SENSOR COVERAGE AND ACTORS RELOCATION IN WIRELESS SENSOR AND ACTOR NETWORKS (WSAN): OPTIMIZATION MODELS AND APPROXIMATION ALGORITHMS

A Thesis presented to
the Faculty of the Graduate School
at the University of Missouri

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

by
PRAVEEN K. NAGILLA
Dr. Esra Sisikoglu, Thesis Supervisor
DECEMBER 2010
The undersigned, appointed by the Dean of the Graduate School, have examined the thesis entitled:

SENSOR COVERAGE AND ACTORS RELOCATION IN WIRELESS SENSOR AND ACTOR NETWORKS (WSAN): OPTIMIZATION MODELS AND APPROXIMATION ALGORITHMS

presented by PRAVEEN K. NAGILLA,
a candidate for the degree of Master of Science and hereby certify that, in their opinion, it is worthy of acceptance.

__________________________

Dr. ESRA SISIKOGLU

__________________________

Dr. MUSTAFA Y. SIR

__________________________

Dr. TIMOTHY MATISZIW
DEDICATION

I dedicate this thesis to all my family members, especially to my MOM, without her support and love, the completion of this work would not have been possible.
ACKNOWLEDGMENTS

I would like to acknowledge Dr. Esra Sisikoglu (Adviser) for all her advice in developing the optimization models, supporting financially during this research and helping me to complete my thesis. I thank Dr. Mustafa Sir (Co-Adviser) for his encouragement in taking up this project initially, his guidance in developing the models and also for his assistance in learning different kind of computational languages required for this project.

I thank Dr. Timothy Matisziw (Committee Member) for sharing his thoughts regarding my thesis. I thank Dr. Kemal Akkaya for providing the WSAN problems. I thank Dr. Sarah Root, for modifying mixed integer models in Actor Relocation Problem and also for her help in preparing the slides during INFORMS Conference of 2009.

I would like to thank Industrial and Manufacturing Systems Engineering department at the University of Missouri, for providing financial assistance during the initial stage of my masters. Finally, I would like to thank Ms. Paula McDonald for all her administrative help throughout my master studies.
# TABLE OF CONTENTS

ACKNOWLEDGMENTS .......................................................... ii

LIST OF TABLES ............................................................ v

LIST OF FIGURES ........................................................... vi

ABSTRACT ................................................................. viii

CHAPTER .................................................................

1 Introduction ............................................................. 1

1.1 WSAN and WSN: Application Areas, Capabilities and Limitations . . 2

1.1.1 Wireless Sensor and Actor Networks .............................. 2

1.1.2 Wireless Sensor Networks ....................................... 4

2 Optimal Relocation of Actors to Restore Connectivity [Actor Re-
location Problem] ......................................................... 7

2.1 Introduction ............................................................. 7

2.2 Literature Review ..................................................... 8

2.3 Problem Description ................................................ 10

2.3.1 A Mixed-Integer Formulation .................................. 12

2.3.2 Using Powers of the Adjacency Matrix to Ensure Connectivity 13

2.4 A Polygon Approximation Scheme .................................. 17

2.5 Numerical Results for Single Connectivity ....................... 21

2.5.1 Heuristics to warm start the algorithm for Single Connectivity 29

2.6 Bi-Connectivity Formulation in WSAN ........................... 33

2.7 Numerical Results for Bi-connectivity ........................... 36

3 Minimum Cost Sensor Coverage Problem Over Continuous Area  
[Sensor Coverage Problem] .............................................. 40
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Coordinates of the actors shown in Figure 2.9   21</td>
</tr>
<tr>
<td>2.3</td>
<td>Optimal positions and n-sided regular polygon solutions for 5 actors 26</td>
</tr>
<tr>
<td></td>
<td>single connectivity</td>
</tr>
<tr>
<td>2.4</td>
<td>Initial Coordinates of the 4 actors              27</td>
</tr>
<tr>
<td>2.5</td>
<td>Intuitive and actual relocation positions for the given 4 actor positions 29</td>
</tr>
<tr>
<td>2.6</td>
<td>Initial, Heuristics and Optimal Positions of the 13 actors 33</td>
</tr>
<tr>
<td>2.7</td>
<td>Initial and optimal relocation positions of the T shape network 39</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Two actors that cannot communicate with each other</td>
<td>8</td>
</tr>
<tr>
<td>2.2</td>
<td>The two actors can communicate with each other</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>A connected WSAN. Even though two actors do not directly communicate with each other, they can do so through other actors in the network.</td>
<td>8</td>
</tr>
<tr>
<td>2.4</td>
<td>A simple WSAN consisting of two actors.</td>
<td>11</td>
</tr>
<tr>
<td>2.5</td>
<td>Replacing the smallest circle containing a vector with a regular polygon</td>
<td>18</td>
</tr>
<tr>
<td>2.6</td>
<td>Inside of a polygon as a feasible region</td>
<td>18</td>
</tr>
<tr>
<td>2.7</td>
<td>The percent area of a circle that is outside of a polygon contained in the circle as a function of the number of sides of the polygon.</td>
<td>19</td>
</tr>
<tr>
<td>2.8</td>
<td>Approximation for checking whether the distance between two vectors is less than a certain value.</td>
<td>20</td>
</tr>
<tr>
<td>2.9</td>
<td>Initial Position of the given 5 actors</td>
<td>22</td>
</tr>
<tr>
<td>2.10</td>
<td>Optimal relocation positions of the given actors</td>
<td>23</td>
</tr>
<tr>
<td>2.11</td>
<td>Relocation Position for 8-Sided Polygon</td>
<td>24</td>
</tr>
<tr>
<td>2.12</td>
<td>Relocation Position for 10-Sided Polygon</td>
<td>24</td>
</tr>
<tr>
<td>2.13</td>
<td>Relocation Position for 12-Sided Polygon</td>
<td>25</td>
</tr>
<tr>
<td>2.14</td>
<td>Relocation Position for 20-Sided Polygon</td>
<td>25</td>
</tr>
<tr>
<td>2.15</td>
<td>Initial positions for the 4 actors</td>
<td>27</td>
</tr>
</tbody>
</table>
ABSTRACT

Wireless Sensors and Actor Networks (WSAN) have a wide variety of applications such as military surveillance, object tracking and habitat monitoring. Sensors are data gathering devices. Selecting the minimum number of sensors for network coverage is crucial to reduce the cost of installation and data processing time. Actors in a WSAN are decision-making units. They need to be communicating with their fellow actors in order to respond to events. Therefore, the need to maintain a connected inter-actor network at all times is critical.

In the Actor Relocation Problem (Chapter 2) of this thesis we considered the problem of finding optimal strategies to restore connectivity when inter-actor network fails. We used a mixed integer programming formulation to find the optimal relocation strategies for actors in which the total travel distance is minimized. In our formulation we used powers of the adjacency matrix to generate constraints that ensure connectivity.

In the Sensor Coverage Problem (Chapter 3) we developed a mixed integer programming model to find the minimum number of sensors and their locations to cover a given area. We also developed a bi-level algorithm that runs two separate optimization algorithms iteratively to find the location of sensors such that every point in a continuous area is covered.
Chapter 1

Introduction

The increase in importance to establish a real time communication and decision making tasks in various application domains led to the development of Wireless Sensor and Actor Networks (WSAN), which are a union of wireless sensor networks and mobile ad-hoc networks (which consist of sensors and actors). Sensors are low-cost, low power devices with limited sensing, computation and wireless communication capabilities. Actors, on the other hand, are resource-rich nodes equipped with better processing capabilities, higher transmission powers and a longer battery life.

With the recent improvements in sensor capabilities and as well as the continuous expansion of sensor network application areas, the problems related to coverage by sensors and connectivity of actor networks have gained significant attention. This thesis, studies the optimization models and heuristics of two problems:

1. **Actor Relocation**: The actor relocation problem focuses on minimizing the total distance moved by the actors to establish a connected WSAN network

2. **Sensor Coverage**: The sensor coverage problem focuses on minimizing the number of sensors to cover a given surveillance area by finding optimal locations to install

This thesis is organized as follows; in the remainder of Chapter 1, application areas
and capabilities of Wireless Sensor Networks (WSN) and Wireless Sensor and Actor Networks (WSAN) are explained. In Chapter 2, literature review in the actor relocation problem is provided; then describe the single connectivity and bi-connectivity problems in WSAN, followed by a discussion of solution approaches and numerical results showing authenticity of the models. In Chapter 3, previous research in the sensor coverage problem is described, solution approaches and numerical results. Finally, conclude the thesis by summarizing the results for the problems in WSN and WSAN, and providing insight about future research possibility in this area.

1.1 WSAN and WSN: Application Areas, Capabilities and Limitations

1.1.1 Wireless Sensor and Actor Networks

In WSAN, the phenomenon of sensing is performed by sensors and acting\footnote{Acting means act of decision making} is done by the actors. In other words, actors are decision making units that collect information from the sensors and scrutinize it. Therefore, actors need to constantly communicate with sensors to collect data. They also need to communicate among themselves to make a collective decision. Having actor nodes around the sensors, makes a WSAN more efficient compared to WSN, which in most cases is inefficient because of the large distances between sensors and the sink (i.e. task manager) node.

The application areas of actors are more flexible than those of sensors, since sensors are not capable of making complex decisions. For example, in military surveillance applications, sensors (like security cameras and radars) are deployed to detect enemy troops; this information is spread out among the actors, whereupon actors communicate among themselves to destroy enemy troops. In another example, urban search
and rescue operations (Akkaya et al.), survivors of disasters such as fires or earthquakes may be in desperate need of oxygen, water or medical supplies. A WSAN would be valuable in this environment, since actors in the network can communicate and collaboratively decide how and where to deploy the sensors and actors to maximize the efficiency of rescue operations.

In order to perform the applications mentioned above, actors should be able to respond rapidly to sensor input; and there must be a constant sensor-actor and actor-actor communication. In WSAN, actors either make a decision among themselves or route data back to the sink, which then issues an action command to actors. These communications should consume low energy, support real time traffic and be dispensed so that the detection of the events is in the correct sequence to ensure proper action on the environment.

In WSAN, actors communicate among themselves as well as communicating with sensors. Communication among actors is required for the following reasons:

1. One actor may not be sufficient to perform the required action
2. If many actors receive same information from the sensors, then they may have to communicate with each other to perform the correct action
3. In some applications when multiple actors are covering same area, it is necessary to ensure that their actions are consistent
4. In case of military applications, it may be necessary to ensure multiple actors act on the environment at the same time
5. They can play the role of sensors by transferring data to each other if possible

In order to maximize the performance of a task, multiple actors must be employed in a WSAN. In a multi-actor networks, actors need to communicate among themselves to make to collective decision. To establish a connected network actors should relocate to establish an inter-connected\(^2\) network. While relocating they need to move as little as possible to minimize energy consumption.

\(^2\)To be connected with each other directly or indirectly
In Chapter 2, the importance of actor relocation problem and mathematical model for establishing a connected network will be explained.

1.1.2 Wireless Sensor Networks

Wireless Sensor Networks (WSN) consists of large number of sensors. Sensors are devices which can sense, measure and gather information like intensity of light, temperature, humidity, barometric pressure, etc, from their surroundings. After collecting the information, sensors send data to the decision makers like a central machine operated by a human or an actor (see Section 1.1.1). With the recent advancements in technology, the application area of the sensors and therefore sensor networks has significantly expanded. In the current generation of sensor networks, an on-board processor is installed to carry out simple computations and transmit the required data, rather than sending the raw data to the user.

Due to their compact size and ability to perform different sets of tasks; sensors are suitable for a wide variety of applications such as military surveillance, security camera coverage, habitat monitoring, object tracking, emergency medical services, supply chain and traffic monitoring, etc. In a military surveillance application studied by the He et al. [2004], sensors are used for tracking the movement of enemy troops and sending the collected information to the military command to aid them in making decisions in the hostile zones. In a habitat monitoring study, Mainwaring et al. [2002] considered the sensors which are deployed in a forest to detect foreign chemical agents in the air and water to provide localized measurements. In emergency medical services studied by Gaynor et al., wireless sensors can be used to send the data on a patient condition and vital signs to the hospital he/she is headed to, so that physicians and nurses could better anticipate the patient’s needs.

In order to perform the applications mentioned above, sensor networks require wireless ad-hoc networking techniques (Perkins [2000]). The differences between WSN
and Ad-Hoc\(^3\) networks are as follows:

1. The Number of nodes in sensor networks is much larger than that of ad-hoc networks
2. The topology of WSN changes quickly, unlike ad-hoc networks
3. WSN are more prone to failures; however, ad-hoc networks are robust
4. WSN are limited in power, computational capacities, and memory

Depending on the environment, sensors might have different resource constraints (Yick et al. [2008]), such as a limited amount of energy, shortened communication ranges, limited processing, storage capabilities and low bandwidths. Battery power is limited in terrestrial WSN (Akyildiz et al. [2002]); thus solar panels need to be installed to conserve energy. Establishing a communication network is very difficult for WSN underground (Li and Liu [2007]); since sensors that can communicate through soil, rocks, water and other mineral components are very expensive and cannot be recharged once they are out of energy. Moreover, WSN deployed underwater must be able to adapt to the harsh environmental conditions like tides and cyclones (Akyildiz et al. [2004]).

The number of sensors employed in WSN depends on the size of the monitoring area. Indoor environments require fewer sensors to cover limited space, whereas outdoor environments require more sensors to cover a large area. Dense sensor placement provides high degrees of coverage with less energy consumption; however, with a sparse sensor placement, energy consumption significantly increases. The degree of WSN coverage varies depending on the application domain and environment. A higher degree of coverage is important in the case of military tracking and security camera surveillance, since the majority of the area may need to be covered from different angles. For house and traffic monitoring, a lower degree of coverage would be acceptable. Research on coverage can be classified into a selection of the mini-

\(^3\)Ad-Hoc can be also referred to as Actors
minimum of sensors, relocation of sensors for restoring connectivity, energy conservation of sensors, etc. Here, selecting the minimum number of sensors can reduce the data redundancy and the cost of installation. Reducing the redundancy of data will reduce the processing and decision making time.

Chapter 3 will focus on Sensor Coverage Problem, which focus on finding minimum number of sensors to cover a given surveillance area entirely.
Chapter 2

Optimal Relocation of Actors to Restore Connectivity [Actor Relocation Problem]

2.1 Introduction

Actor relocation problem focuses on minimizing the total distance moved by the actors to establish a communication network among them. To enable communication, actors need to stay within the transmission range (distance up to which it can communicate directly) of each other. Figure 2.1 shows when actors cannot communicate and Figure 2.2 shows when they can communicate with each other. Figure 2.3 shows a connected inter-actor network. An actor failure can cause the loss of multiple inter-actor communication links and may partition the network if alternate paths among the affected actors are not available. Such a scenario will hinder the actors’ communication capabilities and cause the failure of WSAN. Therefore, it is necessary to maintain a connected inter-actor network at all times.
2.2 Literature Review

Most of the research work done on actor relocation problem focuses on developing an algorithm or heuristic to establish a communication network among actors. Melodia et al. [2005] presents a coordination framework for WSAN to decide which actors should respond to an event in a particular region. In this work, they considered inter-actor connectivity in the context of actor placement ignoring potential breaks in the network connectivity. Wang et al. [2005] studied sensor relocation in mobile sensor networks (which is similar to actor relocation problem), and developed an
algorithm for relocating mobile sensors in a timely, efficient, and balanced manner, and at the same time maintaining the original sensing topology as much as possible. In their algorithm, sensor relocation consists of two phases: the first phase is to find the redundant sensors in the sensor network; the second phase is to relocate them to the target location. In their model, they did not consider connectivity of the whole network.

Abbasi et al. [2007] considered the problem of relocating the actors to establish an inter-actor communication network while minimizing the total traveling distance and therefore reducing the energy overhead of the actors. They developed a heuristic approach called Distributed Actor Recovery Algorithm (DARA) which opts to efficiently restore the connectivity of a partitioned inter-actor network. Basu et al. [2004] studied the configuration of an ad-hoc network that can tolerate failures while allowing recovery. They developed algorithms for achieving bi-connectivity. In the bi-connectivity problem it is required to have at least two different paths between any two sensors.

It is not hard to see the importance of maintaining a connected WSAN considering the various application areas of sensors such as habitat monitoring and military services (see Section 1.1). To enable rapid response to potential natural disaster or enemy threat immediately it is crucial to have a continuous and reliable communication between the nodes of a WSAN. As the nodes of a network can randomly fail, we should be prepared to relocate sensors and actors to re-establish connectivity in the most efficient way. In this study, mixed integer programming model for relocating actors to establish single-connectivity\(^1\) and bi-connectivity\(^2\) using the concepts of adjacency matrix from graph theory are developed.

In the rest of the chapter, problem description and then model notation are explained. Next the mathematical models developed to solve single-connectivity and

\(^1\)Failure of one actor causes entire network to get disconnected  
\(^2\)Can sustain at most one actor failure
bi-connectivity are explained. Finally, this chapter is concluded with numerical results and conclusions section.

2.3 Problem Description

Suppose that we are given a WSAN with \( n \) actors and let \( \mathcal{A} = \{1, 2, \ldots, n\} \) be the set of indices of these actors. Let \( p_i = \begin{bmatrix} p^1_i \\ p^2_i \end{bmatrix} \) be the position coordinates and \( r_i \) be the transmission range of actor \( i, i \in \{1, 2, \ldots, n\} \). Our goal is to relocate actors in the WSAN network using as little total travel distance as possible to restore connectivity when the inter-actor network is partitioned due to a failure of an actor.

To gain some insight into the problem, consider the simple WSAN shown in Figure 2.4, which consists of two actors, indexed by 1 and 2. Where, \( p^1 \) and \( p^2 \) denote the original coordinates of actors, \( x^1 \) and \( x^2 \) denote the decision variables (i.e., coordinates of the actors after relocating). The transmission range of both actors is equal to \( r \). As can be seen in the figure, the distance between the two actors is given by

\[
d = \|p^1 - p^2\| = \sqrt{(p^1_1 - p^2_1)^2 + (p^1_2 - p^2_2)^2}.
\]
The optimal relocation problem for this simple WSAN is given by

$$\min_{x^1, x^2} \|x^1 - p^1\| + \|x^2 - p^2\|$$

subject to $\|x^1 - x^2\| \leq r$.

of actor 1 and actor 2, respectively.

It can be readily shown that the optimal relocation strategy is

- if $\|(p^1 - p^2)\| \leq r$, then $(x^1)^* = p^1$ and $(x^2)^* = p^2$ (i.e., we do not need to move the actors if they are already communicating with each other),

- if $\|(p^1 - p^2)\| > r$, then

$$
(x^1)^* = p^1 + \lambda (p^2 - p^1) \left(1 - \frac{r}{\|p^2 - p^1\|}\right)
$$

$$
(x^2)^* = p^2 + (1 - \lambda) (p^1 - p^2) \left(1 - \frac{r}{\|p^1 - p^2\|}\right),
$$

where $\lambda = [0, 1]$ (i.e., if the actors are outside of their respective transmission
ranges, all we need to do is to move them towards each other along the line
connecting them so that the distance between them becomes equal to $r$.

### 2.3.1 A Mixed-Integer Formulation

Let $x_i = \begin{bmatrix} x^1_i \\ x^2_i \end{bmatrix}$ be the position coordinates of actor $i$, $i = \{1, 2, \ldots, n\}$, after it is relocated.

The total distance traveled by all the actors during the relocation is given by

$$\sum_{i=1}^{n} \|x_i - p_i\| = \sum_{i=1}^{n} \sqrt{(x^1_i - p^1_i)^2 + (x^2_i - p^2_i)^2},$$

where $\| \cdot \|$ is the Euclidean distance.

Let for all $i, j \in A$, $i \neq j$,

$$y_{ij} = \begin{cases} 
1, & \text{if actor } i \text{ can communicate directly with actor } j \ (i.e., \ |x^i - x^j| \leq r_i), \\
0, & \text{otherwise.}
\end{cases}$$

Here, transmission ranges of all sensors are assumed to be same, i.e., $r_i = r_j$, therefore $y_{ij} = y_{ji}$.

Using the ideas from the Traveling Salesman cut-set formulation; a straightforward formulation can be written as follows:
\[
\min \sum_{i=1}^{n} \|x^i - p^i\| \tag{2.1}
\]

subject to
\[
\|x^i - x^j\| \leq r_j y_{ij} + M(1 - y_{ij}), \quad \forall i, j \in \mathcal{A}, \ i \neq j, \tag{2.2}
\]
\[
\sum_{i \in S} \sum_{j \notin S} y_{ij} \geq 1, \quad \forall S \subset \mathcal{A}, S \neq \emptyset, \tag{2.3}
\]
\[
x^i \in \mathbb{R}^2, \quad i \in \mathcal{A}, \tag{2.4}
\]
\[
y_{ij} \in \{0, 1\}, \quad i, j \in \mathcal{A}, \ i \neq j, \tag{2.5}
\]

where the constraint (2.2) determine whether an actor communicates directly with another actor; constraint (2.3) enforce that every possible partition of the network by nodes is connected with at least one node outside of the partition and \(M\) is a large positive number. Unfortunately, the number of constraints in this formulation is non-polynomial (there are \(2^{n-1}\) constraints in equation (2.3). Therefore, we need a more tractable formulation.

### 2.3.2 Using Powers of the Adjacency Matrix to Ensure Connectivity

**Definition 1** (Adjacency Matrix). Let \(Y\) be the adjacency matrix after the actors are relocated:
\[
Y = \begin{bmatrix}
y_{11}(1) & y_{12}(1) & \cdots & y_{1n}(1) \\
y_{21}(1) & y_{22}(1) & \cdots & y_{2n}(1) \\
\vdots & \vdots & \ddots & \vdots \\
y_{n1}(1) & y_{n2}(1) & \cdots & y_{nn}(1)
\end{bmatrix}
\]

**Definition 2** (Walk). A walk is an alternating sequence of vertices and edges, beginning and ending with a vertex. A graph is said to be connected if each pair of vertices is joined by a walk (Biggs [1993]).
**Theorem 1.** The \((i, j)\) entry of the \(k\)th power of the adjacency matrix \(Y\) is \(Y^k\), which equals to the number of walks of length (at-most \(k\) steps) between actor \(i\) to actor \(j\).

**Proof.** Refer to Pemmaraju and Skiena [2003]

**Corollary 1.** A WSAN with \(n\) actors is connected if and only if

\[
[Y^{n-1}]_{ij} \geq 1, \quad i \in \{1, 2, \ldots, n\}, \ j \in \{1, 2, \ldots, n\}.
\]

**Proof.** Where, \([Y^{n-1}]_{ij}\) is \((n-1)\)th power of Adjacency Matrix \(Y\). Therefore, the value of \((i, j)\) entry of \([Y^{n-1}]_{ij}\) will be at most equal to \((n-1)\) walks. From the definition of Walk, if there exists a walk between two vertices, then they both are connected. Therefore, if every entry in \([Y^{n-1}]_{ij}\) is greater than 1, which means there exits at least one walk between \(i\) and \(j\), then WSAN is connected.

Motivated by Corollary 1, an alternative nonlinear mixed-integer programming formulation can be written as

\[
\begin{align*}
\min & \quad \sum_{i=1}^{n} \|x_i - p_i\| \\
\text{subject to} & \quad \|x_i - x_j\| \leq r_j y_{ij}(1) + M(1 - y_{ij}(1)), \quad \forall i, j \in A, \ i \neq j, \\
& \quad [Y^{n-1}]_{ij} \geq 1, \ i \in \{1, 2, \ldots, n\}, \ j \in \{1, 2, \ldots, n\}, \\
& \quad x_i \in \mathbb{R}^2, \ i \in A, \\
& \quad y_{ij}(1) \in \{0, 1\}, \ i, j \in A, \ i \neq j.
\end{align*}
\]

Letting, for all \(i, j \in A, i \neq j\), and \(k \in \{1, 2, \ldots, n-1\},\)

\[
y_{ij}(k) = \begin{cases} 
1, & \text{if actor } i \text{ and actor } j \text{ can communicate using at most } \\
& k \text{ intermediary steps}, \\
0, & \text{otherwise},
\end{cases}
\]
an equivalent formulation is obtained:

\[
\begin{align*}
\text{min} & \quad \sum_{i \in \mathcal{A}} \|x_i - p_i\| \\
\text{s.t.} & \quad \|x_i - x_j\| \leq r_j \ y_{ij}(1) + M(1 - y_{ij}(1)), \quad \forall i, j \in \mathcal{A}, i \neq j, \\
& \quad y_{ij}(k) \geq y_{il}(k-1)y_{lj}(1), \quad \forall i, j, l \in \mathcal{A}, i \neq j \neq l, \\
& \quad k \in \{2, 3, \ldots, n-1\}, \\
& \quad y_{ij}(k) \leq \sum_{l \in \mathcal{A}\setminus\{i, j\}} y_{il}(k-1)y_{lj}(1), \quad \forall i, j \in \mathcal{A}, i \neq j, \\
& \quad k \in \{2, 3, \ldots, n-1\} \\
& \quad y_{ij}(n-1) \geq 1, \quad \forall i, j \in \mathcal{A}, i \neq j, \\
& \quad x_i \in \mathbb{R}^2, \quad \forall i \in \mathcal{A}, \\
& \quad y_{ij}(k) \in \{0, 1\}, \quad \forall i, j \in \mathcal{A}, i \neq j, k \in \{1, 2, \ldots, n-1\},
\end{align*}
\]

Here, objective function (2.11) minimizes the total distance moved by the actors from their initial positions. Constraint (2.12) determines whether an actor communicates directly with another actor; the constraints (2.13)–(2.15) capture multiplication of adjacency matrix and ensure connectivity. Constraints (2.13) and (2.14) make sure that \(y_{ij}(k) = 1\), when there exists a path from one actor to another actor through a neighbor actor within the \(k\)-th step. Finally, constraint (2.15) is needed to confirm that all nodes are connected in at most \((n-1)\) steps.

The constraints (2.13)-(2.14) involve multiplication of binary variables, which is non-linear and therefore are difficult to handle. We can linearize these constraints by defining the following additional variables, for all \(i, j, l \in \{1, 2, \ldots, n\}, i \neq j\), and \(k \in \{2, 3, \ldots, n-1\}\),
\[
v_{ij}(k) = \begin{cases} 
1, & \text{if actor } i \text{ and actor } j \text{ can communicate through actor } l \\
& \text{within exactly } k \text{ steps,} \\
0, & \text{otherwise.}
\end{cases}
\]

Using these variables, we obtain the following formulation, which is equivalent to the formulation (2.11)-(2.17):

\[
\min \sum_{i=1}^{n} \|x_i - p_i\| \tag{2.18}
\]

s.t. \[
\|x_i - x_j\| \leq r_j y_{ij}(1) + M(1 - y_{ij}(1)), \quad \forall i, j \in \mathcal{A}, \ i \neq j, \tag{2.19}
\]

\[
v_{ij}(k) \leq y_{il}(k-1), \quad \forall i, j \in \mathcal{A}, i \neq j, \text{ and} \\
\forall l \in \mathcal{A}, l \neq i, l \neq j, \ \forall k = \{2, 3, \ldots, n-1\} \tag{2.20}
\]

\[
v_{ij}(k) \leq y_{ij}(1), \quad \forall i, j \in \mathcal{A}, i \neq j, \text{ and} \\
\forall l \in \mathcal{A}, l \neq i, l \neq j, \ \forall k = \{2, 3, \ldots, n-1\}, \tag{2.21}
\]

\[
v_{ij}(k) \geq y_{il}(k-1) + y_{ij}(1) - 1, \forall i, j \in \mathcal{A}, i \neq j, \text{ and} \\
\forall l \in \mathcal{A}, l \neq i, l \neq j, \ \forall k = \{2, 3, \ldots, n-1\}, \tag{2.22}
\]

\[
y_{ij}(k) \geq v_{ij}(k), \quad \forall i, j \in \mathcal{A}, i \neq j, \text{ and} \\
\forall l \in \mathcal{A}, l \neq i, l \neq j, \ \forall k = \{2, 3, \ldots, n-1\}, \tag{2.23}
\]

\[
y_{ij}(k) \leq \sum_{l \in \mathcal{A} \setminus \{i,j\}} v_{ij}(k), \quad i, j \in \mathcal{A}, i \neq j, \ \forall k = \{2, 3, \ldots, n-1\}, \tag{2.24}
\]

\[
y_{ij}(n-1) = 1, \quad \forall i, j = \mathcal{A}, i \neq j, \tag{2.25}
\]

\[
x_i \in \mathbb{R}^2, \quad \forall i \in \mathcal{A}, \tag{2.26}
\]

\[
v_{ij}(k) \in \{0, 1\}, \quad \forall i, j, l \in \mathcal{A}, i \neq j \neq l, \ \forall k = \{2, 3, \ldots, n-1\}, \tag{2.27}
\]

\[
y_{ij}(k) \in \{0, 1\}, \quad \forall i, j \in \mathcal{A}, i \neq j, \ k = \{1, 2, \ldots, n-1\}, \tag{2.28}
\]

In the above formulation, constraints (2.20) - (2.22) make sure that \(v_{ij}(k) = 1\) if
\( y_{il}(k - 1) \) and \( y_{lj}(1) \) are both equal to ‘1’, and if any one of these variables equals to ‘0’ then \( v_{ilj}(k) \) will become equal to ‘0’. Constraints (2.24) and (2.25) make sure that \( y_{ij}(k) = 1 \), if \( v_{ilj}(k) = 1 \) (for any in \( l \in \mathcal{A} \)). Constraint (2.25) make sure that every element in \( y_{ij}(n - 1) \) is equal to ‘1’, this confirms that actor \( i \) is connected to actor \( j \) (\( \forall i, j \in \mathcal{A} \)).

In this formulation the objective function (2.18) and the constraints (2.19) involve Euclidean distance calculations and therefore they are nonlinear. Next, approximation scheme used to linearize Euclidean norms in our model is explained.

### 2.4 A Polygon Approximation Scheme

The Euclidean norm of a vector (i.e., its length) is equal to the radius of the smallest circle centered at the origin containing the entire vector. One approximation to the Euclidean norm can be obtained using the concepts explained in Earl and D’Andrea [2007], by replacing this circle with a regular polygon centered at the origin, as shown in Figure 2.5. In this approximation scheme, the Euclidean norm of a vector is approximated by the apothem\(^3\) of the smallest polygon containing the entire vector.

Let \( s \) be the number of sides of the polygon used in the approximation. Then;

\[
\mathbf{a}^h = \begin{bmatrix}
\cos \left( \frac{2\pi h}{s} \right) \\
\sin \left( \frac{2\pi h}{s} \right)
\end{bmatrix}
\]

represents the normal vector to side \( h, h \in \{1, 2, \ldots, s\} \)

---

\(^3\)The apothem of a regular polygon is a line segment from the center to the midpoint of one of its sides.
By considering the inside of a polygon as the feasible region (see Figure 2.6), the problem of finding the apothem of the smallest polygon containing a vector $\mathbf{x}$ can be easily modeled as a linear programming problem (LP):

\[
\begin{align*}
\min_{d} & \quad d \\
\text{subject to} & \quad (a^h)^T \mathbf{x} \leq d, \quad h \in \{1, 2, \ldots, s\} \\
& \quad d \geq 0.
\end{align*}
\]
Here, $d$ is the apothem of a smallest circle that can cover the given vector $\mathbf{x}$. Since apothem of a circle is smaller than radius of circle, there is a slight margin of error in the feasible region. Which is approximately equal to the difference between area of circle and area of the polygon with apothem $d^*$, where $d^*$ is the optimal solution of the model given above ($(2.29) - (2.31)$). In Figure 2.7, 'maximum percent error' \textit{vs.} 'number of sides of the polygon' is plotted using equation (2.32). From the figure it is evident that as the number of sides polygon increase, percentage of error decreases exponentially and for polygons with more than 30 sides, the error is negligible.

$$\text{Maximum Percent Error}(s) = \frac{\text{Area of Circle}(r) - \text{Area of Polygon}(s)}{\text{Area of Circle}(r)} \quad (2.32)$$

![Figure 2.7: The percent area of a circle that is outside of a polygon contained in the circle as a function of the number of sides of the polygon.](image-url)
Figure 2.8: Approximation for checking whether the distance between two vectors is less than a certain value.

In order to check whether the distance between two vectors, say \( \mathbf{x}' \) and \( \mathbf{x}'' \), is less than a certain value, say \( r \), a similar approximation scheme can be used for finding the largest polygon inside of a circle with radius \( r \) as shown in Figure 2.8. The apothem of this polygon is equal to \( r \cos \left( \frac{\pi}{s} \right) \). The problem of determining whether the distance between \( \mathbf{x}' \) and \( \mathbf{x}'' \) is less than \( r \) can be modeled as an integer programming problem:

\[
\begin{align*}
\max_y & \quad y \\
\text{subject to} & \quad (a^h)^T (\mathbf{x}' - \mathbf{x}'') \leq \left( r \cos \left( \frac{\pi}{s} \right) \right) y + M(1 - y) \\
& \quad y \in \{0, 1\},
\end{align*}
\]

where \( M \) is a very large number. Inequality (2.19) forces \( y \) to be '0', if the distance between \( \mathbf{x}' \) and \( \mathbf{x}'' \) is greater than '1'. If the optimal solution \( y^* = 1 \) then it infers that the distance between \( \mathbf{x}' \) and \( \mathbf{x}'' \) is less than \( r \). Otherwise, it infers that the distance is greater than \( r \).

Using the approximation schemes, the formulation given in (2.18) - (2.28) can be linearized as follows:
\[
\min \sum_{i=1}^{n} d_i \quad (2.36)
\]
subject to
\[
(a^h)^T (x^i - p^i) \leq d_i, \quad \forall i \in \mathcal{A}, \forall h = \{1, 2, \ldots, s\}, \quad (2.37)
\]
\[
(a^h)^T (x^i - x^j) \leq \left( r_j \cos \left( \frac{\pi}{s} \right) \right) y_{ij} + M(1 - y_{ij}),
\]
\[
\forall i, j \in \mathcal{A}, \ i \neq j, \forall h = \{1, 2, \ldots, s\}, \quad (2.38)
\]
\[
[y_{ij}: \text{Equations (2.20) - (2.28)]} \quad (2.39)
\]
\[
d_i \geq 0, \quad \forall i \in \mathcal{A}. \quad (2.40)
\]

Here (2.36) and (2.37) are used to linearize the objective function (2.18) and the constraint set (2.37) is used to linearize the inequality (2.19).

### 2.5 Numerical Results for Single Connectivity

To test optimization algorithm given in Section 2.4, different set of unconnected WSAN to relocate them to establish connectivity is considered. In the first set of experiments 5 actors are used, the coordinates and the transmission ranges of which are shown in Figure 2.9. Table 2.1 lists the initial coordinates of these actors. Note that, in these networks the transmission ranges of all the actors are chosen to be equal to two units. Therefore, actors 2, 3 and 4 are connected, while 4 and 5 are not connected. Note that, the solution of this small problem can easily be determined, therefore this example is used to validate the optimization model.

<table>
<thead>
<tr>
<th>Actor i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1^i$</td>
<td>-2</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>$p_2^i$</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 2.1: Coordinates of the actors shown in Figure 2.9
In order to form a connected inter-actor network all the actors should be connected directly or indirectly. The intuitive solution of this problem is to move either actor 1 or actor 2 along the straight line joining the positions of these two actors, (see Figure 2.4). We cannot move actor 2, since it gets disconnected from actor 3, so we should move actor 1 till it reaches the transmission range of actor 2.

In this solution, the distance moved by actor 1 will be,

\[
d_1 = \sqrt{(-2 - 0)^2 + (2 - 0)^2} - 2 = 0.828
\]

The new position of actor 1 is,

\[
\mathbf{x}_1 = \left(\frac{-2 \times 2 + 0 \times 0.828}{2.828}, \frac{2 \times 2 + 0 \times 0.828}{2.828}\right) = (-1.414, 1.414)
\]

Similarly, the closest actor to 5 is actor 4. Therefore, the distance it moves is,

\[
d_5 = \sqrt{(6 - 3)^2 + (0 - 0)^2} - 2 = 1
\]

New position of actor 5 would be,

\[
\mathbf{x}_5 = \left(\frac{3 \times 1 + 6 \times 2}{3}, \frac{0 \times 1 + 0 \times 2}{22.3}\right) = (5, 0).
\]

Figure 2.9: Initial Position of the given 5 actors
Note that, we can also move actor 4 by one unit instead of moving actor 5, since actor 4 will be still connected to actor 3. But, the total distance moved by the actors will be the same.

Figure 2.10: Optimal relocation positions of the given actors

Figure 2.10 shows the optimal relocation positions for the actors. Here, straight circles represent the transmission range of the corresponding actors and arrows show the direction in which actors have been relocated. And, dotted circles are used to represent the Euclidean distance that the actors moved in this solution. To validate the model given in Section 2.4, the model for the 5-actors problem is coded in AMPL and obtained the results using CPLEX solver.

For the polygon approximation, polygons with 8, 10, 12 and 20 sides are used. The results obtained for each of these experiments are shown in Figure 2.11 - 2.14 and their respective solutions are shown in Table 2.3.
Figure 2.11: Relocation Position for 8-Sided Polygon

Figure 2.12: Relocation Position for 10-Sided Polygon
Figure 2.13: Relocation Position for 12-Sided Polygon

Figure 2.14: Relocation Position for 20-Sided Polygon
Table 2.3: Optimal positions and n-sided regular polygon solutions for 5 actors single connectivity

<table>
<thead>
<tr>
<th></th>
<th>$i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initial Position</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p^1_i$</td>
<td>-2.00</td>
<td>0.00</td>
<td>2.00</td>
<td>3.00</td>
<td>6.00</td>
<td></td>
</tr>
<tr>
<td>$p^2_i$</td>
<td>2.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td><strong>Optimal Solution</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p^1_i$</td>
<td>-1.41</td>
<td>0.00</td>
<td>2.00</td>
<td>3.00</td>
<td>5.00</td>
<td></td>
</tr>
<tr>
<td>$p^2_i$</td>
<td>1.41</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>$d_i$</td>
<td>0.83</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td></td>
</tr>
<tr>
<td>$EDM$</td>
<td>0.83</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td></td>
</tr>
<tr>
<td>$\sum EDM = 1.83$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8-Sides Polygon</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p^1_i$</td>
<td>-1.02</td>
<td>0.00</td>
<td>1.85</td>
<td>3.00</td>
<td>4.85</td>
<td></td>
</tr>
<tr>
<td>$p^2_i$</td>
<td>1.59</td>
<td>0.00</td>
<td>0.06</td>
<td>0.00</td>
<td>0.48</td>
<td></td>
</tr>
<tr>
<td>$d_i$</td>
<td>0.98</td>
<td>0.00</td>
<td>0.15</td>
<td>0.00</td>
<td>1.15</td>
<td></td>
</tr>
<tr>
<td>$EDM$</td>
<td>1.06</td>
<td>0.00</td>
<td>0.16</td>
<td>0.00</td>
<td>1.24</td>
<td></td>
</tr>
<tr>
<td>$\sum EDM = 2.47$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>10-Sides Polygon</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p^1_i$</td>
<td>-1.45</td>
<td>0.00</td>
<td>1.90</td>
<td>3.00</td>
<td>4.90</td>
<td></td>
</tr>
<tr>
<td>$p^2_i$</td>
<td>1.24</td>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>-0.36</td>
<td></td>
</tr>
<tr>
<td>$d_i$</td>
<td>0.89</td>
<td>0.00</td>
<td>0.08</td>
<td>0.00</td>
<td>1.10</td>
<td></td>
</tr>
<tr>
<td>$EDM$</td>
<td>0.94</td>
<td>0.00</td>
<td>0.10</td>
<td>0.00</td>
<td>1.20</td>
<td></td>
</tr>
<tr>
<td>$\sum EDM = 2.24$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>12-Sides Polygon</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p^1_i$</td>
<td>-1.41</td>
<td>0.00</td>
<td>1.93</td>
<td>3.00</td>
<td>4.93</td>
<td></td>
</tr>
<tr>
<td>$p^2_i$</td>
<td>1.41</td>
<td>0.00</td>
<td>0.02</td>
<td>0.00</td>
<td>-0.29</td>
<td></td>
</tr>
<tr>
<td>$d_i$</td>
<td>0.80</td>
<td>0.00</td>
<td>0.07</td>
<td>0.00</td>
<td>1.07</td>
<td></td>
</tr>
<tr>
<td>$EDM$</td>
<td>0.83</td>
<td>0.00</td>
<td>0.07</td>
<td>0.00</td>
<td>1.11</td>
<td></td>
</tr>
<tr>
<td>$\sum EDM = 2.00$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>20-Sides Polygon</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p^1_i$</td>
<td>-1.41</td>
<td>0.00</td>
<td>2.00</td>
<td>3.00</td>
<td>4.98</td>
<td></td>
</tr>
<tr>
<td>$p^2_i$</td>
<td>1.41</td>
<td>0.00</td>
<td>0.00</td>
<td>3.00</td>
<td>0.16</td>
<td></td>
</tr>
<tr>
<td>$d_i$</td>
<td>0.82</td>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1.02</td>
<td></td>
</tr>
<tr>
<td>$EDM$</td>
<td>0.83</td>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1.03</td>
<td></td>
</tr>
<tr>
<td>$\sum EDM = 1.86$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In these figures straight polygons represent the approximate transmission ranges of actors, dotted line polygons are used to approximate Euclidean distance moved by the actors denoted by $d_i$. From these figures it is evident that the actors are moving to the corner of either the straight polygon or the dotted polygon. This is due to the well known result in linear programming which states “If an optimal value exists then the value must occur at one or more of the corner points of a feasible region (Luenberger and Ye [2008])”. In Figure 2.14, polygons with 20 sides/edges are used, therefore they look like circles. From the solutions shown in the Table 2.3, it is evident that as the number of sides of the polygons increase the solution obtained is improving and getting closer to the optimal solution of the original nonlinear model.
In the second set of experiments, 4 actors are used which are positioned as shown in Figure 2.15, coordinates of those actors are given in Table 2.4.

Table 2.4: Initial Coordinates of the 4 actors

<table>
<thead>
<tr>
<th>Actor i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1^t$</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$p_2^t$</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 2.15: Initial positions for the 4 actors

As can be seen in Figure 2.15; in this experiment actors 1 and 2 are connected and so are actors 3 and 4. However, actors 1 and 2 cannot communicate with 3 or 4. One could argue that an intuitive solution for this problem is to move actor 2 and 3 on horizontal line between actors 1 and 4 as shown in Figure 2.16. In this solution the total distance moved will be 5.66. Note that if we chose to relocate actors 1 and 4 on a horizontal line between actors 2 and 3, the total movement distance will still be the same due to the symmetry.
To confirm this intuitive solution, the approximate model ((2.36) - (2.40)) with 32-side polygon is implemented, so that the final solution will be almost equal to the actual optimal solution with negligible approximation error. As discussed earlier, increasing the number of sides of the polygons used for approximation reduces the approximation errors. Again used AMPL with CPLEX solver, to implement this example using the approximate model. A comparison of the intuitive solution and the solution from the approximate model are shown in Table 2.5. The results show that according to the approximate model solution the actors 1 and 4 are moved a little bit in the upward direction. Actors 2 and 3 are moved to get connected along an arc between actors 1 and 4 (see Figure 2.17). The total movement obtained with the approximate model was 4.96 units (significantly lower than intuitive solution which was 5.66 units). This example shows that the optimal solution of the relocation problem can be significantly better than an anticipated solution. Therefore, such optimization models are valuable to build better intuition to design heuristics for large scale problem instances,
Figure 2.17: 32-Sided polygon relocation position for 4-actors

Table 2.5: Intuitive and actual relocation positions for the given 4 actor positions

<table>
<thead>
<tr>
<th>i</th>
<th>( p_i^1 )</th>
<th>( p_i^2 )</th>
<th>( x_i^1 )</th>
<th>( x_i^2 )</th>
<th>( EDM )</th>
<th>( x_i^1 )</th>
<th>( x_i^2 )</th>
<th>( EDM )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.27</td>
<td>0.15</td>
<td>0.31</td>
</tr>
<tr>
<td>2</td>
<td>0.00</td>
<td>2.00</td>
<td>2.00</td>
<td>0.00</td>
<td>2.83</td>
<td>2.14</td>
<td>0.86</td>
<td>2.41</td>
</tr>
<tr>
<td>3</td>
<td>6.00</td>
<td>2.00</td>
<td>2.00</td>
<td>4.00</td>
<td>2.83</td>
<td>4.13</td>
<td>0.99</td>
<td>2.11</td>
</tr>
<tr>
<td>4</td>
<td>6.00</td>
<td>0.00</td>
<td>6.00</td>
<td>0.00</td>
<td>0.00</td>
<td>5.89</td>
<td>0.06</td>
<td>0.12</td>
</tr>
</tbody>
</table>

\[ \sum EDM = 5.66 \quad \sum EDM = 4.96 \]

2.5.1 Heuristics to warm start the algorithm for Single Connectivity

Large number of binary variables used in the MIP model makes it difficult to solve the relocation problem for larger networks. One way to decrease the solution time of the MIP model is to provide an initial feasible solution to the model. If we can provide a good starting point to the solver, then the MIP model is expected to find the solution for larger networks. To find a good starting point a heuristic method is developed, which finds a feasible solution for the given actors in no time.
Warm Start Heuristic

For the WSAN relocation problem the simplest heuristic would be to move all the actors towards the center, but that would be a bad idea since it does not take into account the actor ranges and their positions. Therefore, in order to make it more efficient the following algorithm steps are used, which can reduce the movement of actors while moving them towards the center.

Algorithm Steps

1. Calculate the average of all the actor positions along x and y directions and determine the center of these positions. We’ll refer to this points as ‘Center Point’

2. Maintain two data sets, old set and new set. Old set consists of initial actor positions; new set consists of new actor positions. Initially new set is empty.

3. Measure the distance of all the actors from their position to the center point. Rank the actors in the ascending order based on their distances from this point.

4. Select the lowest ranked actor and move it to the center point.

5. Remove the actor in consideration from old set and put it in the new data set, and record its new position coordinates. If old set is empty go to step 7.

6. Measure the distance between the actors in the old data set and actor that was added to the new data set in step 5. Select the closest actor from old data set. If its distance is greater than the transmission range, move it towards the actor from the new data set along the straight line that connects them. If the distance is less than the transmission range, do move the actor and set its new coordinates to be the original coordinates. Go to step 5.

7. Terminate the algorithm.
To realize the heuristic we run it for 13 actors whose positions are shown in Figure 2.18. The total movement distance obtained with this heuristic was 14.9 units. Figure 2.19 shows the relocation position for the 13 actors. From the figure we can see that the heuristic is first moving the actor that is closest to the center, and then it is moving the actor which is next closest to it, to the boundary of that actor. Thereby, all the actors are moving one after the other to whichever actor it is closest in its new positions until all the actors are connected.

Figure 2.18: Initial Position of 13 actors
Figure 2.19: Relocation position for 13 actors using the Heuristics

This solution is used to warm start the MIP model for the given 13 actor example, with transmission range approximated by polygons of 12 edges. The optimal solution is shown in Figure 2.20. Table 2.6 shows the initial positions, heuristic solution and approximate optimal solution coordinates for the 13 actor example. From this example, the total movement distance obtained with heuristics was 14.9 units, whereas the optimal solution was 13.1 units. Even though these values may seem close, the set of actors that were relocated according to heuristic solution was significantly different compared to the set of actors that were relocated in the optimal solution. The exact model could not able to find good feasible solution without warm start, but with warm start it found the optimal solution in approximately 86 hrs with a optimal gap of 0.0096%. However, the Gurobi solver has found the optimal solution within few hrs, but at the gap\(^4\) of 75% , and to make sure that it is the best solution (to reach 0% gap) it took lot of time.

\(^4\)A gap is a difference between the upper and lower bounds of the solution. If the gap is 0%, then it means the model has reached the best solution.
Table 2.6: Initial, Heuristics and Optimal Positions of the 13 actors

<table>
<thead>
<tr>
<th>i</th>
<th>( p_i^1 )</th>
<th>( p_i^2 )</th>
<th>( x_i^1 )</th>
<th>( x_i^2 )</th>
<th>EDM</th>
<th>( x_i^1 )</th>
<th>( x_i^2 )</th>
<th>EDM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3</td>
<td>4.8</td>
<td>0.30</td>
<td>4.80</td>
<td>0.00</td>
<td>3.7</td>
<td>5.9</td>
<td>3.5</td>
</tr>
<tr>
<td>2</td>
<td>2.5</td>
<td>2.3</td>
<td>2.50</td>
<td>2.30</td>
<td>0.00</td>
<td>2.5</td>
<td>2.3</td>
<td>0.0</td>
</tr>
<tr>
<td>3</td>
<td>1.0</td>
<td>4.1</td>
<td>1.03</td>
<td>4.08</td>
<td>0.03</td>
<td>1.0</td>
<td>4.1</td>
<td>0.0</td>
</tr>
<tr>
<td>4</td>
<td>2.2</td>
<td>3.1</td>
<td>2.77</td>
<td>3.10</td>
<td>0.57</td>
<td>2.9</td>
<td>4.0</td>
<td>1.2</td>
</tr>
<tr>
<td>5</td>
<td>3.5</td>
<td>1.5</td>
<td>4.77</td>
<td>3.11</td>
<td>2.05</td>
<td>3.5</td>
<td>2.2</td>
<td>0.7</td>
</tr>
<tr>
<td>6</td>
<td>5.1</td>
<td>8.5</td>
<td>6.01</td>
<td>4.68</td>
<td>3.92</td>
<td>4.8</td>
<td>7.5</td>
<td>1.0</td>
</tr>
<tr>
<td>7</td>
<td>6.8</td>
<td>8.7</td>
<td>6.40</td>
<td>6.65</td>
<td>2.09</td>
<td>6.8</td>
<td>7.6</td>
<td>1.1</td>
</tr>
<tr>
<td>8</td>
<td>7.5</td>
<td>9.3</td>
<td>7.16</td>
<td>8.49</td>
<td>0.87</td>
<td>8.4</td>
<td>6.4</td>
<td>3.0</td>
</tr>
<tr>
<td>9</td>
<td>4.9</td>
<td>9.5</td>
<td>5.34</td>
<td>9.30</td>
<td>0.48</td>
<td>4.9</td>
<td>9.5</td>
<td>0.0</td>
</tr>
<tr>
<td>10</td>
<td>10.8</td>
<td>3.2</td>
<td>7.92</td>
<td>4.09</td>
<td>3.02</td>
<td>10.8</td>
<td>3.2</td>
<td>0.0</td>
</tr>
<tr>
<td>11</td>
<td>10.9</td>
<td>0.7</td>
<td>10.90</td>
<td>0.70</td>
<td>0.00</td>
<td>10.9</td>
<td>1.2</td>
<td>0.5</td>
</tr>
<tr>
<td>12</td>
<td>11.2</td>
<td>3.5</td>
<td>9.87</td>
<td>3.74</td>
<td>1.34</td>
<td>9.6</td>
<td>4.8</td>
<td>2.1</td>
</tr>
<tr>
<td>13</td>
<td>11.4</td>
<td>1.7</td>
<td>11.08</td>
<td>2.13</td>
<td>0.54</td>
<td>11.4</td>
<td>1.7</td>
<td>0.0</td>
</tr>
</tbody>
</table>

\[ \sum \text{EDM} = 14.91 \quad \sum \text{EDM} = 13.1 \]

2.6 Bi-Connectivity Formulation in WSAN

The model given in Section 2.3.1, provides an optimal solution for relocating actors with minimum total movement distance to restore connectivity. However, in certain application domains, i.e. the task carried out by the actors is critical for the success
of the whole network, a more robust design may be desirable. For example, having an alternative connection path between any two actors on the network will help to make the network function without any disruption even if one of the actors fails and destroys connectivity. We refer to the problem of finding the new positions of the actors, such that there are at least two disjoint paths (i.e. the paths do not have any actors in common) exists between any two actors as the bi-connectivity problem in WSAN.

To solve the bi-connectivity problem an optimal mixed integer linear programming model is developed based on the single connectivity model from Section 2.4. In the bi-connectivity model single connectivity formulation is executed for 'n' times by removing actor $i$ each time, $\forall i \in A$. This is done to examine, whether remaining $(n - 1)$ actors will be still connected when an actor fails. Therefore, the resulting formulation is similar to that of single connectivity formulation except that we run it for $n$ times. Equations (2.41) - (2.52) represent the bi-connectivity formulation. The notation used in this formulation is same as the original relocation model except for the additional variable $v_{hilj}(k)$ defined as follows;

$$v_{hilj}(k) = \begin{cases} 
1, & \text{if sensor } i \text{ and sensor } j \text{ can communicate through sensor } l, \\
& \text{within } k \text{ steps, when sensor } h \text{ fails,} \\
0, & \text{otherwise.}
\end{cases}$$

$\forall h, i, l, j \in A, h \neq i \neq j \neq l, \forall k = \{1, 2, \ldots, n - 1\}$
min \sum_{i=1}^{n} d_i \quad (2.41)

s.t. \quad (a_m)^T(x_i - p_i) \leq d_i, \quad \forall i \in \mathcal{A}, \forall m = \{1, 2, \ldots, s\}, \quad (2.42)

(a_m)^T(x_i - x_j) \leq \left( r_j \cos \left( \frac{\pi}{s} \right) \right) y_{ij}(1) + M(1 - y_{ij}(1)),

\forall i, j \in \mathcal{A}, i \neq j, \forall m = \{1, 2, \ldots, s\}, \quad (2.43)

v_{hilj}(k) \leq y_{il}(k - 1), \quad \forall h, i, j, l \in \mathcal{A}, h \neq i \neq j \neq l \text{ and } \forall k = \{2, 3, \ldots, n - 1\}, \quad (2.44)

v_{hilj}(k) \leq y_{ij}(1), \quad \forall h, i, j, l \in \mathcal{A}, h \neq i \neq j \neq l \text{ and } \forall k = \{2, 3, \ldots, n - 1\}, \quad (2.45)

v_{hilj}(k) \geq y_{il}(k - 1) + y_{ij}(1) - 1, \quad \forall h, i, j, l \in \mathcal{A},

h \neq i \neq j \neq l \text{ and } \forall k = \{2, 3, \ldots, n - 1\}, \quad (2.46)

y_{ij}(k) \geq v_{hilj}(k), \quad \forall h, i, j, l \in \mathcal{A}, h \neq i \neq j \neq l \text{ and } \forall k = \{2, 3, \ldots, n - 1\}, \quad (2.47)

y_{ij}(k) \leq \sum_{l \in \mathcal{A} \setminus \{h, i, j\}} v_{hilj}(k), \quad h, i, j \in \mathcal{A}, h \neq i \neq j, \text{ and } \forall k = \{2, 3, \ldots, n - 1\}, \quad (2.48)

y_{ij}(n - 1) = 1, \quad \forall h, i, j = \mathcal{A}, h \neq i \neq j, \quad (2.49)

x_i \in \mathbb{R}^2, \quad \forall i \in \mathcal{A}, \quad (2.50)

v_{hilj}(k) \in \{0, 1\}, \quad \forall h, i, j, l \in \mathcal{A}, h \neq i \neq j \neq l,

\forall k = \{2, 3, \ldots, n - 1\}, \quad (2.51)

y_{ij}(k) \in \{0, 1\}, \quad \forall h, i, j \in \mathcal{A}, h \neq i \neq j, k = \{1, 2, \ldots, n - 1\}, \quad (2.52)

The equations in this model are similar to single connectivity model, Equations (2.41)-(2.43) are similar to (2.36) - (2.38) and Equations (2.44) - (2.52) are similar to (2.20) - (2.28), except for their indices.
2.7 Numerical Results for Bi-connectivity

To test the bi-connectivity model, two test problems are designed with three and four actors.

The initial locations of actors for our first test problem is given in Figure 2.21. In this example all actors are connected; however if actor-2 fails then actor 1 and 3 will not be able to communicate. Our aim is to relocate these actors, so that there are at least two unique paths between any two actors. The intuitive solution of this problem would be to move either actor-1 or actor-3 to the coordinate (2,0) with a total movement distance of 2 units. To test the bi-connectivity model, 32-sided polygons is used for Euclidean norm approximation and run the model in AMPL with CPLEX solver. The final positions of the actors obtained in the solution are shown in Figure 2.22. According to this solution, actor-1 is moved towards actor-2 (consistent with our intuitive solution) with a total movement distance of 2.01 units. This difference is caused by the approximation scheme we use for calculating Euclidean distances.

Figure 2.21: Initial position of the actors which are connected
Figure 2.22: Optimal bi-connectivity relocation positions for the actors approximated 32-sides polygon

For our second test case, a network with 4 actors are used as shown in Figure 2.23, which are positioned in T shape. It can easily be seen from the figure that the network is connected, but if actor-2 fails then the entire network gets disconnected. To find the final positions of actors from the bi-connectivity model 32-side polygons is used for the approximation scheme and run the model in AMPL with CPLEX solver. The results of the bi-connectivity model are shown in Figure 2.24. From the figure, it can easily be seen that if actor-1 fails then actors 2, 3 and 4 are still connected, same is the case for all the actors.
Figure 2.23: Initial position for actors connected in T-shape

Figure 2.24: Optimal relocation position for actors connected in T-shape
Table 2.7: Initial and optimal relocation positions of the T shape network

<table>
<thead>
<tr>
<th>i</th>
<th>$p_i^1$</th>
<th>$p_i^2$</th>
<th>$x_i^1$</th>
<th>$x_i^2$</th>
<th>$EDM$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>0.236</td>
<td>-0.13</td>
<td>0.27</td>
</tr>
<tr>
<td>2</td>
<td>2.00</td>
<td>0.00</td>
<td>2.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>4.00</td>
<td>0.00</td>
<td>3.76</td>
<td>-0.13</td>
<td>0.27</td>
</tr>
<tr>
<td>4</td>
<td>2.00</td>
<td>-2.00</td>
<td>2.00</td>
<td>-1.07</td>
<td>0.93</td>
</tr>
</tbody>
</table>

$\sum EDM = 1.47$
Chapter 3

Minimum Cost Sensor Coverage Problem Over Continuous Area [Sensor Coverage Problem]

3.1 Introduction

WSN’s are used for a wide variety of applications, such as coverage, localization, tracking, synchronization and data aggregations. Coverage is an important application among them, as it is a measure of quality of service. The sensor coverage problem focuses on identifying the positions of the sensors, which can observe the surveillance area efficiently and effectively. In order to be efficient, the minimum number of sensors should be installed. To be effective, sensors should be positioned in such a way that every point in the surveillance area will be covered by at least one sensor.

3.2 Literature Review

In this section, we will discuss previous research that has been done related to the sensor coverage problem. The sensor coverage problem has its relevance to problems
in many other fields, such as art galleries, geographic information systems (GIS), ocean coverage, and coverage in robotic systems.

In an art gallery setting (ORourke [1987]), the goal is to determine the minimum number of guards that are sufficient to cover the interior of an art gallery. This problem has a linear solution in 2D; for 3D cases the problem becomes non-linear. Ghosh [2009] explained different approximate algorithms for art gallery problems in polygons.

Meguerdichian et al. [2001] studied best and worst case coverage in deterministic and stochastic coverage problems surrounding WSN. Worst case coverage involves quantifying the quality of service by finding areas of lower observability and detecting the breaching regions. In best-case coverage, areas of high observability will be located in order to identify the best support and guidance regions. This study analyzed the performance of algorithms and heuristics, based on Voronoi diagrams and graph theoretical concepts. Dhillon and Chakrabarty [2003] proposed a probabilistic approach to optimize the number of sensors in a distributed sensor network (where initially the sensors are deployed randomly). They considered uncertainty associated with sensor detections by discretization of the area into grid points.

Xing et al. [2009] explains the performance of the sensing ranges of sensors in WSN by using a stochastic data fusion model, which is useful in testing whether the sensors are making proper detections or not. Zou and Chakrabarty [2004] explained the coverage problem for distributed sensors networks (using the virtual force algorithm) and developed a probabilistic localization algorithm that can reduce the energy consumption for target detection and location. Huang and Tseng [2005] determined whether every location in the service area is monitored by at least \( k \) sensors, by looking at the coverage of the perimeter of each sensor’s sensing range, rather than looking at the coverage of area. They claimed that the whole area would be sufficiently covered if the perimeters of sensors were sufficiently covered. Chakrabarty et al. [2002] considered
a sensor field with grids and developed an integer program to minimize the cost of sensors, while ensuring that all the target points in the surveillance area are covered. However, they did not consider the coverage of a continuous surveillance area.

In Geographic Information Systems (GIS) field, Coverage Problem focus on identifying the facility locations for sitting (installing) the sirens, in order to cover the demand points (populated places) in the surveillance area. Here, a major distinction has been between discrete and continuous (or planar) sitting methods. Church and ReVelle [1974] has developed maximal covering location problem (MCLP), which cover's maximum demand points in the continuous space using limited resources. However, demand cannot always be represented in the form as points. There is also a need for modeling approaches capable of serving lines or polygons, as suggested by Miller [1996]. In order to answer those problems, Murray and Tong [2007] focused on facility placement in the continuous plane, using the polygon intersection point set (PIPS), which is an extension of the point-based approach discussed in Church [1984]. Murray et al. [2008] developed a non-linear model that maximizes regional coverage with limited resources. They have developed a geo-computational approach to solve their model. Matisziw and Murray [2009b] discussed about approach for covering continuously distributed demand in a continuous facility location with a single service facility. Matisziw and Murray [2009a] developed a heuristic approach for sitting a facility in any shape of continuous service area (transmission range).

Installing large numbers of sensors will increase the installation costs and increase the size of data for scrutinization, a process that can be very tedious and complex. Therefore, finding the optimal number of sensors for covering the entire area will reduce the cost of installation of sensors and reduce the complexity of the gathered data. Finding an efficient algorithm to minimize the number of installations of sensors while at the same time covering the entire surveillance area can be crucial for designing WSN. In our research, we developed a mixed integer programming coverage
model (MIPCM) to find the minimum number sensors and their locations to cover a continuous, closed and convex surveillance region, using a two stage bi-level algorithm called master-problem and sub-problem.

### 3.3 Sensor Coverage Problem in WSN

Consider a network of wireless sensors deployed in a rectangular area $A$. Let $r_i$ denote the sensing range of sensor $i$ in the network. Here, we consider a sensor coverage problem such that the continuous region $A$ is covered, while the total number of sensors deployed in the network is minimized. Some of the optimization models on sensor coverage problem discretized the region $A$ and solve an integer programming model to find the location of sensors. Simply discretizing fails to address the actual problem since the region in consideration is continuous. To find the best deployment strategy in a continuous region, we designed a bi-level algorithm that runs in two stages which are referred as *master-problem* and *sub-problem* in the rest of the chapter.

We initially discretize the region $A$ and use the master-problem to find the minimum number of sensors, which cover all the discrete points identified in the region. Next, we run the sub-problem to find a point in the continuous region which is not covered by the sensors located according to the solution of the master-problem. Both master and sub problems are run iteratively; every time the master-problem finds optimal sensor locations, sub-problem is run to check if there are any uncovered points in the continuous region $A$. If it finds an uncovered point, it is added to the original set of discrete points that represent region $A$ and this is used to run the master-problem. These two models are run until the sub-problem becomes infeasible. The infeasibility of the sub-problem suggests that there are not any uncovered points left in the continuous region $A$. It is important to note that in the master-problem the sensors are placed in a continuous region and in the secondary stage an uncovered
point is identified from the continuous region. Therefore, the approach proposed in this chapter overcomes the shortcomings of discretising region $A$.

Next, we will provide the details on algorithmic steps and optimization models run in each phase of the algorithm.

### 3.3.1 The Bi-level Algorithm

We run the master-problem and the sub-problem iteratively until the sub-problem becomes infeasible. The steps of this algorithm can be summarized as follows:

1. **Initialization**
   
   (a) Randomly select a discrete point from region $A$.
   
   (b) Set iteration count $t = 1$

2. **Iteration $t$**

   (a) **Master – Problem**: Run the master-problem, find the minimum number of sensors and their locations to cover all the points in $P$, where $P$ is points set to be covered in region $A$.

   (b) **Sub – problem**: Run the sub-problem to find an uncovered point $u$ in region $A$. Where the sensors positioned according to the master-problem.

   (c) If the sub-problem is feasible, then set $P = P \cup \{u\}$ and go to step (2a). Else, terminate and return the most recent solution of the master-problem.

Figure 3.1 - Figure 3.5 shows how the bi-level algorithm works on a simple example. Here, *circular dots* represent discrete points in set $P$, *square dots* represent locations of the sensors and circles surrounding them represent their sensing ranges. Figure 3.1 shows that initially a point is selected in region $A$. Figure 3.2 shows that after running the master problem a sensor is placed to cover the point given in region $A$. 

Figure 3.3, shows that sub problem found a new point which is not covered in region $A$. From Figure 3.4, we can see that the initial position of the sensor is adjusted to cover the two points according to the intersection of those points using MCLP approach. Figure 3.5, show the final solution obtained with the bi-level algorithm in which all the points in the continuous region $A$ are covered.

Figure 3.1: Randomly selected point in the given Surveillance area

Figure 3.2: A Sensor is selected to cover the uncovered point

Figure 3.3: New uncovered point is found
3.3.2 Master-Problem

In the master-problem, we find the minimum number of sensors needed to cover a given finite set of discrete points. The notation and the optimization model formulation of this problem is as follows:

**Parameters:**

- $N$: Maximum number of sensors to be deployed. We assume that there is a large number of sensors available and the subset of these sensors are to be selected to cover the finite set of discrete points
- $T_p$: Total number of points that need to be covered
- $P = \{p_j, j \in \{1, 2, ..., T_p\}\}$: Discrete points set to be covered from region $A$
- $M$: A very large number
Decision Variables:

- $x_i$: Final position of the sensor $i$, $i \in \{1, 2, ..., N\}$
- $y_{ij} = \begin{cases} 1, & \text{if sensor } i \text{ covers point } j \\ 0, & \text{otherwise} \end{cases}$, $i \in \{1, 2, ..., N\}, j \in \{1, 2, ..., T_p\}$.
- $z_i = \begin{cases} 1, & \text{if sensor } i \text{ is used} \\ 0, & \text{otherwise} \end{cases}$

The Model:

$$\min \sum_{i=1}^{N} z_i \quad (3.1)$$

s.t. $\|x_i - p_j\| \leq r \cdot y_{ij} + M(1 - y_{ij}), \forall i \in \{1, 2, ..., N\}, \forall j \in \{1, 2, ..., T_p\}$,

$$\sum_{i=1}^{N} y_{ij} \geq 1, \forall j \in \{1, 2, ..., T_p\} \quad (3.3)$$

$$y_{ij} \leq z_i, \forall i \in \{1, 2, ..., N\}, \forall j \in \{1, 2, ..., T_p\} \quad (3.4)$$

$$y_{ij}, z_i \in \{0, 1\} \quad \forall i \in \{1, 2, ..., N\}, \forall j \in \{1, 2, ..., T_p\} \quad (3.5)$$

Objective function (3.1) minimizes the total number of sensors to be installed. Constraint (3.2) verifies if sensor $x_i$ is capable to cover point $p_j$. Constraint (3.3) checks if every point in the region is covered by at least one sensor. Constraint (3.4) makes sure if a point is covered by a sensor, then that sensor is selected to install.

The model given above selects the sensor positions from the continuous region $A$, to cover the points set of $l_j$. However, it is shown in Church and ReVelle [1974], that in order to find the solution that may be some of the optimal positions of sensors to cover a discrete set of points, we only need to inspect a finite set of critical points rather
than searching the whole region. Therefore mixed integer model is replaced by MCLP problem discussed in Church and ReVelle [1974], since its solution matches with at least one optimal solution. Next, we will explain the details of finding intersection points of the circles and mixed integer model which identifies the minimum number of sensors to deploy using the critical points. These critical points are defined to be the intersection points of the circles drawn with a radius \( r \) centered at point \( p_j \). If the circle drawn is not intersecting with any other circle then critical point of that circle is point \( p_j \) itself.

**Finding Critical/Intersection Points:**

Here, we will explain how to find intersection points of two circles, if they intersect with each other. In Figure 3.6, two circles centered at \( p_1 \) and \( p_2 \) are shown. In this figure \( r_1 \) and \( r_2 \) represent the radii of these circles and \( ip_1 \) and \( ip_2 \) denote the critical points i.e. intersection points of the circles. We use the approach explained by Darby [2009] to calculate the coordinates of the critical points \( ip_1 \) and \( ip_2 \) as follows:

- \( ip_c \) : center point of \( ip_1 \) and \( ip_2 \)
- \( h \) : distance between \( ip_1 \) and \( ip_c \)
- \( a \) : distance between \( p_1 \) and \( ip_c \)
- \( b \) : distance between \( p_2 \) and \( ip_c \)
- \( d_x \) : difference in the x-coordinates of points \( p_1 \) and \( p_2 \)
- \( d_y \) : difference in the y-coordinates of points \( p_1 \) and \( p_2 \)
- \( d \) : distance between points \( p_1 \) and \( p_2 \) (i.e. \( d = (a + b) \))
Figure 3.6: Critical points of circles centered at $p_1$ and $p_2$

Intersection Point 1:

\[
ip_1^1 = p_1^1 + (a \cdot d_x - h \cdot d_y)/d \\
ip_1^2 = p_1^2 + (h \cdot d_x + a \cdot d_y)/d
\] (3.6) (3.7)

Intersection Point 2:

\[
ip_2^1 = p_1^1 + (a \cdot d_x + h \cdot d_y)/d \\
ip_2^2 = p_1^2 + (-h \cdot d_x + a \cdot d_y)/d
\] (3.8) (3.9)

Using the above equations, we find intersection points for the given points set. From these intersection points, we can identify the minimum number of sensors to cover the region $A$ using the following mixed integer formulation,

**Parameters:**

- $T_p$: Total number of points to be covered in region $A$
• $T_c$: Total number of critical points available for selection

• $c_{ij} = \begin{cases} 
1, & \text{if sensor at critical point } i \text{ can cover point at location } j \\
0, & \text{otherwise}
\end{cases}$

where, $i \in \{1, 2, ..., T_c\}$ and $j \in \{1, 2, ..., T_p\}$

**Decision Variable:**

• $z_i = \begin{cases} 
1, & \text{if a sensor is placed at critical point } i \\
0, & \text{otherwise}
\end{cases}$

\[
T_n = \min \sum_{i=1}^{T_c} z_i \quad (3.10)
\]

subject to \[
\sum_{i=1}^{T_c} c_{ij} \cdot z_i \geq 1, \quad \forall j \in \{1...T_p\}, \quad (3.11)
\]

Objective function (3.10), finds minimum number of critical points for installing sensors. Constraint (3.11) will verify if point $p_j$ is covered by at least one sensor.

### 3.3.3 Sub-Problem

In the sub-problem our aim is to identify a point in region $A$ that is not covered by the sensors deployed according to the master-problem solution. The optimization model formulation of this problem is as follows:

**Parameters:**

• $T_n$: Total number of sensors selected to deploy in the Master-Problem

• $x_i$: Position of sensor $i, \forall i \in \{1, 2, ..., T_n\}$

**Decision Variables:**
• **u**: Uncovered point in the given region \( A \)

• \( v_i = \begin{cases} 
1, & \text{if point } u \text{ is not covered by sensor } x_i \\
0, & \text{otherwise} 
\end{cases} \)

**The Model:**

\[
\text{subject to } \| u - x_i \| \geq r \cdot v_i + M(v_i - 1), \quad \forall i \in \{1, 2, ..., T_n\} \quad (3.12)
\]

\[
v_i \geq 1, \quad \forall i \in \{1, 2, ..., T_n\} \quad (3.13)
\]

Note that in the sub problem we aim to find an uncovered point so we set up the model as a feasibility problem. Therefore, the model for the sub problem does not have an objective function. Moreover, in the model constraint (3.12) verifies if there is at least one point in the continuous region \( A \) which is not covered. In constraint (3.13), binary variable is used to check if the point is outside of all the sensors.

In the above formulation, constraint (3.12) is not linear. We will use the approximation schemes explained in Section 2.4 to linearize the Euclidean distances in this constraint. Using the approximation method from Chapter 2, the approximate model of the sub problem can be set up as follows:

\[
\text{subject to } (a^k)^T (u - x_i) \geq r \cdot v_{ik} + M(v_{ik} - 1),
\begin{align*}
&\forall i \in \{1, 2, ..., T_n\}, \forall k \in \{1, 2, ..., S\} \\
&\sum_{k=1}^{S} v_{ik} \geq 1, \quad \forall i \in \{1, 2, ..., T_n\}, \forall k \in \{1, 2, ..., S\} 
\end{align*} \quad (3.14, 3.15)
\]

Where, \( a^k = \begin{bmatrix} \cos \left( \frac{2\pi k}{s} \right) \\
\sin \left( \frac{2\pi k}{s} \right) \end{bmatrix} \) and \( S \) is the number of polygon edges used in the
3.4 Numerical Results

To test the performance of the bi-level algorithm, we designed a small case problem. The objective of which is to cover a surveillance area of 7x7 units using sensors with transmission radius equal to 2 units. We used programming language Python and the optimization solver Gurobi on a computer with 32 GB RAM CPU for our test. Figure 3.7 - Figure 3.13, shows the iteration steps of bi-level algorithm. Here, rectangle plane is the surveillance area, red dots are the points that need to be covered, green triangular dots are the uncovered points found by sub-problem, blue circles and their centers represent the positions and transmission radii of sensors. Initially 4 corner points of the surveillance area are provided in the points set which needs to be covered. Figure 3.7 shows the master problem solution of iteration 1. Here four sensors are selected to cover the four corner points. Figure 3.8 shows the uncovered point, found by sub-problem in the iteration 1. After 30 iterations, bi-level algorithm has found 29 points apart from the 4 initial points and it has selected 6 sensors to cover those 33 points (see Figure 3.9). After 90 iterations, the points that need to be covered have spread all over the area and bi-level algorithm selected 7 sensors to cover those points (see Figure 3.11). After 145 iterations, bi-level algorithm was able to cover points in the entire surveillance area using 7 sensors. Figure 3.13 show the total number of sensors and their final positions. Note that in the final solution of the bi-level algorithm, the master problem placed sensors to cover a finite set of points, from the surveillance area. However, these sensors were placed so that all the points in the area were covered. This fact was confirmed when the sub problem was infeasible and therefore the algorithm was terminated.
Figure 3.7: Iteration 1 - master problem found the initial points to cover the given points

Figure 3.8: Iteration 1 - sub problem found an uncovered point
Figure 3.9: Iteration 30

Figure 3.10: Iteration 60
Figure 3.11: Iteration 90

Figure 3.12: Iteration 120
Figure 3.13: Iteration 145 - complete surveillance area is covered by WSN

Figure 3.14 shows the time taken by the master problem at each iteration level, from it we can see that solving time of the algorithm increases after each iteration, because the number of points that need to be covered increases, which leads to increase in number of constraints. Figure 3.15 shows the time taken by the sub problem at each iteration level, it is evident that solution time is almost constant at each iteration level, since increase in number of constraints is less. Figure 3.16 shows the cumulative total solving time at each iteration level. Due to the increase in solving time of master problem, cumulative solving time increased exponentially.
Figure 3.14: Solving time of Master Problem for 7x7 surveillance area at each Iteration

Figure 3.15: Solving time of Sub Problem for 7x7 surveillance area at each Iteration
Figure 3.16: Cumulative solving time for 7x7 surveillance area
Chapter 4

Conclusion and Future Research

4.1 Conclusions

In Chapter 2, optimal relocation model for the actors is developed to establish single connectivity and bi-connectivity in WSAN, using linear mixed integer programming techniques, concept of adjacency matrix from graph theory and matrix multiplication. None of the previous papers which dealt with this problem developed an optimization model and to the best of our knowledge, this model is first of its kind to consider the concepts like adjacency matrix and matrix multiplication together in developing mixed integer programming model. However, there are certain drawbacks to this model. The number of constraints increases proportional to $n^3$ in the case of single-connectivity and $n^4$ in the case of bi-connectivity, where $n$ is the number of actors. Therefore, solving time increases significantly for large scale problems.

In Chapter 3, problems in WSN and importance of coverage are discussed. A bi-level algorithm is developed in order to find minimum number of sensors to cover a given surveillance area completely. The bi-level algorithm runs two optimization models, namely main problem and sub problem iteratively. Main problem is used
to find minimum number of sensors to cover given set of discrete points. In the sub problem, the points which are uncovered by the sensors located according to the points set which again is fed to the master problem. The bi-level algorithm also takes extensive amount of time when applied, for large scale problem. From numerical experiments, it is found that the majority of time is taken by the master problem which becomes harder to solve with large points sets. Whereas, the time requirement of sub problem stays constant in each iteration.

4.2 Future Research

Even though the models developed for relocation of actors in WSAN and coverage of sensors in WSN are optimal, there are issues with the solving time of the models. To overcome these problems, we are planning to further research on methods to make our model more efficient and also to develop heuristics whose solution is near to the optimal solution.
Bibliography


