Prolonging Stability and Energy Conservation of CDS-based WSN using Nature-inspired Techniques

A Dissertation submitted in partial fulfillment of the requirement for the award of Degree in

Master of Technology

in

Computer Science Engineering

Submitted By

Bhavya Saggi

(Roll No.- 2K15/CSE/04)

Under the guidance of

Prof.(Dr.) Daya Gupta

Department of Computer Science & Engineering

Delhi Technological University

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-110042

2015 - 2017

CERTIFICATE

This is to certify that Dissertation entitled "**Prolonging Stability and Energy Conservation of CDS-based WSN using Nature-inspired Techniques**" has been submitted by **Mr. Bhavya Saggi** (Roll No. **2K15/CSE/04**), in partial fulfillment of the requirement for the award of degree in Master of Technology in Computer Science & Engineering. This is a record of the candidate work carried out by him under my supervision and has not been submitted earlier for the award of any degree or diploma in any university to the best of my knowledge.

Prof.(Dr.) Daya Gupta
Department of Computer Science & Engineering
Delhi Technological University

DECLARATION

I hereby declare that the dissertation entitled "**Prolonging Stability and Energy Conservation of CDS-based WSN using Nature-inspired Techniques**" which is being submitted to Delhi Technological University, in partial fulfillment of requirements for the award of degree in Master of Technology (Computer Science and Engineering) is a bona fide report of Major Project-II carried out by me. The material contained in the report has not been submitted to any university or institution for the award of any degree.

Bhavya Saggi
2K15/CSE/04
M.Tech (Computer Science & Engineering)
Department of Computer Engineering
Delhi Technological University Delhi – 110042

ACKNOWLEDGEMENT

I would like to express my deep sense of respect and gratitude Dr. Daya Gupta (Professor, Computer Science & Engineering) for providing the opportunity of carrying out this project and being the guiding force behind this work. I am deeply indebted to her and all the faculty members of the Computer Science Engineering Dept. of DTU for the support, guidance, and encouragement for the project.

I would also like to express gratitude to Mrs. Shruti Jaiswal (Research Scholar, Delhi Technological University) for providing me continuous support and guidance during this project.
I would also like to acknowledge Delhi Technological University library and staff for providing the right academic resources and environment for this work to be carried out. Last but not the least I would like to express sincere gratitude to my parents and friends for constantly encouraging me during the completion of work

Bhavya Saggi
2K15/CSE/04
M.Tech (Computer Science & Engineering)
Department of Computer Engineering
Delhi Technological University Delhi – 110042

ABSTRACT

Wireless Sensor Networks is a collection of tiny, battery-operated but smart sensors in a distributed environment for sensing tasks.
Being the foundation of Internet-of-Things (IoT), the battery powered nodes in WSN are eligible to Energy Constraints. As the Energy is consumed during the communication in a WSN, this makes construction of a large-scaled and efficient WSN a difficult task.

A popular technique for regulation the energy costs and to prolong the lifespan of a wireless sensor network is Clustering. “Clusters” consist of a Cluster-Head and Border Nodes (or Edge Nodes), where all Cluster-Heads are conjoined together to aggregate and deliver data to the Base Station (or Sink).
The connected Cluster-Heads form a multi-hop Virtual-Backbone, using which any message can be sent from any source to any destination, directly or indirectly. As there is no predefined physical backbone infrastructure, a virtual backbone can be formed by constructing a Connected Dominating Set (CDS).

In order to reduce the Energy costs of a WSN, focus remains on constructing a Minimum CDS (MCDS). However, construction of a minimum connected dominating set falls under the set of NP-Hard problems. With the advancement of various soft computing techniques, nature-inspired algorithms have shown much promise to resolve many optimization problems.
Concentration is focused on using such algorithms (Artificial Bee Colony, & Grey-Wolf Optimizer) to solve Energy-Constraints in construction of a CDS to prolong the stability and form even Cluster-Heads (Dominator Nodes). For confirmation of results obtained, they are compared with the previously researched algorithms and proving that with tweaks in fitness function and choosing appropriate nature-inspired optimization algorithm, better results can be found.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

- **WSN**: Wireless Sensor Network
- **DS**: Dominating Set
- **CDS**: Connected Dominating Set
- **mCDS**: minimally Connected Dominating Set
- **CH**: Cluster Head
- **VB**: Virtual Backbone
- **EA**: Evolutionary Algorithm
- **GA**: Genetic Algorithm
- **SAECDS**: Stability Aware Evolutionary CDS
- **CDS-ABC**: CDS with Artificial Bee Colony
- **CDS-GWO**: CDS with Grey Wolf Optimizer
- **|D|**: Number of Dominators
- **D_{DE}**: Total Energy Dissipation in Dominators
- **NW_{DE}**: Total Energy Dissipation in Network
- **D_{die}**: Difference between first and last Dominator Node failure

Chapter 1. Introduction

Recent advances in technology and wireless communications have led the emanation of wireless sensor networks (WSNs). This chapter contains overview and detail of the WSN and its' related technologies, with their impact. Also, a brief description of the optimization problems and the 'Nature inspired Algorithms' is mentioned.

1.1. Overview

Large number of independent nodes in environment performing various tasks such as sensing, processing, and transmitting information, are collectively known as Wireless Sensor Networks (WSNs). Member nodes of a WSN can either homogeneous or heterogeneous but all have limited communication and computational capacity; but they can perform magnificently in dangerous or remote areas with appropriate connectivity and programming.

With the wide range of technologies available, there is a requirement to solve basic topological and energy constraints problems inherent to the WSN system. To understand where the problem arises, we need to understand the working and potential solutions to these problems.

The following section delves into the WSN, its shortcomings (primarily in construction of a mCDS which is a NP-Hard problem), and how they can be overcomed (with use Nature inspired optimization techniques).

1.2. Basic Concepts

This section begins with description of what a Wireless Sensor Network is and what are its basic essential components. Later it is described, how these components work together and find that hierarchical routing (using clustering) proves to be an efficient way to do so.

In the next sub-section, aforementioned clustering is explained in detail, with its merits and demerits and how to achieve it algorithmically with use of Connected Dominating Sets

The last sub-section, describes the Nature-Inspired Algorithms (and various common meta-heuristics) which would later be utilized to achieve the goal of this thesis.

1.2.1. WSN

A wireless sensor network (WSN) is an autonomous wireless network of spatially distributed independent devices equipped with sensors, radio systems and a battery to survey its surroundings and unanimously pass the collected data to a central location (named Base Station or Sink). An overview of a basic model of wireless sensor network is shown in Figure 1.

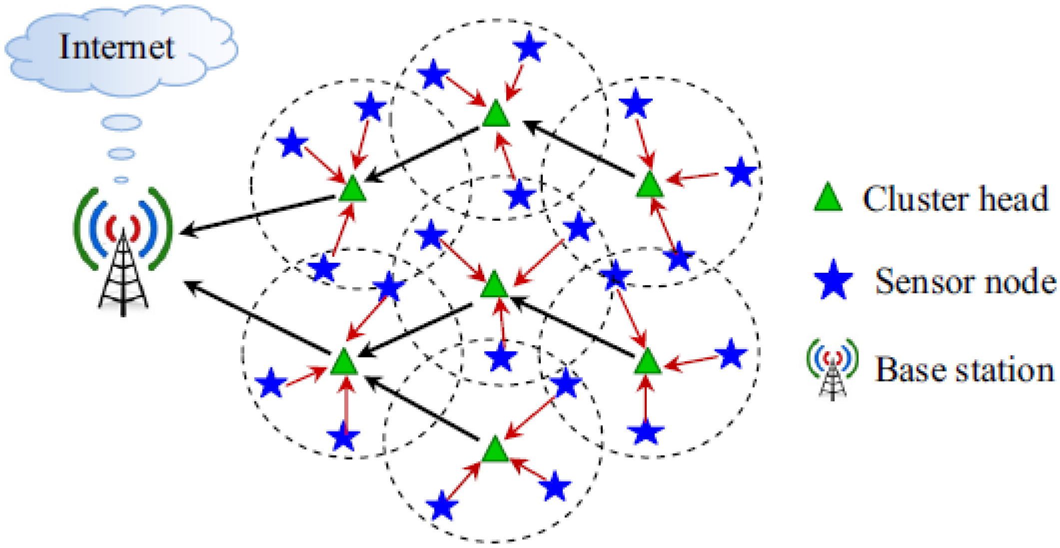


Figure 1. A Wireless Sensor Network Model

Integral component of a WSN is a “node”, each fitted with a radio transceiver with an antenna, a microcontroller, an electronic circuit (for sensor interfacing), and an energy source (usually a battery). Having varying size and cost, results in nodes having constraints on its resources such as energy, computational speed, communications bandwidth, and memory.

Like wired networks, varying topologies can be used for WSN (e.g. star network, mesh network) with different propagation techniques (e.g. routing, flooding). This could be achieved via two different solutions from existing works. [1]

- First being Flat Routing, where all sensor nodes are homogenous and send their data to Base Station (or Sink Node) directly; this generally leads to excessive data-redundancy and faster energy consumption.
- Other being Hierarchical Routing, where the entire network is segmented into distinct and diverse Clusters. [2]

Due to lack of data-redundancy and efficient energy consumption, Hierarchical Routing proves to be a popular choice. For making use of hierarchical routing clusters are required to be constructed.

1.2.2. Clustering

Clusters are a collection of nodes which work together in unison. Each cluster has a Cluster head dominating over Source Nodes. Source nodes are sensor nodes which cumulate information and deliver it to cluster head. The cluster head is an elected node via some criteria having responsibility of assembling and aggregating data from sensor node and delivering it to sink node.

Merits of Clustering includes efficient use of energy and improved network lifetime, and furthermore with adequate formation they may also provide load balancing.

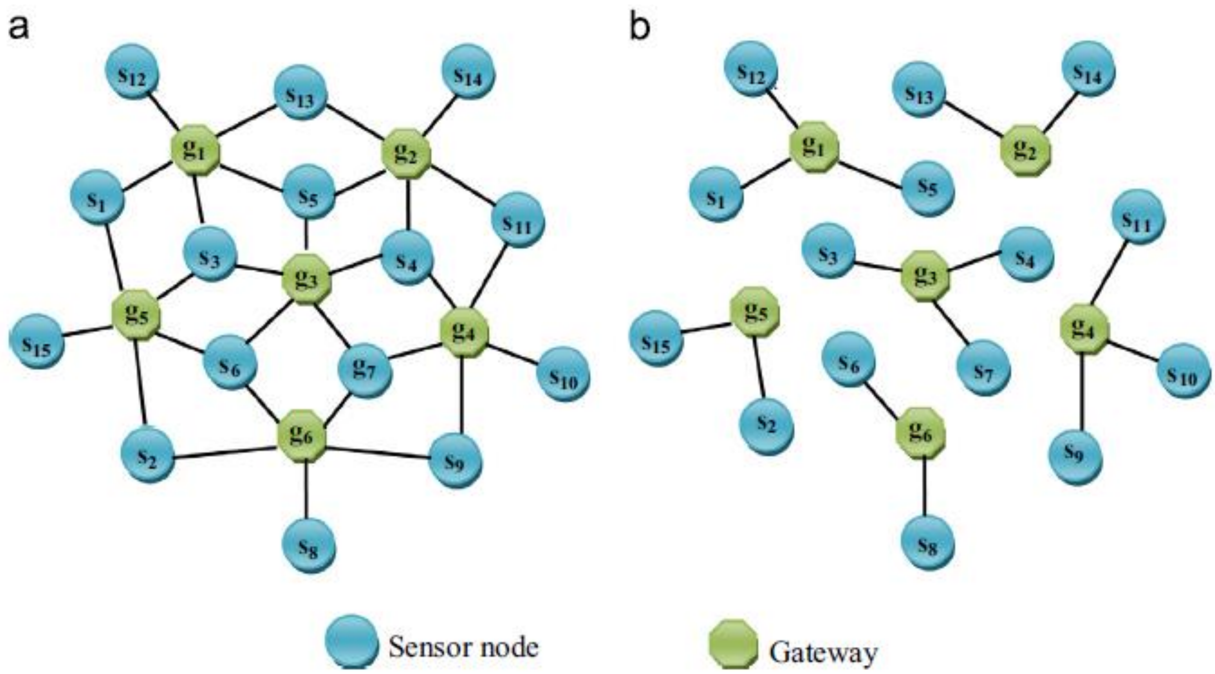


Figure 2. A WSN a) before clustering b) after clustering

As shown in an example in Figure 2, while clustering provides better energy efficiency, there exist few issues with it.

- Distance: Higher distance between nodes causes a cluster to have fewer nodes in it, making the clusters less energy efficient.
- Energy: Reducing the transmission cost and decreasing the number of messages delivered leads to lesser energy consumption.
- Density: Increase in sensor density makes the network overloaded, which may cause communication latency and dissatisfactory performance.

While clustering has many variations, but common element in all is that each node transfers its message to its cluster head and the respective cluster head has the responsibility to convey the message to sink (directly or indirectly via intermediate cluster heads). To perform in such manner, a Virtual Backbone is formed to cover the network, comprising of the cluster heads.

Virtual backbones are a set of procedures mirroring the infrastructure in the traditional network model. It is generated by choosing a subset of network nodes and / or communication channels Without presence of predefined physical backbone infrastructure, a virtual backbone can be constructed through use of a Connected Dominating Set (CDS) as shown in Figure 3.

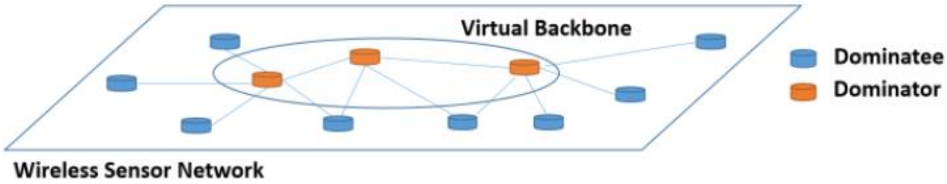


Figure 3. CDS-based Virtual backbone in a Wireless Sensor Network

Since, by definition, any node in the network is less than 1-hop away from a connected dominating set (CDS) node, a connected dominating set (CDS) provides a viable solution towards formation of a virtual backbone based topology control for WSNs. It holds much importance because, messages are relayed in network through backbone nodes only.

Connected Dominating Sets

For a graph G, Connected Dominating Set is a set D of vertices with two properties:

- Every vertex in G either is adjacent to a vertex in D or belongs to D.
- D induces a connected subgraph of G. Meaning, any node in D can be reached via any other node in D by a path that stays entirely within D.

The nodes in the CDS are called as dominators and the rest of the nodes of the network are called as dominatees.

To test whether there exists a connected dominating set with size less than a given threshold is a NP-complete problem. Therefore, it is believed that the minimum connected dominating set problem cannot be solved in polynomial time.

To solve the NP-Complete problem, we need meta-heuristic algorithms or approximation / optimization solutions to the problem.

One recently popular solution to this is found in the Nature-inspired Optimization algorithms. In this the meta-heuristics of approximation and exploration of search space is inspired through the study of existing flora or fauna. A detailed description for nature-inspired algorithm is provided in the next section.

1.2.3. Nature Inspired Optimization

Nature-inspired algorithms are high-level procedures designed to navigate and produce a viable result to an optimization problem. While not guaranteeing globally optimal solution, they implement some form of stochastic optimization leading solutions to depend on the randomness of the variables generated.

Gaining momentum in past two decades, they have been worked upon theoretically and applied in different fields of study by scientists. Reasons for such includes:

- **Simplicity** arises from the derivation & inspiration of the metaheuristics from physical phenomena, animal behavior, and evolutionary concepts. The simplicity enables scientists to adapt, improve and apply them to their problems.
- **Flexibility** enhances flexibility of metaheuristics to diverse problems since they comprise mostly of black-boxes; i.e. only inputs and outputs are important for a metaheuristic.
- **Derivation-free mechanisms** allow metaheuristics to optimize problems stochastically as it navigates through random solutions and saves calculating derivative of entire search space to find optimal, making them highly eligible for real problems.
- **Local-optima avoidance** is achieved through stochastic nature of metaheuristics allowing them to avoid stagnation. Real problems have an extensive the search space which includes massive (complex) number of local optima.

Basic requirements for gaining a solution through use of nature-inspired algorithm is

- Representation of search-space in discrete solution-sets
- Fitness function to evaluate the proximity of solution-set to best answer

For combinatorial optimizations, with less effort metaheuristics can often find good solutions with by searching over a set of feasible solutions.

The representation of search-space and designing of apt fitness function depends upon the meta-heuristic chosen along which the entire algorithm would be designed. It fundamentally depends upon the core designing of the problem e.g. "if the solution to the problem has a discrete solution or continuous solution" or "if the problem is a combinatorial problem or a real-value single-solution problem".

Following subsection describes more about the meta-heuristics with few examples of popular algorithms and their meta-heuristics.

1.2.3.1. Meta-heuristics for Nature Inspired Algorithms

There is no metaheuristic best apt for resolving all optimization problems. Hence, a metaheuristic may need to be fine-tuned for a problem, but some other algorithm may even surpass it on another set of problems.

Generally, metaheuristics can be classified along two main classes:

- Single-solution based: A single candidate is improved upon over a course of iterations.
- Population-based: A set of solutions (population) is enhanced over course of iterations.

Meta-heuristics may be classified into three main classes:

- Physics-based Algorithms
- Evolutionary Algorithms
- Swarm Intelligence Algorithms

Broadly, optimization is achieved through evolution of initial random solution; where individuals in each population has higher probability of being better than previous generation(s). This guarantees optimization of random population over generations.

A few popular techniques are discussed as follows:

i. Genetic Algorithm

Inspired by the concepts of evolution of in nature, most popular Evolutionary Algorithm is Genetic Algorithm. Simulating Darwinian Evolutionary concepts, it was proposed by Holland [3] in 1992. Genetic algorithm (GA) is a metaheuristic inspired by the process of natural selection. Being an evolutionary algorithms (EA), genetic algorithms generate optimization solutions and search problems by banking on bio-inspired operators such as mutation, crossover and selection.

Requirement of a Genetic Algorithm includes:

- A genetic representation of the solution domain
- A fitness function to evaluate the solution domain.

ii. Artificial Bee Colony

By mimicking social behavior of swarms, herds, flocks, etc. Swarm Intelligence Algorithms uses navigation of search agents using simulated collective and social intelligence of natural creatures.

Artificial bee colony algorithm (ABC) is an optimization metaheuristic based on the intelligent foraging behavior of honey bee. Simulating the bees, they are classified into 3 groups:

- Employed Bees: Each employed bee is assigned to a source. They 'dance' around the said source till it exhausts and then comeback as scouts.
- Onlooker Bees: They follow employed bees and look for new sources.
- Scout Bees: Help finding new sources upon exhaustion of previous one.

The Bees keep scavenging the entire search-space for better solutions and keep shifting responsibilities when a "solution source" is depleted.

1.3. Motivation

When an enormous quantity of nodes in a WSN work together unanimously, a single fault occurrence at any node can be hard to dismiss. Also, being in a dynamic environment makes recharging or replacement of batteries a difficult task. Therefore, WSN protocols are required to be designed in a manner to mitigate fault occurrence and minimalize power consumption.

For a large network, most stable result is achieved through minimization of the number of Clusters and the length of Virtual-Backbone; albeit by reducing number of Dominator Nodes. Furthermore, the reduction of number of Dominator Nodes should be done in such a way that it also reduces the energy consumption of the dominator nodes as well as they energy consumption of the entire network. The energy functions were considered following the simple first-order ratio model [4];

Transmission of ***k***-bit message over a distance ***d*** expends:

$$E_{TX}(k,d) = E_{elec} * k + \varepsilon_{amp} * k * d^2$$

Reception of message, it expends:

$$E_{RX} = E_{elec} * k$$

where,

E_{elec} = energy consumption rate for radio transceiver

ε_{amp} = energy consumption rate for transmitter amplifier

Hence, the optimization / fitness function that needs to be designed for best solution is designed in a way that it includes:

- Sum of "Energy consumption in a Cluster" for all clusters.
- "Energy for transmitting aggregated data" from each Cluster Head to Base station
- "Distance of each cluster" from Base Station

Formally,

$$F = \left(\sum_{\forall Clusters} Energy\ consumption\ in\ a\ Cluster \right) * \left(\sum_{\forall Clusters} Energy\ Factor * Distance\ of\ Cluster\ from\ BS \right)$$

Since the Optimization of above has a major influence over the network performance but a major part these problems are categorized under the NP-Hard problems they require further research on how to obtain optimized results. This is explored further in the thesis and fitness / objective function is given a more mathematical definition to provide a viable and decipherable solution.

1.4. Related Work

Because of the widespread utility of the Wireless Sensor Networks, much research has been done on optimizing it along various bounds i.e. topological, energy constraints, etc.

Generally, algorithms to solve the mCDS problem contain two types:

- After forming a Maximal Independent Set (MIS), connecting independent nodes by finding connecters. [2]
- Pruning redundant nodes after building a CDS.

In brief, Major research results are enumerated as follows:

- Centralized greedy algorithms [5] has the greedy function which is number of unconnected neighbors of each node and with each iteration, the node with largest number of unconnected neighbors is designated as a dominator.
- In [6] genetic algorithms were implemented to construct mCDS while addressing reliability concept in CDS to maintain and measure probabilistic connectivity between nodes
- In [7] three-stage algorithm was used to construct mCDS. Dominating sets were identified, connectors were found, and then pruning was done to obtain a mCDS.
- If [8] , mr-CDS protocol was used which considers the residual energy of nodes. Focus was that when energy of a dominator drops below acceptable threshold, another node picks the dominator responsibility, resulting in energy efficiency.
- In SAECDS [9] centralized protocol was used, utilizing Evolutionary Algorithm (EA) with goal of prolonging the stability period of CDS by iteratively exploring through the search space and finding appropriate optimization solution

A more detailed description of all are described in the next chapter.

1.5. Problem Statement

For a WSN of ***N*** sensor nodes, a CDS solution is considered as ***I*** i.e. a fixed-length array of size ***N***, where each element can either be 0 (if dominate) or 1 (if dominator). So, each element (***j***) of the solution $I_j = \begin{cases} 1 \; if \; node \; j \; is \; dominator \\ 0 \; if \; node \; j \; is \; dominatee \end{cases}$

Therefore, the possible combinations for CDS are exponential for a set of sensor nodes in a large-sized WSN. Also, as described by Gary & Johnson et.al. [10], the dominating set problem is a classical NP-Hard decision problem in computational complexity theory.

Furthermore, to test whether there exists a connected dominating set with size less than a given threshold [i.e. testing whether domination number $\gamma(G) \leqslant K$ for a given graph ***G*** and input ***K***] is a NP-complete problem.

Hence, the results need to be obtained through more complex and intelligent algorithms which finds a CDS solution to a WSN in polynomial time. This gave way to implement Evolutionary Algorithms onto the mCDS problem. One of the most recent solution towards this is the SAECDS algorithm, which applies Genetic Algorithm to find a CDS for a WSN, and considered only the energy consumption of the member nodes for generation of a CDS. But, certain important factors were overlooked such as the cluster distance from the base station, energy requirements of a node in a cluster and effective network energy consumption in the WSN.

Introduction of these factors remains the focus of this research. To fit in these factors, new emerging algorithms, Artificial Bee Colony and Grey Wolf Optimization have been considered.

Artificial Bee Colony algorithm was implemented as it been proved to be efficient in solving complex problems (e.g. solving Sphere function & Rastrigin function) and performed exceedingly well in application in standard NP-Complete problems such as “Travelling Salesman Problem”.

Furthermore, implementation of “Grey Wolf Optimizer” was done as it has shown promising and more than satisfactory results in the various domains a training algorithm for Multi-layer perceptron (Feedforward Neural Networks) [11], Feature Subset Selection Approach by Emary et al. [12], and optimizing key values in the cryptography algorithms [13].

Consequently, in this thesis more mature nature-inspired algorithms i.e. Artificial Bee Colony (ABC) and Grey Wolf Optimization (GWO) are applied onto the NP-Hard problem set of finding the Connected Dominating Set with energy conservation constraints. Then, results are compared with existing algorithms to achieve a promising solution to prolong stability and extend the lifetime of the Wireless Sensor Network. The research question posed in this thesis is thus:

"**To apply Evolutionary Algorithms to find an optimized Connected Dominating Set for a Wireless Sensor Network, satisfying the energy constraints for member nodes and reduction in network energy consumption, with reduction in Euclidian topological distance between member nodes**"

1.6. Scope of work

The Artificial Bee Colony (ABC) and Grey Wolf Optimization (GWO) provide a novel approach towards finding a solution to the combinatorial NP-Hard problems and are designed in such a way that they require just initial viable solution and number of Search Agents (a Bee in case of ABC and a Wolf in case of GWO). Additionally, GWO has two main parameters to be adjusted (***a*** & ***C***) which makes it easier to fine-tune the algorithm, making it better than other nature-based meta-heuristics.

The energy cost of each node, and energy consumption of entire network in WSN is considered in this thesis with Euclidian distance of cluster-head from Base station for construction of an efficient Connected Dominating Set.
Also, their performance of GWO and ABC were empirically analyzed with existing work (SAECDS) based on the total number of dominators, total energy loss in network, and difference between first Dominator failure and last Dominator failure.

'python' was chosen as the platform for simulating the WSN. WSN data was generated randomly and comparisons were drawn by taking average of 5 random instances. The SAECDS was implemented along with the proposed ABC and GWO algorithms.

To this effect, scope of the work can be summarized as:

- Implement Artificial Bee Colony Algorithm for constructing a minimal Connected Dominating Set, using energy consumption and distance of cluster-heads from base station as additional parameters.
- Adapting Grey Wolf Optimizer for construction of a minimal Connected Dominating Set, using similar energy consumption and distance of cluster-heads from base station as parameters.

- To apply the new algorithms onto a simulation, and compare them with existing work

1.7. Thesis Organization

The rest of this dissertation is organized as follows:

- **Chapter 2** provides details about the past research done on improving optimization results towards solving mCDS problem in WSN.
- **Chapter 3** gives description of the recent popular Nature-inspired algorithm named "Grey-Wolf Optimizer"
- **Chapter 4** shows the proposed framework and describes the algorithm in detail on how GWO is used to solve the mCDS problem in WSN.
- **Chapter 5** gives information about the Implementation and Simulation performed, with further details on the simulation environment, results, and analysis.
- **Chapter 6** is about the conclusion and future work, where all work is summarized, and followed by references where all research articles which contributed to this research are listed

Chapter 2. Literature Review

In this chapter, the description of the related work and the research necessary to devise more efficient method of achieving the goal of prolonged stability and extended lifetime in WSN is provided.

Later, a recent solution [9] is described.

2.1. Solutions for Minimum-Connected Dominating Set

Conceptualized by Claude Berge [1], mathematical concept of domination is utilized to find CDSs (chiefly mCDSs). Generally, algorithms to solve the mCDS problem contain two types:

- After forming a Maximal Independent Set (MIS), connecting independent nodes by finding connecters. [2]
- Pruning redundant nodes after building a CDS.

Top three popular approaches towards finding solution to a minimum Connected Dominating Set problem are discussed below. Afterwards a more recent solution to the problem [9] is discussed.

2.1.1. Centralized greedy Algorithm

Centralized greedy algorithms proposed by Guha et al. [5] has performance ratios of $2(H(\Delta) + 1)$ and $H(\Delta) + 2$ respectively, where H is a harmonic function. Here, the function designed greedily around the number of unconnected neighbors of each node and along each iteration, the node with largest number of unconnected neighbors is designated as a dominator.

Presented in [14] centralized algorithms for CDS Problem in Unit Disk Graphs (UDGs), the algorithm initiates from starting node and using the spanning tree it adds dominators to set. Finally, the algorithm confirms if a dominator node could be omitted while condition for a CDS is satisfied.

2.1.2. Distributed mCDS Algorithm

Following above, numerous distributed algorithms were proposed for construction of a minimally Connected Dominating Set (mCDS).

In [6] genetic algorithms were implemented to construct mCDS while addressing the concept of reliability in CDS to maintain and measure probabilistic connectivity between nodes; Through assignment of Transmission Success Ratio (TSR) to each link connecting a pair of nodes in CDS and a single-objective fitness function to maximize CDS reliability.

It is evident that through increase in number of dominators, the reliability of CDS decreases. Hence, in [7] and [8] this was addressed through a localized repair algorithm.

2.1.3. Three-Stage mCDS Algorithm

In [7] three-stage algorithm was used to construct a minimally Connected Dominating Set (mCDS). First, dominating sets were identified. Second, connectors were found (connected via Steiner Tree). Third, pruning to obtain a mCDS.

If [8] , mr-CDS protocol was used which considers the residual energy of nodes. Focus was that when energy of a dominator node drops below acceptable threshold, another node picks the dominator responsibility, resulting in energy efficiency.

SAECDS [9] is another centralized protocol, utilizing Evolutionary Algorithm (EA) with goal of prolonging the stability period of CDS.

2.2. Stability Aware Evolutionary CDS Algorithm

Stability Aware Evolutionary CDS Algorithm (SAECDS) is a central protocol, running at Base Station (BS), with knowledge of sensors' areas and unlimited energy resource. Using the Evolutionary Algorithm (EA), it starts with initial population of viable solutions and obliges diverse forms of selection, crossover and mutation to iteratively deliver new deliver new solutions.

Better individuals have more probability of being chosen, which is measured by its' fitness. The various operations provide variation and evolution of the population; such as Crossover is exchange of good genetic material between parent to generate new solutions, and mutation provides minute simple changes with user-defined flexible likelihood to explore the entire space and maintain diversity.

The SAECDS considers population ***p*** of ***K*** solutions, since problem solution is highly sensitive to evolutionary operations and solution representation.

A WSN of ***N*** sensor nodes is considered with each individual solution $\boldsymbol{I^i}$; $1 \leq \boldsymbol{i} \leq K \in \boldsymbol{p}$

i.e. a fixed-length of size ***N***, where each element can either be 0 (if dominate) or 1 (if dominator).

A population of ***K*** individual solutions,

$$\forall i \in \{1, \dots, K\} \; and \; \forall j \in \{1, \dots, N\}$$

$$I_j^i = \begin{cases} 1 \; if \; node \; j \; is \; dominator \\ 0 \; if \; node \; j \; is \; dominatee \end{cases}$$

After designing the essentials of the system, the subsequent subsection provides a deeper look into the algorithm, describing the stages of the algorithm and how it progresses to find the optimal solution to the problem.

2.2.1. Algorithm Framework

Using the initialized chromosome, the population is extended using Inheritance Population Initialization (IPI) [15], where first individual is obtained by running mCDS algorithm [5].
Each individual in population has a fitness (objective) attached to it, which numerically quantifies its' effectiveness.

2.2.1.1. Fitness Function

For SAECDS, objective function is evaluated through minimization of total dissipated energy in network, measured through summation of total energy dissipated of network dominates to transmit data via connected dominators and total energy used by dominators to aggregate and send data to Base Station, with total energy dissipation along the path.

Formally,

$$\varphi_{SAECDS}(I) = \left(\sum_{i=1}^{D}\left(\sum_{j=1}^{k} E_{TX_{j,i}} + E_{RX} + E_{DA}\right)\right) + \sum_{i=1}^{D} E_{D_i,BS}$$

where,

D = Total number of dominators

k = Total number of associated dominates

$E_{TX\,(node1,\,node2)}$ = Energy dissipation for transmitting data from node1 to node2

E_{RX} = Energy dissipated for receiving data

E_{DA} = Energy dissipated for aggregating data

$E_{D,\,BS}$ = Total Energy dissipated by all dominators to Base Station

2.2.1.2. Algorithm Procedure

After initialization of population with individuals with respective fitness values, solutions are iterated through a series of operations to evolve the said population with better solutions.

First is Selection operator, which chooses partners from current population and keeps them in a mating pool for reproduction. Using "Binary Tournament", the best individual from two randomly selected individuals of population set is chosen. This process is repeated ***K*** times to produce a mating pool of ***K*** parents.

Formally,

$$S: I^2 \rightarrow I'$$

$$I_{i,r1}, I_{i,r2} \; \forall i \in \{1, \dots, K\}$$

$$I_i' = \begin{cases} I_{i,r1} \; if \; \varphi(I_{i,r1}) \le \varphi(I_{i,r2}) \\ I_{i,r2} \; ; if \; otherwise \end{cases}$$

where,

r1, r2 ~ U{1,…,K} ; two uniformly distributed random numbers from Set {1,…,K}

Second component is Crossover operator. A $\boldsymbol{p_c}$ percentage of pairs of parents in population set are chosen for merger. From the selected pairs, two random points are selected from range {1,…,N} and alleles of parents are swapped between these points.

Formally,

$$I_1' = (I_{1,1}, \dots, I_{1,r1}, I_{2,r2}, \dots, I_{2,r2}, I_{1,r2+1}, \dots, I_{1,N})$$

$$I_2' = (I_{2,1}, \dots, I_{2,r1}, I_{1,r2}, \dots, I_{1,r2}, I_{1,r2+1}, \dots, I_{2,N})$$

Third component is Mutation operator. With a probability of pm, each allele is mutated in population set i.e. its inverted from 1 to 0 and vice-versa.

Formally,

$$(\forall i \in \{1, \dots, n\} \; \& \forall j \in \{1, \dots, N\}) :$$

$$I_j^{i\prime} = \begin{cases} I_j^i \; if \; random > \; p_m \\ 1 - I_j^i \; otherwise \end{cases}$$

Because of generated randomness, the new population set needs to be validated for generated solutions i.e. the population chromosome may not be a valid Connected Dominating Set.

After validation and filtering of unwanted solutions, the algorithm iterates till termination condition of Evolutionary algorithm is satisfied.

Towards the end, the solution with best fitness / objective (minimum value in this case) is chosen as Final solution.

After going through the existing works, it was found that some aspects of cluster formation have been skipped and there can be more work done over this. A primary factor that was realized was that while a cluster is formed, there is no consideration of the topological distance of the cluster from the base station.

Hence, a new algorithm is needed to be designed so as the correct chosen dominators need to have a well-formed trade-off between the number of connected dominators and energy constraints over them.

The next chapter provides details of a new nature-inspired optimization algorithm with a hunting-based meta-heuristic which is very well suited to our needs.

Chapter 3. Grey Wolf Optimizer Algorithm

The GWO algorithm [16] impersonates the hierarchy and hunting mechanism of grey wolves in nature to achieve optimization for problems.

It falls under the category of Evolutionary Algorithm (implementing Swarm Intelligence) and requires an initialized (or random) population to begin with.

Four types of grey wolves: alpha, beta, delta, and omega simulates the hierarchy.

Being a hunting algorithm, its steps include:

- Searching
- Encircling
- Attacking

3.1. Social Hierarchy of Grey Wolves (Inspiration)

- The pioneers are called alphas and mostly part in charge of settling on choices which are directed to the pack.
- Second is Beta Wolf, who assists the alpha in basic leadership or other pack exercises.
- Most minimal positioning is of Omega Wolf, which appear to be not a critical individual in the pack. Generally, the omega is an additional sitter in the pack.
- Lastly, Delta wolves lie somewhere between Beta wolf and Omega wolf.

3.2. Mathematical Model

The mathematical models of the social hierarchy, tracking, encircling, and attacking prey are as follows:

3.2.1. Social Hierarchy

From a set of solutions or initial population, the fittest solution is considered as the alpha (**α**). Consequently, the second and third best solutions are considered as beta (**β**) and delta (**δ**). Remainder of solutions are assumed to be omega (**ω**).

In the GWO, calculations are led by **α**, **β**, and **δ**; whereas **ω** wolves take after these three wolves.

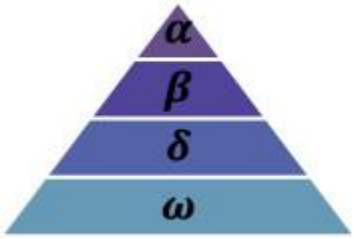


Figure 4. Grey wolf social hierarchy

3.2.2. Encircling prey

To mathematically model encircling behavior, the following equations are followed:

$$\vec{D} = \left| \vec{C}.\overrightarrow{X_p}(t) - \vec{X}(t) \right|$$

$$\vec{X}(t+1) = \overrightarrow{X_p}(t) - \vec{A}.\vec{D}$$

where **t** indicates the current iteration, and **A** and **C** coefficient vectors, $\mathbf{X_p}$ is the position vector of the prey, and **X** indicates the position vector of a grey wolf.

To calculate vectors **A** and **C**:

$$\vec{A} = 2.\vec{a}.\overrightarrow{r_1} - \vec{a}$$

$$\vec{C} = 2.\overrightarrow{r_2}$$

where components of **a** are linearly decreased from 2 to 0 over the course of iterations and $\mathbf{r_1}$, $\mathbf{r_2}$ are random vectors in [0,1].

3.2.3. Hunting

First three best solutions are considered by the other search agents to update their positions accommodating the status of the best search agent.

The following formulas are utilized:

$$\overrightarrow{D_\alpha} = \left| \overrightarrow{C_1}.\overrightarrow{X_\alpha} - \vec{X} \right|, \quad \overrightarrow{D_\beta} = \left| \overrightarrow{C_2}.\overrightarrow{X_\beta} - \vec{X} \right|, \quad \overrightarrow{D_\delta} = \left| \overrightarrow{C_3}.\overrightarrow{X_\delta} - \vec{X} \right|$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1}.\left(\overrightarrow{D_\alpha}\right), \quad \overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2}.\left(\overrightarrow{D_\beta}\right), \quad \overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3}.\left(\overrightarrow{D_\delta}\right)$$

$$X(t+1) = \frac{X_1 + X_2 + X_3}{3}$$

This can be visualized in following Figure on how the next search agent updates its position utilizing the alpha, beta, and delta in 2D search-space. Hence, the resulting position would lie within a random location within the circle.

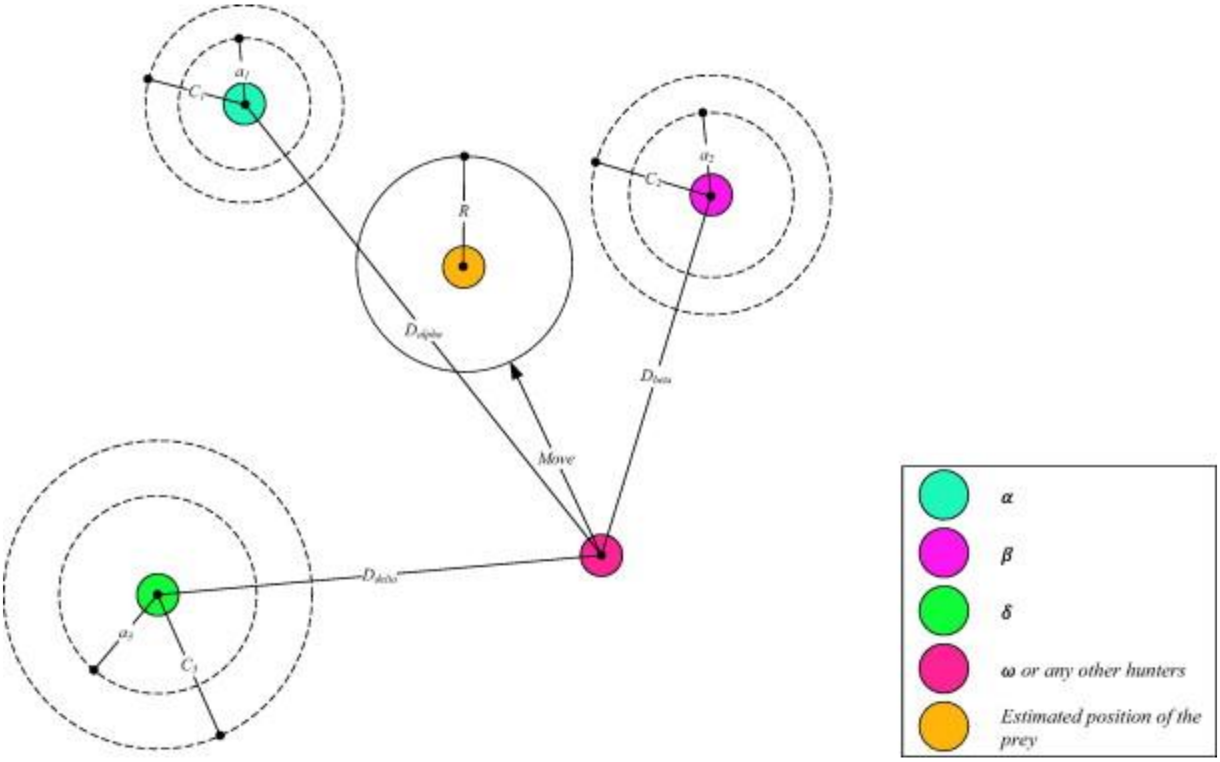


Figure 5. Updating Position (Hunting) for next search agent using best 3 solutions.

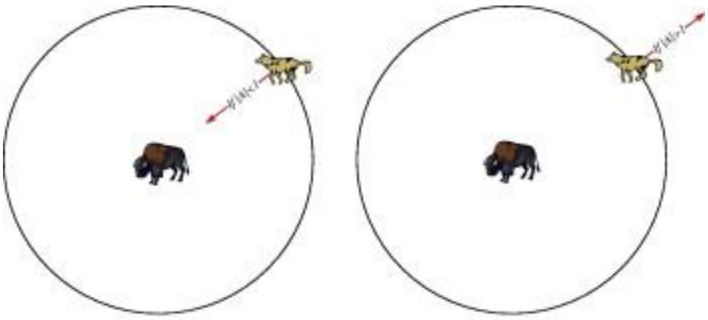
3.2.4. Attacking prey (exploitation)

When prey stops, the grey wolves finish hunting. To model stopping of prey, value of **a** is decreased, causing the variation range of **A** to be decremented by **a**.

While **a** is decremented from 2 to 0 over the course of iterations, **A** becomes a random value in the interval [-2a,2a].

3.2.5. Search for prey (exploration)

Grey wolves move with respect to the position of the alpha, beta, and delta. Wolves diverge from one another to search for prey, and converge to attack prey. In order to mathematically model divergence from the prey, **A** is set with random values greater than 1 or less than -1.



(a) (b)

Figure 6. Effects of **A** on the exploration and exploitation of GWO

3.3. Algorithm Framework

This section provides details onto the how the previously mentioned stages and steps are modelled and used in an algorithmically format to provide a solution to an optimization problem.

Algorithm: Grey-Wolf Optimization

1) Initialize population $\mathbf{X_i}$ (i = 1, 2, ..., n)
2) Initialize **a**, **A**, and **C**
3) Initialize fitness of all search agents.
4) $\mathbf{X_\alpha}$=the best search agent
5) $\mathbf{X_\beta}$=the second-best search agent
6) $\mathbf{X_\delta}$=the third best search agent
7) while (**t** < Maximum iterations)
    a) Update positions of all search agents using $\mathbf{X_\alpha}$, $\mathbf{X_\beta}$, and $\mathbf{X_\delta}$
    b) Update **a**, **A**, and **C**
    c) Calculate fitness of all search agents.
    d) Update $\mathbf{X_\alpha}$, $\mathbf{X_\beta}$, and $\mathbf{X_\delta}$
    e) Increment **t** by1
8) end while
9) finish

In brief description,

First a population is initialized with probably solutions while setting the exploration/exploitation parameters (a, A, and C) is done and they are fine-tuned for algorithm to reach better result. Choosing the top 3 best solutions as our guides, the population space is iterated through the maximum number that is required which uses the fine-tuned parameters to reach better results.

Chapter 4. Proposed Framework

This chapter gives an insight and description of the proposed framework and incorporated algorithm (which was inspired by previous works and with use of nature-inspired algorithms) to improve upon the existing optimization approaches to prolong stability and lifetime of a WSN.

4.1. Introduction

The proposed framework extends the previously explained SAECDS [9] through additional use of more efficient nature-inspired Algorithms.

Since finding a minimal Connected Dominating Set belongs to a class of NP-Hard problems, it can be solved with traditional optimization solutions and to get the best answer in polynomial time application of ‘approximation’ algorithms is necessary. While much work has been done before as mentioned in previous sections, but much work still needs to be done with more development in nature-inspired optimization techniques.

Most the recent technique being the Grey-Wolf Optimizer, which mimics the hunting mechanism of the Grey Wolves. Belonging to a class of Evolutionary Algorithms, it iteratively moves through the search space to find the best answer possible for the problem.

Proposed algorithm builds over existing research work done for constructing SAECDS algorithm, and therefore uses similar System Models and Assumptions. With basics covered with SAECDS, the core algorithm progression was changed from Genetic Algorithm to Grey-Wolf Optimizer. This change brought 2 major differences:

- Updating Fitness function: incorporating the distance of dominator node from Base station.
- Replacing Genetic Algorithm stages in SAECDS with Grey-Wolf Optimizer stages

As it is commonly known that nature-inspired algorithms have different performance on different problems, hence it was necessary to confirm that the GWO is best suited for the current scenario. To this effect, the use of two nature-inspired algorithms was considered to provide a much better comprehension and comparison between them and to visualize the improvement in efficiency and lifetime of the WSN. Therefore, along the proposed algorithm which makes use of Grey-Wolf Optimizer, another algorithm was designed with Artificial Bee Colony at its’ core.

Subsequent sections explain the System Model and Algorithm Framework, upon which furthermore assumptions, energy models, and procedural stages have been described.

4.2. System Model

This section described the mathematical assumptions and modelling used to mimic a WSN network for simulation purposes and result comparisons. The assumptions include upon how we believe the system would work, whereas energy approximations give hint towards the interaction loss by WSN nodes in a network.

4.2.1. Assumptions

The WSN system is modelled as a 2D sensing area with ***n*** homogenous sensors, of transmission radius ***r***.

The sensors spread across area of known size (X_{max}, Y_{max}). Formally,

$$\left(1 \leq i \leq n \mid S_{i_{x,y}} = ([0, X_{max}], [0, Y_{max}])\right)$$

Using probabilistic model, the strength of a signal deteriorates exponentially with distance. Formally,

$$Signal(S_i, D_j) = \begin{cases} 0\,;\, if\ (r + r_u) \leq dis(S_i, D_j) \\ e^{-\lambda\alpha^{\beta}}\ ;\ if\ (r - r_u) < dis(S_i, D_j)\ <\ (r + r_u) \\ 1\,;\, if\ (r - r_u) \geq dis(S_i, D_j) \end{cases}$$

where,

dis (S_i, D_j) = Euclidean distance between the source and destination.

α = dis $(S_i, D_j) - (r - r_u)$

λ, β = probabilistic parameters when within the interval $\{r - r_u, r + r_u\}$.

4.2.2. Energy Expenditure

Using the simple first-order ratio model [4];

Transmission of ***k***-bit message over a distance ***d*** expends:

$$E_{TX}(k, d) = E_{elec} * k + \varepsilon_{amp} * k * d^2$$

Reception of message, it expends:

$$E_{RX} = E_{elec} * k$$

Also, for data-aggregation energy expended:

$$E_{DA} = 5nJ/bit/message$$

where,

E_{elec} = energy consumption rate for radio transceiver

ε_{amp} = energy consumption rate for transmitter amplifier

4.3. Fitness Function

Inspired by the fitness / objection function used in the SAECDS [9] algorithm, with further fine-tuning, following fitness function was designed:

$$\varphi_{.}(I) = \left(\sum_{i=1}^{D}\left(\sum_{j=1}^{k} E_{TX_{j,i}} + E_{RX} + E_{DA}\right)\right) * \left(\sum_{i=1}^{D} \varepsilon_{amp} * dis(D_i, BS)\right)$$

where,

D = Total number of dominators

k = Total number of associated dominates

$E_{TX\ (node1,\ node2)}$ = Energy dissipation for transmitting data from node1 to node2

E_{RX} = Energy dissipated for receiving data

E_{DA} = Energy dissipated for aggregating data

$E_{D,\ BS}$ = Total Energy dissipated by all dominators to Base Station

dis (D_i, BS) = Euclidean distance between the dominator and Base Station

ε_{amp} = energy consumption rate for transmitter amplifier

4.4. Algorithm Framework

WSN of ***N*** sensor nodes is considered with each individual solution $\boldsymbol{I^i}$; $1 \leq \boldsymbol{i} \leq \mathrm{K} \in \boldsymbol{p}$

i.e. a fixed-length of size ***N***, where each element can either be 0 (if dominate) or 1 (if dominator). A population of ***K*** individual solutions,

$$\forall i \in \{1, \dots, K\} \ and \ \forall j \in \{1, \dots, N\}$$

$$I_j^i = \begin{cases} 1 \ if \ node \ j \ is \ dominator \\ 0 \ if \ node \ j \ is \ dominatee \end{cases}$$

Using the initialized chromosome, the population is extended using Inheritance Population Initialization (IPI) [15], where first individual is obtained by running mCDS algorithm [5].
Each individual in population has a fitness (objective) attached to it, which numerically quantifies its' effectiveness. Each allele in the individual chromosome represents a dimension for the Search Agent.

To solve for most optimized CDS following two algorithm frameworks were implemented:

4.4.1. Algorithm Framework using ABC

In the ABC algorithm, the first half of the swarm consists of employed bees, and the second half constitutes the onlooker bees. The number of employed bees or the onlooker bees is equal to the number of solutions in the swarm. The ABC is fed with an initial population of viable solutions (through mCDS algorithm and extended using IPI). To model the ABC according to the problem, the algorithm proceeds in 3 stages.

Stage 1: Initialization

In first stage, a Bee is designed to store a 'solution' and 'fitness' and all generated Bees are stored in a hive. The hive has a ***best_fitness*** value associated with it, which is calculated through the 'waggle dance' of the Bees i.e. all bees share their data between themselves and decide the best solution found. The 50% of total bees in hive are designated as Active Bees, 25% of total bees are designated as Scout Bees, and remainder of bees are designated as Inactive bees. Therefore, this stage includes following steps:

- First individual is obtained by running mCDS algorithm [5].
- Population is extended using Inheritance Population Initialization (IPI) [15]
- Each Bee is assigned an individual solution in the population.
- All Bees are collected in a Hive
- For all Bees, fitness is evaluated and best_fitness is saved in the Hive.
- 50% Bees are designated as Active Bees, 25% Bees are designated as Scout Bees, and remaining Bees are designated as Inactive Bees.

Stage 2: Iterations

During the second stage, each bee performs its function towards a better solution for a specified number of iterations or till the best solution is encountered.

- For Active Bees, they check their neighborhoods for a better solution. They do so, by swapping any random two adjacent value in the solution array and estimating its fitness. If the fitness of new combination is found to be lower, the bee accepts the new combination as its solution.
- For Scout Bees, they move over to a random combination array and estimates its fitness. If the fitness of new combination is found to be lower, the bee accepts the new combination as its solution.

To introduce a level of randomness in the algorithm, if a new solution found is worse than existing one for a bee [i.e. has higher/worse fitness] still there is a probability of 5% that the Bee will accept the new solution [but will update its previous solution configuration to the Hive]. It is done so that the algorithm can explore more solutions and avoid getting stuck at local minima.

Formally, steps can be described as:

- For total number of iterations
    - For each Bee ($\boldsymbol{B_i}$)
        - if ($\boldsymbol{B_i}$ is active bee)
            - Find a neighbor solution by swapping two consecutive random locations of Bee's solution
                - r = random(N)
                - swap ($\boldsymbol{B_i}$ [r], $\boldsymbol{B_i}$ [r+1])
            - if (new solution has better fitness than previous one)
                - Accept the new solution for the Bee with a probability of 5%
        - if ($\boldsymbol{B_i}$ is scout Bee)
            - Send the Bee to a random solution
                - $B_i[k] = random(0,1);\ \forall k \epsilon [0, N]$
            - if the new solution is valid and has better fitness than previous one
                - Accept the new solution for the Bee with a probability of 5%
    - Make Bees do ***'Waggle Dance'*** [i.e. all bees share their fitness and best one is saved]

Stage 3: Find best solution in Hive

The best solution in the Hive is calculated and returned as the outcome of the algorithm.

4.4.2. Algorithm Framework using GWO

The key driving factors in GWO algorithm are $\boldsymbol{a}$, $\boldsymbol{A}$, and $\boldsymbol{C}$. Using the description provided in [16] the factors were calculated as:

- ***a***: The value of 'a' decreases linearly with more iterations from 2 to 0. Mathematically, $a = 2 - i * \left(\frac{2}{maximum\_iterations}\right)$
- ***A***: The value of 'A' is determined through value of 'a' and a random instantaneous value. Mathematically, $\vec{A} = 2.\vec{a}.\vec{r_1} - \vec{a}$
- ***C***: The value of 'C' is solely determined through a random instantaneous value. Mathematically, $\vec{C} = 2.\vec{r_2}$

This allows the algorithm to initially have more exploration chances using the solution set, but as the algorithm moves towards the end, existing solutions and their nearby search-space is exploited for better results. Can be mathematically be seen as iteration increases, value of '***a***' decreases. Decrease in '***a***' causes a drop in the value of '***A***'. Mixed with appropriate low random value |A|<1, which by definition in [16] causes exploitation of solution-set's search-space.

The progression of algorithm is detailed in following stages

Stage 1: Initialization

During this step, all required information is gathered and a population is built of possible solutions, named as 'Individual'

Formally, steps can be represented as follows:

- First individual is obtained by running mCDS algorithm [5].
- Population is extended using Inheritance Population Initialization (IPI) [15]
- Each Search Agent is assigned an individual in the population.
- For all Search Agents, fitness is evaluated.
- Best 3 search agents are named X_α, X_β, and X_δ respectively.

Stage 2: Iterations

During this step, the algorithm iterates till a specified number of time and updates each Individual in the population set with the help of functions described in previous sections of GWO. [16]

Formally, the steps can be represented as follows:

- For each Search Agent,
    - Re-evaluate fitness
    - Choose best 3 Search Agents as X_α, X_β, and X_δ respectively.
- Linearly degrade ***a*** i.e.
    - $a = 2 - i * \left(\frac{2}{maximum\_iterations}\right)$

- For each dimension (***k***) of every Search Agent ($\boldsymbol{I_i}$), calculate
  - #for Alpha
    - r1, r2 = random ()
    - A1 = 2*a*r1-a ;
    - C1 = 2*r2
    - D_Alpha = abs (C1* $\boldsymbol{X_\alpha}$ [***k***]- $\boldsymbol{I_i}$ [***k***])
    - X1 = $\boldsymbol{X_\alpha}$ [***k***]-A1*D_Alpha
  - #for Beta
    - r1, r2 = random ()
    - A2 = 2*a*r1-a ;
    - C2 = 2*r2
    - D_Beta = abs (C2* $\boldsymbol{X_\beta}$ [***k***]- $\boldsymbol{I_i}$[***k***])
    - X2 = $\boldsymbol{X_\beta}$ [***k***]-A2*D_Beta
  - #for Delta
    - r1, r2 = random ()
    - A3 = 2*a*r1-a ;
    - C3 = 2*r2
    - D_Delta = abs (C3* $\boldsymbol{X_\delta}$ [***k***]- $\boldsymbol{I_i}$ [***k***])
    - X3 = $\boldsymbol{X_\delta}$ [***k***]-A3*D_Delta
  - if $\left(\frac{X1+X2+X3}{3}\right) < 0.5$:
    - $\boldsymbol{I_i}$ [***k***] = 0
  - else:
    - $\boldsymbol{I_i}$ [***k***]= 1
- For each Search Agent, confirm if $\boldsymbol{I_i}$ is a valid Dominating Set by
  - Checking if all dominatees have a dominator nearby
  - Checking if all dominators are connected (through minimum spanning tree algorithm)

Stage 3: Find Best Individual in population as Result

Towards the end, the last stage includes to gather the best solution (the Alpha) from the population-set as the outcome of the algorithm.

A visual flowchart description is given below in Figure 7.

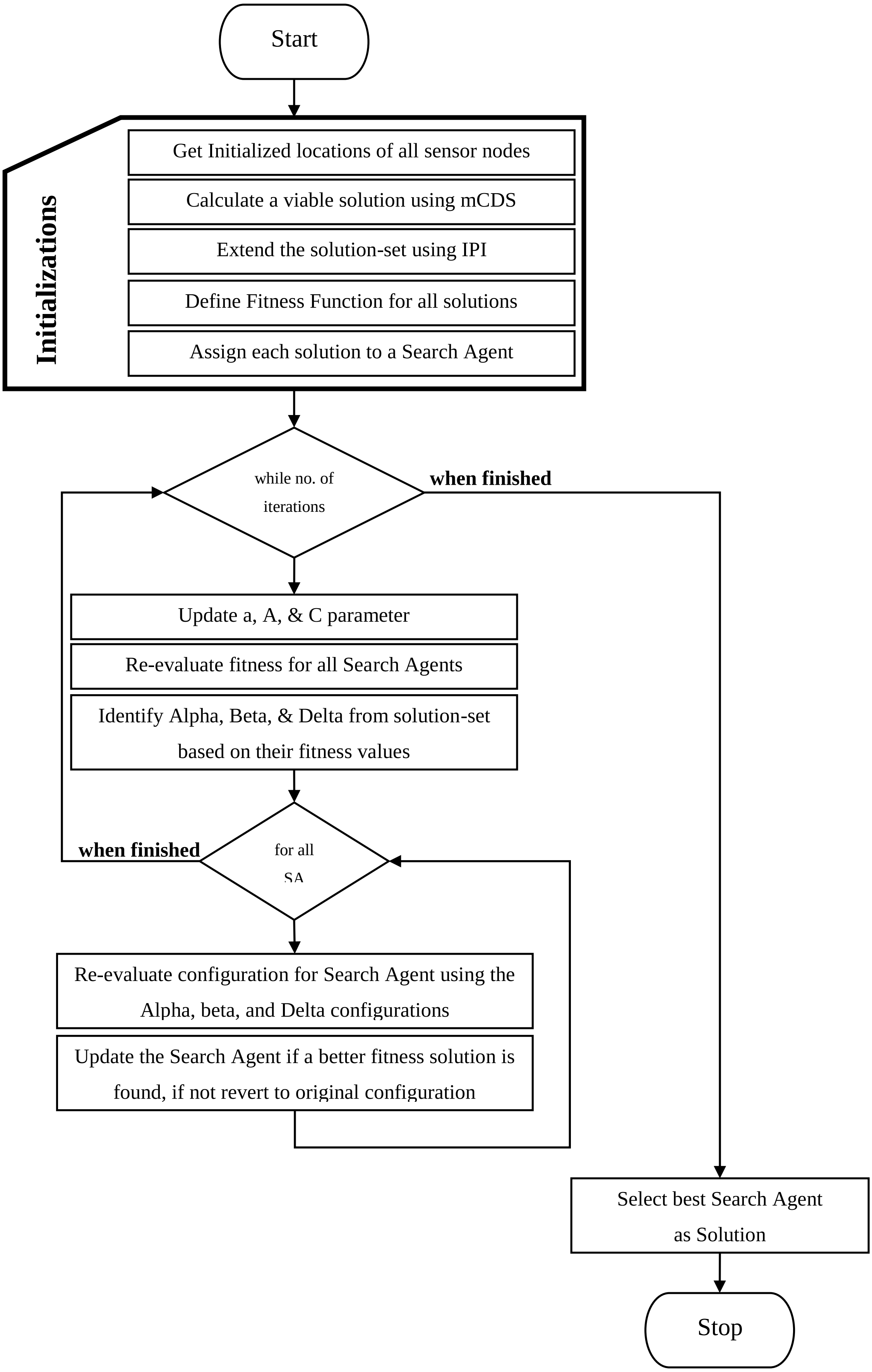


Figure 7. Flowchart description of the proposed algorithm

For better understanding on how the algorithm progresses, a simple sample simulation example is explained.

4.5. Discussion on Proposed Framework

The collection of ***N*** sensor nodes distributed randomly in a ***M*M*** area with a base station at middle of the area is considered. A viable solution is found using a mCDS algorithm [15] and extended to a required population size.

Each viable solution is assigned to a Search Agent which stores the solution information in a bit-array of length ***N***, where each bit represents if its respective node is a dominator (assigned a value of 1) or dominatee (assigned a value of 0). Furthermore, each Search Agent is assigned with an evaluated Fitness Value.

For better understanding of the algorithm, an instance is considered and explained mathematically on how it progresses. E.g. A single Search Agent "$\boldsymbol{I_i}$" (if we consider 20 nodes with 5 dominators) it'd look like as described in following Figure 8.

$\boldsymbol{I_i}$	1 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 1 0 0	: 0.45
↓	↓	↓
Search Agent 'i'	Bit-array representation of a solution	Fitness of Search Agent

Figure 8. Representation of a viable solution for a Search Agent

Now, for execution of the algorithm, a peek into an instance of an iteration is analyzed. For every iteration of the algorithm, the best three Search Agents (lowest fitness value) are designated as Alpha, Beta and Delta of the solution set, and are utilized for further evaluations. From the above example, for snapshot at [***N***=20, ***maximum_iterations***=20, ***k***=1] the calculations would proceed as follows

- For 1st iteration out of 20 iterations:
  - $a = 2 - i * \left(\frac{2}{maximum\_iterations}\right) \ = 2 - 1 * \left(\frac{2}{20}\right) \ = \ 1.9$
  - For 1st allele of Search Agent $\boldsymbol{I_i}$
    - For Alpha
      - Assuming r1=**0.3** & r2=**0.6**
      - A1 = 2*a*r1-a = 2*1.9*0.3-1.9 = **-0.76**
      - C1 = 2*r2 = 2*0.6 = 1.2

- D_alpha = abs (C1*$\boldsymbol{X_\alpha}$ [1]- $\boldsymbol{I_i}$ [1]) = abs (1.2*1-1) = **0.2**
    - X1 = $\boldsymbol{X_\alpha}$ [1]-A1*D_Alpha = 1-(-0.76)*0.2 = **1.152**
  - For Beta
    - Assuming r1=**0.1**, r2=**0.2**
    - A2 = 2*a*r1-a = 2*1.9*0.1-1.9 = **-1.52**
    - C2 = 2*r2 = 2*0.2 = **0.4**
    - D_Beta = abs (C2* $\boldsymbol{X_\beta}$ [1]- $\boldsymbol{I_i}$[1]) = abs (0.4*1-1) = **0.6**
    - X2 = $\boldsymbol{X_\beta}$ [1]-A2*D_Beta = 1-(-1.52)*0.6 = **1.912**
  - For Delta
    - Assuming r1=**0.7**, r2=**0.9**
    - A3 = 2*a*r1-a = 2*1.9*0.7-1.9 = **0.76**
    - C3 = 2*r2 = **1.8**
    - D_Delta = abs (C3* $\boldsymbol{X_\delta}$ [1]- $\boldsymbol{I_i}$ [1]) = abs (1.8*0-1) = **1**
    - X3 = $\boldsymbol{X_\delta}$ [1]-A3*D_Delta = 0-0.76*1 = **-0.76**
  - $X_i = \left(\frac{X1+X2+X3}{3}\right) = \left(\frac{1.152+1.912-0.76}{3}\right) = 0.768$:
  - Since Xi > 0.5
    - $\boldsymbol{I_i}$ [1] = 1
  - Repeat the above procedure for all ***N*** allele of all Search Agents
- Repeat the procedure till ***maximum_iterations.***

In the following Figure 9, a description of the above process is depicted graphically for better understanding. The mathematical part is encapsulated and abstracted, and it can be seen that from the existing solution-set, top 3 Search Agents are chosen as Alpha, Beta, and Delta. Using whose values [mixed with some random values, difference with current Search Agent and fine-tuned driving factors], the resultant Search Agent's new value is estimated.

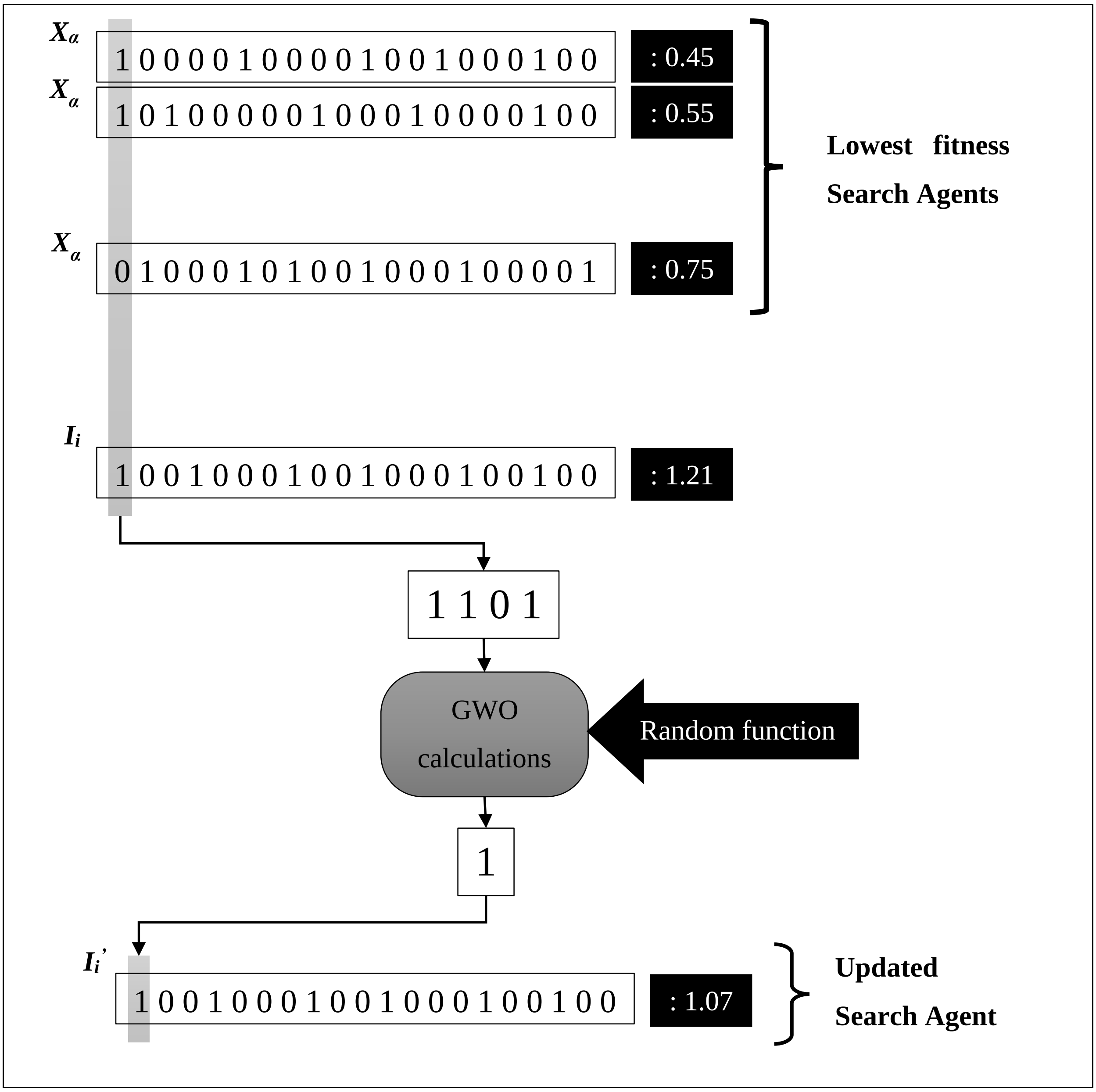


Figure 9. An instance of algorithm progression

Following the discussion, the algorithm continues for all allele of all Search Agents for a specified number of iterations. Through the fine-tuning of driving factors [***a***, ***A***, & ***C***], initially the search-space is explored more using the best 3 solutions as guides, but in later stages of the algorithm the search-space near the solution-sets are exploited.

In the next chapter, the implementation of the proposed framework and algorithm is detailed providing and insight on how it works and progresses to find the optimal solution to the WSN energy-constraint problem.

Chapter 5. Implementation & Simulation

As it is commonly known that nature-inspired algorithms have different performance on different problems, hence it was necessary to validate the proposed algorithm for the problem and to compare it with other existing ones while confirming if it any other nature-inspired optimization algorithm would be better suited in this scenario.

To this effect, 3 Algorithms were implemented

- **Stability Aware Evolutionary CDS**
- **Constructing CDS using Artificial Bee Colony**
- **Constructing CDS using Grey Wolf Optimization**

Implemented Algorithms were simulated and made run through a number of cycles and were compared along 4 different parameters:

- Number of Dominators ($|D|$)
- Total Energy Loss of Dominators (D_{DE})
- Total Energy Loss in Network (NW_{DE})
- Difference between first Dominator failure and last Dominator Failure (D_{die})

Total number of Dominators represents the number of clusters in the network. The lower this value is, the less susceptible the network is to failure (as the dominator nodes are generally the critical point of failures).
Total Energy Loss in Dominators and Network represents the per-round (each sensor node sends at least one message to base station). The lower this value is, the better network performs as the cost of transmission is reduced, putting less strain on the network. Hence, less Loss of Energy means more Energy Conservation.
Lower the Difference between first Dominator failure and last Dominator failure, more stable the network as. As reducing this value, the network's unstable time gets reduced, in turn prolonging the stability of the WSN.

Since, in previous papers it was already proved that SAECDS gave better results in terms of stability and helped prolong the lifetime of a WSN [9].Using that as a base, we try to surpass the performance of the SAECDS through more fine-tuning and using more appropriate nature-inspired algorithm for this scenario.

5.1. Implementation Constraints

Implement was done in python2.7, with numpy (for large dataset handling) and matplotlib (for generation output graphs) as add-ons.

5.2. Simulation Settings

To effectively and efficiently simulate, ***N*** sensor nodes were randomly deployed in the area of ***M*M*** with a centrally located Base Station (***BS***).
A test-bed of 5 randomly generated deployments with following configurations were considered:

- **Communication Range** = [20, 25, 30, 35, 40]
- **Sensor Field Dimension** = [40*40, 60*60, 80*80, 100*100]
- **Number of Sensor Nodes** = [100,120,140,160]

With normal value of each configuration being

- **Communication Range** = [20]
- **Sensor Field Dimension** = [100*100]
- **Number of Sensor Nodes** = [100]

Hence each protocol / algorithm is compared over 65 random test instances, with identical characteristics of radio mode, probabilistic communication, and other system model assumptions for fair comparison.

The other constants required were initialized according to a simple first-order radio model [4].

- E_{elec} = 50 nJ / bit
- ε_{amp} = 100 pJ / bit / m2
- E_0 = 0.25 J
- E_{DA} = 5 nJ / bit / report
- k (set size) = 4000 bits

Other required probabilistic communication parameters were initialized as:

- R_u = 5 units
- λ, β = 0.5

The Simulation was allowed to be run till every node in network transmits at least one message to Base Station. Then simulation is allowed to continue from "1st Dominator failure" till "Last Dominator failure" to evaluate the length of unstable period of Wireless Sensor Network.

Furthermore, for 20 Generations, a Population size of 20 individuals is allowed to evolve. Other parameters for the SAECDS algorithm were unchanged and followed as described in [8].

5.3. Evaluations

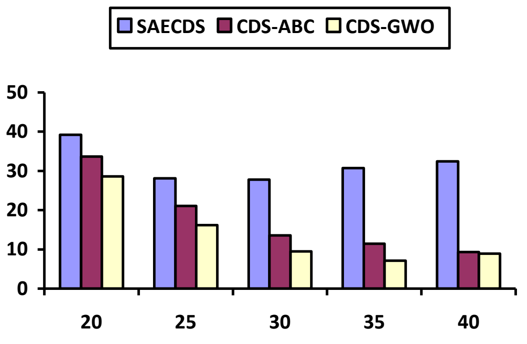
To comprehend performance simulation result of various network test instances and test the behavior of different algorithms, Table 1, Table 2, and Table 3 quantify statistically performance of proposed framework and algorithm ***CDS-GWO,*** with existing ***SAECDS*** and another algorithm ***CDS-ABC***.

The performance of SAECDS, CDS-ABC, and CDS-GWO for different communication ranges are given in Table 1. [A network field of 100 × 100 with 100 sensor nodes is assumed to be constant]

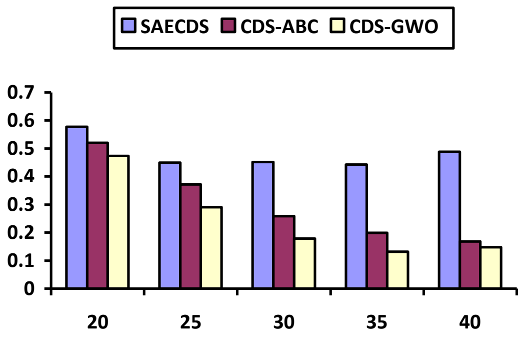
R	SAECDS				CDS-ABC				CDS-GWO			
	$\|D\|$	D_{DE}	NW_{DE}	D_{die}	$\|D\|$	D_{DE}	NW_{DE}	D_{die}	$\|D\|$	D_{DE}	NW_{DE}	D_{die}
20	39.2	0.578	1.563	7.3	33.7	0.520	1.440	6.4	28.6	0.474	1.330	5.1
25	28.1	0.450	1.430	5.4	21.1	0.372	1.160	4.7	16.2	0.291	0.942	4.3
30	27.8	0.452	1.550	6.7	13.6	0.259	0.928	4.4	9.5	0.178	0.644	3.9
35	30.7	0.443	1.759	5.1	11.4	0.199	0.807	3.7	7.1	0.131	0.529	3.4
40	32.4	0.488	2.122	6.2	9.3	0.168	0.730	4.1	8.9	0.148	0.679	2.5

Table 1. Performance comparison over different Communication Ranges

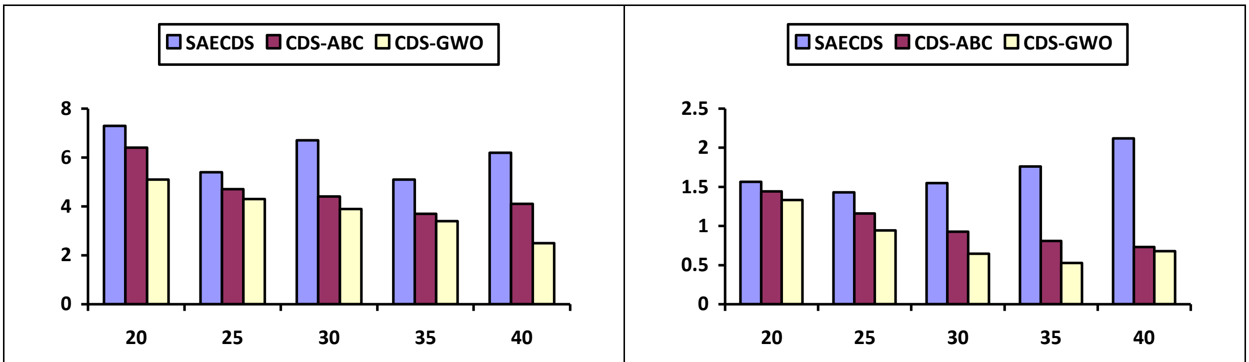
Figure 7 provides a visual representation of the results for quick overview for Performance analysis over different communication ranges (***r***).



a) Analysis of |***D***| over ***r***



b) Analysis of D_{DE} over ***r***



c) Analysis of D_{die} over r d) Analysis of NW_{DE} over r

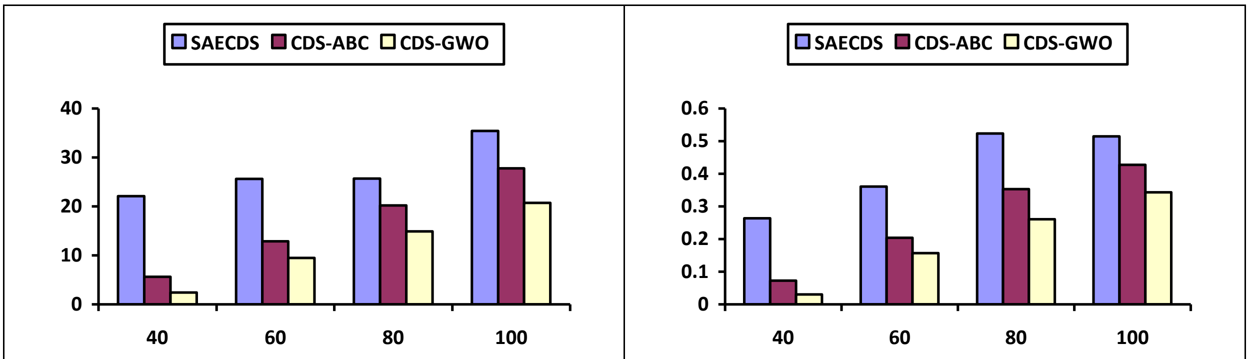
Figure 10. Performance Analysis over r

The performance of SAECDS, CDS-ABC, and CDS-GWO for different network field sizes are given in Table 2. [A communication range of 20 with 100 sensor nodes is assumed to be constant]

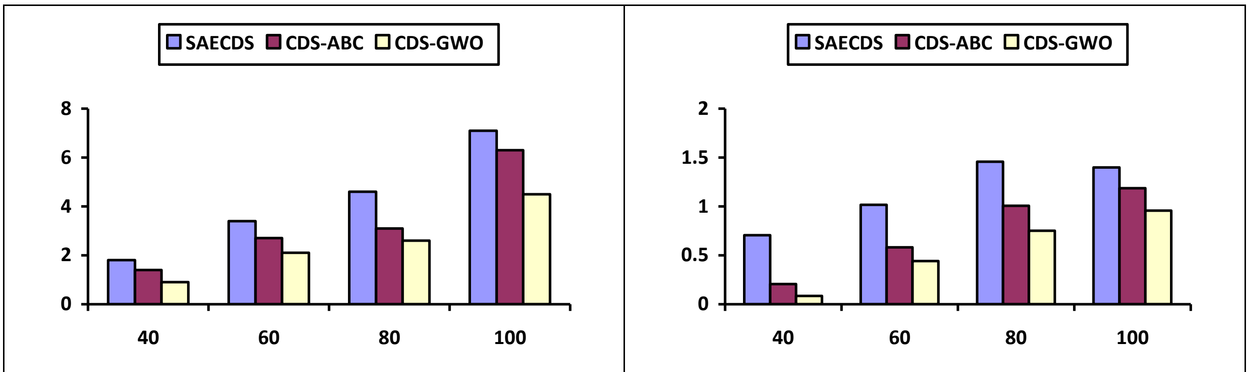
M	SAECDS				CDS-ABC				CDS-GWO			
	$\|D\|$	D_{DE}	NW_{DE}	D_{die}	$\|D\|$	D_{DE}	NW_{DE}	D_{die}	$\|D\|$	D_{DE}	NW_{DE}	D_{die}
40	22.1	0.264	0.707	1.8	5.6	0.073	0.207	1.4	2.4	0.030	0.085	0.9
60	25.6	0.361	1.015	3.4	12.9	0.204	0.582	2.7	9.5	0.157	0.442	2.1
80	25.7	0.524	1.458	4.6	20.2	0.353	1.006	3.1	14.9	0.261	0.753	2.6
100	35.4	0.515	1.399	7.1	27.8	0.427	1.185	6.3	20.7	0.343	0.958	4.5

Table 2. Performance comparison over different Network Field Sizes

Figure 8 provides a visual representation of the results for quick overview for Performance analysis over different network field sizes (***M*M***).



a) Analysis of $|D|$ over $M*M$ b) Analysis of D_{DE} over $M*M$



c) Analysis of $\boldsymbol{D_{die}}$ over $\boldsymbol{M*M}$

d) Analysis of $\boldsymbol{NW_{DE}}$ over $\boldsymbol{M*M}$

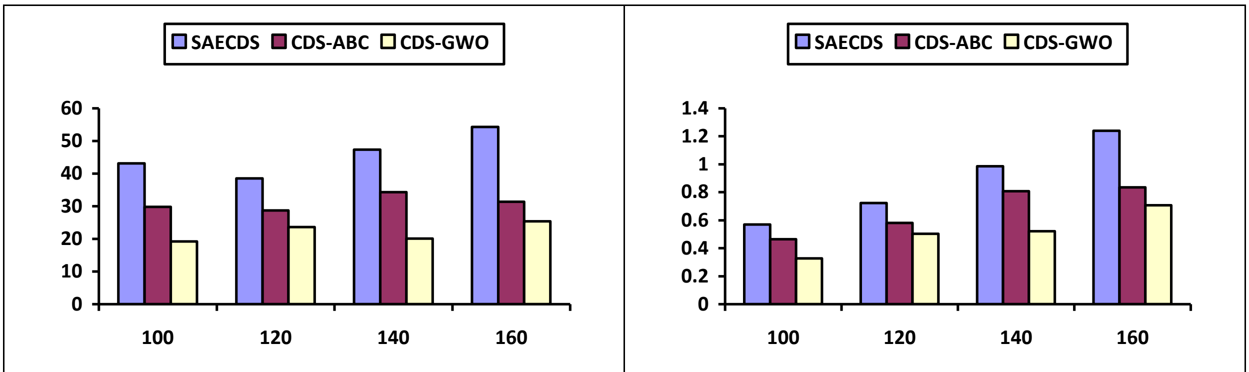
Figure 11. Performance Analysis over $\boldsymbol{M*M}$

The performance of SAECDS, CDS-ABC, and CDS-GWO for different number of Nodes are given in Table 3. [A network field of 100 × 100 with communication range of 20 is assumed to be constant]

N	SAECDS				CDS-ABC				CDS-GWO			
	|D|	**D_{DE}**	**NW_{DE}**	**D_{die}**	**|D|**	**D_{DE}**	**NW_{DE}**	**D_{die}**	**|D|**	**D_{DE}**	**NW_{DE}**	**D_{die}**
100	43.1	0.570	1.551	9.4	29.8	0.464	1.301	6.1	19.2	0.328	0.922	4.9
120	38.5	0.722	2.021	4.8	28.7	0.580	1.650	4.4	23.6	0.504	1.431	4.1
140	47.4	0.985	2.845	3.1	34.3	0.808	2.342	2.7	20.1	0.522	1.527	2.4
160	54.3	1.240	3.557	3.2	31.4	0.835	2.435	2.5	25.4	0.706	2.072	2.2

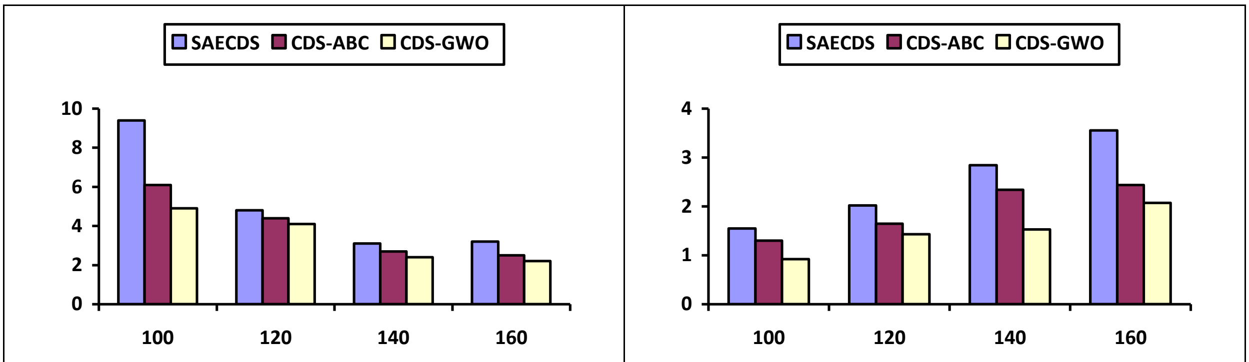
Table 3. Performance comparison over different Node Numbers

Figure 9 provides a visual representation of the results for quick overview for Performance analysis over different number of nodes (***N***).



a) Analysis of $|\boldsymbol{D}|$ over $\boldsymbol{N}$

b) Analysis of $\boldsymbol{D_{DE}}$ over $\boldsymbol{N}$



c) Analysis of D_{die} over N

d) Analysis of NW_{DE} over N

Figure 12. Performance Comparison over N

5.4. Results

During comparison, it is found that compared to the existing work [9] the proposed framework and algorithm consumes less energy in terms of both total network dissipated energy dominators dissipated energy.

The proposed algorithm succeeds where the previous work failed i.e. energy dissipation is reduced in the proposed work, while at same time reducing the number of dominators in the network.

Chapter 6. Conclusion & Future Work

Since there are various limitations over a Wireless Sensor Network e.g., sensor node energy, geographical topology, routing bottlenecks, it has triggered a boost in research over development of more effective and energy-aware protocols and algorithms for WSNs.

Amongst all works, it was found that a Connected Dominating Set based topology control proved to be an effective technique. But still there were few weaknesses that still needed to be overcome such as, incorporating energy needs of the network, dominator failure and topology changes.
Finding an appropriate balance between both proved to be a challenging effort but a novel solution in form of proposed work was found, which incorporated a minimally Connected Dominating Set and enhanced it considering the energy constraints.

The proposed work, prolonged the network stability through reduction of un-stability period (the time between first and last dominator failure) and provided efficient energy conversation (reduced per round network energy dissipation).
Also, the simulations reported that the proposed work gave better results than recent works.

6.1. Future Work

While many issues were overcome, there is still a need for more focus to explore more complex models as the major requirement is hindered by the fact the problem in itself is NP-Hard at its core, and there exists no polynomial time solution towards this.
The Future research work could incorporate more constraints into the equation to prolong stability and overall network lifetime because of different demands of diverse WSNs according to the environment they are implemented in.

First, other fitness and objective functions could be designed in a singular or conjoined manner and can address additional factos. Second, evaluation and evolution of the solutions could be improved through further fine-tuning of algorithms e.g. varying location of Base Station may have an impact on more efficient solution.

REFERENCES

[1] a. S. T. H. a. P. J. S. T. W. Haynes, Fundamentals of Domination in Graphs, New York: Marcel Dekker Inc., 1998.

[2] X. J. a. H. Y. Z. Yuanyuan, "Energy efficient distributed connected dominating sets construction in wireless sensor networks.," *ACM International Conference on Communications and Mobile Computing,* p. 797–802, 2006.

[3] J. H. Holland, "Genetic Algorithms," *Scientific american,* vol. 267, p. J. H. Holland, 1992.

[4] A. C. a. H. B. W. R. Heinzelman, "Energy efficient communication protocol for wireless microsensor networks," in *33rd Annual Hawaii International Conference on System Sciences*, 2000.

[5] a. S. K. S. Guha, "Approximation Algorithms for Connected Domimating Sets," *Algorithmica,* vol. 20, pp. 374-387, 1998.

[6] Z. C. S. J. R. R. a. Y. P. J. He, "A Genetic Algorithm for Constructing a Reliable MCDS in Probabilistic Wireless Networks," in *WASA: 6th international conference on Wireless algorithms, systems, and applications*, Chengdu, China, 2011.

[7] S. V. S. T. M. Rai, "A power aware minimum connected dominating set for wireless sensor networks," *Journal of Networks,* vol. 4, no. 6, pp. 511-519, 2009.

[8] M. I. S. a. L. T. Y. S. Hussain, "Constructing a CDS-Based Network Backbone for Energy Efficiency in Industrial Wireless Sensor," in *12th IEEE International Conference on High Performance Computing and Communications (HPCC '10)*, Melbourne, Australia, 2010.

[9] S. O. Enan A. Khalil, "Prolonging stability period of CDS based WSNs," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2015 International*, Dubrovnik, Croatia, 2015.

[10] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979, p. 190.

[11] S. Mirjalili, "How effective is the Grey Wolf optimizer in training multi-layer perceptrons," in *Applied Intelligence,* 2015, pp. 1-12.

[12] E. Emary, "Feature Subset Selection Approach by Gray-Wolf Optimization," in *Afro-European Conference for Industrial Advancement*, 2015.

[13] P. E. K. Shankar, " A secure visual secret share (VSS) creation scheme in visual cryptography using elliptic curve cryptography with optimization technique.," *Australian Journal of Basic & Applied Science,* vol. 9(36), pp. 150-163, 2015.

[14] Z. M. T. Y. Z. M. N. X. a. S. K. Lu G, "Approximation algorithms for the connected dominating set problem in unit disk graphs," *Journal of Electronic Science and Technology of China,* vol. 3, no. 7, pp. 214-222, 2009.

[15] J. P. a. C. M. Mohanty, "A distributed greedy algorithm for construction of minimum connected dominating set in wireless sensor network," *Applications and Innovations in Mobile Computing (AIMoC), IEEE,* 2014.

[16] S. M. M. a. A. L. S. Mirjalili, "Grey Wolf Optimizer," *Advances in Engineering Software,* vol. 69, pp. 46-61, 2014.

[17] J. K. a. R. Eberhart, "Particle swarm optimization," *IEEE International Conference on Neural Networks,* pp. 1942-1948, 1995.

[18] M. B. a. T. S. M. Dorigo, "Ant colony optimization," *Computational Intelligence Magazine, IEEE,* vol. 1, pp. 28-39, 2006.