Be the change you want to see in the world.

– Mahatma Gandhi.

University of Alberta

DESIGNING HIERARCHICAL WSNS FOR HETEROGENEOUS OUTDOOR ENVIRONMENTS

by

Seyed Hossein Mortazavi

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

I lovingly dedicate this thesis to my Mother and Father
For teaching me about the importance of knowledge and the significance of endeavour.

Abstract

In this project we study the problem of wireless sensor network (WSN) node placement in a modelled environment. Although various optimal and sub-optimal techniques with different objectives and constraints such as maximizing coverage, network lifetime and reliable data transfer have previously been proposed for different variations of the node placement problem, many of these methods make various simplifying assumptions such as homogeneous transmission ranges among nodes. In our work we model the real environment and then based on our model we specifically design two node placement algorithms for two-tiered heterogeneous networks that aim to solve problems such as minimizing the number of RNs and SNs needed to satisfy a coverage constraint while maintaining connectivity (with fault-less connections) of the SNs to the base station and also maximizing the area covered by a specific number of RNs and SNs.

Acknowledgements

I would like to express my sincerest gratitude to my supervisor, Prof. Mike MacGregor, whose support and encouragement was an inspiration for me throughout this thesis. Without his invaluable patience, priceless assistance and excellent knowledge this thesis would not have been possible. I would also like to thank Mr. Cassidy Rankine for his appreciated recommendations and suggestions which directed me towards a correct path in this research.

My friends and my family are my most valuable assets. I would like to thank my family for their everlasting love and support throughout my life and for their absolute confidence in me. I doubt that I would ever be able to properly convey my appreciation and express my gratitude fully to them.

My sincere thanks go to my life-long friend Mr. Mohammad Salehe whose incredible knowledge and unique insight assisted me throughout this project. I am blessed to have such a brilliant, kind and trustworthy friend. I would also like to thank my exceptional friends Mr. Mostafa Vafadoost and Mr. Amir Malekzadeh for their special support and encouragement. It is always easier and better with them.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over the past decade, the quickly growing technology of Wireless Sensor Networks (WSNs) has been used in a wide range of applications, such as agriculture, transportation, home automation, search and rescue, industrial and environmental monitoring, military applications and medical care [2, 54, 60].

One category of these applications is Outdoor Environment Monitoring (OEM). WSN's have been used in detecting forest fires, monitoring endangered habitats, watching over volcanoes, detecting soil erosion, etc. [8, 39, 48]. However designing efficient applicable networks in outdoor environments faces major challenges such as dealing with limited connectivity, faulty connections, limited lifetime, the cost of the nodes and sometimes a need to cover huge geographical areas. In particular, in static networks where the location of the nodes is not changed frequently, node placement becomes a crucial concern and can affect many properties of the network, as well as the quality of the data sensed about the environment.

In difficult environments such as tropical dry forests, the potential cost of ”cut and try” on the ground is prohibitive. WSN deployments in such environments can't be addressed by sending people out multiple times to adjust the network due to cost, environmental impact on the plants or animals being monitored, and access issues. Planning techniques are required that can predict ahead of time where best to locate sensor nodes (SNs), repeaters and other equipment. These techniques must be able to factor in historically known and fairly predictable seasonal changes, or environmental factors that can be sensed remotely via satellite, aircraft or UAVs. These changes and factors are important because they can affect the RF environment of the network (e.g. reducing RF signal propagation), as well as dictating how densely sensors must be placed to reliably sense variables such as photosynthetically active radiation (PAR).

For example, some WSN nodes may need to be deployed in inaccessible terrain. Reaching these nodes to relocate them (due to changes in RF propagation) or to replace or recharge batteries (due to operating at higher transmit power) is often difficult or impossible [28]. We seek design techniques to address these challenges while meeting sensing coverage requirements, cost constraints, etc. We believe substantial improvements are possible over techniques like random node placement that do

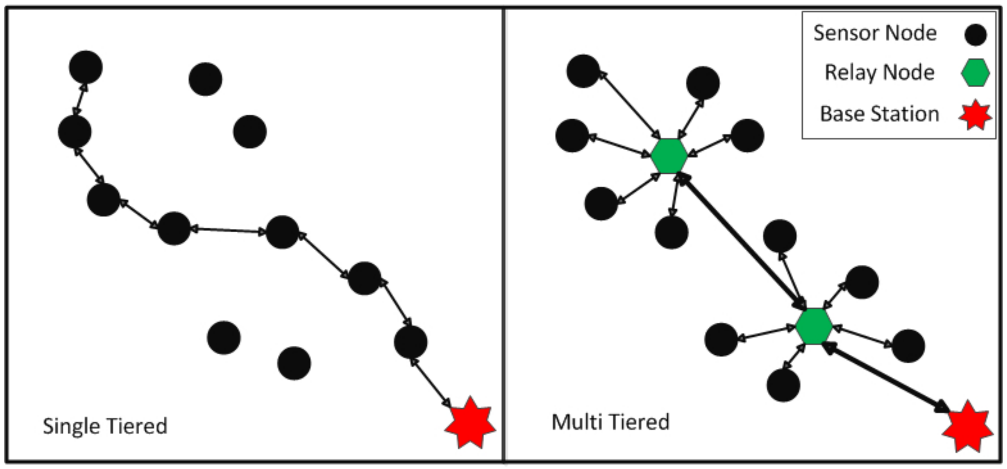


Figure 1.1: Single-tiered network vs Multi-tiered network

not factor in extra knowledge about the area in which a WSN will be placed.

Once the environmental variables of interest have been sensed by the SNs, this data needs to be collected, aggregated and transmitted via a reliable route from the SN to one or more base stations. This is usually done using multi-hop communication from the SNs to the base station. The network architecture can be either single-tiered or multi-tiered. In single-tiered networks, the SNs participate in the process of forwarding data to the base station, while in multi-tiered networks additional nodes called relay nodes (RNs) are needed [28] and SNs do not participate in forwarding the data. These RNs usually have higher maximum transmission power than the SNs which makes them more expensive in terms of cost. While frequently replacing the batteries in many SNs might be difficult, the problem is simpler with the much smaller number of RNs. Including RNs in a network can significantly decrease the frequency of battery changes in SNs, and consequently increase network lifetime in a very effective manner. The difference between a single-tiered and multi-tiered network is illustrated in Figure 1.1.

Although optimal node placement has been proved to be an NP-Hard problem in most conditions [15, 62], various optimal and sub-optimal techniques have been proposed for minimizing the number of nodes needed to create a network while respecting constraints such as connectivity [3], network lifetime [56,58], reliable data transfer [5] and coverage [52] in both single-tiered and multi-tiered networks. These methods often make various simplifying assumptions such as homogeneous transmission ranges among nodes. They also often concentrate on optimizing either the number of SNs [13] or the number of RNs [9] without considering solving the whole problem together.

The objective of this work is to design node placement algorithms for two-tiered heterogeneous networks that aim to solve problems such as minimizing the number of RNs and SNs needed to satisfy a coverage constraint while maintaining connectivity (with fault-less connections) of the SNs to the base station and also maximizing the area covered by specific number of RNs and SNs. We specifically concentrate on static networks. Our main contributions are:

- **Considering heterogeneous signal path loss by modelling the environment**: In this work we consider heterogeneous communications among nodes. This means that a signal that is originated from the same source may experience different amounts of distortion when it reaches points that have a distance r from the source. The amount of distortion depends on the areas the signal propagates through.

- **Deploying both SNs and RNs and satisfying a coverage requirement while maintaining communication between the SNs and the base station in an environment with heterogeneous connections**: Considering the previous work done in this field, our Sensor-Unification-Relay (SUR) algorithm allows users to implement and utilize different SN and RN placement algorithms in an environment that models heterogeneous connections among nodes. Our framework adds an intermediate stage between SN and RN placement algorithms that aims to reduce the faulty links among nodes. This stage transforms a non-uniform field in terms of communication to a uniform field so that SN and RN placement algorithms could perform normally. As the transformation is not perfect, we also measure the quality of each link once the nodes have been deployed and compare our SUR method with situations where the intermediate stage is not used. The SN deployment algorithms should deploy SNs in a way that a specific amount of an given area is covered and the RN placement algorithms must establish connectivity from the SNs to the base station using RNs so that the sensing data could be properly transferred to the base stations.

- **Maximizing the sensing area using a specific number of RNs and SNs**: In the same problem model, we also solve the problem of maximizing the covered sensing area by using a specific number of SNs and RNs. Rather than first placing the SNs, our Max-cover 1-connected algorithm uses a greedy and simulated annealing method to place RNs in locations that maximize the areas SNs can be placed in. We then modify a known SN placement algorithm to fully utilize our available SNs.

1.1 Organization

The rest of the thesis is organized as follows: in Chapter 2 we review the related work in current literature. In Chapter 3, we present the problem model and preliminaries. Then we detail our work about node placement in Chapters 4 through 6. Specifically, Chapter 4 explains our Sensor Unification Relay (SUR) algorithm. In Chapter 5 we describe the Max-cover 1-connected algorithm and in Chapter 6 we present our results and comparisons in detail. We conclude the thesis in Chapter 7.

Chapter 2

Related Work

From the introduction of WSNs, optimizing SN placement has been studied thoroughly [62]. Various methods have been proposed to address different challenges in node placement such as network connectivity [3, 4, 11, 34, 63], network lifetime [55, 58, 59], network coverage [32, 51, 52], and fault tolerance [5, 65, 66], etc. mainly in 2D but also in 3D space [40, 41]. These papers often frame secondary challenges as constraints on the problem. While some of these algorithms concentrate on placing SNs [1, 13, 30], others focus on RN placement [9, 31]. In this section, we review some previous algorithms that optimize the number of SNs and RNs while providing specified network coverage or network connectivity.

2.1 Sensor Node Placement

We categorize static SN placement algorithms based on the problems they aim to solve. The optimization methods for solving these problems include linear programming techniques, greedy algorithms, divide and conquer techniques, AI based algorithms such as simulated annealing and tabu search, etc. The problem that has attracted the most attention in the literature is known as the area coverage problem [16]. Here, the challenge is either maximizing the sensing area covered by a specific number of nodes, or minimizing the number of nodes needed to cover a specific amount or proportion of an area.

One well-known study is that by Dhillon and Chakrabarty [13]. They present a probabilistic optimization framework for placing SNs. They consider a probabilistic detection model for the SNs and consider noise and distance as parameters that can affect the information received by the sensor. Their *Max-Min-Cov* algorithm deploys SNs on predefined grid points and also considers obstacle avoidance. This greedy, iterative algorithm considers a coverage miss probability for each grid square. The miss probability of a cell is the probability that a cell would not be covered by any sensor nodes. In each iteration a SN is deployed on the grid point where it has the maximum impact on minimizing the miss probabilities of the overall grid. After a SN is placed, the miss probabilities of the grid squares are updated. Dhillon and Chakrabarty also discuss similarities between this problem and the *art gallery* [36] problem.

A triangular-grid SN placement architecture for underwater WSNs in both two and three dimensional space was presented by Pompili et al. in [40]. The main objective of this study was to find the minimum number of SNs needed to achieve optimal sensing and communication coverage and this is done using tiling patterns. They propose a triangular grid pattern for the SNs and prove that if the distance between neighbouring nodes is $\sqrt{3} \times$ the sensing range, then an area would be fully covered without any blind spots.

Lin et al. [30] consider the problem of sensor placement for locating targets. They model the sensing field as a grid and propose a simulated annealing min-max optimization algorithm that ensures full coverage in the field. They first place SNs on all grid points and gradually remove unnecessary nodes until a cost constraint is met. In each iteration an attempt is made to remove one unnecessary node, otherwise a node is selected and moved to a random position. The system reaches a cool state once coverage and discrimination requirements are achieved. They compare their solution with the optimal solution in smaller cases and claim that although their algorithm may not always find the best possible result, the solution is the desired solution.

In [64], Zhang et al. address the problem of node placement in situations where the required detection probability thresholds of various locations are different [64]. Similar to the work done by Dhillon and Chakrabarty [13], they also use a probabilistic sensing model for the SNs and model the problem as an integer linear programming problem. They propose an iterative heuristic algorithm (*DIFF-DEPLOY*) for solving this problem. The computational complexity of their algorithm is $O(\frac{4}{3}n^6)$ which is the main drawback for this method.

Aitsaadi et al. [1] propose using a tabu search algorithm to achieve full coverage of a given field. In their work they also consider the differentiated detection probability threshold for different areas and address a probabilistic event detection model for sensor nodes. Geographical characteristics of the monitored events are also considered. They claim that the number of SNs needed to fully cover a field is less than for previous methods, such as the *Max-Min-Cov* algorithm proposed by Dhillon et al. [13] and the *DIFF-DEPLOY* algorithm of Zhang et al [64].

Furthermore, the problem of evaluating coverage in a network was scrutinized by Meguerdichian et al. in [32]. In their study which was on wireless ad-hoc sensor networks, Voronoi diagrams and Delaunay triangulation techniques are used to find spaces not covered between the nodes and compute *maximal breach paths* and *maximal support path*. The *maximal breach path "is a path where its closest distance to any of the sensors is as large as possible"* and the *maximal support path* is defined as *"a path where its farthest distance from the closest sensors is minimized"* [32]. The polynomial time algorithm they present is centralized and is used to determine worst-case (breach) and best-case (support) coverage.

While connectivity is a very important issue in WSNs, the previous studies only discuss the coverage constraint – connectivity among the SNs is not considered. These studies have been extended by including the connectivity constraint. This means that while coverage remains the main objective, the single or k-connectivity constraint among the SNs is satisfied as a side problem. To satisfy the k-connectivity constraint, every SN should be connected to k other SNs in the network.

To achieve both connectivity and coverage, Wang et al. [56] divide a sensing area into subareas. In each area, they deploy SNs row by row such that each row guarantees continuous coverage and connectivity, and so that adjacent rows ensure continuous coverage [56]. Their solution allows arbitrary polygons as the sensing area, with possible existence of obstacles.

The relationship between connectivity and coverage in single-tier networks has been studied in detail. Zhang and Hou [63] use computational geometry to prove that with a radio transmission range of at least twice the sensing range, connectivity is implied by complete coverage of a convex area in the network. In their work they "maintain coverage and connectivity by keeping a minimum number of SNs to operate" and also study the problem of choosing the optimal number of working SNs from a substantially dense network to obtain full coverage.

In [5], various SN placement topologies are discussed by Biagioni and Sasaki. In their work they aim to achieve coverage and also preserve connectivity in case of possible failures or battery depletion of a number of SNs. Various deployment topologies such as circular, hexagonal, star-in-square, triangle and hexagonal grids are studied. They conclude that the deployment depends on the sampling distance and the communication radius of the SNs, which means that each deployment should be customized to its specific location.

Other work includes the analytical study by Kumar et al. [25]. This method covers each point of an area with at least k nodes, and provides analytical bounds on the number of nodes needed to do so. This property is mainly required in intrusion detection and security applications. They also develop the Randomized Independent Scheduling (RIS) algorithm that maximizes the lifetime of the network by using a probabilistic model to control which SNs are functional and which are sleeping.

2.2 Relay Node Placement

RN placement algorithms are categorized into single-tiered and multi-tiered networks. In two-tiered networks only the RNs participate in forwarding data gathered by the SNs to the base station while in single-tiered networks, SNs can also receive and forward data. In both of these problems, the connectivity and fault tolerance of the network depends on the placement of RNs. The main objective is to minimize the number of RNs needed to connect all the SNs to the base station. This problem has also been studied in wireless area networks [44], IEEE 802.16j networks [29] and WiMAX Networks [61]. We assume k is the connectivity level between the RNs and r, R are the radio ranges of SNs and RNs. It has been proven [21] that for $k = 1$ and $r = R$ the RN placement problem is NP hard in single-tiered networks.

While some previous studies have the goal of achieving k-connected networks as a secondary objective [23, 65] ($k \succ 1$), we do not have the same requirement in our work and solve the problem for 1-connected networks. We should also note that efficient RN deployment can increase the size of the areas where it is feasible to place SNs.

Based on the application, different studies use different definitions for lifetime which could be the time when the first node dies, the time when a specific amount of nodes die or the time before loss of coverage happens [10]. Pan et al. [37] were among the first to study the RN placement problem for multi-tiered networks. They intend to maximize the network lifetime of a multi-tiered WSN by deploying base stations and hierarchical clusters with "Application Nodes" functioning as the RNs at specific locations. By maximizing topological lifetime they mean maximizing the time *"from network initialization to a point when the WSN cannot maintain enough ANs alive to continue its given mission"* [37]. Additionally, they provide upper and lower bounds for the maximal topological lifetime.

Tang et al. [49] present an optimization method for networks where the SNs are distributed uniformly. The objective is to deploy the minimum number of RNs so that SNs can connect to 1 or 2 RNs. The RNs are either 1-connected or 2-connected to each other. Using R and r to denote the communication range of the RNs and SNs respectively, they study the case where $R \geq 4r$. Their solution is a 4.5-approximation algorithm for both one-connected and two-connected networks. That is, their solution is guaranteed to be within a factor of 4,5 of the optimal. In their work they mostly concentrate on guaranteeing connectivity and ensuring reliability in case of node failure, rather than increasing the total network lifetime.

Lloyd and Xue [31] present an improved algorithm that only has a constraint of $R \geq r$. Based on whether the SNs can or cannot act as data forwarders, they have modelled two problems algorithms that work for both single-tiered and multi-tiered networks. For the single-tiered problem, the solution is based on a minimum spanning tree and a polynomial time 7-approximation algorithm is proposed. For the two-tiered problem where the RNs should be strongly connected, the algorithm is based on a Steiner tree with minimum number of Steiner points where the Minimum Geometric Disk Cover algorithm is used to connect the SNs to the RNs. The objective of the Minimum Geometric Disk Cover algorithm is to find the minimum number of unit disks needed, whose union covers a number of input points. The algorithm provided by Lloyd and Xue is a polynomial time $(5 + \epsilon)$-approximation algorithm. Furthermore the NP-hardness of the second problem is proved.

Inspired by Lloyd and Xue's work, Zhang et al. [11] develop a polynomial 14-approximation fault-tolerant algorithm that aims to deploy the minimum required number of RNs while providing a two-connectivity network among RNs. The algorithm with base stations has been proved to be a $(20 + \epsilon)$-approximation algorithm.

Srinivas et al. [45] proposed an improved two-stage approximation algorithm with the assumption that $R \geq 2r$. They formulate this as the *Connected Disk Cover* problem in mobile backbone networks. In their study, every regular node should be connected to a backbone node. In the first stage, a strip cover algorithm is used to associate SNs to RNs. In the second phase, they connect the

RNs by creating a Steiner tree and using a minimum number of Steiner points.

In [11], Cheng et al. tackle the problem of relay placement in single-tier networks. Their objective is to deploy additional RNs to maintain global connectivity between every pair of SNs, under the assumption that the SNs can communicate with each other. They also assume the RNs have the same communication range as the SNs ($R = r$). They formulate the problem as an Steiner tree with a minimum number of Steiner points and bounded edge length (*SMT-MSPBEL*). This problem was previously proved to be NP-Hard by Lin and Xue [21].

Furthermore, Chen and Cui [9] recently proposed a polynomial time $(5 + \epsilon)$-approximation algorithm for the RN placement problem in WSNs with a base station. To find the positions of the relay nodes, they first turn the problem into a 1-Geometric Disk Cover problem [20] where each SN needs to be covered by one RN. Then they design an algorithm which is similar to the algorithm used by Cheng et al [11] for solving the *SMT-MSPBEL* problem. The objective is to efficiently connect the RNs deployed at the previous stage (covering SNs) to the base station using the minimum number of RNs. The difference between Chen and Cui's work and Lloyd and Xue's work is that the former considers base stations in the deployment.

Also Misra et al. [34] study two problems in the category of constrained RN placement in single-tiered networks where RNs can only be placed at a set of candidate locations. They propose polynomial time approximation algorithms for solving the problem of connecting each sensor to the BS through a bi-directional path(connectivity) and the problem of connecting each SN to at least 2 base stations (survivability) by modelling the problems into a Steiner Tree Problem with Minimum Steiner Points (STP-MSP).

The aforementioned studies assume homogeneous connections between nodes, and identical transmission radii of nodes. For more realistic networks, Han et al. [19] address the problem of "deploying RNs to provide fault-tolerance with higher network connectivity in heterogeneous wireless sensor networks". They assume a transmission range of T_{relay} for relay nodes and a range $[T_{min}, T_{max}]$ for sensor nodes. They model this asymmetric link communications using one-way and two-way link models. Their main result is an $O(\alpha k^3)$-approximation algorithm for full-tolerance networks under the constraints that $R \geq r$ and $k \geq 2$. However they still assume a disk model for radio ranges which is usually not true in real networks.

Many of the studies discussed above are analytical and the authors have not implemented their methods in real experiments or simulation experiments. This means that detailed parameters such as the communication quality among nodes in different situations or the network lifetime of the proposed topologies has not been well studied in these works. Also a complete framework that can address different problems has not been proposed. None of the works above assume different radio ranges for both SNs and RNs. Assuming different radio ranges for RNs and SNs between different points in the map makes the problem significantly more difficult but also more realistic. In our framework a SN or RN may have a different radio range in different environments. Furthermore we aim to further increase the efficiency of the nodes by placing them together rather than just considering RN placement or SN placement. We also measure how the communications between nodes would be in our modelled environment.

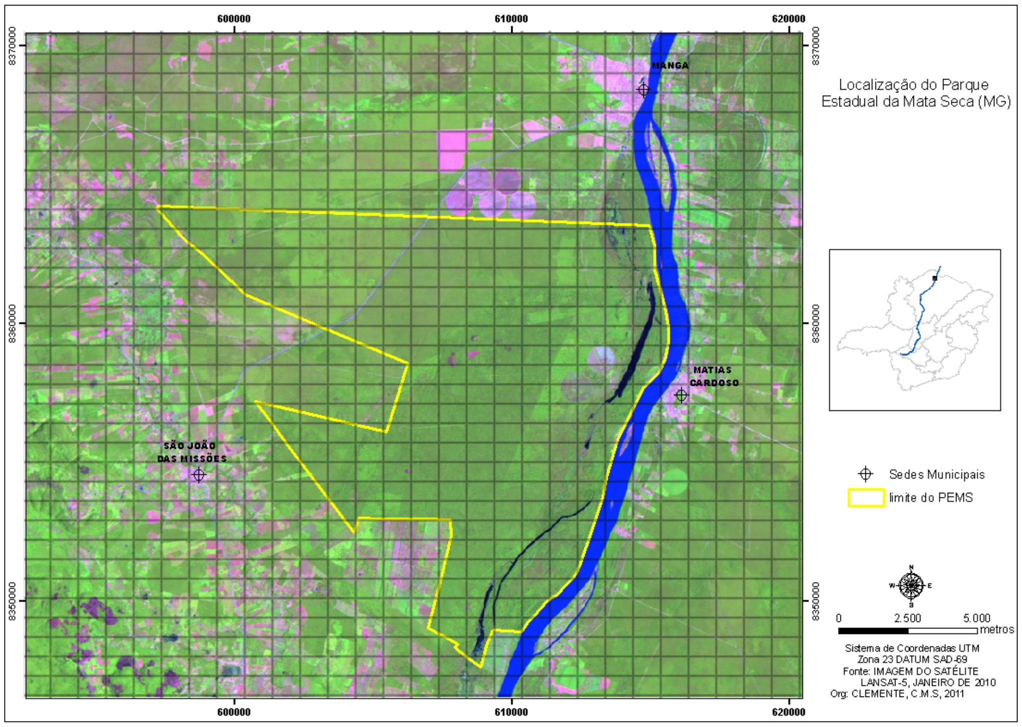
Chapter 3

Problem Model

None of the studies discussed in the previous section consider the problem of SN placement or RN placement in areas with non-uniform communications for both SNs and RNs. However, many studies have scrutinized the RF communication path loss in different environments such as forests [7, 33,42] and found highly variable and heterogeneous conditions. In [7] the receiving signal is affected by trees and vegetation by adding additional attenuation and also indirectly scattering the signal, in [14], the path loss of links in a Brazilian rain forest was measured and modelled. However even these models do not address the problem of signal propagation through different terrains with different characteristics. In our work, we model the spatially-varying RF link conditions in the area of interest. However more comprehensive models with real measurements can be used to increase the accuracy of our model.

For this purpose, we use a grid to subdivide the map of the area of interest, and then we attribute the parameter α as the path loss exponent for each cell. Using maps derived from remote sensing, the number of trees and vegetation density of the cell can be used to better select the α value for each cell. More accurate values for the path loss exponent result in better modelling of the area and a more realistic version of the map. However for our current experiments, we use self-generated values and maps and using real data is proposed as future work. A sample grid map of a tropical forest area in Brazil is depicted in Figure 3.1. Similar models that divide an area based on environmental characteristics have previously been proposed in [43] where a region in Australia with different vegetation characteristics was divided into 2000 zones.

In general, increasing the number of the cells increases the accuracy of the algorithm. However splitting the map into small areas may be a difficult task and it also requires more computation and may not be feasible everywhere. In our model the number of cells depends on the communication range of the RNs and the SNs. Once an RN is deployed, it should at least cover its neighbouring cell.



600000
610000
620000
8370000
8360000
8350000
MANGA
MATIAS CARDOSO
SÃO JOÃO DAS MISSÕES
Localização do Parque Estadual da Mata Seca (MG)
Sedes Municipais
limite do PEMS
0
2.500
5.000
metros
Sistema de Coordenadas UTM
Zona 23 DATUM SAD-69
Fonte: IMAGEM DO SATÉLITE LANSAT-5, JANEIRO DE 2010
Org: CLEMENTE, C.M.S, 2011

Figure 3.1: Sample map with grid

3.0.1 Signal Propagation Model

Using different path loss exponents for different terrains has been previously used in models such as Lee et. al's [26] path loss prediction method for flat terrain and Grimlund and Gudmundson's [18] empirical street corner path loss model [46]. Inspired by these models we have changed the free space signal propagation model [38] to consider the distortion of the transmitted signal as it propagates through cells with varying values of α. Although this model is very simple and imperfect, we can model how a signal approximately behaves in different terrains. Generally, RF signal strength in free space is modelled as:

$$P_r(d) = P_t \times G_1 G_2 (\frac{\lambda}{4\pi d})^2 \tag{3.1}$$

where P_t is the transmitted power in mW, P_r is the received signal strength in mW in d is the distance from the sensor to the point being considered in meters, $G = G_1 \times G_2$ is the antenna gain and λ is the wavelength in meters. We set $K = G \times (\frac{\lambda}{4\pi})^2$ and use $G = 1$ in our experiments.

We use the idea presented by Grimlund and Gudmundson's [18] path loss model and include a path loss exponent α in the free space loss equation so that we can use it in other environments. Their formula is as follows:

$$\Omega_p = \frac{k\Omega_t}{d_c^a(1 + d_c/g)^b} \tag{3.2}$$

where Ω_t is the transmitted power in mW, d_c is the distance from the sensor to the point being considered, $g = 150$ meters, and $a = b =$ the value of the path loss exponent in the cell. Ω_p is the received power, again in mW.

Crossing into the next cell:

$$\Omega_p = \frac{k\Omega_t}{d_c^a(1 + d_c/g)^{b_0}} \frac{1}{(d - d_c)^a(1 + (d - d_c)/g)^{b_1})} \tag{3.3}$$

Although their model is designed to model the signal strength as it changes around corners, our model is influenced by the general ideas of using a path loss exponent and modelling the behaviour of the signal when entering a new cell. We assume the path loss exponent has a value between 2 and 3 (2 for free space and 3 for cluttered outdoor spaces or indoor communication).

The loss caused by the vegetation that is in addition to the free space loss is referred to as "excess loss". To account for the excess loss, we create a map of the area, divide it into a uniform grid, and allocate a path loss exponent to each cell in the grid. To calculate the excess loss, we need to know how much power the signal has when it is leaving a cell. When the signal is crossing into the next cell its power is calculated as:

$$P_r(d) = K \times \frac{P_t}{d_1^{\alpha_1}} \times \frac{d_1^{\alpha_2}}{d_2^{\alpha_2}} \tag{3.4}$$

where d_1 is the distance from the sensor to the edge of its cell and d_2 is the distance the signal travels in the second cell. α_1 is the value of the path loss exponent in the first cell. α_2 is the value of the path loss exponent in the second cell.

This formula allows us to calculate the amount of power that is lost within the second cell. Assuming the signal only has to propagate from the first cell into a second cell with a path loss exponent of α_2, we divide the signal strength at the edge of the first cell by $d_2^{\alpha_2}$ and then multiply the result by $(d - d_2)^{\alpha_2}$ to account for the loss over the distance the signal propagates in the second cell. That is, $d_1^{\alpha_2}/d_2^{\alpha_2}$ represents the power that is lost in propagating through the second cell.

To chain together segments through multiple cells, we include more terms of the form of the second term above as product terms, and use the distance covered in each cell, d_j. α_i is the path loss coefficient in cell i. This gives us:

$$P_r(d) = K \times \frac{P_t}{d_1^{\alpha_1}} \prod_{i=2}^{k} \frac{(\sum_{j=1}^{i-1} d_j)^{\alpha_i}}{(\sum_{j=1}^{i} d_j)^{\alpha_i}} \tag{3.5}$$

for the overall path loss model from the cell in which the transmitter is located, followed by segments across $(k - 1)$ more cells. We name $\lambda(a, b)$ the communication distance function between points a and b and which is defined as:

$$\lambda(a, b) = \frac{1}{d_1^{\alpha_1}} \prod_{i=2}^{k} \frac{(\sum_{j=1}^{i-1} d_j)^{\alpha_i}}{(\sum_{j=1}^{i} d_j)^{\alpha_i}} \mathit{where} \sum d_j = distance(a, b) \tag{3.6}$$

For example in Figure 3.2, node A transmits a signal with a power of $100mW$ and frequency of $2.4 \times 10^9 Hz$ towards B. The received signal's power when it reaches B is calculated as:

$$\begin{aligned}
&100 \times 10^-3 \times (\frac{0.0125}{4\pi})^2 \times (1/44^{2.7}) \\
&\times (44^{2.2}/95^{2.2}) \times (95^{2.6}/168^{2.6}) \\
&\times (168^{2.3}/228^{2.3}) = 7.48 \times 10^{-12} W \\
&\simeq -81.2dBm
\end{aligned}$$

Using this path loss model, we are able to model the communication behaviour in various environments such as tropical forests and flat terrains by dividing the field into cells with different path loss exponents by a grid. We name this grid θ. In this project we assume the SNs can only send data to the RNs and SNs cannot participate in the process of data forwarding, and we define our communications as follows:

- **Relay to relay communications**: If we consider a receiving sensitivity threshold (P_r) for RN B, then node A could connect to node B if the signal power of the signal received from node A at node B is at least (P_r). If both A and B can communicate with each other under the aforementioned condition, then A and B are *connected*.
- **Sensor to relay communications**: Similarly the SN S is connected to RN A if the signal originated at S has a minimum power of P_r when it reaches A.

3.1 Simulated Hardware

For the communications among nodes, we model the transmit power and receiving sensitivity of the Xbee® [22] 802.15.4 radios. For the SNs we model the XBee 802.15.4 radios which have a transmit power of $10dBm$ at a frequency of $2.4 \times 10^9 Hz$ (which is $0.125m$ in wavelength) and receiving sensitivity of $-70dBm$. For the RNs, we model the XBee-PRO model 802.15.4 radios that have a transmit power of around $20dBm$ at a frequency of $2.4 \times 10^9 Hz$ and receiving sensitivity of $-70dBm$.

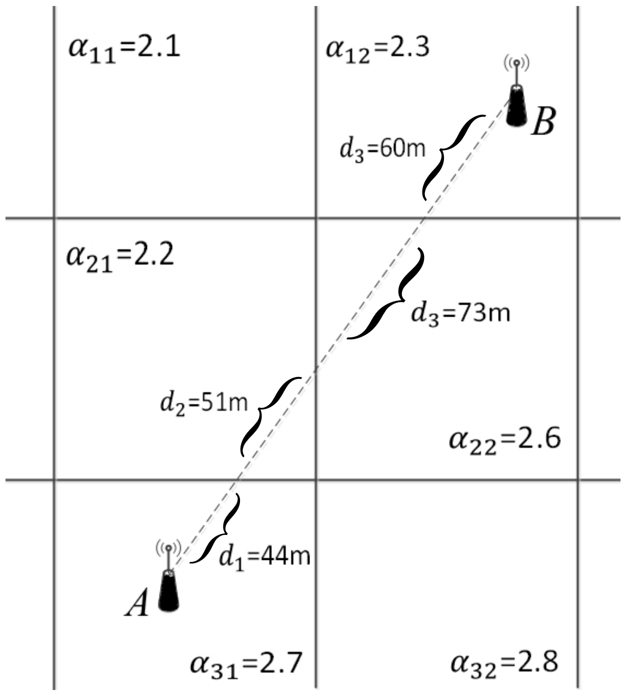


Figure 3.2: Calculating the communication distance function between two points.

Chapter 4

Sensor-Unification-Relay (SUR) Algorithm

Based on the problem and communication models described above, we have designed algorithms that place both SNs and RNs in a multi-tiered WSN. Our purpose is to effectively and efficiently place SNs and RNs so that coverage requirements and connectivity constraints are satisfied in a modelled real environment using a minimum number of nodes. We propose a framework that connects SN placement and RN placement methods and allows these algorithms to run in near real environments by transforming a non-uniform map (in terms of communication) to a uniform area. The effect of this transformation would be apparent in the number of RNs deployed and the quality of links between nodes. We use these two metrics as our main measurements. Hence in general our method for transforming a non-uniform map to a uniform map improves previous RN placement algorithms in terms of communication accuracy in real areas. The whole framework however is designed to increase efficiency and accuracy of both RN placement and SN placement algorithms. Our SUR algorithm is composed of three stages.

- **Sensor node placement**: In the first step, we place SNs to satisfy coverage requirements which is given as an input to our system (for example to sense 70 % of a given area). In future work this can be extended to implement any coverage requirement such as target tracking or maximising coverage using a specific number of SNs. Any sensor deployment can be used in the next steps and the whole system does not depend on this stage.

- **Field unification**: Once the sensors are placed (with any algorithm) then we transform the area such that communications are more uniform than before in all cells of the new region. By uniform connections in an area we mean two main properties: first, any transmitted signal from a source point s will have the same strength when it reaches a point x at distance d from the original source and second, the amount of signal distortion between two points in the original map is equal to signal distortion of the transformed two points in the uniform area. Although our algorithm does not perfectly satisfy these requirements, our deployments have much better performance in this regard. We should note that this part of the algorithm mostly affects the performance of the RN placement algorithms we describe next.

- **Relay node placement**: When the transformation is complete, the locations of the SNs are also transformed. Using these new positions, we deploy the minimum number of RNs required in such a way that a route exists between each SN to the base station possibly through a set of RNs. Once the RN placement is finished on the transformed area, we use a reverse transformation to get the coordinates of the RNs on the real area and the algorithm is complete.

4.1 Mathematical Model

In this section, we formulate the problem model as a combinatorial optimization problem.

Given Parameters:

$R = r_1, r_2, ..., r_n$:	Set of the Relay Nodes.
$S = s_1, s_2, ..., s_m$:	Set of the Sensor Nodes.
$B = b_1, b_2, ..., b_m$:	Set of the Base Stations.
π =	$a_1, a_2, ..., a_{m+n}$: Set of candidate node locations
c_r :	Cost of each RN.
c_s :	Cost of each SN.

Decision Parameters:

$\lambda(a, b)$ = Communication distance between point a and point b. $a, b \in \pi$

$$\mathrm{cc}(\pi) = \begin{cases} 1 & \text{if the coverage constraints are satisfied.} \\ 0 & \text{if the coverage constraints are not satisfied.} \end{cases}.$$

$$A(\pi, R, S) = \begin{cases} A & \text{if the area covered by } S \text{ is } A \text{ and } S \text{ is connected to } B \text{ through } R \\ 0 & \text{if } cc(\pi) = 0 \text{ or there is a } s_i \text{ that is not connected to a } b_j \end{cases}.$$

Objective Function:

$$\min_{c_r m + c_s n} \max_{\pi} \quad A(\pi, R, S).$$

such that $A > C_t$ the coverage threshold, which is given as input

In the next sections we explain in detail each of the steps of our algorithm.

4.2 Sensor Node Placement

The first step in SUR is to deploy the SNs over the field. In this stage the main objective is to satisfy a sensor coverage constraint using an algorithm that minimizes the number of SNs needed. As an

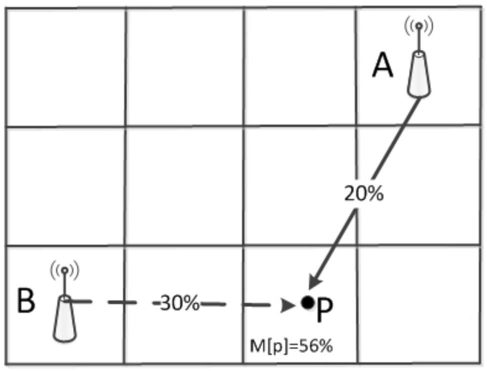


Figure 4.1: Updating the miss probability of a sample point in the MAX_AVG_COV algorithm

example, we use the *MAX_AVG_COV* algorithm suggested by Dhillon et al. [13].

In this algorithm, first the area is divided into a grid μ. With this grid (which is different than the communication grid), the problem space is decreased because only candidate positions can be selected as sensor locations and also the issue of preferential coverage of grid points or targets is modelled. In general, the number of cells in the grid μ depends on the desired accuracy the SN placement would achieve. If the number of the cells n is increased, then the accuracy of the algorithm increases. However the algorithm requires a memory space of N^4 (where n and m are number of the cells on each edge and $N = m \times n$) so m and n cannot get too big. In our work we determine the number of cells by estimating the average sensing range of the SNs. If the estimated sensing range is S_R then the side of each cell will have a maximum size of $a = \sqrt{2} \times S_R$ so that we can guarantee coverage of a cell by a SN placed in the middle of the cell.

SN placement is guided by the "miss probability". This is the probability that a cell is not covered by a given SN. In each iteration i, the algorithm places a node a_i on every possible empty cell and measures the effect of that choice in terms of the miss probability on all of the other cells. The choice with the maximum impact (least accumulated miss probability on all other SNs) is selected, a_i is deployed there, and the miss probability matrix is updated for the next round. The original algorithm continues until a specified number of SNs are deployed or sufficient coverage of the grid is achieved. A miss probability threshold (M_{min}) is defined for each cell. Once the miss probability threshold of each grid point has been reached, the algorithm stops. We modify the algorithm to continue only until a specified percentage coverage of the whole area is reached. The miss probability threshold (M_{min}) is given to the algorithm as an input. For example in Figure 4.1 we first deploy the SN A the field. A senses cell P with a probability of 20%. This updates the miss probability of cell P to 80%. Now if another SN B is deployed and it senses P with a probability of 30%, then overall the miss probability of cell P is updated to $(1 - 0.2) \times (1 - 0.3) = 0.56$.

To model the miss probability matrix, while some algorithms use the binary sensor model (also known as the disk coverage model), these models are not accurate for real environments because the outcome of the sensor readings is not always a binary yes/no and "there is inherent uncertainty

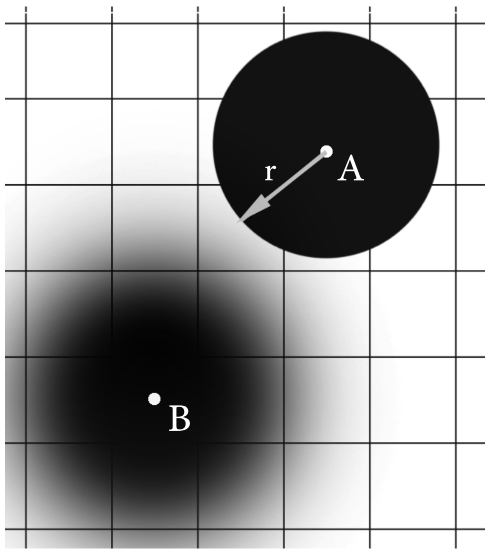


Figure 4.2: Disk coverage model (A) vs. attenuated disk model (B)

associated with sensor readings, hence sensor detections must be modelled probabilistically" [13].

The algorithm we use requires a probabilistic sensor detection model to determine how well a sensor covers a cell. We use the attenuated disk model described in [53] as our probabilistic function. The difference between the attenuated disk model and the disk coverage model is depicted in Figure 4.2. Our probabilistic function that is used to fill the miss probability matrix is:

$$p_{ij} = e^{-\beta \times d} \tag{4.1}$$

where p_{ij} is the probability that a sensor placed in cell i would sense something in cell j. Here, d is the distance between i and j and β is based on the sensing range of the sensors and the unit size of the cells.

As illustrated in Figure 4.3 the algorithm works as follows. The miss probability matrix $E = [e_i]$ defines the probability that a cell is not covered by other SNs. The size of this matrix is the total number of grid points, $n \times m$. Considering E, a distance matrix D is created that has n^2 rows and m^2 columns. D represents the sensing relation of every two cells of the map. In each step a SN is placed and its effect on the miss probability of all other SNs is calculated using D; the results are saved in M_{ik}. Once a node is selected, E is updated and the process continues until a specified proportion (e.g. 80%) of the area is covered. This amount is given to the algorithm as an input. As stated earlier, by coverage we mean that the accumulated miss probabilities of a cell should be less than M_{min}.

We should note that this algorithm does not consider connectivity and only aims to increase coverage. Providing connectivity to some nodes that are far out of the reach of the base station might be difficult and inefficient. One important favourable point about this algorithm is that avoiding obstacles has been considered, and cells that have obstacles between them cannot sense each other.

```
1: $N \leftarrow m \times n$                                ▷ $m$ and $n$ are the number of columns and rows
2: $M \leftarrow$ miss probability matrix
3: **repeat**
4:     $num_sensors \leftarrow 1$
5:     **for** $i$ = 1 to $N$ **do**
6:         $P \leftarrow D_i1 + D_i2 + ... + D_iN$              ▷ calculate impact of a SN on location $i$
7:         **if** $i$ has maximum impact **then**
8:             $k \leftarrow i$
9:     place a SN on $k$
10:    **for** $i$ = 1 to $N$ **do**
11:        $M_{ik} = 1 - D_{ik}$
12:        $E_i = E_i \times M_{ik}$            ▷ update miss probabilities of each cell due to new SN on $k$
13: **until** $E_i < M_min$ for all $i$ $1 < i < N$
```

Figure 4.3: Pseudocode of the MAX_AVG_COV algorithm presented in [13]

The computational complexity of this algorithm is $O(kN)$, where k is the specified number of sensors to be deployed and N the number of cells. The memory complexity of the algorithm is $O(N^4)$ because the distance matrix has $N^2 \times N^2$ columns and rows.

4.2.1 Area transformation

As explained previously, real-world environments are not necessarily uniform in terms of communication. Consequently points within a range r from a transmitter will not necessarily have the same received signal power. These conditions occur where vegetation, buildings and landform variations affect the signal. This means models and algorithms that address heterogeneous communications are required to design WSNs for real environments. We recall the signal loss model, Eq. 3.4. Assuming that the distance between points p and q is d, as mentioned before the communication distance function $\lambda(p, q)$ is defined as:

$$\lambda(p,q) = \frac{1}{d_1^{\alpha_1}} \prod_{i=2}^{k} \frac{(\sum_{j=1}^{i-1} d_j)^{\alpha_i}}{(\sum_{j=1}^{i} d_j)^{\alpha_i}} \tag{4.2}$$

However if we can transform a non-uniform field into a near uniform field, by properly stretching and shrinking areas that have different path loss exponents, then based on the communication distance function, the algorithms that assume homogeneous radio ranges be utilized by implementing them in the transformed area. A uniform area which these algorithms can use should have the following property:

$$\forall a, a' \in T^2, b \in T^2, d(a,b) = d(a',b) \Rightarrow \lambda(a,b) = \lambda(a',b') \tag{4.3}$$

Where $d(a, b)$ is the Euclidean distance of a, b. To reach this objective, we need an injective function that maps the locations of points in the non-uniform field into new points on a uniform field

T^2. Based on our assumptions regarding communications between nodes, the following property should also be satisfied:

$$\forall P, Q \in S^2, f_{S \to T} \to \lambda(P, Q) = \lambda(f(P), f(Q)) \tag{4.4}$$

where S^2 is our initial set of points. This means that the communication distance between two points in the non-uniform field should be equal to the communication distance between the transformed locations of the same two points in the uniform field.

Inspired by the work by Leikien et al. [27] we develop a method to transform a non-uniform field into a uniform field using cartograms. Cartograms [12] are "transformations that that map non-uniform metrics to a near Euclidean metric" [27]. These transformations rescale a region or area based on a given attribute (usually geographic) and also keep some recognizability of the original map [6]. Cartograms are commonly used to illustrate statistical information in geographical maps such as population density, disease spread and election results because using conventional maps with colours can be misleading [17]. Furthermore, other cartograms such as isochronic cartograms "that distorts the cityscape using travel times originated at the observer's location" [35] are used to create travel time maps. Cartograms deal with challenges such as map recognizability, selecting boundaries and minimizing the distortion of shapes [17]. An example of the German population cartogram [57] is depicted in Figure 4.4.

The main difference between the work currently in the literature and our work is that the transformation function $r \to T(r)$ in our work is not a simple function such as the density (population) function used by Gastner and Newman [17]. In their work, the transformed field should have an even density of population in all of the regions. This translates to:

$$\frac{\delta(T_x, T_y)}{\delta(x, y)} \equiv \frac{\delta T_x}{\delta x}\frac{\delta T_y}{\delta y} - \frac{\delta T_x}{\delta y}\frac{\delta T_y}{\delta x} = \frac{\rho(r)}{\overline{\rho}}$$

where $\overline{\rho}$ is the average population density of the map. However, our transformation must be based on the communication distance function. In the desired transformed area, all of the cells should have the same path loss exponent, α. As a result, unlike the density function of the population cartograms, the relocation of the points from our initial map cannot be achieved in 2D Euclidean space, R^2.

This is due to the nature of our communication distance function. Consider the equilateral triangles $\triangle lmn$ and $\triangle mno$ with sides of length d in an initial field S^2 all in one cell (with the same path loss exponent) as depicted in Figure 4.5. Based on Eqs. 4.3 and 4.4 the resulting transformed points in T^2 should also form an equilateral triangle:

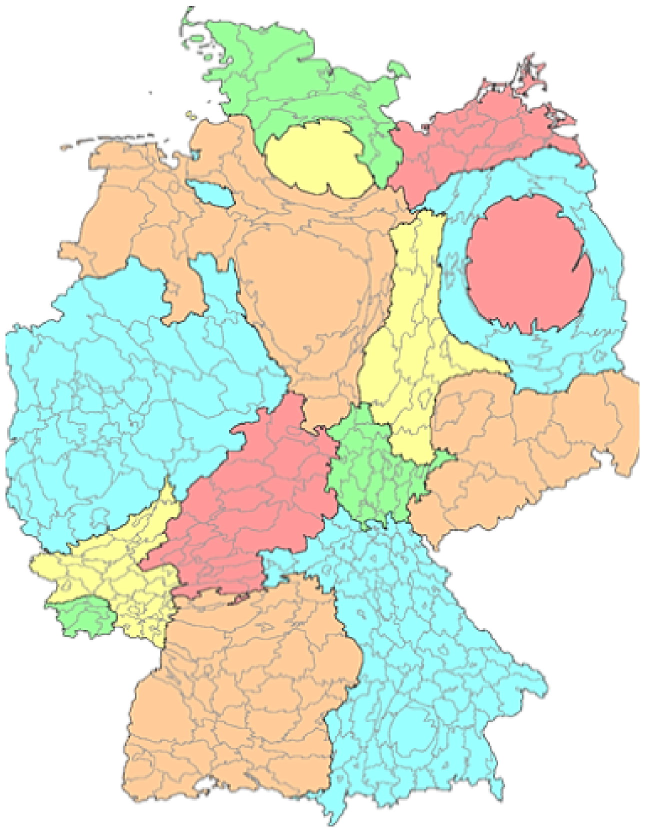


Figure 4.4: German population cartogram (from [57])

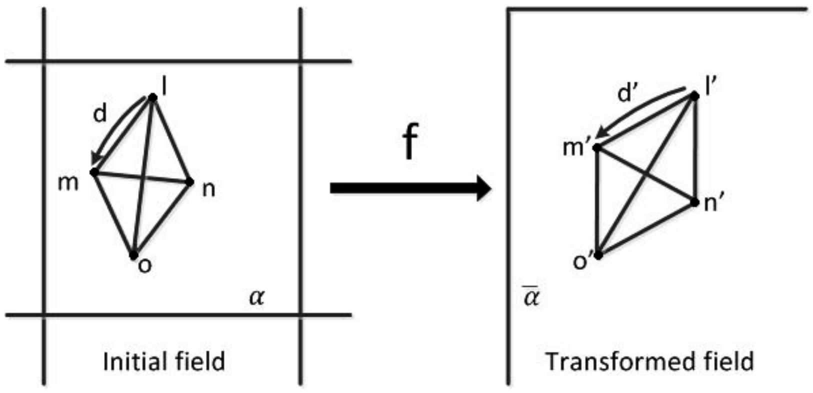


Figure 4.5: Transforming equilateral triangles

$$\begin{aligned}
&\| m-n \| = \| n-l \| = \| l-m \| = d \\
&\Rightarrow \lambda(m,n) = \lambda(n,l) = \lambda(l,m) \\
&f: S^2 \to T^2, f(l) = l' \\
&f(m) = m', f(n) = n', f(o) = o' \\
&\Rightarrow \lambda(m',n') = \lambda(n',l') = \lambda(l',m') \Rightarrow \\
&\| m'-n' \| = \| n'-l' \| = \| l'-m' \| = d'
\end{aligned}$$

Now if we calculate the communication distance function for l, o we will get:

$$\lambda(l,o) = \frac{1}{(\sqrt{3}d)^{\alpha}}$$

In a transformed area where we have a new α for the whole area (which we name $\overline{\alpha}$) this distance is:

$$\lambda(l',o') = \frac{1}{(\sqrt{3}d')^{\overline{\alpha}}}$$

Considering Eq. 4.4, $\lambda(l,o)$ should be equal to $\lambda(l',o')$ so:

$$\begin{aligned}
&\lambda(l',o') = \lambda(l,o) \\
&\Rightarrow \frac{1}{(\sqrt{3}d')^{\overline{\alpha}}} = \frac{1}{(\sqrt{3}d)^{\alpha}} \\
&\Rightarrow (d')^{\overline{\alpha}} = d^{\alpha}
\end{aligned}$$

Obviously, this condition is not satisfied everywhere, so we conclude that a uniform area satisfying Eqs. 4.3 and 4.4 cannot be achieved with an injective and surjective function in R^2. However, we use the idea of cartograms to move points using transformations and develop a method to create uniform areas that have properties close to these constraints. These transformations are not perfect and the resulting points will have errors. However, they will get us close to the area we desire as they will decrease the communication errors in RN placement. We discuss the effects of these transformations in the following sections.

4.3 Area Transformation

We call our transformation algorithm "Rectangular Unification". In brief, this algorithm is based on a Divide and Conquer approach and combines neighbouring cells to produce new cells with new path loss exponents, α', until a single rectangle with a single path loss exponent is achieved. In the process of combining neighbouring cells, points are moved in such a way that the communication

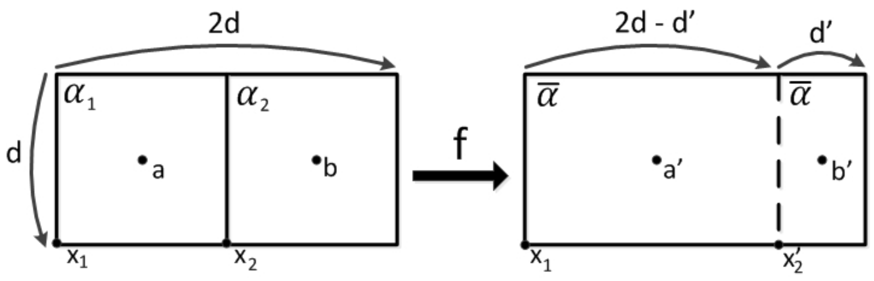


Figure 4.6: Combining two grid squares with two different path loss exponents and creating a new grid square. $\alpha_2 < \alpha)_1$

function distance function λ would be better satisfied in the new area. Currently this method only takes rectangular maps as input. We plan to extend this method for other shapes in future work.

As illustrated in Figure 4.6 our algorithm spreads areas that have higher path loss exponents values and squeezes zones that have a lower α. This is because we select a new path loss exponent α for each two cells we are combining so if we want a signal to lose the same amount of energy that it used to while propagating in the two cells, based on the communication distance function λ we should also change d the distance the signals are travelling through the cells. For the cell that had a higher path loss exponent, the new α is decreased so the distance that the signal is travelling through should increase and points should spread. Similarly, the cell that had a lower path loss exponent would shrink.

As depicted in Figure 4.6, to combine two grid squares, first we need to find x_2'. This point represents and creates a new virtual border between the two previous cells and has the following property:

$$\frac{1}{(d')^{\alpha_2}} = \frac{1}{(2d-d')^{\alpha_1}}$$

Now to perform a transformation between two cells we first need to calculate d':

$$\begin{aligned}
&(d')^{\alpha_2} = (2d-d')^{\alpha_1} \\
&\Rightarrow \alpha_2 \times \ln(d') = \alpha_1 \times \ln(2d-d') \\
&\Rightarrow \frac{\ln(d')}{\ln(2d-d')} - \frac{\alpha_1}{\alpha_2} = 0
\end{aligned}$$

Now assuming $d > 1$, this function above, is a strictly monotone function if $0 < d' < 2d$. We calculate the derivative function to prove so:

$$\begin{aligned}
f(x) &\Longrightarrow \frac{\ln(x}{\ln(2d-x)} - \frac{\alpha_1}{\alpha_2} = 0 \\
\Rightarrow \frac{\delta f}{\delta x} &= \frac{\frac{\log(2d-x)}{x} + \frac{\log(x}{2d-x)}}{\log^2(2d-x)} \\
&= -\frac{x\log(x) - x\log(2d-x) + 2d\log(2d-x)}{x^2\log^2(2d-x) - 2d\log^2(2d-x)x}
\end{aligned}$$

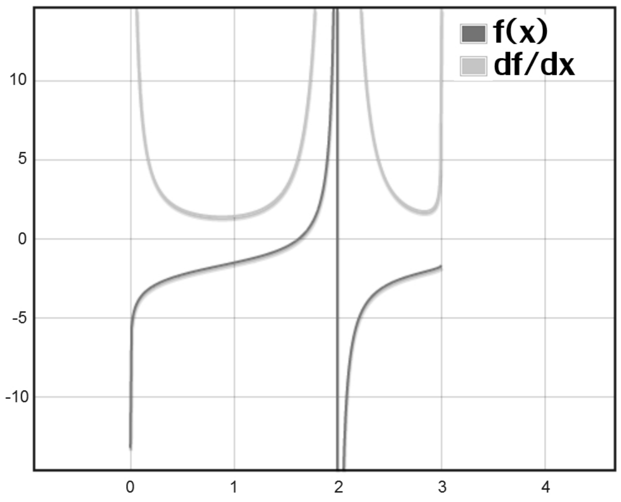


Figure 4.7: $f(x)$ and $\frac{\delta f}{\delta x}$ when $d = 1.5$

As depicted in Figure 4.7, the derivative always has a positive value for $d = 1.5$.

We use the Gauss-Newton algorithm to obtain d' as $f(x)$ has a root between $0 < x < 2d$ and $alpha_1$ and $alpha_2$ have positive values.

Once d^{prime} is calculated, we shift all the points in a cell using the following linear formulas. For points in the left-hand cell:

$$x' = x_1 + \frac{(2d - d') \times (x - x_1)}{d}$$

and for points in the right-hand cell:

$$x' = x'_2 + \frac{d' \times (x - x_2)}{d}$$

Now that all the points in the cells have been rearranged, a new path loss exponent is required to be computed for the combined cell. We suggest various methods from the following list:

- **SUR-Average**: In this method the new path loss exponent α_n is calculated as: $\alpha_n = \frac{\alpha_1+\alpha_2}{2}$ where α_1, α_2 are the path loss exponent of the two cells that are to be combined.
- **SUR-Sqrt**: the new path loss exponent α_n is calculated as: $\alpha_n = \sqrt{(\alpha_1)^2 + (\alpha_2)^2}$.
- **1WayPathLoss**: Based on our path loss model, in this method, the new α is calculated in a way that the amount of energy that a signal loses while going through the two cells (in one direction) would be the same with the new α and is calculated as follows: $\alpha_n = \alpha_2 * log(d1 + d2) - (\alpha_2 - \alpha_1) * log(d1))/log(d1 + d2)$; where $d1, d2$ set the boundaries of the new two virtual cells.
- **2WayPathLoss**: This model calculates the amount of energy that a signal loses in both directions (rather than one direction) and considers the maximum. Then it does the same cal-

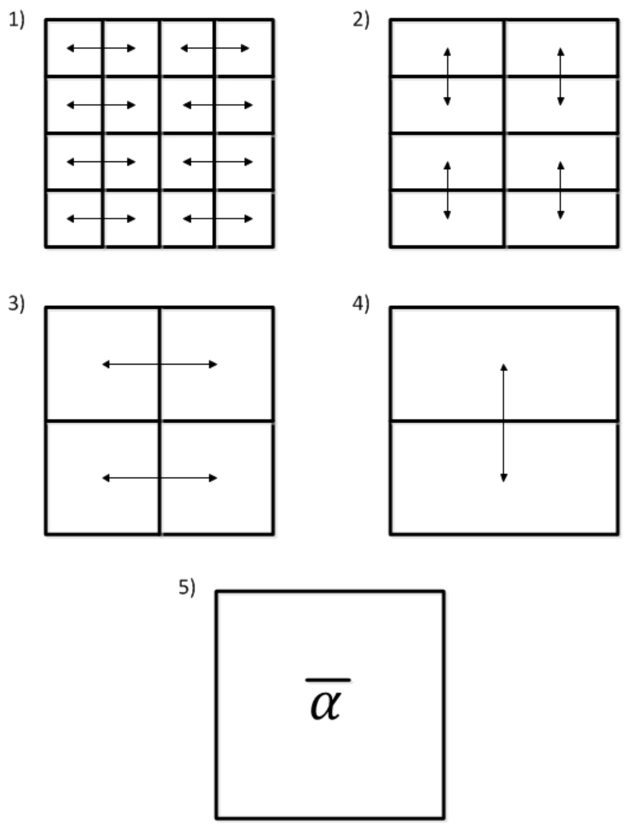


Figure 4.8: Transforming an area

culations similar to the SUR-1WayPathLoss model. This means that the formula is: $\alpha_n = Max(\alpha_2, \alpha_1) * log(d1 + d2) - |(\alpha_2 - \alpha_1)| * log(d1(ord2if\alpha_1 < \alpha_2)))/log(d1 + d2)$;

Neighbouring cells are combined recursively in both dimensions until a single cell is obtained. Figure 4.8 illustrates the process in a 16×16 area.

4.4 Analysis

This algorithm estimates a single path loss exponent, $\overline{\alpha}$, for the whole field, and also relocates points so that the area becomes more uniform relative to the communication distance function. Having a unique path loss exponent means that Eq. 4.3 is automatically true in all of the area. In our algorithm, points in cells that have a higher path loss exponent than $\overline{\alpha}$ stretch and diffuse so that Eq. 4.4 can be better satisfied. Alternatively points in cells that have a lower path loss exponent will compress. These alterations approximately satisfy the second constraint, Eq 4.4. The points that used to be in the same cell, and particularly points that were near the borders of the cell, are most inaccurate as in our method the edges of the cells are kept constant so points near edges can not fully move to desired locations. Despite this inaccuracy, we should emphasize that satisfying Eq. 4.3 is essential in most of the current RN placement algorithms. This is what Rectangular Unification does. Without this constraint, these previous algorithms will not work properly.

When the transformation of the area is complete, we will also obtain the new coordinates of the SNs. Having these coordinates in hand, we place the RNs to connect the SNs we deployed in the first stage. Once the RN deployment is complete, we need to do a reverse transformation to retrieve the original points and the locations where the SNs and the RNs actually need to be deployed. For this reason, we keep the transformations, and as these conversions are injective and surjective, a reverse transformation can be performed.

4.5 Relay Node Placement

The algorithm we use for RN placement is based on the method proposed by Chen and Cui [9]. The objective is to deploy a minimum number of RNs to connect the SNs to the base stations. It is a $(5 + \epsilon)$ polynomial optimization algorithm for two-tiered wireless sensor networks with base stations. We run this algorithm on the transformed area, and once the locations of the RNs have been determined, we do a reverse transformation to find the original coordinates of these points.

The algorithm needs to have two main properties. First, all of the SNs must be able to connect to at least one RN. Second, the RNs should be able to connect to each other and the base stations. We note that the only possible communications in our network are on the links between the RNs and SNs, and on the links between the RNs. SNs cannot participate in forwarding the data.

The algorithm proposed by Chen and Cui [9] solves the problem where every SN is covered by at least one RN, and the network of RNs is 1-connected. This is called 1-coverage, 1-connected RN deployment. This algorithm can be described in two steps. First, an approximation algorithm for RN placement in a single-tiered network with base stations is presented. Using the ideas from this first algorithm, the algorithm to solve 1-coverage, 1-connected RN deployment in multi-tiered networks is explained .

4.5.1 Part 1: Approximate RN placement in a single-tiered network

Inspired by an algorithm proposed by Cheng et al. [11], Chen and Cui presented a solution for RN deployment in a single-tiered network with base stations when $R = r$. Cheng et al. define the problem as follows: Given a set of duty sensors (required sensors) in the plane, place a minimum number of relay sensors to maintain global connectivity such that the transmission range of each sensor is at most R, where R is a constant.

Note that $R = r$ turns the problem into adding nodes to a single-tiered network to establish connectivity between all the SNs and the base station. While Cheng et al. solve the problem by constructing a Steiner tree and obtaining the Steiner points as the result, Chen and Cui present an algorithm similar to Kruskal's algorithm for creating minimum spanning trees which we describe next. We name this algorithm ψ in this manuscript.

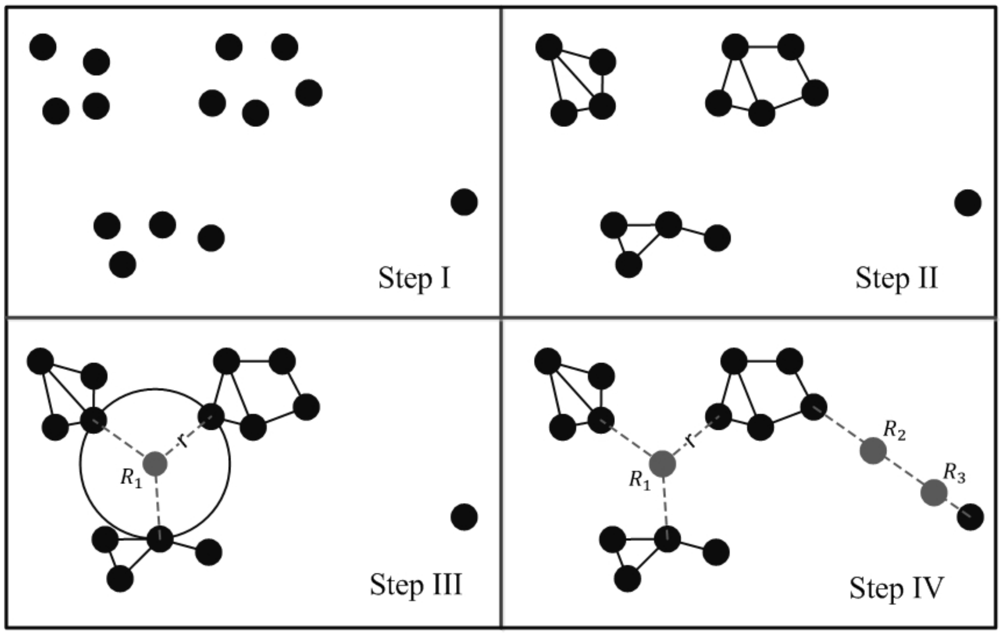


Figure 4.9: Four steps of algorithm ψ for RN placement algorithm in single tier networks

- $S = s_1, s_2, ..., s_n$ is the set of n SNs, $B = b_1, b_2, ..., b_m$ is the set of m base stations (trivially, we can let $m = 1$) and r is the communication range of the SNs . The output of the algorithm, Y, is a set of connectors. Also, T_A a disjoint set containing the vertices of the complete graph.

- The algorithm starts by running Kruskal's algorithm on the complete graph $G = (S \cup B, E)$ where $e_1, e_2, ...e_{(n+m)(n+m-1)/2}$ are edges sorted according to the Euclidean distances between two nodes in the graph.

- Next, for every e_i in the weight-increasing sequence $e_1, e_2, ...e_{(n+m)(n+m-1)/2}$, e_i is selected and the neighbouring vertices are merged in T_A if $e_i < r$. Each e_i selected connects two different components in T_A.

- Third, we select every three vertices $\{e, f, g\}$ in T_A that belong to three different components of T_A, and if possible obtain a point p such that the distances from p to e, f, g are all no more than r. We do so by passing a circumscribed circle through the triangle $\triangle e, f, g$ and calculating the coordinates of its center. If the radius of this circle is less than r, then p is added to Y and the edges pe, pf, pg combine associated components in T_A.

- Finally, for the remaining $e_i \geq r$ in ascending order, if e_i connects two different components or adds a new vertex in T_A, then we divide the edge e_i into small pieces of length at most r by $\lceil \frac{w(e_i)}{r} - 1 \rceil$ relay nodes and add the new vertices to Y. The new components thus connected are merged in T_A.

The pseudo code of the algorithm is provided in Figure 4.10.

```
1: $B \leftarrow Basestation$                                  ▷ $B$ the set of deployed BNs
2: $S \leftarrow RelayNodes$                                    ▷ $R$ a set of deployed SNs
3: $r \leftarrow$ Sensors Range
4: $R \leftarrow$ Relays Range
5: $E \leftarrow$ Set of all weighted edges between all nodes
6: $T_A$ A set of vertices                                     ▷ the Euclidean distance is the weight
7: $E \leftarrow Sort(E)$                                       ▷ The edges are sorted according to their weight
8: for all $e \in E$ do
9:     if $e_{weight} < r$ & $e$ connects two different components of $T_A$ then
10:        $T_A \leftarrow$ Vertices of $e$                     ▷ put the vertices of $e$ in $T_A$
11:        update connected components in $T_A$
12: for all $e \in T_A$ do
13:     for all $f \in T_A$ do
14:         for all $g \in T_A$ do
15:             if $e, f, g$ are from three different components then
16:                 $q \leftarrow$ the circumscribed circle of $\triangle efg$
17:                 if $q_r \preceq r$ then                      ▷ $q_r$ is the radius of the circle
18:                     $Y \leftarrow q_c$                       ▷ $q_c$ is the center of the circle
19: for all $e \in E$ do                     ▷ remaining edges not processed in the previous stages
20:     if $e$ connects two different components in $T_A$ or adds a new vertex to $T_A$ then
21:         divide $e$ into pieces of length at most $r$
22:         add dividing points to $Y$
23: Return $Y$
```

Figure 4.10: The Relay deployment algorithm presented by Chen and Cui [9]

As stated in [9], the time complexity of this process for the first step is $O(n^2 logn)$. For step 2 and step 4 the complexity is $O(n^2)$. For step 3, the complexity is $O(n^3)$. The overall time complexity of the algorithm is $O(n^3)$.

This algorithm is used in RN placement, which we describe next.

4.5.2 Part 2: An algorithm for the 1-coverage 1-connected problem

As explained previously, our problem is to place the minimum number of RNs so that all of the already placed SNs are connected to the base station. While RNs can connect to other RNs and SNs, SNs cannot communicate among themselves. We explain an algorithm that was proposed by Chen and Cui [9], and also consider its performance.

Similar to the previous algorithm, we have the following assumptions. $S = s_1, s_2, ..., s_n$ is the set of n SNs. $B = b_1, b_2, ..., b_m$ is the set of m base stations (trivially, we can set $m = 1$). We assume r is the communication range of the SNs, and R is the communication range of the RNs. The set of locations of RNs, Y, is the algorithm's output. We also assume the RNs have a larger communication range than the SNs, $R \geq r$. Furthermore, φ is an approximation algorithm based on Hochbaum and Maass's algorithm [20] for the 1-Geometric Disk Cover problem.

Assuming that $S \subset Z \times Z$ and given a rational number r, the minimum Geometric Disk Cover problem for (S, r) finds the minimum number of cardinality points $C \subset Q \times Q$ in a Euclidean plane such that every point in S is covered by a circle with the radius of r and the center being a point in

C. In other words for all of the points s_i in S, the distance between s_i and one of the points in C is less than r [9]. We have implemented the solution proposed by Hochbaum and Maass [20] which is a polynomial time approximation for this problem. The result of this process is a set of RNs that cover a given input set of SNs in a modelled environment.

The algorithm provided by Hochbaum and Maass for the problem of covering S points with the minimal number of disks with of diameter r (φ) works as follows: First we call the value l the ”shifting parameter”. Now assuming that we have some points that we need to cover in a square cell with the edge of $l \times r$, we are able to fully cover all the space in that square with $2 \times l^2$ disks. Using exhaustive search we calculate the minimum number of disks required to cover the points we need in that square by in turn removing disks from the initial state. However this process is computationally expensive.

To overcome this problem, we subdivide the area containing the points into vertical strips that have a width of $l \times r$. We again divide these vertical strips into horizontal grids that have a width of $l \times r$ (We created a grid where each cell has a edge size of $l \times r$). Now we run the exhaustive search algorithm of finding the minimum number of disks required to cover the points in a unit square for each cell and obtain a set of points C_i. This leads to an optimal solution in any stripe separately, but does not necessarily lead to an optimal solution of entire rectangles. So in turn horizontal and vertical strips are shifted and a new set of results is obtained for each cell. This bounds the error of the divide and conquer scheme. The outcome of the algorithm is the minimum number of disks obtained from one of the shifts.

Now the rest of the algorithm of RN placement works in two stages.

- **Step 1** The algorithm φ is applied to the set of the already placed SNs, S. The output is a set C, which is a feasible solution of the minimum Geometric Disk Cover problem.
- **Step 2** The set $C \cup B$ and radius R is given as input to the algorithm ψ which was described previously. Y is obtained as output.

In step 1, this algorithm satisfies the condition that each SN should connect to at least one RN. The second condition that the RNs should be connected to the base station is satisfied in step 2. Together, the result is a feasible solution for the 1-coverage 1-connected problem. The overall result of the algorithm is $C \cup Y$.

In their paper, Chen and Cui prove that the resulting set Y has a performance ratio of no worse than $(5 + \epsilon)$ of the optimal solution.

The algorithm suggested by Hochbaum and Mass [20] has a time complexity of $O(n^{1/\epsilon^2})$ for a given $\epsilon > 0$ and a $(1 + \epsilon)$-approximation algorithm. The value for ϵ is usually chosen as 0.3, so the complexity of the Hochbaum and Mass algorithm is typically about $O(n^{11})$. Considering the time complexity of algorithm ψ which was $O(n^3)$, the total time complexity to obtain the result is $O(n^{1/\epsilon^2})$ [9].

We run this algorithm on the transformed area, given the locations of the SNs from the previous step, to find the positions of the RNs. Once this process is complete, we have the coordinates of the needed RNs in the transformed map. However as we need the real coordinates of these points so that we can deploy them in the original area, we have to perform a reverse transformation to obtain the coordinates of the RNs in the original non-uniform area. To do so we keep track of the transformations that are performed on the cells during the transformation from the non-uniform area so that after the RN placement, we are able to do the reverse transformations accordingly.

4.6 Discussion

In the following we discuss the performance, strengths and weaknesses of our SUR method.

4.6.1 Number of RNs

We should note that applying the transformation does not necessarily decrease the number of RNs in the RN placement stage. Based on the map and the path loss exponents, increased or decreased number of RNs in comparison with a situation where we don't run the transformation and the second step in general, can result. While the objective of our transformation method is to consider the irregularities of the communication map, a model without the transformation method would deploy more nodes in areas that have lower path loss exponents and place fewer nodes in areas that have higher path loss exponents. As illustrated in Figure 4.11 an RN placement algorithm without the transformation algorithm might consider using an average path loss exponent of, for example $\alpha = 2.5$ for the whole field. However, if nodes R_1, R_2 and R_3 have an area with a path loss exponent of $\alpha = 2$ between them, then a redundant RN might have been placed which we remove with our algorithm. Similarly if R_2 is in an area with a path loss exponent of $\alpha = 3$ then additional RNs might be needed so that communications between R_1 and R_2 and R_3 would not be faulty.

4.6.2 Number of Cells

As mentioned previously, increasing the number of the cells in the grid will increase the accuracy of the RN placement in the next round. However this might lead to numerous small cells that each need a path loss exponent. Estimating this value for each of them might be impractical.

4.6.3 Sensing model

The algorithm that we used for SN deployment, requires a probabilistic sensor detection model to determine how well a sensor covers a cell. This means that while we use a uniform attenuated disk model, other models that necessarily do not presume uniform sensing by nodes in all directions can be used as a model for SN deployment. The second and third stage of our model are not specific for the current algorithm we use.

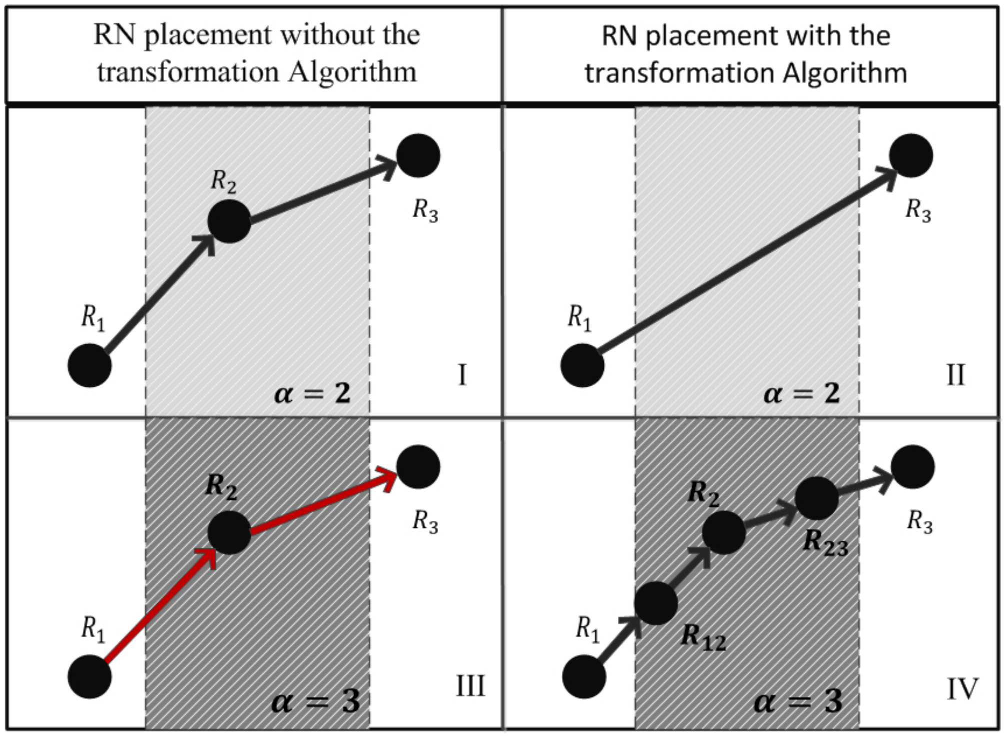


Figure 4.11: RN placement in different conditions

4.6.4 Other applications

In this thesis, we use the transformation in step 2 of our algorithm to increase the communication quality of links in real environments using our model. The effect of this proposed method is apparent in stage 3 of our framework. However this transformation method and model may be used in any real area and with any wireless communication algorithm that assumes homogeneous connections among nodes. The applications are not limited to node placement in WSNs, and fields such as data routing and network topology algorithms could benefit from this method. In addition, although the transformation of cells depends on the signal propagation formula we introduced in the previous section, even with different models, new transformations can be proposed that can have the same effect and decrease the communication error rate of algorithms that assume homogeneous connections among nodes.

4.6.5 Suggested Improvements

This framework can be improved in various ways. In general, improved algorithms can be used for SN placement. Also other algorithms that concentrate on parameters other than coverage, such as target detection or network longevity and data fidelity can be implemented. Furthermore the RN placement algorithms can also be enhanced as the framework does not depend on only one algorithm. Additionally, using cartograms that better satisfy the requirements of a desired uniform field (especially the second constraint) will improve the overall performance of the framework because they better estimate the connections among nodes.

Chapter 5

Max-cover 1-connected Algorithm

In the previous chapters we explained that many simple node placement algorithms cannot be utilized because of the communication model and our non-uniform field (in terms of communication). With our SUR algorithm, we proposed a solution to overcome this problem by creating a uniform field using a transformation on our communication model. Here, we solve another problem in node placement and suggest a three stage solution for it. While in the previous chapter the objective was to cover a specific proportion of an area with the minimum number of RNs and SNs, herein we solve the problem of maximizing the area covered by a limited number of SN and RNs. Our algorithm aims to fully utilize each node so that the sensing coverage area would be increased the connectivity maintained with a specific number of SNs and RNs.

We formulate our problem as follows:

Given Parameters:

$R = r_1, s_2, ..., r_n$:	Set of the Relay Nodes.
$S = s_1, s_2, ..., s_m$:	Set of the Sensor Nodes.
$B = b_1, b_2, ..., b_o$:	Set of the base stations.
π =	$a_1, a_2, ..., a_{m+n+o}$: Set of candidate node locations
$\mu(r_i)$ =	$(c_1, c_2, ..., c_t)$: Set of points where SNs can be deployed in and connect to a r_i.
$d(i, j)$ =	Euclidean distance between location i and location j. $i, j \in \pi$
$\lambda(i, j)$ =	Communication distance between location i and location j. $i, j \in \pi$
n :	Number of RNs.
m :	Number of SNs.
w :	Width of each cell
h :	Height of each cell

Decision Parameters:

$$\text{cc}(\pi) = \begin{cases} 1 & \text{if all SNs are directly connected to at least one RN and all RNs are connected to at least one base station .} \\ 0 & \text{if the conditions above do not stand.} \end{cases} .$$

$$E(\pi, R) = \begin{cases} A_s = & |\mu(r_1) \cup \mu(r_2) \cup ... \cup \mu(r_n)| \\ 0 & \text{if } cc(\pi) = 0 \end{cases} .$$

$$\text{Neighbour}(\pi) = \{< a'_1, a'_2, ..., a'_{m+n} >| \; a'_i \in neighbors(a_i)\}.$$

$$\text{neighbours}(a_i) = \quad a_j \in \pi \mid d(a_i, a_j) \preceq \sqrt{w^2 + d^2}$$

Objective Function:

$$\underset{\pi}{argmax} \quad A(\pi).$$

where m, n the number of SNs and RNs is given as input.

Analogous to our previous algorithm, our max-cover 1-connected method is also composed of three main stages:

- **Greedy RN placement**: First, we present a iterative greedy algorithm for RN placement. In each step this algorithm places RNs in such a way that the candidate locations for SN placement and also RN placement in the next round would be maximized while also preserving connectivity among previous deployed RNs. This algorithm does not necessarily lead to suitable results.

- **Improving the RN placement by using Simulated Annealing**: Once RNs are initially deployed, we try to improve the placement by running a stimulated annealing algorithm on a defined objective function. As the objective in general is to maximize the area sensed by a specific number of SNs, an efficient and effective RN deployment algorithm increases the areas which the the SNs can potentially be deployed (places which a SN can communicate with a RN and base stations) and also preserves the connectivity among the RNs and the base stations.

- **Sensor node placement**: Finally we need to deploy SNs in candidate locations that the RN placement algorithms have provided in previous steps. Consequently this stage turns into a normal SN placement algorithm where SNs can be deployed only in specific locations. We utilize and modify a method presented by Pompili et al. in [40] for covering uniform sensing areas. The candidate locations already have the ability to establish a connection to the base station through RNs as this property was already provided in the previous steps. So as long as nodes are deployed at points obtained in the previous stages, the connectivity would be established.

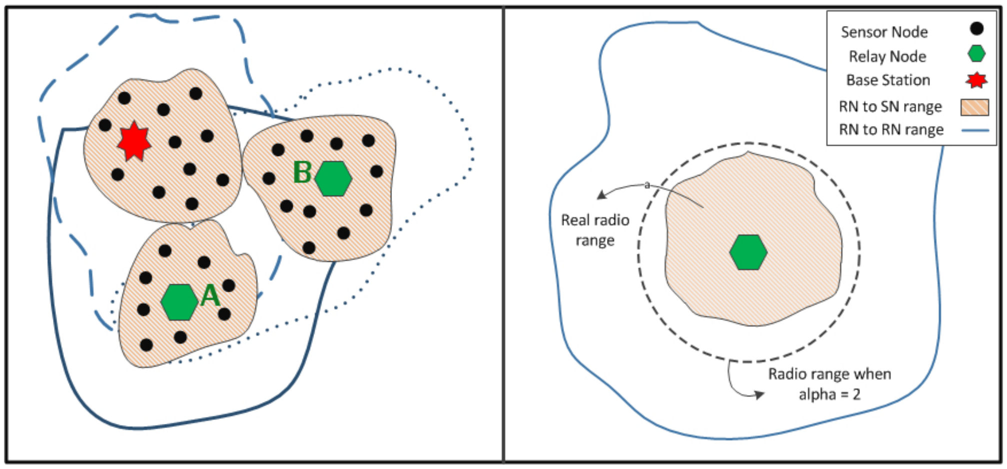


Figure 5.1: Radio range of relay nodes and sensor nodes in a real environment.

5.1 Greedy Relay Node placement

Unlike other node placement algorithms, our max-cover 1-connected method aims to create an algorithm that deploys both RNs and SNs in a multi-tiered network. Other work usually either places SNs or deploys RNs according to the locations of the SNs. Based on our communication model presented in Chapter 3, we suggest to first deploy the RNs rather than the SNs and then based on the areas which the RNs are able to communicate over we can choose the locations for the SNs.

Before explaining the mechanism of the first stage of our algorithm, we give two definitions:

- **Relay to relay communication area**: Is the area in which RN B can connect to RN or base station A. This means both A and B should be able to receive signals from each other that have a power of more than P_r. In Figure 5.1 nodes A and B are connected to each other but node B is only partially connected to the base station. If we define the communication areas of nodes A, B as $CA(A), CA(B)$ then the communication area of this pair of nodes $CA(a, b)$ is $CA(A) \cap CA(B)$.

- **Sensor to relay communication area)**: Similarly the sensor to relay communication area is the area where sensor node S can connect to RN or base station A. In Figure 5.1 the shaded areas are ranges where the sensors can connect to respective RNs or base stations.

Note that based on the map and our communication model, the resulting shape of both the sensor to relay and relay to relay communication range of a node in all directions is not necessarily a circle. In fact circles are created only when the the communication distance is equal in all directions which usually is not the case when we have various path loss exponents as in real life.

Based on these definitions the algorithm operates as follows:

First a preprocessing step finds the areas where the SNs and RNs can be deployed for each point. A grid ω is created and based on the nodes radio ranges, two set of points, one for the relay to relay communication area and one for sensor to relay communication area is obtained for each vertex of the grid. This determines the points that the specific point can communicate to if there was a SN or RN at that specific point. For each vertex a set of the candidate points for both the SNs $(p_1, p_2, ..., p_n)$ and RNs $q_1, q_2, ..., q_n)$ is kept. This leads to an estimated area of the sensor to relay and the relay to relay communication range. The number of cells of this grid depends on the desired accuracy level. This grid is also used in the next stages of the algorithm. The complexity of finding this set is $O(N^2 \times n)$ where N is the number of cells of grid ω and n is the number of cells of the θ grid (we have to calculate λ between every two points). In Figure 5.1 each RN has a hypothetical maximum radio range when $\alpha = 2$. The shaded areas in Figure 5.1 represent points where SNs are able to communicate with the corresponding RN.

The algorithm starts with a given set of base stations. In each step, we iterate on the base stations and RNs and for each point of the RN candidate point set q_i of these RNs and base stations, the SN candidate points p_i are pre-calculated. An RN is placed on a point that maximizes the total sensing area *i.e.* the number of candidate points p_i. In Figure 5.1 points A and B have been selected based on the shaded area they have added. The algorithm continues until all the RNs have been placed.

Analysis

Because of the greedy nature of the algorithm, the method is prone to missing RN deployment locations that may lead to increased sensing areas. For example if the method reaches a region that has a path loss exponent of $\alpha = 3$, regardless of the thickness and extent of that zone, it tries to go around it rather than attempting to place RNs in that area. If the region is small enough, it may be feasible to place some RNs inside that region and find more suitable locations to place other RNs afterwards. We try to confront this shortcoming in the next stage of the algorithm. The complexity of the algorithm in each step for each RN is $O(RR \times RS)$ where RR is the available number of candidate locations for RN placement (we have to check every point) and RS is the number of candidate locations created for SN placement in that stage (the maximum points added to the candidate sensor locations set).

5.2 Simulated Annealing

To improve the performance of our RN placement, we propose using a Simulated Annealing (SA) algorithm on the results of the greedy algorithm. Simulated annealing delivers notable solutions on combinatorial optimization problems. It is suitable for our use because of the large solution space of our problem. Although SA does not necessarily reach the best possible solution of a given function in a large search space in a certain amount of time, it locates a good approximation. Solving problems with SA is relatively simple to formulate and requires low memory to run [47]. The idea of SA in computer science originates from a paper in annealing solids presented by Metropolis et al. [50] in 1953. Kirkpatrick et al [24] developed the idea of applying the technique to solve optimisation problems in 1983.

One of the most important properties of SA-based algorithms is their ability to escape from local maxima and minima by sometimes choosing worse moves. The probability of accepting a worse move depends on the current state (temperature) of the system and the change in cost function. The SA algorithms select a random move from the available neighbouring moves rather choosing the best available move. If the move is better than the current position, then the move is certainly taken. If not, worse moves are selected based on the following probability [47]:

$$p = e^{-(\frac{\Delta E}{t})} \succ R(0,1)$$

In this function, ΔE is the change in the evaluation function, T is the current temperature of the system and R is a random number between 0 and 1.

Starting from a higher temperature $T = 0.9$, to cool our system, we use the following cooling function:

$$F(T) = c/1 + log(1 + k)$$

where $c = 10$ is a constant and k is the current iteration number. In each step $T = F(T)$. Using the mentioned function, the temperature dynamically changes as the algorithm progresses.

Preprocessing

From the previous round, we have an RN deployment as input. Again we grid the area and keep the set of points covered in the relay to relay and sensor to relay communication areas for each point. The grid is named ω. Also we move the RNs to the nearest grid point in ω. This means that if the number of cells in ω is not big enough, then the connectivity among RNs might be broken and reaching a suitable answer would not be an easy task.

Energy Function and Cooling Schedule

In each step the SA algorithm selects one RN randomly and moves it to a random neighbour location. Then we examine the change in the energy of the system. Maximizing the number of points in the sensor to relay communication areas of all RNs and base stations (A_s) is our objective so the current energy of the system E in each state is the number of points covered by the sensor to relay communication area of the RNs. The difference in this value works as the ΔE. If the answer is better than the best yet achieved result, then we update the best achieved results and move on.

Maintaining connectivity among RNs and the base stations is an important requirement. If the connectivity among RNs and the base station is broken in a step, re-establishing the connections might require many steps. In our implementation we do not allow our RNs to move to locations where it disconnects them or other RNs from the base stations. But doing so still does not prevent

```
B ← $Basestation$                                    ▷ $B$ the set of BNs
R ← $RelayNodes$                     ▷ $R$ the set of RNs from the Greedy algorithm
T ← $CurrentTempreture$
while $true$ do        ▷ Or other constraints or conditions on the run time of the algorithm
    $r \leftarrow$ random RN
    $SelectMove(r)$          ▷ Selects a move for the relay node to a neighbouring position
    if $CalculateNextTempreture() \succ 0$ then     ▷ smaller than zero means no connectivity
        $E \leftarrow CalculateNextTempreture$
        $\Delta E \leftarrow E - T$
        if ( then$\Delta E \succ 0$)
            $Move(r)$
        else
            if ( then$e^{-\frac{\Delta E}{t}} \succ 0$)
                $Move(r)$
        $T \leftarrow E$
function PREPROCESS
    for all $n \in Cells$ do
        $n_{RelayCover} \leftarrow$ covering relay points
        $n_{SensorCover} \leftarrow$ covering sensor points
    for all $r \in RelayNodes$ do
        Move $r$ to closest grid point
function CALCULATENEXTTEMPRETURE
    if a RN is not connected to a base station then       ▷ using DFS and $n_{RelayCover}$
        return $-1$
    else
        return Calculate number of all covered points using $n_{SensorCover}$
```

Figure 5.2: The Simulated Annealing Algorithm

us from reaching the global maximum as the best possible answer would still be in the search space range. In each step we check the connectivity by running a depth-first search algorithm from the base station and also calculate the change in the objective function by calculating the number of points covered by the RNs.

In our implementation, we always keep the best achieved answer. Our algorithm finishes when a specific number of iterations has been performed. That is, until the system has reached a specific temperature T_d. However, other constraints such as continuing until a specific amount of sensing coverage is achieved, can be used to limit the runtime.

The complexity of each iteration of our SA algorithm is $O(R \times N^2 + R^2)$ where R is the number of RNs available and N is the number of cells in the grid. The R^2 is because we have to check the connectivity between the RNs and the base station and the $R \times N^2$ is for calculating the SN candidate location set. This makes our method scalable for bigger maps.

We present the pseudo code for the algorithm in Figure 5.2.

Once the algorithm is finished, we have obtained a set of RNs that are connected to the base station and maximize the sensor to relay communication area. In the next stage, we must place the SNs. We have candidate locations that are suitable for node placement and we should choose among

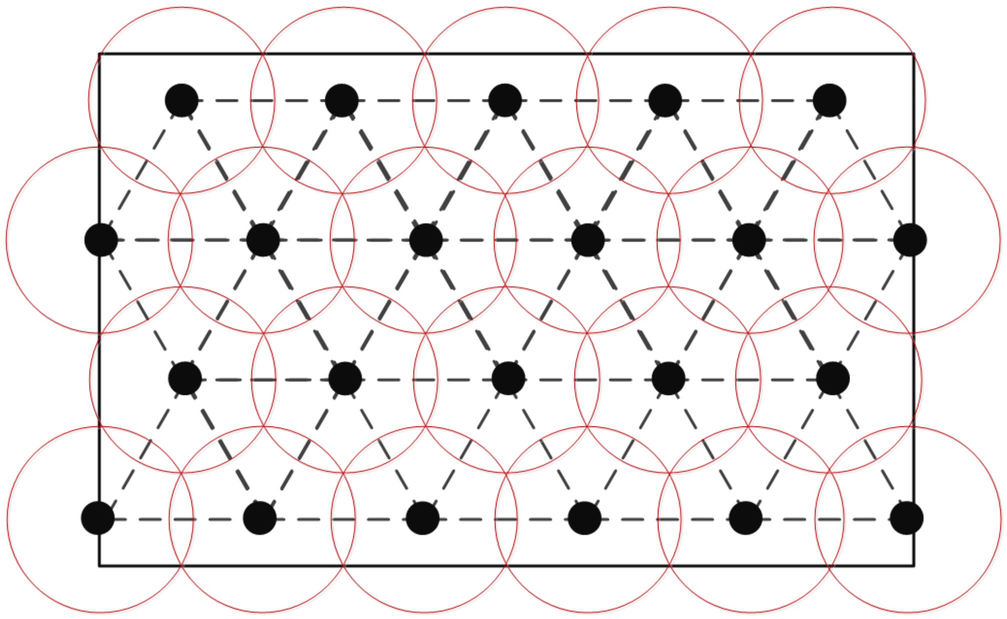


Figure 5.3: Full coverage of an area is achieved when $d = \sqrt{3}S_r$.

these points.

5.3 Sensor Placement

The last stage of our algorithm is SN placement. SNs must be deployed according to the placement of RNs. One of the benefits of our algorithm is that in this stage, an SN placement algorithm does not have any connectivity concerns. The SN placement algorithm in this stage could be maximizing the covered area with the minimum number of SNs where the we only have candidate areas for SN placement. The provided areas are not necessarily connected to each other. We should note that as we only have candidate locations for the SNs, an upper bound for the maximized area that can be covered exists. Here we solve the problem of maximizing the area covered with a specific number of SNs and present a greedy algorithm for it. We assume a uniform sensing field and consider a disk covering model for the SNs. This means that the SNs either fully sense an area or do not sense it at all.

We modify the SN placement algorithm proposed by Pompili et al. [40] to maximize the coverage with the minimum number of nodes in an area where the sensing model is a disk covering model. Their method is presented for underwater WSN applications and is proposed for both two and three dimensional space. They assert that if the distance between adjacent nodes (d) in the grid is $\sqrt{3} \times S_r$, (S_r is the sensing range of the SNs) then full coverage of an area can be achieved. Figure 5.3 illustrates such a deployment.

However, if the distance between the adjacent SNs in the grid is $2 \times S_r$ or more, then the sensing cover disks of neighbouring nodes would not have any junctions and the sensing covered area by

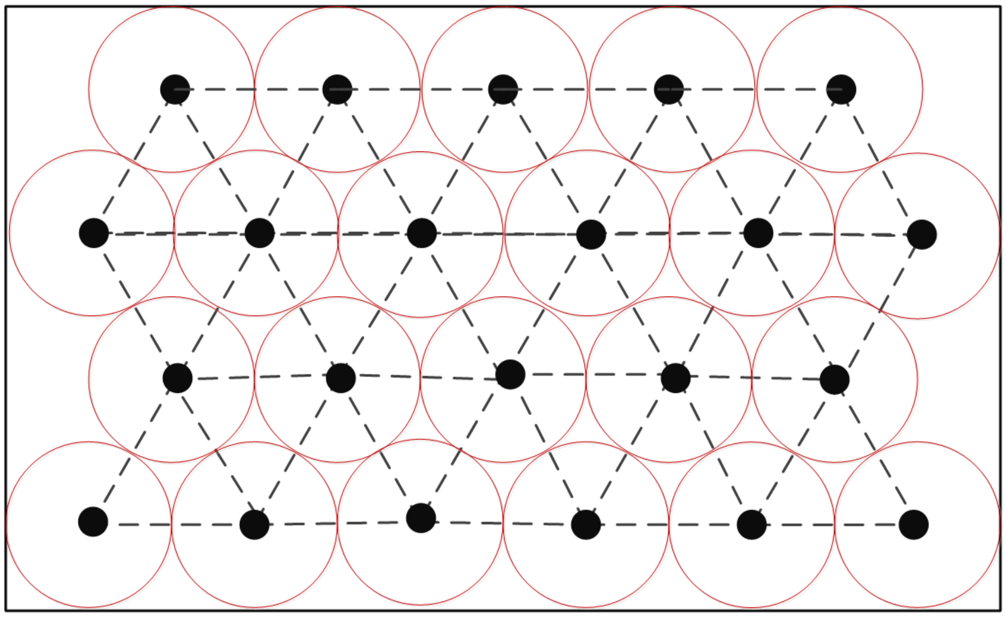


Figure 5.4: Coverage is maximized with the same amount of SNs when $d = 2 \times S_r$

each disk is maximized. This requires SNs to be apart from each other and requires space which means that the previous RN algorithm should create large sections of candidate locations for the SNs. Figure 5.4 depicts a deployment where $d = 2 \times S_r$.

However as we want to minimize the covered communal space between the neighbouring SNs and also want to place the maximum number of nodes in a specific area available, we need to find the best possible distance d between the adjacent SNs and then create a triangular grid τ accordingly. In this way the sensing range disks of the SNs would have minimum interference while remaining as close as possible to each other so that additional SNs can fit into the area. Hence the sensing coverage efficiency would increase.

So to find the best distance d we test different values for d between $d = \sqrt{3}S_r$ and $d = 2 \times S_r$ and select the value that maximizes the area covered by the SNs. If the step size is w, then the total number of times that we need to test a deployment is:

$$t = \frac{(2 - \sqrt{3})S_r}{w} \tag{5.1}$$

In each step, our SN placement algorithm works as follows:

- **Creating a grid:** In each step i we create a triangular grid τ for the whole field and set the distance of adjacent vertices $\sqrt{3} + (w \times i)$.
- **Placing SNs:** According to the RNs deployed in the field from the previous stage and the areas created for SN placement, if a vertex of the grid τ is in the areas covered by the RNs, we

select the location for SN deployment. This process continues until all the SNs are deployed or we run out of space.

- **Examining the placement:** Based on the deployed SNs, we calculate the total area covered by the SNs. This is done by creating another grid ς that has smaller cell sizes to increase the accuracy of the measurement. We examine whether a point in the grid ς or not. If a point is covered by two or more SNs, we only consider that point covered once.

We keep the deployment that maximizes the sensing area covered and alongside the locations of the RNs, we return it as the output of our max-cover 1-connected algorithm.

With our method, areas that are available for node placement, will be fully utilized and if there is not enough room for SN placement (even with $d = \sqrt{3}S_r$) then the algorithm returns the surplus number of SNs. As illustrated in Figure 5.5, the area is tiled into a triangular grid with $d = 2 \times S_r$ and then points that are in the sensing communication range of the RNs, are selected for SN deployment. The complexity of placing the SNs is $O(n)$ where n is the number of candidate locations where the SNs can be placed.

5.4 Discussion

In this section we discuss the performance, strengths and weaknesses of our max-cover 1-connected method.

5.4.1 Performance in different maps

The objective of our max-cover 1-connected algorithm is to maximize the areas where potential SNs can be deployed. This means that it searches for areas that have a lower path loss exponent. However this might not be the objective of the user. In many cases SNs need to be placed everywhere, and uniformly rather than in specific points that have lower path loss exponents. In that case we suggest using our SUR algorithm because considering the SNs, it tries to connect them to the base stations. Our max-cover 1-connected algorithm has better use when the vegetation in an area is rather uniform and the average path loss exponent does not have a high variance.

5.4.2 Obstacle Avoidance

One of the main benefits of our max-cover 1-connected algorithm, is its capacity of considering obstacles and forbidden regions. These areas can easily be specified by allocating a very high path loss exponent α for them. In this way the algorithm tries to avoid the obstacles and places nodes around them.

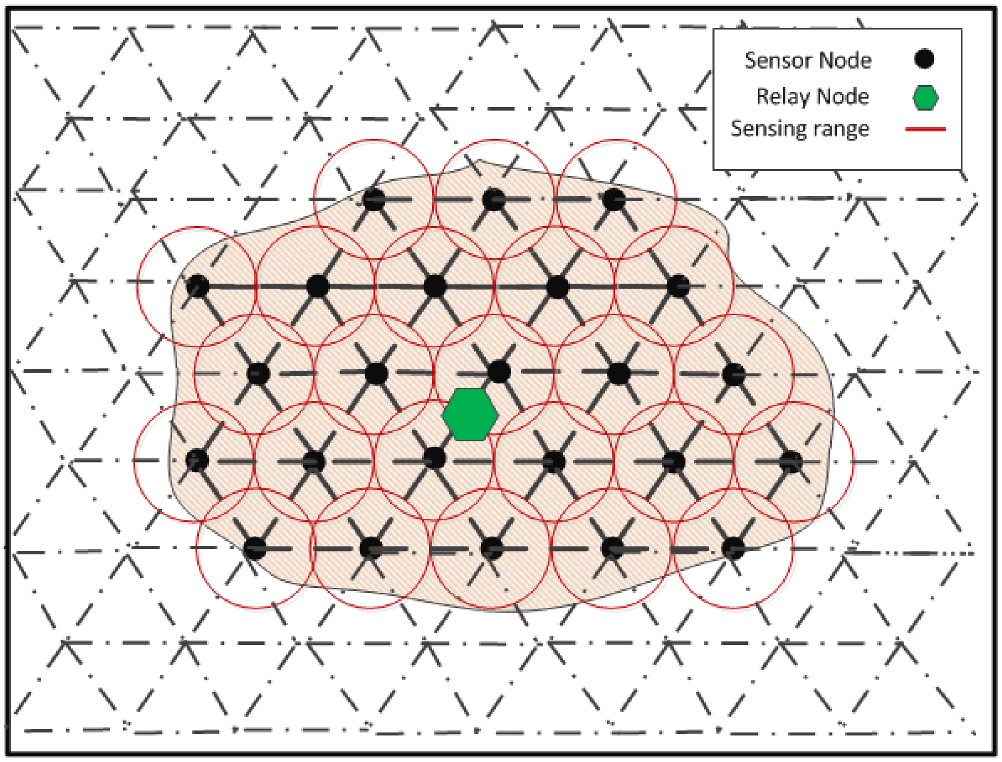


Figure 5.5: SN placement in candidate points.

5.4.3 Preprocessing

Before the SA algorithm, we use a preprocessing step that for each point in the grid ω finds the set of vertices that are in the relay to communication and sensor to communication areas of that point. The complexity of this process is $O(n^2 \times m^2)$ where n is the number of cells in the grid ω and m is the number of cells in the communication grid θ. Finding $\lambda(i, j)$ between every two points i, j is $O(m^2)$ so the whole process becomes $O(n^2 \times m^2)$ as for each point we need to check whether other points are in its sensor to relay or relay to relay communication distance or not (we recall that $\lambda(i, j)$ is not necessarily equal to $\lambda(j, i)$). But because we have a upper bound for the sensor to relay and relay to relay communication ranges (when $\alpha = 2$), we improve the performance of the pre-process by only checking the points that are in the upper bound distance of each point. In general the high time complexity of the pre-processing limits the number of cells ω which means reduced accuracy for the SA.

5.4.4 Number of SNs

To better realize the full effects of the algorithm, a sufficient number of SNs is required. Otherwise with only a few SNs (compared to the number of RNs) the placement of the RNs would not really matter as we might be able to deploy all the SNs regardless of how well the RNs are distributed. The effect of our algorithm is apparent when the sensor to relay communication areas of the RNs is fully utilized by a sufficient number of SNs.

5.4.5 Communication

One of the main benefits of our max-cover 1-connected algorithm is that it considers the communication function λ in its node placement. Theoretically, there should not be any errors in the communication between nodes. Furthermore the algorithm is not dependent on the function λ and other communication models can also be used.

5.4.6 Other variation of the SN problem

Once the RNs are placed and deployed, a good estimate of the maximum number of SNs that can be fully utilized in that area can be developed. In other words, while we solve the problem of using a specific number of SNs to maximize coverage, the problem can be turned into finding the minimum number of SNs that achieve maximal coverage, because the candidate areas that SNs can be deployed in is limited.

Chapter 6

Results

In this section we present the results from the experiments of our SUR and max-cover 1-connected algorithm. We explain the results in two main sections. First we present the results from the max-cover 1-connected algorithm.

6.1 Results of the max-cover 1-connected algorithm

In the previous chapter, we presented the max-cover 1-connected algorithm. This algorithm takes a number of SNs and RNs as input and maximizes the total area covered by the SNs by also maintaining connectivity among the SNs to the base stations through the RNs. We presented a three stage algorithm that consisted of 1) a greedy RN placement algorithm, 2) a simulated annealing algorithm that enhanced and improved the results of the first stage and finally 3) the SN placement algorithm. We implemented our proposed solutions and in this section we talk about their performance.

6.1.1 The performance of the max-cover 1-connected algorithm in specific cases

The first stage of our algorithm places RNs in an iterative and greedy manner. This behaviour leads the algorithm to select points that result in the maximum sensor to relay communication areas. In Figure 6.1, a $1000m \times 1000m$ map is depicted with the shaded areas representing parts with a high path loss exponent ($\alpha = 3$). The rest of the map has a path loss exponent of $\alpha = 2$ and the base station is located at the top left corner of the map. In this figure the green dots represent the candidate locations of SNs and the blue circles are the sensor to relay communication ranges of the RNs. Furthermore the relay to relay communication ranges are shown as orange circles. The RNs are represented by these two circles. Note that the radio ranges that cross borders are not accurate due to the change in α between cells.

As illustrated in Figure 6.1, the algorithm starts placing nodes by selecting points that maximize the sensor to relay communication areas so that maximum number of SNs can be deployed. This leads the algorithm to go around the shaded area and reach the regions that have lower path

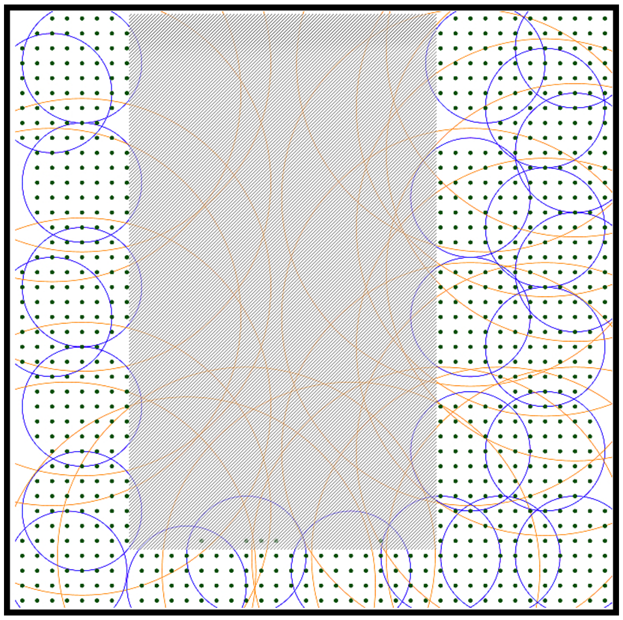


Figure 6.1: Greedy RN Algorithm placing RNs in suitable locations.

loss exponent which leads to achieving maximum coverage. Choosing locations that maximize our candidate locations fulfills the objective of the algorithm. Figure 6.1 depicts the benefits of such a method.

However the greedy algorithm may not always find its way to better positions for RN placement. Because of its greedy nature, it does not explore alternative paths. As illustrated in Figure 6.2, the greedy algorithm gets trapped in the region with higher path loss exponent because it cannot find better variations in its small search space.

To overcome this problem we use the second stage of the algorithm, simulated annealing (SA), to reach the white areas that have lower path loss exponents so that the RNs can be better utilized. As shown in Figure 6.3, the SA easily finds a route to the clear side and with the same number of RNs, the area provided for the SN placement is significantly increased. As depicted in the figure, nodes that are deployed in areas with lower path loss exponents have larger radio ranges compared to nodes deployed in areas with $\alpha = 3$.

Once the SA algorithm places the RNs, candidate locations are available for SN placement. Inspired by the method used by Pompili el al. [40], we place SNs on a triangular grid. The distance d between two neighbouring vertices on this grid may result in different deployments and total area covered by these deployments. We test different values for d and select the deployment that maximizes the total area covered by the SNs. Table 6.1 summarizes the results for different values for a sample deployment of 10 RNs in a $1000m \times 1000m$ map with random path loss exponents for its cells. S_r is the sensing radius of each SN, which we assume is $10m$ here.

As seen in Figure 6.1, the best result is achieved when $d = S_r \times \sqrt{3}$ however, this placement

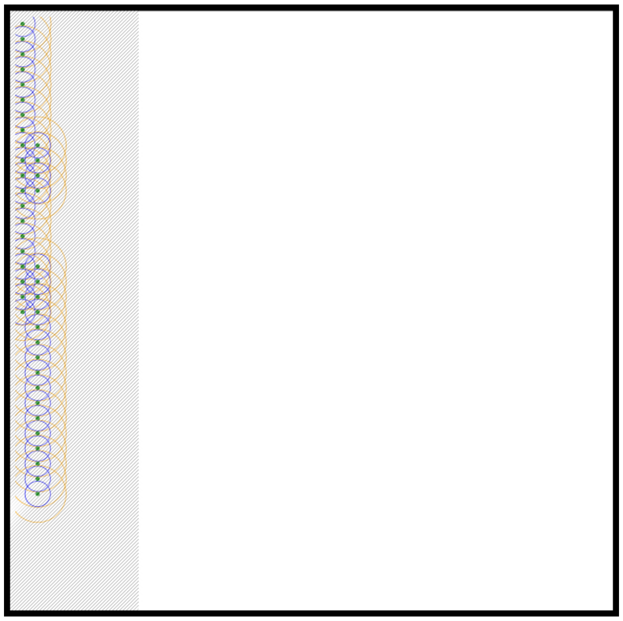


Figure 6.2: Greedy RN Algorithm not able to place RNs in suitable locations.

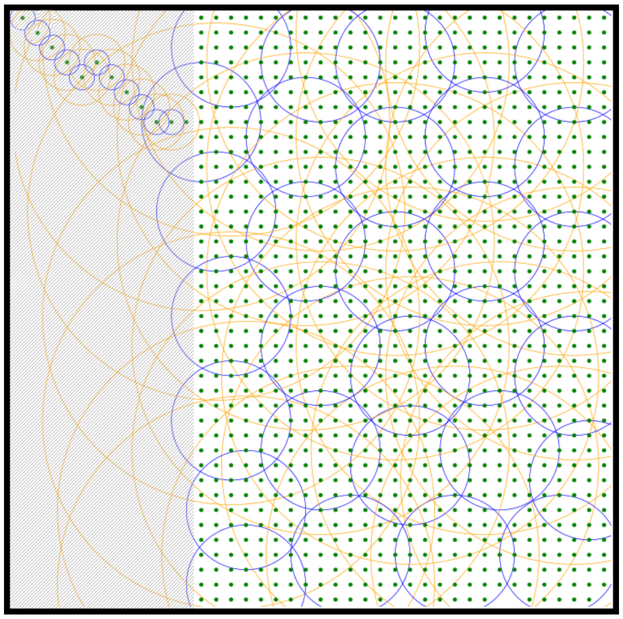


Figure 6.3: SA algorithm improves the performance of the initial greedy algorithm.

Table 6.1: Comparing different SN deployment for the max-cover 1-connected algorithm

Distance Between Neighbouring vertices	Total Number of Placeable SNs	Total Area Covered (%)
$\sqrt{3} \times S_r$	626	16.80%
$1.78 \times S_r$	603	16.66%
$1.83 \times S_r$	563	16.30%
$1.88 \times S_r$	520	15.56%
$1.93 \times S_r$	503	16.30%
$2.0 \times S_r$	478	14.86%

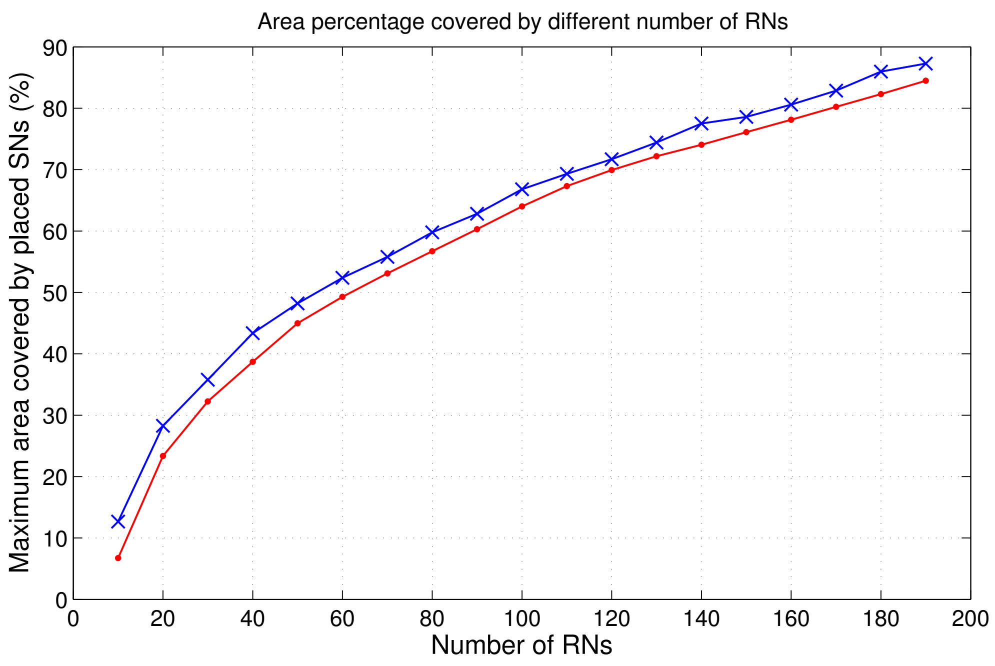


Figure 6.4: Comparison of the SA and greedy algorithms.

needs 626 SNs. If we want to deploy slightly fewer SNs, 603 for example, the covered area is maximized by $1.78 \times S_r$

6.1.2 Comparing the greedy algorithm with the SA algorithm

To better compare the performance of our SA and greedy RN placement algorithms, we compare the percentage of area that their deployed SNs can cover. To do so, we vary the number of available RNs and run the algorithm on a $1000m \times 1000m$ field. The communication grid θ is a $10x10$ grid and has random α values for its cells. The grid ω that is used by the SA and greedy algorithms is a 50×50 grid.

In Figure 6.4, we present the performance of these two algorithms. In our experiments the SA algorithm improves the results of the greedy algorithm by 3.15% on average. The map was a $1000m \times 1000m$ map and the path loss exponent of the cell of the θ grid were chosen randomly (values between 2 and 3. As mentioned earlier in special cases where the greedy algorithm is not able to proceed and produce desired results, the beneficial effects of the SA algorithm are better observed.

6.1.3 Comparing our algorithms with the optimal solution.

Finally, we compare the results of our max-cover 1-connected algorithm with an exhaustive search back track algorithm that examines all possible placements of the RNs in the field and returns the optimal result. The back track algorithm looks into all variations and possibilities of RN placement

Table 6.2: Comparing the maximum sensing area of the max-cover 1-connected algorithm (with and without SA) with a back track exhaustive search algorithm (optimal)

Map Size (ω size)	RN Num	Optimal	SA	Greedy
$200m \times 300m(6 \times 8)$	4	39.40%	39.40%	37.09%
$200m \times 300m(6 \times 8)$	5	44.30%	44.30%	38.03%
$200m \times 300m(6 \times 8)$	6	48.19%	48.19%	43.58%
$200m \times 300m(6 \times 8)$	7	54.67%	54.67%	48.21%
$250m \times 250m(8 \times 8)$	5	58.47%	58.47%	56.48%
$250m \times 250m(8 \times 8)$	6	62.57%	62.57%	62.57%
$250m \times 250m(8 \times 8)$	7	69.00%	69.00%	67.18%
$300m \times 300m(9 \times 9)$	5	46.5%	46.5%	46.5%
$300m \times 300m(9 \times 9)$	6	54.9%	54.9%	45.2%
$300m \times 300m(9 \times 9)$	7	62.5%	61.2%	51.3%

and returns the best result. So the purpose of this test is to compare our SA and greedy algorithms with the optimal solutions. Because of the huge search space, the back track (BT) algorithm can only be tested on smaller ω grids. The results of the evaluations between the three different algorithms, greedy, SA and backtrack, are presented in in Table 6.2. In these tests, the number of RNs (RNNum) vvaried and the map is again a random map.

We examine the performance of the algorithms in different maps with different sizes of ω. In Table 6.2 we present the results that distinguish their performance. Our SA algorithm achieves the optimal result in all but one case. In this case the BT algorithm has a better performance but it needed 37 minutes to do so. The SA algorithm only operated for 10 seconds and by increasing its number of iterations, we also achieved identical results in this case too.

6.1.4 Summary

In this section we presented the performance of our max-cover 1-connected algorithm. The purpose of our algorithm was to maximize the areas covered by SNs in a multi-tiered network by conveniently placing the RNs and SNs. The greedy stage of the algorithm was compared with the SA stage and the benefits of each strategy was explained. We argued that the combination of these two methods leads to a proper algorithm and evaluated the performance of our algorithm by comparing it to the optimal solution. In small cases where finding the best possible answer was possible, the SA algorithm achieved nearly identical results with the optimal solution.

6.2 Results of the SUR algorithm

As explained in Chapter 4, the main purpose of the SUR algorithm is to create a framework that allows different SN and RN algorithms to be deployed in a modelled real environment. It also, on average increases the sensor to relay and relay to relay connectivity. This means that we expect the SUR to consider different path loss exponents of different cells and place RNs more accurately in terms of communication links.

6.2.1 Experimental Setup

In our experiments, we consider a $1000m \times 1000m$ area that is divided into a $2^{\lceil \frac{d}{2} \rceil} \times 2^{\lfloor \frac{d}{2} \rfloor}$ grid to create the communication grid θ of the map. In some of our experiments we set the path loss exponent of the cells to the same value to show the performance of our algorithm while in other experiments it is chosen randomly. We use d, depth, as one of the properties of the communication grid; d also indicates the number of cells of the grid.

In our experiments we measure different metrics about the links between the nodes. We use the following definitions:

- **Faulty Link:** A link between an RN-SN or RN-RN communication link is faulty if the received signal strength is less than a certain threshold. Considering our signal propagation model and simulated hardware, this amount is $-70dBm$. This means that considering the constant parameters in our signal propagation model and the transmit power of the RNs, the result of the communication function $\lambda(i, j)$ between every two RNs that need to communicate should be more than $\frac{P_r}{KP_t}$. As $K = G(\frac{\lambda}{4\pi})^2 = 1 \times (\frac{0.125}{4\pi})^2 = 9.89 \times 10^-5$, then the threshold is : $\frac{1}{98946.46}W$. This value is $\frac{1}{9894.64}W$ for the links between an RN and SN. If this condition is not satisfied, then the link is a faulty link.

- **Fault Coefficient:** As the previous definition of a troubled link would not best reflect the quality of the communication link we present a second. If a link is faulty then to measure the amount of the fault compared to the threshold, we define the fault coefficient that is:

$$F_c = \frac{\frac{1}{\lambda(a,b)} - \frac{1}{threshold}}{\frac{1}{threshold}} \tag{6.1}$$

  As we only calculate this value for faulty links, the value is always positive and the closer this amount is to 0, the link is more reliable. This means that although the link is faulty, the received signal strength would be closer to the threshold value if F_c is closer to 0. The following example clarifies our definitions.

  Assuming that RN a needs to communicate to RN b in a single cell with path loss exponent of $\alpha = 3$, if the Euclidean distance between the two nodes is 50 meters, then the communication distance function would be calculated as:

$$\lambda(a, b) = (\frac{1}{50})^3 = 8 \times 10^{-8} \tag{6.2}$$

which is less than the $\frac{1}{98946.46}$ threshold. This means that the link between a, b is faulty. We can also calculate the received signal strength at b:

$$P_r = 10^{-1} \times (\frac{0.125}{4\pi})^2 \times (\frac{1}{50})^3 = 7.91 \times 10^{-11} \tag{6.3}$$

$$\tag{6.4}$$

This value is less than $-70dBm(10^{-10}W)$ and affirms that the link is faulty.

Now the fault coefficient of the link is calculated as follows:

$$F_c = \frac{125000 - 98946}{98946} = 0.26 \tag{6.5}$$

If the Euclidean distance between the two nodes was 70 meters, then this value would have been:

$$F_c = \frac{125000 - 98946}{98946} = 2.46 \tag{6.6}$$

By comparing the two values we can conclude that although both links are faulty, the fault coefficient of the condition where the Euclidean distance of a, b was $50m$, indicates that the received signal strength is closer to the threshold rather than when the distance between a, b is $70m$.

Sensor Node placement

We have two methods to deploy SNs in the first stage of our SUR algorithm. In our first method based on the algorithm proposed by Dhillon et al. [13] we deploy 634 SNs in the $1000m \times 1000m$ map. The sensors cover 70% of the map. The probability function that we use for the miss probability matrix is:

$$p_{ij} = e^{-0.05 \times d}$$

This means that cell i is able to cover j with the probability of $e^{-0.05 \times d}$. A cell is covered if its miss probability is less e_i is less than 20%. If two cells a, b cover a third cell c with a probability of 30% each, then the miss probability of cell c, will be $e_c = 0.49$. This algorithm completely spreads the SNs into the field in a way that neighbouring SNs would have minimal covered areas in common.

The other method that we used is random placement where we randomly deploy a specific amount of RNs into the field.

Other algorithms

To better study and compare the performance of our SUR algorithm, we create another algorithm *Assumed Alpha (AA)*, that does not have the second stage of SUR and jumps to the third stage. As the third stage requires an R and r for the ranges of the RNs and SNs, we assume a path loss exponent (for example 2.5) for the whole map and compute the range of the RNs and SNs if this path loss exponent was to be used. This means that for this algorithm, all the SNs would have the same radio range (around $100m$ if $\alpha = 2.5$ is assumed) and the RN-SN radio range for all the RNs would be the same (around $40m$ if $\alpha = 2.5$ is assumed).

Comparing our algorithm with AA(2.0) algorithm (that assumes the whole field has a path loss exponent of $\alpha = 2$) allows us to understand how estimating the average path loss exponent of the whole field would result in fewer or more faulty links and also how the value would influence the number of RNs in the field. Furthermore by comparing our algorithm with AA(2.5) on maps that have an average path loss exponent of 2.5, we are able to estimate the effect of the area transformation stage in our SUR-Average algorithm.

6.2.2 The performance of SUR in specific cases

To better understand how SUR operates and behaves, we designed maps in which the performance of the SUR algorithm could be tested. In our first experiment, to study the behaviour of our SUR algorithm on the RN-SN communication links, we create a map with depth $d = 1$, which means the θ grid divides the map into two cells. The left cell of this grid which has a path loss exponent of $\alpha = 2$ and the right cell has a path loss exponent of $\alpha = 3$. Each SN connects to its closest RN so we examine the best possible SN-RN link for each SN.

In Figure 6.5, the behaviour of AA(2.5) is shown. The SNs are depicted as small green nodes and the blue circles represent the assumed SN to RN communication distance. Also the RN-RN communication distance is orange. The base station is placed at $(0, 0)$ in the top left corner of the figure. The SNs that are not properly connected to an RN are shown as red nodes. As depicted in the top figure, the AA(2.5) algorithm assumes that the whole field has a path loss exponent of 2.5 and adjusts the range of the RNs accordingly. However the real map has different path loss exponents for each half. This means that the actual RN placement of the AA(2.5) algorithm would look like the bottom figure. In this figure many faulty links are created because of the misjudged RN placement. The total percentage of faulty links between the SNs and RNs in this placement is 37%.

In Figure 6.6 however, the SUR-Average algorithm works more intelligently because it places more RNs in the area that has a higher path loss exponent value because the range of the RNs is smaller in these areas. Also, there are fewer nodes in the area that has lower path loss exponent because the radio range of RNs is bigger in these areas. Although it does not perfectly solve the problem because it does not completely consider where the nodes go afterwards, the percentage of faulty links created by the SUR-Average algorithm is 26% less than the AA(2.5) placement algo-

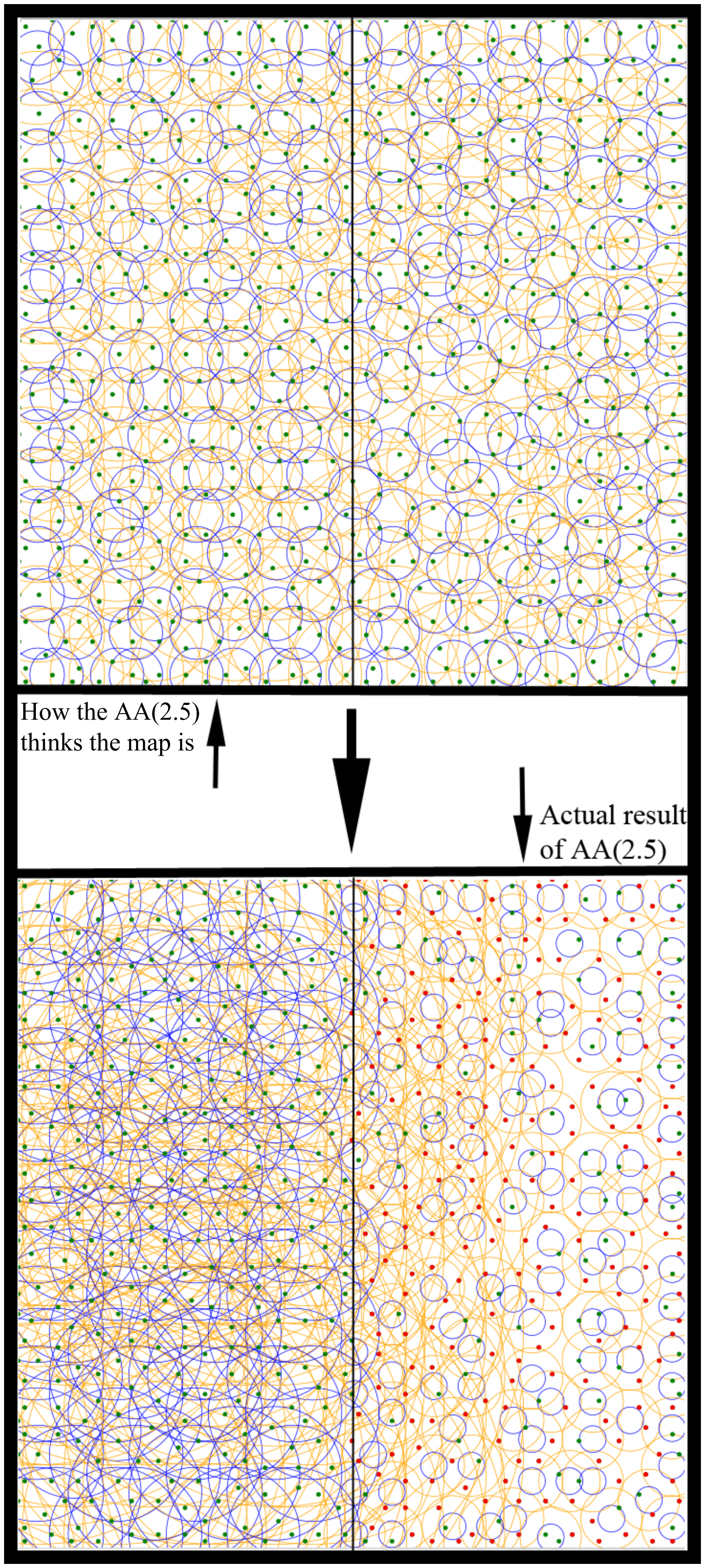


Figure 6.5: Node placement of the AA(2.5) algorithm in a sample map with depth 1.

rithm. Also the fault coefficient at the RNs is 1.29, which means that on average for faulty links, λ is 2.19 times the threshold. This value for AA(2.5) is 3.05. Based on the fault coefficient, we can conclude that in general, the SNs are placed closer to the RNs compared to the AA(2.5) method.

To understand the effects of our SUR algorithm on the RN-RN communication links, we design a $1000m \times 1000m$ map with depth $d = 3$ where a single SN is placed at point $(1000, 500)$ and the base station is located at point $(0, 500)$. In between these two nodes from $x = 250$ to $x = 750$, we consider an area that has a higher path loss exponent ($\alpha = 2.7$). In Figure 6.7 and 6.8 the shaded parts depict these areas. The other parts of the map have a path loss exponent of $\alpha = 2.3$. The AA(2.5) and SUR-Average algorithms should connect the single SN to the base station by deploying intermediate RNs. The orange radio ranges that cross border are again inaccurate due to the change in α.

As illustrated in Figure 6.7, AA(2.5) uses 11 RNs to connect the SN to the base station. However, as it does not consider cells with higher or lower path loss exponents, an RN is placed every 100 meters, and 4 out of the total 11 resulting intermediate links are faulty. These faulty links are depicted as red links in the figure. The average fault coefficient of these four faulty links is 2.71.

However as shown in Figure 6.8 our SUR-Average algorithm stretches the areas that have a higher path loss exponent so that more RNs are placed in these areas. However as the overall path loss exponent of the whole map is not calculated in the best way, again 11 RNs are deployed and still 4 links between the borders of the cells are faulty. Two of these four links have a very low fault coefficient (0.3) which means they are very close to the threshold. In an ideal deployment however, an additional RN would correct the faulty links of the SUR-Average deployment. The average fault coefficient of the four faulty links is 0.8, so based on what we see in the figures and the two comparing parameters, we can say that our SUR-Average algorithm has a better performance because it considers places that have higher path loss, and places more RNs inside them. However, it misjudges the locations somewhat and places the RNs just slightly over the border in the low path loss area. Also we can say that our algorithm might underestimate the number of RNs required in areas that have lower path loss exponents, and that this issue can also be solved by selecting a better average for the path loss exponent of the map.

Based on the observations presented in this section, we conclude that both selecting an appropriate value for the overall α of the map and moving points to suitable locations play an important role in having fault-less links. However the number of RNs used for the deployment should also be considered. In the next section we examine the performance of our algorithms in more general maps.

The effect of depth(d)

In our third experiment, we differentiate the depth of the map (2^d = number of the cells in the map) to observe the behaviour of our algorithm.

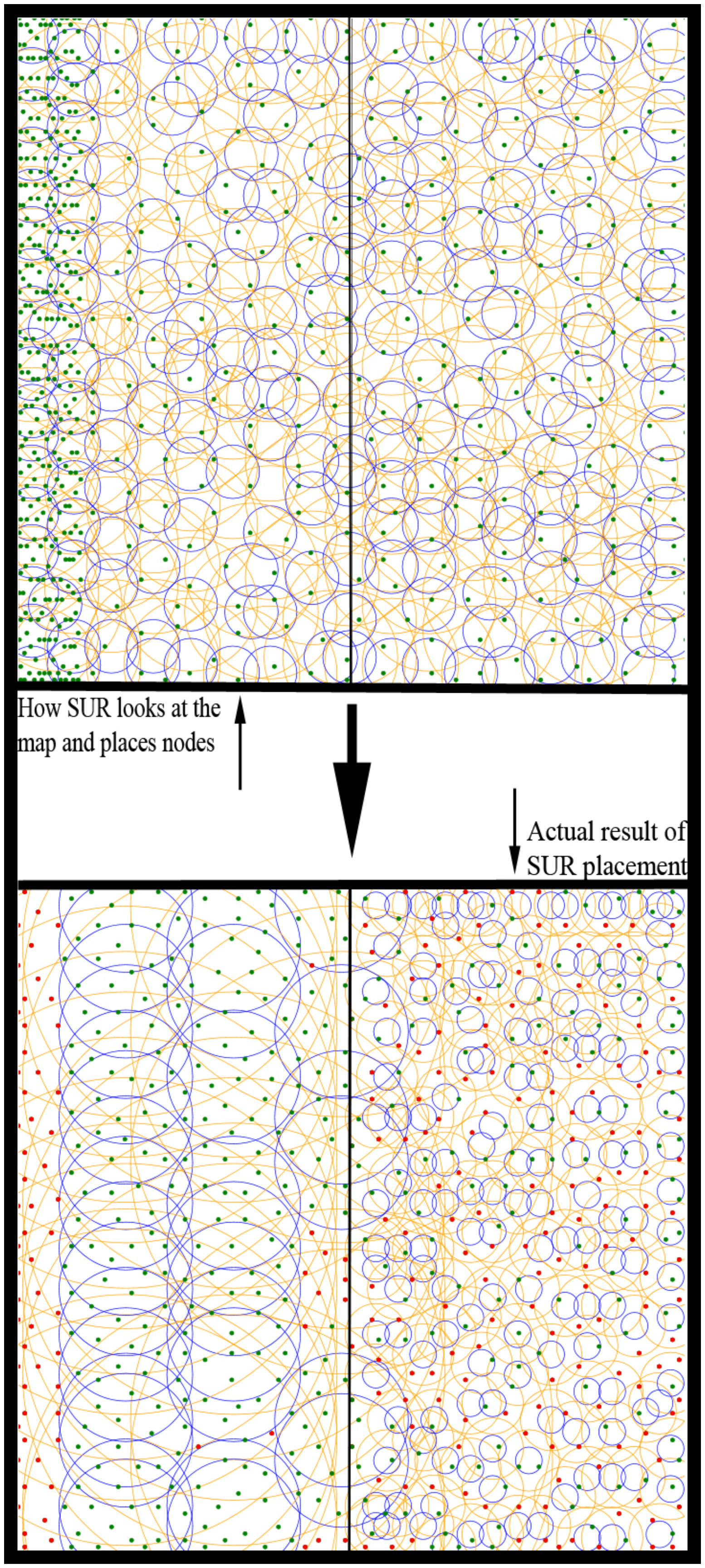

How SUR looks at the map and places nodes
Actual result of SUR placement

Figure 6.6: Node placement of the SUR-Average algorithm in a sample map with depth 1.

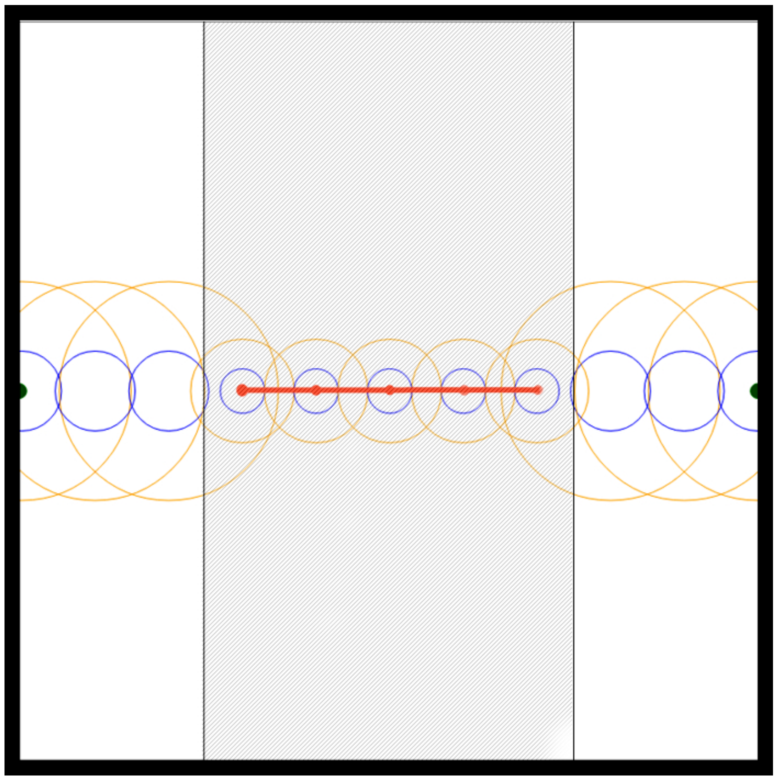


Figure 6.7: RN placement of the AA(2.5) algorithm in a sample map with depth 3.

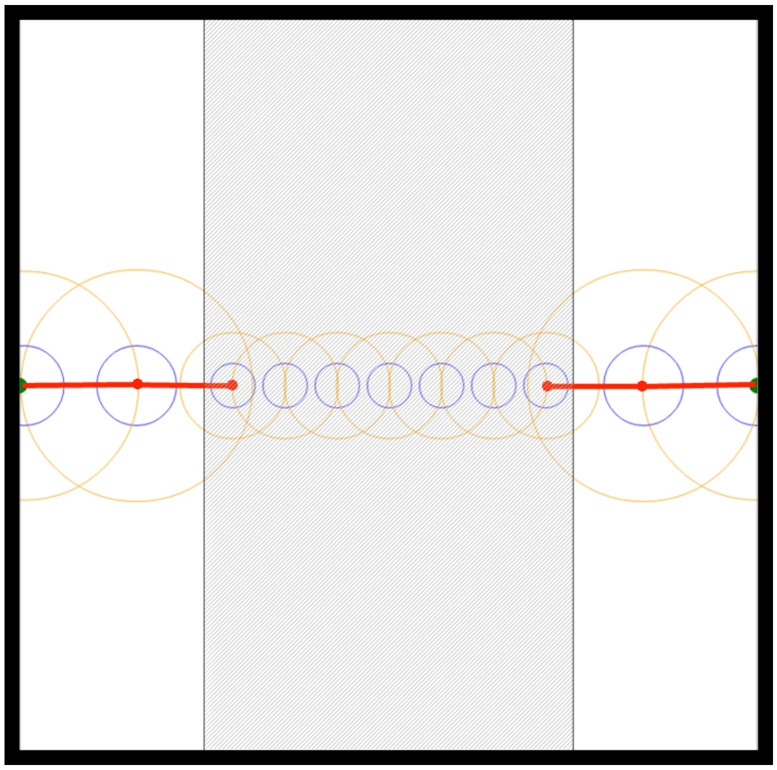


Figure 6.8: RN placement of the SUR-Average algorithm in a sample map with depth 3.

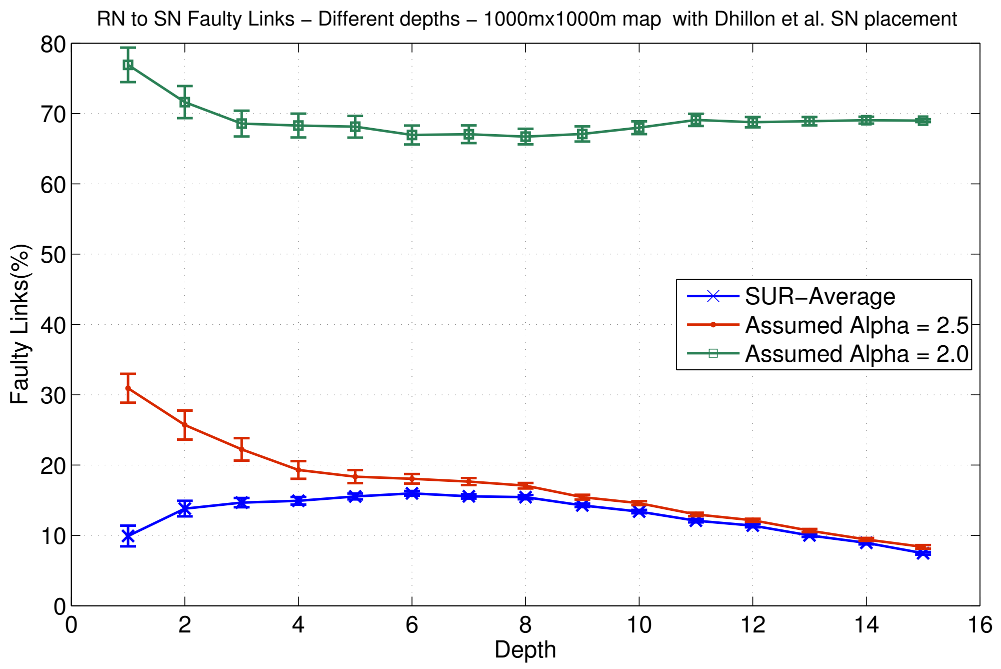


Figure 6.9: Percentage of faulty links of SN to RN links in a 1000x1000m^2 field with different depths.

We test each of our algorithms in 100 different random maps. In Figure 6.9, the communication links between RNs and SNs are examined. In Figure 6.9 the y-axis depicts the percentage of the faulty links while the x-axis depicts the depth.

Compared to situations where the "Rectangular Unification" is not used, in general, our SUR algorithm decreases the total number of faulty links between SN and RN. This difference is more visible in depths less than 10.

The green line depicts the errors of the AA(2.0) algorithm when the assumed path loss exponent is 2. This means that communication range is approximately 100 meters for RN to SN and 316 meters for RN to RN connections. As the diagram illustrates, this leads to large numbers of faulty links because the real path loss exponents of the cells are not 2. However the number of RNs placed by this algorithm is much less than the other two algorithms. The algorithm's performance gets slightly better because the probability that connections would be correct due to path loss exponent values (closer to 2.5), increases as the number of cells increases.

The red line represents the AA(2.5) algorithm when the assumed path loss exponent is 2.5. This time the error rates are much less but still more than our SUR algorithm. In average, for all depths less than 8, the average number of faulty links of SUR-Average is 14.8% while this value for the AA(2.5) is 21.5%. The reason that the errors of this algorithm decrease when the number of cells is increased, is that the average of the overall path loss exponent between each two cells and points in the $\lambda(i, j)$ formula are closer to 2.5 when we increase the number of cells and make them smaller.

With lower depths, the areas that have different path loss exponent values are bigger. As a result, the percentage of faulty RN-SN links is significantly higher due to misjudged average path loss exponent. Furthermore the bigger confidence intervals for lower depths are because when the depth is for example 1, then with a probability of 0.5 the cells will have path loss exponents less than the assumed 2.5 and with the same probability, have higher values. This leads to a higher or lower percentage of faulty links for lower values which means higher variance in the recorded values.

The performance of our SUR-Average algorithm is illustrated by the blue line. The reason that the algorithm has fewer faulty links in the very low depths ($1, 2$ for example) is that in these depths, the probability of having a total path loss exponent average of 2.5 is much more for other depths. This means that for example at depth $d = 1$ where we only have 2 cells, the probability that both path loss exponents of these two cells is more than 2.75 for example, is 4096 times more compared to when the depth is 3. In these situations, our SUR-algorithm selects more appropriate values for α (values near 2.75 for example) and the number of faulty links is reduced. When depth increases, these special beneficial situations for our SUR-average algorithm are reduced so the percentage of the faulty links is increased on average. However although the assumed average of the path loss exponent for the SUR-average also approaches 2.5, the algorithm performs better than the AA(2.5) algorithm because it moves and transforms points and places RNs in better locations accordingly. Similar to the AA(2.5) algorithm, the percentage of faulty links is further reduced at higher depths because the average path loss exponent between each two points is closer to 2.5. As shown in the figure, the error bars of the values of our SUR-Average algorithm are tighter than the other two algorithms and in general, we have fewer faulty links. For example for a depth of 3, which means an 8×8 communication grid (θ), the percentage of faulty links in the SUR-Average algorithm is 14.7% with a confidence interval of 0.61 while the AA(2.5) algorithm has an error rate of 22.2% with a confidence interval of 1.59 and AA(2.0) has higher errors of 66.62% with confidence intervals of 0.4.

In terms of the fault coefficient of the faulty links, we compare the values obtained by the SUR-Average and AA(2.5) algorithms in Table 6.3.

Table 6.3: Comparison of the fault coefficient of algorithms SUR-Average and AA(2.5)

Algorithm	d=1	d=2	d=3	d=4	d=5
SUR-Average	1.22	1.47	1.56	1.51	1.45
AA(2.5)	1.93	1.70	1.62	1.65	1.49

Algorithm	d=6	d=7	d=8	d=9	d=10
SUR-Average	1.39	1.33	1.46	1.54	1.63
AA(2.5)	1.41	1.35	1.60	1.73	1.79

Algorithm	d=11	d=12	d=13	d=14	d=15
SUR-Average	1.686	1.79	1.76	1.78	1.70
AA(2.5)	1.81	1.84	1.80	1.83	1.77

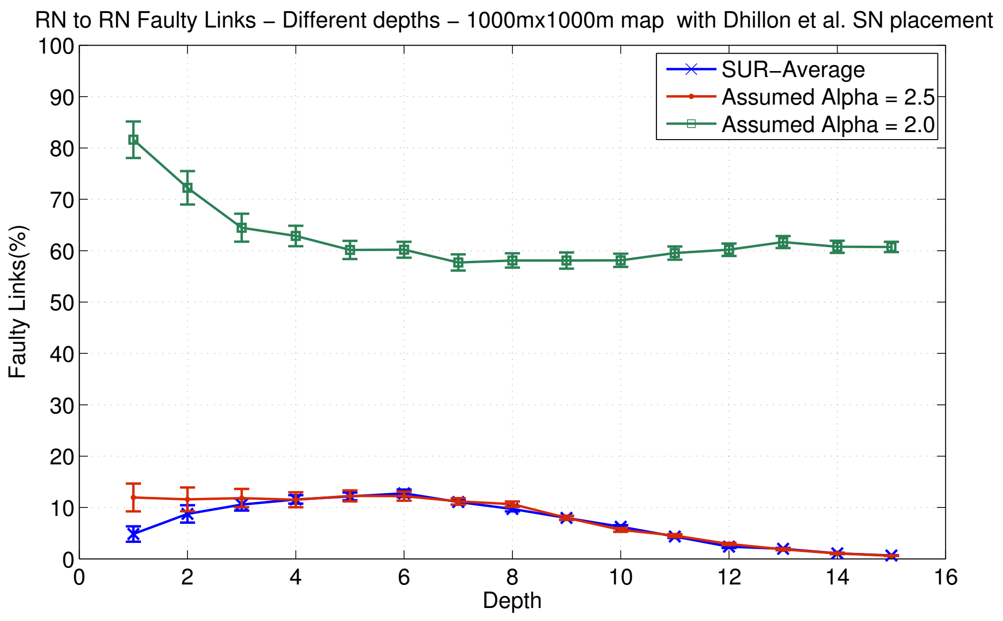


Figure 6.10: Percentage of faulty links of RN to RN links in a 1000x1000m^2 field with different depths.

As seen in the table, the fault coefficient of the faulty links have lower values for the SUR-Average algorithm which means that the faulty SNs are closer to RNs in the SUR-Average compared to AA(2.5).

Overall we can say that our SUR-Average algorithm is more beneficial both in terms of total number of faulty links and the fault coefficient of the faulty links. This is especially true in lower depths and in situations where there is difference between path loss exponent in neighbouring cells. Our algorithm also has an advantage of calculating more accurate values for the average path loss exponent of the whole map rather than just choosing or guessing a value as required for the AA algorithms.

In Figure 6.10, we illustrate the percentage of faulty links from RNs. As explained in Chapter 4, the final phase of the Chen and Cui RN placement algorithm is ensuring that every RN is connected to the base station with a direct link or through other RNs. Figure 6.10 examines the quality of these connections.

These figures indicate similar behaviour for the RN to RN communications as shown for the SN to RN communications depicted in Figure 6.9. Again we see a higher percentage of faulty links for AA(2.0) because of the very poor estimation of the average path loss exponent.

The performance of our SUR-Average algorithm in this figure again is close to the red line that represents AA(2.5). The same reasons that we stated for behaviour of the RN-SN faulty links in Figure 6.9, also stands here. The algorithms have lower error rates compared to the SN-RN

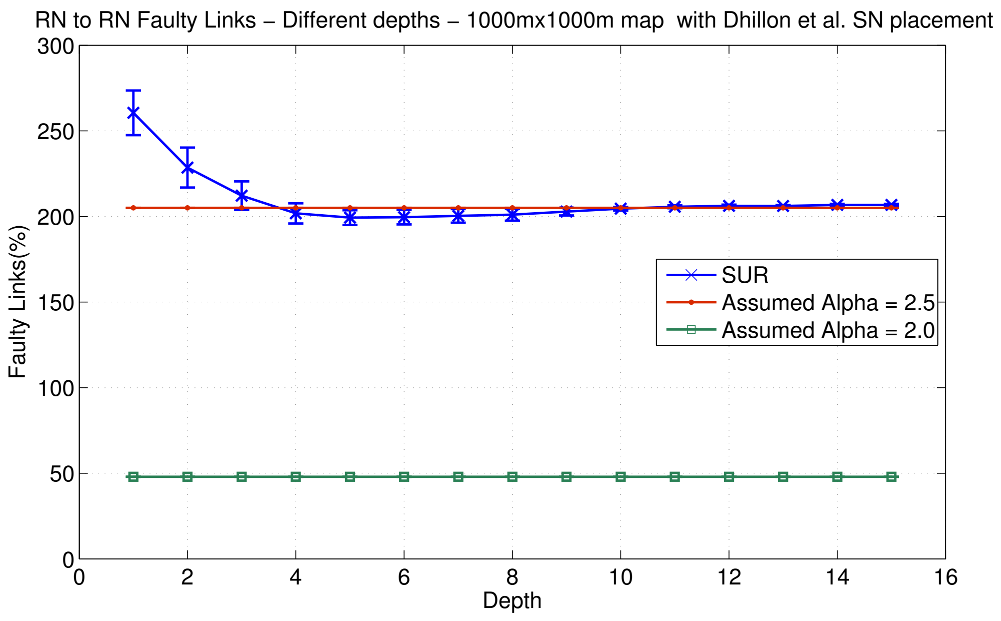


Figure 6.11: Number of RNs deployed in a 1000x1000m^2 field with different depths.

communication links because there are many RNs deployed due to the need to cover the 635 SNs, which means higher density of RNs.

In Figure 6.10, we depict the number of RNs required for the experiment.

Finally in Figure 6.11, we compare the number of RNs used for these three algorithms. As illustrated in this figure, the number of RNs for both the AA(2.0) and AA(2.5) algorithms is constant. This is because these algorithms consider the map to be uniform. In this figure AA(2.0) uses significantly fewer RNs but as explained in the previous two figures, this algorithm also has significantly more faulty links. Our SUR-Average algorithm however, uses more RNs at lower depths to reduce the percentage of faulty links. As the average path loss exponent between each two points and cells gets closer to 2.5, the number of RNs deployed also gets closer to the number of RNs used by the AA(2.5) algorithm.

6.2.3 The effect of number of SNs on the performance of the SUR algorithm

In this section, we compare the results of the SUR-Average algorithm to the AA(2.5) and AA(2.0) algorithms by varying the number of SNs deployed between 100 SNs to 900 SNs in the map. These SNs have been deployed randomly in the map so that the second and third stage of our algorithm could be tested on other SN deployments rather than just the Dhillon et al's SN placement algorithm. The depth of the map is 5 which means an 8×4 grid (θ).

In Figure 6.12 the percentage of faulty links created by the three different algorithms is observed. As depicted in this figure, the SUR-Average and AA(2.5) algorithms have a lower percentage of faulty links when a fewer number of SNs are deployed because the RN placement stage of the

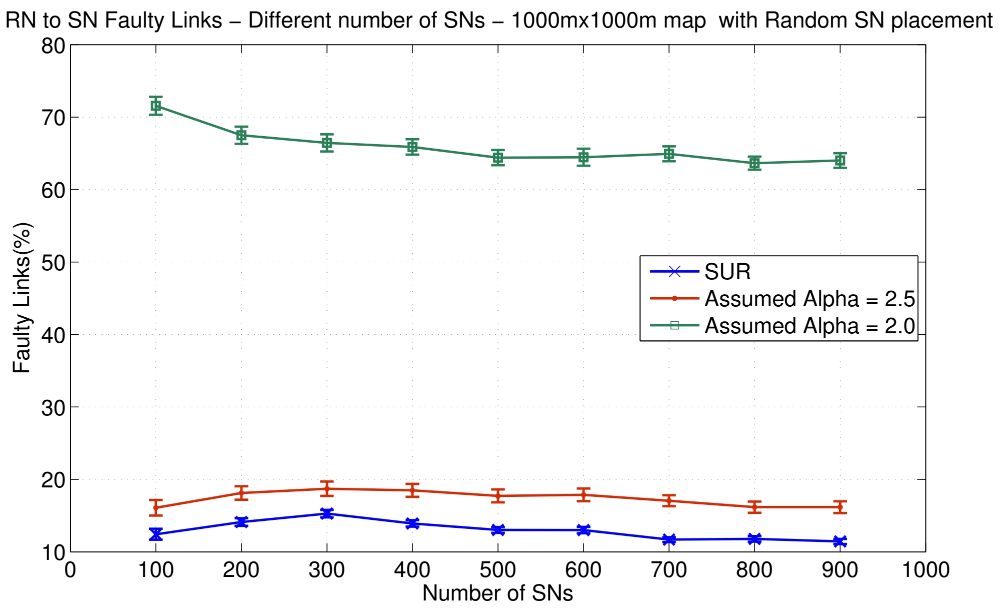


Figure 6.12: Percentage of faulty links of SN to RN links in a 1000x1000m^2 field with different number of SNs.

algorithms has to place an RN close to each SN because of the low density of the RNs in the map which results in fewer faulty links. However the AA(2.0) algorithm tries to cover multiple SNs with a single RN (because it assumes a bigger radio range for the RNs) which results in a higher percentage of faulty links.

The percentage of faulty links is slightly reduced at higher densities because when the number of SNs is increased, the density of RNs is also increased so with a higher probability, an SN can be covered by an alternative neighbouring RN.

On average our SUR-Average algorithm has 5.5% fewer faulty links than the AA(2.5) algorithm. And as Table 6.4 shows, the fault coefficient is also less for our algorithm which means that the SNs that have faulty links are closer to RNs compared with AA(2.5).

Table 6.4: Comparison of the fault coefficient of algorithms SUR-Average and AA(2.5)

Algorithm	SN=100	SN=200	SN=300	SN=400	SN=500
SUR-Average	2.06	2.01	1.88	1.83	1.82
AA(2.5)	2.04	1.99	1.99	1.96	1.95

Algorithm	SN=600	SN=700	SN=800	SN=900
SUR-Average	1.73	1.71	1.71	1.68
AA(2.5)	1.93	1.93	1.91	1.90

Furthermore in Figure 6.13, the percentage of the faulty links between the RNs is depicted. The

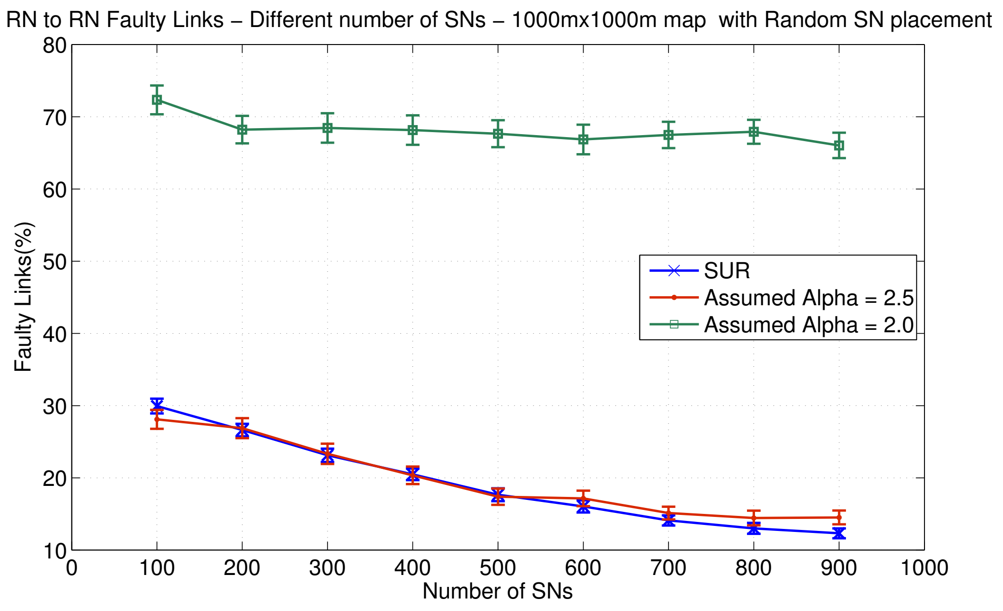


Figure 6.13: Percentage of faulty links of RN to RN links in a 1000x1000m^2 field with different number of SNs.

reason that the algorithms have more faulty links with fewer SNs is that in these depths intermediate RNs are needed to connect covering RNs to the base station. In the RN placement stage of the algorithms these additional RNs are placed on the radio range boundaries of RNs, so the likelihood that the links between these RNs is faulty, is increased. The AA(2.0) algorithm again has high error rates due to placing fewer RNs than needed to properly connect the RNs to each other and the base station. Similar to Figure 6.10, the SUR-Average and AA(2.5) algorithms have similar percentage of faulty links. At depth $d = 5$ which is the default depth of this experiments, we expected that these two algorithms would have similar behaviours in terms of RN-RN connections.

Finally in Figure 6.14, we present the number of RNs needed for this RN placement. As seen in this figure, additional RNs are needed to cover additional SNs. In this figure the number of RNs added in each stage is reduced because RNs are able to cover multiple SNs. After reaching a certain threshold based on the assumed radio range of the RNs, the map would be fully covered regardless of the number of SNs. For example in our SUR-Average algorithm an average of 19.5 additional RNs is needed to cover 200 SNs compared to 100 SNs. This value is reduced to 8 RNs between 800 and 900 RNs.

While the SUR-Average and AA(2.5) algorithm deploy similar number of RNs due to the same estimation of the path loss exponent of the field, the AA(2.0) utilizes significantly less RNs which leads to higher faulty links.

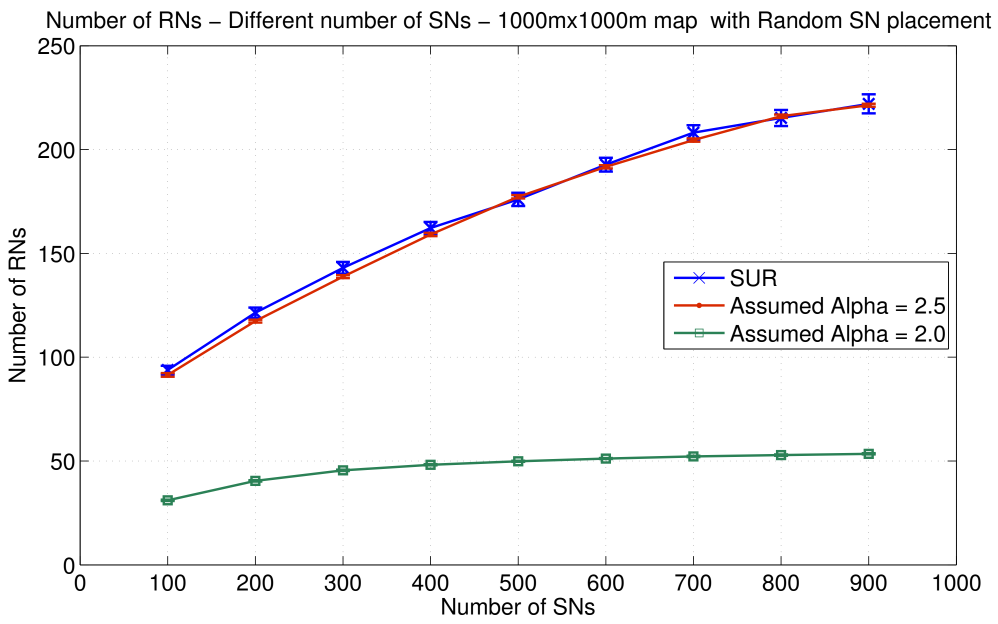


Figure 6.14: Number of RNs deployed by each algorithm in a 1000x1000m^2 field with different number of SNs.

Comparison between different variants of the SUR algorithm

In Chapter 4, we claimed that alternative methods can be used to find an suitable path loss exponent for two combining cells. In the previous experiments we compared the SUR-Average algorithm with AA(2.0) and AA(2.5) algorithms and argued that even with a simple combination averaging function, our method works better than the AA algorithm.

But our SUR-Average algorithm has its own limitations and as seen in Figure 6.8, it fails to appropriately select the number of RNs required because of its average path loss exponent estimation of the whole map. For example if we use the SUR-2WayPathLoss algorithm, for the same problem explained in 6.8, 14 RNs are used and the number of errors is reduced to 0.

In general the problem of reducing the number of faulty links can be solved using additional RNs, but the main challenge is using the minimum number of RNs that do so. Selecting an appropriate function that finds an suitable path loss exponent of two combining cells, gets us closer to a acceptable result.

We also test different variations of the SUR algorithm on random maps with different depths (constant number of SNs). Comparing the number of faulty links between the RNs and SNs, our results show similar behaviour between the SUR-Average and SUR-Sqrt algorithm but the percentage of faulty links of the SUR-Average is a 1% less than the SUR-Sqrt variation. Our tests also indicate that in average the SUR-2WayPathLoss has error rates close to 1.6% but also uses 1.8 times more RNs compared to SUR-Average to do so. Furthermore the SUR-1WayPathLoss algorithm has an average of 9.5% faulty links compared to 14.8 of the SUR-Average algorithm but it uses 1.3 times more RNs in average.

6.2.4 Summary

In this section we presented the results of our SUR algorithm. We displayed and explained its performance in different situations with different metrics. We emphasized the importance of selecting proper values for the average path loss exponent by comparing our SUR-Average algorithm with AA(2.0) to show that selecting an incorrect average may result in many faulty links. Furthermore, by comparing our algorithm with AA(2.5) we showed that placing nodes in proper locations also plays an important role in having fewer faulty links. Our SUR-Average algorithm aims to both select proper values for the average path loss exponent and place nodes at better locations by moving points to different locations. Our algorithm performs better in terms of total percentage of faulty links and the average fault coefficient of the faulty links. However, the performance of our algorithm can be further improved by introducing better functions in the ”Rectangular Unification” stage.

If we change the way we calculate the path loss exponent of two combined cells and in general the average path loss exponent of the whole map (rather than our simple average function), the number of RNs used might be increased or decreased. This change would reflect itself in the percentage of faulty links between nodes in the map. So based on the requirements of the problem, a better function would get us closer to the desired objective.

Chapter 7

Conclusions and future work

7.1 Problem model

In this thesis we studied the problem of node placement in a modelled real environment. First in Chapter 3 we suggested a simple model for signal propagation so that the fading of the signal in different terrains with different path loss exponents would be considered. We also modelled the map as a grid and set the different path loss exponent as a property of each cell. Our model can be improved in various ways.

- **Creating real maps:** Based on real environments, grid maps can be designed that divide the real map into smaller cells and considering the properties of each cell, appropriate path loss exponents can be associated.
- **Signal propagation model:** A more accurate and complete model of how a signal propagates through different terrains can be developed and tested in real environments.
- **Modelling uneven terrain:** Our current model only considers flat terrain, however this can be extended to model uneven terrain. The signal propagation model should also be adjusted accordingly.

7.2 Sensor-Unification-Relay (SUR) algorithm

This model was used to solve two different problems. In the first problem, the problem of covering a specific amount of area (which is given as input) with the minimum number of SNs and connecting them to the base stations through the minimum number of RNs was addressed. Using Dhillon et. al's algorithm [13] we satisfied the coverage requirement with the minimum number of SNs, and for RN placement we used the method proposed by Chen and Cui [9]. As these algorithms were designed to work in uniform areas (areas where the radio range of the nodes is the same in all directions), they work inaccurately in our modelled environment so we added a ”Rectangular Unification” stage to overcome this problem. The purpose of our ”Rectangular Unification” stage was to improve the quality of the links between nodes. We did so by transforming points from the previous non-uniform area to a new map with a uniform path loss exponent. Our results indicated that our ”Rectangular Unification” stage on average reduced the number of RN-SN and RN-RN faulty links and also decreased the fault coefficient of these faulty links compared to the situation where this stage was not used. In the results chapter we showed how our algorithm places more RNs in areas that have a higher path loss exponent and how it makes RN placement more efficient by using fewer nodes in areas that had lower path loss exponents and increased radio ranges. We also compared its performance in random maps and showed its ability to place the right number of RNs based on the field path loss exponents and also choosing more appropriate locations for RN placement. We argued that the way that we choose a new path loss exponent of neighbouring cells influences the overall estimated path loss exponent of the area and consequently results in deploying different numbers of RNs. We suggested different functions for finding the new path loss exponent.

The SUR algorithm can also be improved in various ways:

- **Testing other RN and SN placement algorithms:** As future work we plan to implement and test different deployment algorithms. In particular the performance of different RN placement algorithms in our modelled environment can be compared to each other.
- **Implementing our transformation algorithms on other types of maps:** Rather than only using rectangular maps, our transformation method should be modified in such a way that all maps with all kinds of shapes could be given to the framework as input.
- **Improving the transformation of points:** Better and more accurate functions can be used to move our points to desired locations. These functions do not necessarily need to be limited to a rectangular unification and new cells can have different shapes. Even a map to non-convex shapes with bends can be suggested, however the reverse transformation should also be considered.
- **Improving the path loss exponent estimation:** More sophisticated and accurate functions that better estimate a new path loss exponent based on how a signal propagates through the cell from different angles is also suggested as future work.
- **Using the idea of Ray Tracing:** In this work we noticed that the problem of placing nodes in a non-uniform environment is similar to the problem of refraction of light in different environments. The ideas and techniques used in ray tracing in physics can be utilized for node placement too. Using a algorithm based on ray tracing can replace our cartogram based method.

As mentioned in previous chapters, a perfect rectangular transformation is not necessarily achievable in two dimensions but improved transformations can be suggested and implemented. In general our idea allows previous suggested RN placement algorithms to work with better quality among links.

7.3 Max-cover 1-connected Algorithm

In Chapter 5 we solved the problem of maximizing the sensing coverage by also maintain connectivity among SNs to the base stations through RNs with a specific number of RNs and base stations. In this problem we proposed a three stage algorithm that consisted of greedy RN placement, simulated annealing improvement and an SN placement algorithm. We showed how our simulated annealing stage improves our greedy RN placement method in terms of candidate locations provided for the SN placement stage and also compared the performance of our algorithm with the optimal result that was obtained by an exhaustive search. Our algorithm had identical results compared to the optimal result for smaller test cases where the exhaustive search would return results in a feasible amount of time. Our results show that our algorithm can effectively find a high-quality solution and because of its suitable complexity, it is scalable and robust.

We suggest the following improvements as future work:

- **Using other cooling functions:** Other cooling functions that can better cool our system can be tested. In our model the cooling depends on the number of iterations. If we increase the number of iterations, more states would be examined in higher temperatures. Also the constant parameter in our cooling function could be tuned based on the size of the grid and number of RNs.

- **Sensor Placement:** Our current method can be improved if we create the triangular grids from different starting positions (with different shifts). This means that rather than aligning the $0, 0$ points of our ω and τ grids, we can shift the triangular grid by $\frac{S_r}{2}$ so that other SN deployments based on the shifted grid can be examined too.

- **Quantum annealing**: Rather than using thermal functions, quantum fluctuations can be utilized to find the global minima of our system. Because of our discrete search space, using this method may lead to faster and better results.

- **Stochastic tunneling:** To better escape local minima points, stochastic tunneling based on Monte Carlo sampling can be used. In this way instead of only moving to neighbouring positions, we are able to jump to new states.

- **Dynamic step size:** Rather than always keeping the step size of 1, dynamic step sizes can be used for moving to neighbouring nodes. If we are moving towards a better answer, we decrease the step sizes and if we are far from a good solution, we increase the step sizes.

- **Genetic algorithms:** We also propose testing the efficiency of other optimization algorithms such as genetic algorithms. These algorithms may lead us to better results in special conditions.

7.4 Further future work

- Examining the case of dense sensor deployments in relatively small areas ($100x100m$ plots) where only small numbers of RNs are economically feasible (say 1 to 5 RNs maximum).
- Including the constraint of the spatial autocorrelation of the biophysical variables being sensed in the sensor node placement algorithm
- Extending both methods to account for seasonal environmental variations that affect the radio propagation in the area being monitored
- Also implementing a interactive graphical user interface is suggested as future work.

One of the main objectives of this project was to take a different approach in solving the node placement problems in WSNs. We tried to look into different aspects of the problem especially from a more realistic point of view and also define objectives and problems based on real demands. We tried to fully utilize the current solutions for different subsets of the problem and create a framework that can be more effectively used in real applications. Because of the scale of our approach a lot of improvements that require more specification can be included. This project took a step in viewing the problem differently and used various methods from different fields of computer science, mathematics and engineering to make node placement more efficient and feasible. We proposed various methods for solving different problems in the our model.

Bibliography

[1] N. Aitsaadi, N. Achirt, K. Boussetta, and G. Pujolle. A Tabu Search Approach for Differentiated Sensor Network Deployment. In *CCNC IEEE Consumer Communications and Networking Conference*, pages 163–167, 2008.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422, 2002.

[3] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai. Deploying wireless sensors to achieve both coverage and connectivity. In *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '06, pages 131–142, New York, NY, USA, 2006. ACM.

[4] X. Bai, D. Xuan, Z. Yun, T. H. Lai, and W. Jia. Complete optimal deployment patterns for full-coverage and k-connectivity (k $\preceq$ 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, 2008. ACM.

[5] E. Biagioni and G. Sasaki. Wireless sensor placement for reliable and efficient data collection. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, page 10 pp., jan. 2003.

[6] K. Buchin, B. Speckmann, S. Verdonschot, K. Buchin, B. Speckmann, S. Verdonschot, P. Bose, S. Verdonschot, M. de Berg, A. Khosravi, et al. Evolution strategies for optimizing rectangular cartograms. In *the 7th International Conference on Geographic Information Science (GIScience 2012).*, pages 117–128, 2012.

[7] R. Caldeirinha and M. Al-Nuaimi. A novel fdtd based model for prediction of bistatic rcs of single leaves and trees. In *Antennas and Propagation, 2001. Eleventh International Conference on (IEE Conf. Publ. No. 480)*, volume 2, pages 783 –786 vol.2, 2001.

[8] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: application driver for wireless communications technology. *SIGCOMM Comput. Commun. Rev.*, 31(2 supplement):20–41, Apr. 2001.

[9] G. Chen and S. Cui. Relay node placement in two-tiered wireless sensor networks with base stations. *Journal of Combinatorial Optimization*, pages 1–10. 10.1007/s10878-012-9451-5.

[10] Y. Chen and Q. Zhao. On the lifetime of wireless sensor networks. *Communications Letters, IEEE*, 9(11):976 – 978, nov. 2005.

[11] X. Cheng, D.-Z. Du, L. Wang, and B. Xu. Relay sensor placement in wireless sensor networks. *Wireless Networks*, 14:347–355, 2008. 10.1007/s11276-006-0724-8.

[12] B. D. Dent, J. Torguson, and T. W. Hodler. *Cartography : thematic map design / Borden D. Dent, Jeffrey S. Torguson, Thomas W. Hodler*. McGraw-Hill Higher Education, New York :, 6th ed. edition, 2009.

[13] S. S. Dhillon and K. Chakrabarty. Sensor placement for effective coverage and surveillance in distributed sensor networks. In *Proc. of IEEE Wireless Communications and Networking Conference*, pages 1609–1614, 2003.

[14] M. H. C. Dias, A. Rotava, F. G. Andrade, R. A. Alem, M. A. K. Melo, and J. C. A. Santos. Path Loss Measurements of HF/VHF Land Links in a Brazilian Atlantic Rainforest Urban Site. *IEEE Antennas and Wireless Propagation Letters*, 10:1063–1067, 2011.

[15] A. Efrat, S. Har-Peled, and J. Mitchell. Approximation algorithms for two optimal location problems in sensor networks. In *Broadband Networks, 2005. BroadNets 2005. 2nd International Conference on*, pages 714 – 723 Vol. 1, oct. 2005.

[16] S. Gabriele and P. Di Giamberardino. Mobile sensors networks under communication constraints. *WTOS*, 7(3):165–174, Mar. 2008.

[17] M. T. Gastner and M. E. J. Newman. Diffusion-based method for producing density-equalizing maps. *Proceedings of the National Academy of Sciences of the United States of America*, 101(20):7499–7504, May 2004.

[18] B. Gudmundson and O. Grimlund. Handoff in microcellular based personal telephone systems. In S. Nanda and D. J. Goodman, editors, *Third Generation Wireless Information Networks*, chapter 12, pages 187–203. Kluwer Academic Publishers, Boston, 1992.

[19] X. Han, X. Cao, E. Lloyd, and C.-C. Shen. Fault-tolerant relay node placement in heterogeneous wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 9(5):643 –656, may 2010.

[20] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of The ACM*, 32:130–136, 1985.

[21] G. hui Lin and G. Xue. Steiner Tree Problem with Minimum Number of Steiner Points and Bounded Edge-Length. *Information Processing Letters*, 69:53–57, 1999.

[22] D. I. Inc. Xbee/xbee-pro zb rf modules data sheets, howpublished = `http://www.digi.com/pdf/chart_xbee_rf_features.pdf`.

[23] A. Kashyap, S. Khuller, and M. A. Shayman. Relay Placement for Higher Order Connectivity in Wireless Sensor Networks. In *IEEE INFOCOM*, pages 1–12, 2006.

[24] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220, 4598:671–680, 1983.

[25] S. Kumar, T. H. Lai, and J. Balogh. On k-coverage in a mostly sleeping sensor network. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, MobiCom '04, pages 144–158, New York, NY, USA, 2004. ACM.

[26] W. C. Y. Lee. *Mobile Communications Design Fundamentals*. John Wiley & Sons, Inc., New York, NY, USA, 2nd edition, 1992.

[27] F. Lekien and N. Leonard. Nonuniform coverage and cartograms. *SIAM Journal on Control and Optimization*, 48(1):351–372, 2009.

[28] J. Li, L. L. Andrew, C. H. Foh, M. Zukerman, and H.-H. Chen. Connectivity, coverage and placement in wireless sensor networks. *Sensors*, 9(10):7664–7693, 2009.

[29] B. Lin, P. han Ho, L. liang Xie, and X. Shen. Relay Station Placement in IEEE 802.16j Dual-Relay MMR Networks. In *IEEE International Conference on Communications*, pages 3437–3441, 2008.

[30] F. Lin and P. Chiu. A near-optimal sensor placement algorithm to achieve complete coverage-discrimination in sensor networks. *Communications Letters, IEEE*, 9(1):43 – 45, jan. 2005.

[31] E. L. Lloyd and G. Xue. Relay Node Placement in Wireless Sensor Networks. *IEEE Transactions on Computers*, 56:134–138, 2007.

[32] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava. Coverage Problems in Wireless Ad-hoc Sensor Networks. In *IEEE INFOCOM*, volume 3, pages 1380–1387, 2001.

[33] Y. S. Meng, Y. H. Lee, and B. C. Ng. STUDY OF PROPAGATION LOSS PREDICTION IN FOREST ENVIRONMENT. *Progress in Electromagnetics Research B*, 17:117–133, 2009.

[34] S. Misra, S. D. Hong, G. Xue, and J. Tang. Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirements. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 281 –285, april 2008.

[35] T. Nagakura, C. Ratti, X. Chen, et al. *Seeing differently: cartography for subjective maps based on dynamic urban data*. PhD thesis, Massachusetts Institute of Technology, 2011.

[36] J. O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, Inc., New York, NY, USA, 1987.

[37] J. Pan, L. Cai, Y. T. Hou, Y. Shi, and S. X. Shen. Optimal base-station locations in two-tiered wireless sensor networks. *IEEE Transactions on Mobile Computing*, 4:458–473, 2005.

[38] J. D. Parsons. *The Mobile Radio Propagation Channel, 2nd Edition*. Wiley, 2 edition, Nov. 2000.

[39] J. Polastre, R. Szewczyk, A. Mainwaring, D. Culler, and J. Anderson. Analysis of wireless sensor networks for habitat monitoring. *Wireless sensor networks*, pages 399–423, 2004.

[40] D. Pompili, T. Melodia, and I. F. Akyildiz. Deployment analysis in underwater acoustic wireless sensor networks. In *Proceedings of the 1st ACM international workshop on Underwater networks*, WUWNet '06, pages 48–55, New York, NY, USA, 2006. ACM.

[41] V. Ravelomanana. Extremal properties of three-dimensional sensor networks with applications. *IEEE Transactions on Mobile Computing*, 3(3):246–257, July 2004.

[42] N. C. Rogers, A. Seville, J. Richter, D. Ndzi, N. Savage, R. F. S. Caldeir-inha, A. K. Shukla, M. O. Al-nuaimi, K. Craig, E. Vilar, and J. Austin. A generic model of 1-60 GHz radio propagation through vegetation-final report. Technical report, Radiocommunications Agency, 2002.

[43] S. Shukla, N. Bulusu, and S. Jha. Cane-toad monitoring in Kakadu national park using wireless sensor networks. In *Networks Research Workshop*, 2004.

[44] A. So and B. Liang. Enhancing WLAN Capacity by Strategic Placement of Tetherless Relay Points. *IEEE Transactions on Mobile Computing*, 6:522–535, 2007.

[45] A. Srinivas, G. Zussman, and E. Modiano. Mobile backbone networks –: construction and maintenance. In *Mobile Ad Hoc Networking and Computing*, pages 166–177, 2006.

[46] G. Stüber. *Principles of mobile communication*. Springer Verlag, 2011.

[47] B. Suman and P. Kumar. A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of The Operational Research Society*, 57:1143–1160, 2006.

[48] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin. Habitat monitoring with sensor networks. *Commun. ACM*, 47(6):34–40, June 2004.

[49] J. Tang, B. Hao, and A. Sen. Relay node placement in large scale wireless sensor networks. *Comput. Commun.*, 29(4):490–501, Feb. 2006.

[50] E. Teller, N. Metropolis, and A. Rosenbluth. Equation of state calculations by fast computing machines. *J. Chem. Phys*, 21(13):1087–1092, 1953.

[51] D. Tian and N. D. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *Mobile Computing and Networking*, pages 32–41, 2002.

[52] B. Wang. Sensor Placement for Complete Information Coverage in Distributed Sensor Networks. *Journal of Circuits, Systems, and Computers*, 17:627–636, 2008.

[53] B. Wang. Coverage problems in sensor networks: A survey. *ACM Comput. Surv.*, 43(4):32:1–32:53, Oct. 2011.

[54] N. Wang, N. Zhang, and M. Wang. Wireless sensors in agriculture and food industryrecent development and future perspective. *Computers and Electronics in Agriculture*, 50(1):1 – 14, 2006.

[55] Q. Wang, K. Xu, H. Hassanein, and G. Takahara. Minimum cost guaranteed lifetime design for heterogeneous wireless sensor networks (wsns). In *Performance, Computing, and Communications Conference, 2005. IPCCC 2005. 24th IEEE International*, pages 599 – 604, april 2005.

[56] Y.-C. Wang, C.-C. Hu, and Y.-C. Tseng. Efficient placement and dispatch of sensors in a wireless sensor network. *IEEE Trans. Mob. Comput.*, 7(2):262–274, 2008.

[57] Wikipedia. Cartogram — Wikipedia, the free encyclopedia, 2012. [Online; accessed 29-July-2012].

[58] K. Xu, H. Hassanein, G. Takahara, and Q. Wang. Relay node deployment strategies in heterogeneous wireless sensor networks: single-hop communication case. In *Global Telecommunications Conference, 2005. GLOBECOM'05. IEEE*, volume 1, pages 5–pp. IEEE, 2005.

[59] K. Xu, Q. Wang, H. Hassanein, and G. Takahara. Optimal wireless sensor networks (wsns) deployment: minimum cost with lifetime constraint. In *Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005), IEEE International Conference on*, volume 3, pages 454 – 461 Vol. 3, aug. 2005.

[60] N. Xu. A survey of sensor network applications. *Survey Paper for CS694a Computer Science Department, University of Southern California*, 2002.

[61] D. Yang, X. Fang, G. Xue, and J. Tang. Relay Station Placement for Cooperative Communications in WiMAX Networks. In *Global Telecommunications Conference, . GLOBECOM . IEEE*, pages 1–5, 2010.

[62] M. Younis and K. Akkaya. Strategies and techniques for node placement in wireless sensor networks: A survey. *Ad Hoc Netw.*, 6(4):621–655, June 2008.

[63] H. Zhang and J. C. Hou. Maintaining Sensing Coverage and Connectivity in Large Sensor Networks. *Ad Hoc & Sensor Wireless Networks*, 1, 2005.

[64] J. Zhang, T. Yan, and S. Son. Deployment strategies for differentiated detection in wireless sensor networks. In *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on*, volume 1, pages 316 –325, sept. 2006.

[65] W. Zhang, G. Xue, and S. Misra. Fault-tolerant relay node placement in wireless sensor networks: Problems and algorithms. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1649 –1657, may 2007.

[66] X. Zhang and S. B. Wicker. How to distribute sensors in a random field? In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, IPSN '04, pages 243–250, New York, NY, USA, 2004. ACM.