih Li. Palm: A partition avoidance lazy movement protocol for mobile sensor networks. In *Proceedings of Wireless Communications and Networking Conference*, pages 2484–2489, 2007.
- Sobeih, Ahmed and Chen, Wei-Peng and Hou, Jennifer C. and Kung, Lu-Chuan and Li, Ning and Lim, Hyuk and Tyan, Hung-ying and Zhang, Honghai. J-Sim: a simulation and emulation environment for wireless sensor networks. *IEEE Wireless Communications*, 13(4):104–119, 2006.
- Ignacio Solis and Katia Obraczka. Isolines: Energy-efficient mapping in sensor networks. In *Proceedings of the 10th IEEE Symposium on Computers and Communications*, pages 379–385, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2373-0.
- Andrey Somov, Vinay Sachidananda, and Roberto Passerone. A self-powered module with localization and tracking system for paintball. In *Proceedings of the 3rd International Workshop on Self-Organizing Systems, IWSOS '08*, pages 182–193, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-92156-1.

Anand Srinivas, Gil Zussman, and Eytan Modiano. Construction and maintenance of wireless mobile backbone networks. *IEEE/ACM Transactions on Networking*, 17:239–252, February 2009. ISSN 1063-6692.

Sumana Srinivasan, Krithi Ramamritham, and Purushottam Kulkarni. Ace in the hole: Adaptive contour estimation using collaborating mobile sensors. In *Proceedings of the 7th international conference on Information processing in sensor networks*, IPSN '08, pages 147–158, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3157-1.

G S Sukhatme and G S Sukhatme. Adaptive sampling for marine microorganism monitoring. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1115–1122, 2004.

Piotr Szczytowski, Abdelmajid Khelil, and Neeraj Suri. Asample: Adaptive spatial sampling in wireless sensor networks. In *Proceedings of the 2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, SUTC '10, pages 35–42, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4049-8.

Di Tian and Nicolas D. Georganas. Connectivity maintenance and coverage preservation in wireless sensor networks. *Ad Hoc Networks*, 3:744–761, November 2005. ISSN 1570-8705.

vast. Space Time Toolkit. <http://vast.uah.edu/>.

Serdar Vural and Eylem Ekici. Analysis of hop-distance relationship in spatially random sensor networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '05, pages 320–331, New York, NY, USA, 2005. ACM. ISBN 1-59593-004-3.

Serdar Vural and Eylem Ekici. On multihop distances in wireless sensor networks with random node locations. *IEEE Transactions on Mobile Computing*, 9:540–552, April 2010. ISSN 1536-1233.

Chu-Fu Wang and Jen-Wen Ding. The optimum sensor redeployment scheme using the most frangible clusters set. *Computer Communications*, 31:3492–3502, September 2008. ISSN 0140-3664.

Guiling Wang, Guohong Cao, and Tom La Porta. Proxy-based sensor deployment for mobile sensor networks. In *Proceedings of IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, pages 493–502, 2004.

- Guiling Wang, Guohong Cao, Tom La Porta, and Wensheng Zhang. Sensor relocation in mobile sensor networks. In *Proceedings of IEEE INFOCOM*, pages 2302–2312, 2005.
- Guiling Wang, Guohong Cao, and Thomas F. La Porta. Movement-assisted sensor deployment. *IEEE Transactions on Mobile Computing*, 5:640–652, June 2006a. ISSN 1536-1233.
- Xiaorui Wang, Guoliang Xing, Yuanfang Zhang, Chenyang Lu, Robert Pless, and Christopher Gill. Integrated coverage and connectivity configuration in wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 28–39, New York, NY, USA, 2003. ACM. ISBN 1-58113-707-9.
- You-Chiun Wang, Wen-Chih Peng, Min-Hsien Chang, and Yu-Chee Tseng. Exploring load-balance to dispatch mobile sensors in wireless sensor networks. In *Proceedings of the 16th International Conference on Computer Communications and Networks*, pages 669–674, 2007.
- Yue Wang, Jie Gao, and Joseph S.B. Mitchell. Boundary recognition in sensor networks by topological methods. In *Proceedings of the 12th annual international conference on Mobile computing and networking*, MobiCom '06, pages 122–133, New York, NY, USA, 2006b. ACM. ISBN 1-59593-286-0.
- Rebecca Willett, Aline Martin, and Robert Nowak. Backcasting: adaptive sampling for sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, IPSN '04, pages 124–133, New York, NY, USA, 2004. ACM. ISBN 1-58113-846-6.
- Xiaobing Wu and Guihai Chen. On the energy hole problem of nonuniform node distribution in wireless sensor networks. In *Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 180–187, 2006.
- Yiwei Wu and Yingshu Li. Construction algorithms for k-connected m-dominating sets in wireless sensor networks. In *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '08, pages 83–90, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-073-9.
- Wenwei Xue, Qiong Luo, Lei Chen, and Yunhao Liu. Contour map matching for event detection in sensor networks. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, SIGMOD '06, pages 145–156, New York, NY, USA, 2006. ACM. ISBN 1-59593-434-0.

- Yunjiao Xue, Ho Sung Lee, Ming Yang, Priyantha Kumarawadu, Hamada H Ghenniwa, and Weiming Shen. Performance evaluation of ns-2 simulator for wireless sensor networks. *Performance Evaluation*, pages 1372–1375, 2007.
- Chang Wu Yu, Elvis Chen, and Chun-Cheng Fang. Deploying mobile nodes to connect wireless sensor networks using novel algorithms. In *Proceedings of the International Conference on Wireless Algorithms, Systems and Applications, WASA '07*, pages 199–204, Washington, DC, USA, 2007a. IEEE Computer Society. ISBN 0-7695-2981-X.
- Fucaai Yu, Euisin Lee, Younghwan Choi, Soochang Park, Donghun Lee, Ye Tian, and Sang-Ha Kim. A modeling for hole problem in wireless sensor networks. In *Proceedings of the 2007 international conference on Wireless communications and mobile computing, IWCMC '07*, pages 370–375, New York, NY, USA, 2007b. ACM. ISBN 978-1-59593-695-0.
- Yuanyuan Zeng, Cormac J. Sreenan, Naixue Xiong, Laurence T. Yang, and Jong Hyuk Park. Connectivity and coverage maintenance in wireless sensor networks. *The Journal of Supercomputing*, 52:23–46, April 2010. ISSN 0920-8542.
- Bin Zhang and G. S. Sukhat. Controlling sensor density using mobility. In *Proceedings of the 2nd IEEE workshop on Embedded Networked Sensors*, pages 141–149, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7803-9246-9.
- Bin Zhang and G.S. Sukhatme. Adaptive sampling for estimating a scalar field using a robotic boat and a sensor network. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 3673–3680, 2007.
- Jerry Zhao, Ramesh Govindan, and Deborah Estrin. Residual energy scan for monitoring sensor networks. In *Proceedings of the Wireless Communications and Networking Conference*, pages 356–362, 2002.
- Zongheng Zhou, Samir R. Das, and Himanshu Gupta. Variable radii connected sensor cover in sensor networks. *ACM Transactions on Sensor Networks*, 5: 8:1–8:36, February 2009. ISSN 1550-4859.



# Index

- angular ordering, 45
- bottleneck links, 82
- connectivity, 17
- coverage, 14
- energy holes, 27, 35
  - profiling, 38
- fault tolerance, 6, 73
- framework, 3
- holes, 27
- k-connectivity, 19, 73
- maintenance, 3
- map, 119
- Minimum Connected Dominated Set,
  - 76
- reconfiguration, 46, 65, 84, 100
- relay nodes, 31
- sampling, 7, 23, 91
  - over-sampling, 91, 97
  - under-sampling, 91, 95
- self-sustainability, 84
- simulation, 121
  - design, 129
  - modeling, 127
- thesis
  - contributions, 9
  - research questions, 8
    - conceptual, 8
    - resulted publications, 10
    - structure, 11
- topology, 5
  - holes, 28
  - irregularity, 55
  - regularity, 55
- topology control, 21
- Voronoi, 94



# Curriculum Vitae

## Personal Data

**Name:** Piotr Szczytowski

**Date of birth:** August 25th, 1980

**Place of birth:** Pyskowice, Poland

## School Education

**1987-1995** Primary School, Gliwice, Poland

**1995-1999** High School, "II Liceum Ogólnokształcące", Gliwice, Poland

## University Education

**1999-2004** *Master of Science, Engineer in Computer Science* – Faculty Of Automatic Control, Electronics And Computer Science, Silesian University of Technology, Gliwice, Poland

**2008-2011** *Ph.D. in Computer Science* – Technische Universität Darmstadt, Darmstadt, Germany

## Professional Occupation

**2011-present** *Research Scientist* – NEC Laboratories Europe, Heidelberg, Germany

## Publications

### Conference Publications

- Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *LEHP: Localized Energy Hole Profiling in Wireless Sensor Networks*, in Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Riccione, pp. 100–106, 2010
- Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *ASample: Adaptive Spatial Sampling in Wireless Sensor Networks*, in Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), Newport Beach, pp. 35–42, 2010
- Piotr Szczytowski**, Faisal Karim Shaikh, Vinay Sachidananda, Abdelmajid Khelil, Neeraj Suri, *Mobility Assisted Adaptive Sampling in Wireless Sensor Networks*, in Proceedings of the International Conference on Networked Sensing Systems (INSS), Demo session, Kassel, 2010
- Piotr Szczytowski**, Abdelmajid Khelil, Azad Ali and Neeraj Suri, *TOM: Topology Oriented Maintenance in Sparse Wireless Sensor Networks*, in Proceedings of the IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), Salt Lake City, 2011
- Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *DKM: Distributed k-Connectivity Maintenance in Wireless Sensor Networks*, in Proceedings of the Annual Conference on Wireless On-demand Network Systems and Services (WONS), Courmayeur, 2012
- Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *Map-based modeling and design of Wireless Sensor Networks with OMNeT++*, in Proceedings of International Symposium on Performance Evaluation of Computer & Telecommunication Systems (SPECTS), Istanbul, pp. 162–169, 2009
- Piotr Szczytowski**, Abdelmajid Khelil and Neeraj Suri, *MAP++: Support for Map-Based WSN Modeling and Design with OMNeT++*, in Proceedings of the 2nd International Workshop on OMNeT++, 2009
- Azad Ali, Abdelmajid Khelil, **Piotr Szczytowski** and Neeraj Suri, *An Adaptive and Composite Spatio-Temporal Data Compression Approach for Wireless Sensor Networks*, in Proceedings of ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM), 2011

Faisal Karim Shaikh, Abdelmajid Khelil, Brahim Ayari, **Piotr Szczytowski** and Neeraj Suri, *Generic Information Transport for Wireless Sensor Networks*, in Proceedings of the Third IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), 2010

### **Book Chapters**

Abdelmajid Khelil, Christian Reinl, Brahim Ayari, Faisal Karim Shaikh, **Piotr Szczytowski**, Azad Ali, Neeraj Suri, *Wireless Sensor Cooperation for a Sustainable Quality of Information*, Chapter 6 in the Book "Pervasive Computing and Networking" (Eds. M.S. Obaidat, M. Denko and I. Wounggang), John Wiley, ISBN 978-0-470-74772-8, pp 71-100, 2011

Abdelmajid Khelil, Christian Reinl, Brahim Ayari, Faisal Karim Shaikh, **Piotr Szczytowski**, Azad Ali, Neeraj Suri, *Sensor Cooperation for a Sustainable Quality of Information*, in "Pervasive Computing and Networking", Wiley, Edited by Prof. Mohammad S. Obaidat and Dr. Mieso Denko, 2010

Abdelmajid Khelil, Faisal Karim Shaikh, **Piotr Szczytowski**, Brahim Ayari and Neeraj Suri, *Map-based Design for Autonomic Wireless Sensor Networks*, in "Autonomic Communication", Springer-Verlag, Berlin/Heidelberg/New York, Edited by A. Vasilakos, M. Parashar, S. Karnouskos, W. Pedrycz, ISBN: 978-0-387-09752-7, 2009

