DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2019

Topology optimization for distributed consensus in multi-agent networks

Oskar Hahr and Johan Niklasson

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Topology optimization for distributed consensus in multi-agent networks

Authors

Oskar Hahr
Information and Communication Technology
KTH Royal Institute of Technology

Johan Niklasson
Computer Science and Engineering
KTH Royal Institute of Technology

Swedish title

Topologioptimering för distribuerad konsensus i multiagent-nätverk

Date

June 7, 2019

Supervisors

Robert Lagerström
KTH Royal Institute of Technology

Carlo Fischione
KTH Royal Institute of Technology

Hossein Shokri Ghadikolaei
KTH Royal Institute of Technology

Examiners

Örjan Ekeberg
KTH Royal Institute of Technology

Carlo Fischione
KTH Royal Institute of Technology

Abstract

Distributed networks, meaning a network in which several agents work together unanimously to perform some task in order to reach goals has become a field with a wide range of applications. One such applications may exist in the form of drones with a purpose of observing and detecting forest fires.

In such applications it can be of paramount importance to be able to agree over some opinions or values between the agents. This value could be something such as event detection or a general direction to fly in. However in such a network there might not exist a central hub and it would not be possible for all drones to communicate directly with each other. In order for such a network to be able to reach consensus or agreement, values have to be exchanged between the agents.

This thesis focuses on a subset of this problem known as distributed averaging. In the thesis it is investigated how a networks ability to detect forest fires and communicate both efficiently and quickly can change when the number of agents are adjusted in the network.

The results showed that, when operating in a fixed area, for a small network of drones the increasing effective energy cost per drone were higher, than that of a larger network. It was also discovered that the speed at which a network could reach an agreement was not necessarily affected by the size of the network. But as the field area being observed was increased, adverse effects were observed in terms of communication and event detection.

Keywords

Multi-agent networks, Distributed systems, Wireless Sensor Network, Consensus averaging, Unmanned Aerial Vehicle, Graph-Theory.

Sammanfattning

Distribuerade nätverk bestående av flera agenter som har som uppgift att tillsammans nå gemensamma resultat har blivit allt mer populärt. Ett sådant användningsområde är hur drönare kan användas för att observera och upptäcka skogsbränder över en given yta.

I en sådan tillämpning är det av stor vikt att drönarnätverket kan kommunicera och kongruera över värden nätverket delar med varandra. Dessa värden kan representera händelser som nätverket har som uppgift att upptäcka eller en riktning för drönarna att flyga i. Det är inte alltid garanterat att det finns en central kommunikationscentral för sådana nätverk, utan blir beroende på att kommunicera med varandra för att utbyta och kongruera över värden.

Den här rapporten fokuserar på en avgränsad del av det ovanstående problemet som kallas för distribuerat konsensusvärde (eng. distributed averaging). Rapporten undersöker hur ett sådant nätverks konvergeringsförmåga, totala energikostnad samt täckning påverkas när fler drönare tillförs till nätverket.

När arbetsytan var satt till statisk storlek visade resultaten att den tillförda energikostnaden per drönare var högre för små nätverk än för större nätverk. Det visades också att hastigheten som nätverket når ett kongruerande värde inte nödvändigtvis påverkas av storleken av nätverket. När arbetsytan ökade i takt med storleken på nätverket observerades däremot motsatt effekt för energikostnad och hastigheten för att nå ett konsensusvärde.

Nyckelord

Multiagent-nätverk, Distribuerade system, Trådlösa sensornätverk, Konsensusvärde, Drönare, Grafteori.

Acknowledgements

We want to thank Hossein Shokri Ghadikolaei and Rong Du for the invaluable support and supervision in the form of suggestions, encouragement and feedback that we have received during the work on this report.

List of Notations

$V(G)$	All the nodes in the graph $G(V, E)$, such as $\forall v \in V$. Used when emphasizing on *all the nodes in the graph* G.
$E(G)$	All the edges in the graph $G(V, E)$, such as $\forall e \in E$. Used when emphasizing on *all the edges in the graph* G.
$\|G\|$	The number of nodes in the graph $G(V, E)$.
$d(v, y)$	The shortest distance, in terms of number of edges, between the two nodes v and u, where $v, u \in V(G)$.
$\mathcal{E}(v)$	Eccentricity of the node $v \in V(G)$. The longest shortest distance between the node v and any other node in $V(G)$. Declared at page 7.
$D(G)$	Diameter of the graph $G(V, E)$. The longest shortest distance between any two nodes in the graph G. Declared at page 8.
$c(e)$	Edge cost of the single edge $e \in E(G)$. Declared at page 9.
$C(G)$	Total edge cost of all edges in the graph $G(V, E)$. The accumulated cost of all edges in the graph G. Declared at page 9.
$\mu(G), \lambda$	Convergence rate for the graph G. A value that determines how fast the graph (network) G can reach an average consensus. Declared at page 9.
$W(G), w$	Consensus energy for the graph G. The total energy needed for a network to reach a consensus average. Declared at page 10.

Contents

1 Introduction

In Distributed Computing, it is sometimes necessary to coordinate several agents or systems in order to reach a common goal by agreeing on some opinion across the entire network. These agents or systems can exist in many different application areas and perform a variety of different tasks such as: Wireless Sensor Networks, Unmanned Aerial Vehicles (UAVs) or Distributed Machine Learning [1, 2].

Consider that some agents working together in a network for: security-, geographical-, or energy constrains cannot communicate with all the other agents. The problem of how such a network can agree to a common opinion is known as the Distributed Consensus Problem [2, 3]. This report will focus on a subset of that problem, known as the Distributed Averaging Problem. The Distributed Averaging Problem entails that a number of agents are independently performing some tasks, in which they will reach local results [2]. These results may be measurement of temperatures, relative-position or an image [3]. The agents then need to share their local results with other agents through communication, in order to find out the average of their own result and the results of all other agents in the network. When all agents have found the average, the result is said to be reflective of the entire system [2].

The result that is agreed upon can for example represent: a direction or speed that a UAVs should move in, or a specific time that a network of clocks should synchronize to [2]. It is of paramount importance that such decisions are able to be made quickly, to not let the performance of the application suffer or malfunction.

1.1 Distributed Averaging Problem

This report will focus on an iterative algorithm that can be used to solve the Distributed Averaging problem, where each node communicates its value to all its neighbours [2]. More specifically each node in the network starts with an initial value x_0, and over a number of iterations each node will update its value to be the

average of its own value, and the values of the neighbouring nodes. This process is repeated until all the nodes in the network have agreed (converged) to a common value which represents the consensus for the entire network. See Section 2.2 for a more in depth look at how the algorithm is able to converges to a value.

The number of iterations which are required for the network to reach consensus is determined by the network structure such as how the nodes are connected to each other [4]. If a node is left without any connections the network will never reach an agreement, while a complete graph will reach consensus after only one iteration. There is however a trade-off for each connection in the network, since every connection between two nodes has a cost related to the energy consumption for communicating between each other. If a node has more connections, it is able to communicate its value with a larger number of the other nodes in the network, but the many different connections could lead to higher levels of energy consumption. For a network consisting of many different agents such as drones or UAVs, this could lead to agents more frequently having to be brought down for recharging. A goal for such a network should be to optimize the speed at which the network can communicate a result, while at the same time making sure that the up time of the network does not suffer.

For example, Figure 1.1 represents a network where a number of drones have been placed in a fixed area without any connections between them. In Figure 1.2 the same network has been optimized by adding connections between drones in such a way that they can quickly reach a consensus while also keeping the energy costs for communicating down. Adding more connections can improve the convergence rate but worsen the energy consumption. Removing connections will improve the energy consumption but can worsen the convergence rate.

1.1.1 Optimization problem

An optimization problem is a problem where the goal is to find the best solution over some given objective function $f(x)$. The goal of most optimization problems can often summarized as minimizing or maximizing the objective function $f(x)$ over some other function $g(x)$ as constraints [5].

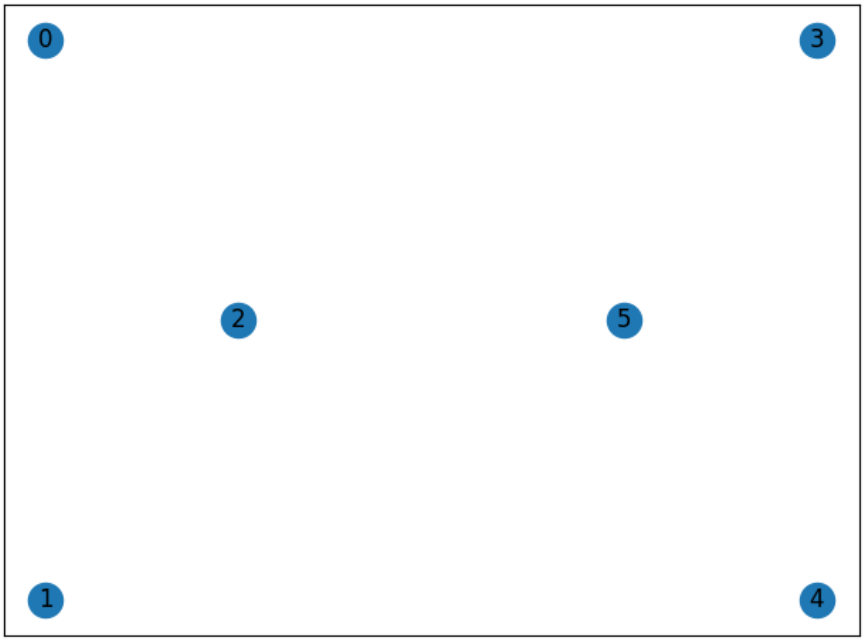

Figure 1.1: An example of an empty network containing 6 nodes

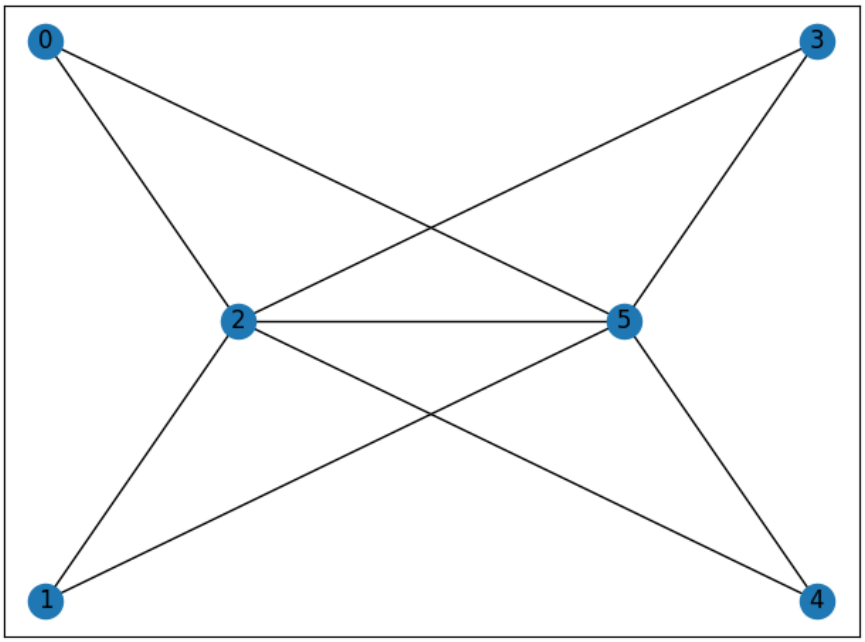

Figure 1.2: An optimized version of the network presented in Figure 1.1

A classical optimization problem is the Travelling salesman problem, where the objective is to find the shortest path between n number of cities without visiting any city twice [6, p. 1583]. There are then $n!$ different ways to visit every city, and the objective function to minimize is the total distance travelled.

1.2 Problem

Given a distributed network tasked with observation of some area, such as a forest. The agents could be instructed to collect and agree upon some information, they could then on top of this be tasked to detect if a certain event occurs. This event could be something like a wildfire, in which it is crucial that the network first and foremost detects the fire and then is able to inform the rest of the network. In an ideal world, all the agents in the network would have an unlimited detection range and be able to communicate with other parties in the network no matter the distance or number of other agents. In such a scenario this type of task would be trivial, as it would be as simple as just placing an agent in charge of observing the forest and have it communicate its results.

However in real world applications, most of the time such a solution would be impossible to implement. Restrictions such as detection range, communication range, or energy cost may be imposed on the network . In a such a scenario, all agents may not be able to communicate directly with each other, as a direct results of said restrictions. Trade-offs have to be considered, where the number of agents may increase the coverage of the forest, while at the same time having a negative effect on convergence rate and energy costs. How does one create a topology for such a network, to allow for the network to quickly detect the event of a fire, while at the same time taking things such as converge speed, and energy cost constraints into consideration? Improving on the aforementioned metrics, would potentially allow for strides to be made in a networks overall ability to monitor wildfires. Being able to perform this type of monitoring is key when presented with the difficult challenge of fighting a large scale wildfire[7].

1.3 Purpose and goal

The purpose of this report is to evaluate how introducing more agents to a network affects the overall performance on the network in terms of: energy cost, coverage and convergence speed, which would show how network topologies is affected by when expanded. The goal of the report is to be able to identify how the metrics described in the purpose affects topology structures that allow for a network to have a high probability of picking up an event in the area being observed, while at the same time being optimized in terms of convergence rate and energy costs. If these results shows any specific patterns it can be identified how forming these network structures can be done for a better convergence rate or consensus energy with respect of keeping the coverage of the network.

1.4 Research question

The research question for this report is based on the purpose and goal which aims to identify the change of metrics that is affected by expanding a network that is optimized on consensus energy. Therefore the research question is defined as *How does introducing more agents to a multi-agent network affect the energy cost, coverage and convergence speed?*

1.5 Related work

There has been a wide range of work done on the area of distributed network optimization [7–11]. The work made from two reports within the area will be presented below.

In a study by Zhongjie Lin & Hugh H. T the problem of monitoring wildfires is investigated. A solution to solve that problem is given in the form of a multi-agent UAV network [7]. In the report the UAVs deploy in a distributed fashion, and consensus based methods are applied to allow the UAVs to cooperate on a group level [7]. The UAVs were assumed to not be able to communicate outside of certain ranges and were deployed within groups. Several groups were deployed in which

the UAVs cooperated locally to observe the perimeter of a wildfire [7]. Whenever two groups were within distance of one another, local results would be able to be exchanged in a consensus based manner between the groups [7]. Mathematical models were built to optimize both how the UAVs worked within the groups as well as between different groups [7]. The results of the models were then validated by simulation scenarios, in which a wildfire was placed within a certain area and a fixed number of UAVs were deployed in groups to observe the fire. Some of the contributions made by this report is to explore how group based communications can be used to improve the mission performance of the individual UAVs and how wildfire observation can be seen as a distributed consensus problem [7].

Another study by Ho Dac-Tu, et al, explores how a network consisting of wireless sensors that collect and relay information to UAVs can be optimized [8]. The study focuses on a scenario in which data is being gathered by a number of wireless sensors, and UAVs are tasked with travelling around the network to collect that information from the sensors [8]. The study evaluates how two different optimization algorithms perform in terms of optimizing energy consumption and the travel time of the UAVs [8]. The authors state that energy conservation becomes far more difficult when dealing with large scale deployment of networks [8]. The results concluded that energy consumption of a network and conversely life-time of the network can be extended [8]. This was related to one of the optimization techniques being superior to the other in regards to the metrics chosen, and that the gap between them became more evident for larger networks [8].

1.6 Outline

Chapter 2 presents in-depth theoretical background of the report. In Chapter 3, the methodologies are presented and motivations are given for why a particular method was used to answer the problem statement. Chapter 4 contains the results that came from the method and Chapter 5 contains the discussion on the interpretation and analysis of the results. Finally Chapter 6 will present the conclusions drawn from the report including ethical concerns, suggestions for future work, and some final words.

2 Background

The background defines the various terms and definitions that's used throughout the report.

2.1 Graph Theory

2.1.1 Graph Representation of a Network

A graph is a data structure that can be described as two sets $G = (V, E)$ which represent a number of nodes V and the edges E which represent connections between these nodes [12]. There are many ways to represent different forms of graphs, the one that this report will delve further into is known as a "Simple Undirected Graph". What this entails in simple terms is [12]:

- Edges can be traversed in both directions.
- No pair of nodes may have more than one edge directly connecting them together.
- No node may have an edge leading to itself.

A common way to represent different types of networks is by representing them as graphs. This can be applied in areas such as social networks or to represent data, electricity, or water flow in some form of network [12, 13]. The reason that this is done is to make it easier both to visualize the network, as well as perform analytically tasks on them such as to test network connectivity [14].

2.1.2 Eccentricity

The eccentricity is the longest path from a given node v in a graph $G = (V, E)$ where $v \in V(G)$ to any other node connected to the same component. This can be seen as the following:

$$\mathcal{E}(v) = \forall u \in V(G) : max(d(v, u)),$$

where $d(v, u)$ is the shortest distance between the two nodes v and u in the graph G.

2.1.3 Graph Diameter

The graph diameter of a graph describes the longest shortest path between two nodes v, and u in a graph $G = (V, E)$ where $v, v \in V(G)$. The diameter D seen as the function of the eccentricity $\mathcal{E}$ can be seen as the following:

$$D(G) = \forall v \in V(G) : max(\mathcal{E}(v)).$$

2.2 Distributed Averaging

Under the assumption that the distributed network can be represented as an bi-directional simple graph, in which the agents are nodes and communication between the agents are bi-directed edges. Each node i then starts out with an initial opinion, or value $x_i(t)$. A vector can then be constructed to contain all the initial values of the network $\boldsymbol{x}(t) = \{x_1(t), ..., x_n(t)\}$. The values of the network at a given time t can then be retrieved using the following formula:

$$\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{x}(t-1),$$

Where $\boldsymbol{A}$ is a stochastic square matrix containing information concerning the topology of the network, also referred to as a Weighted Adjacency Matrix (WAM). Each row i in the matrix $\boldsymbol{A}$ represents an adjacency vector for the node corresponding to i. Each value in the rows is either 0, if no edge exists between those nodes or $\frac{1}{n}$ if there is an edge connecting them, where n is the number of its neighbouring nodes.

If the network continuously applies the algorithm a number of times, the values for each node will converge to a common value eventually, assuming three conditions [2, 15]:

1. **Matrix $\boldsymbol{A}$ is stochastic**: All the values in the matrix $\boldsymbol{A}$ exist in the space:

$[0, 1]$, and that for each row $\boldsymbol{a}$ the sum of the values in that row add up to 1.

2. **Strong Connectivity**: For every point in time $t \geq 0$, the network is strongly connected. Meaning that there has to exist a path of communication directly or indirectly, between all the agents in the network.

3. **Bounded communication intervals**: The time $t \geq 0$ can be divided into several time intervals $\mathcal{B} : \{\mathcal{B}_0, ..., \mathcal{B}_i\}$, where each node is required to send its current value to its neighbours *at least* once per time interval.

2.3 Edge cost

Each edge e between any two nodes (v, u) is represented with a weight, or in this paper referred as cost. The meaning of this cost, is seen as the energy cost for each communication needed between any two nodes. The formula for this cost is given as the distance between the nodes squared:

$$\begin{aligned} c(e) = (v, u) \in e : \sqrt{(v_x - u_x)^2 + (v_y - u_y)^2}^2 \\ = \ldots : (v_x - u_x)^2 + (v_y - u_y)^2. \end{aligned}$$

Given a graph G, the total edge cost is then the sum of all the edges in the graph:

$$C(G) = \sum_{\forall e \in E(G)} c(e).$$

2.4 Convergence rate

The convergence rate of a graph G is denoted as the rate of the nodes that holds different values can reach a consensus such as all the nodes in the graph contains the same value. In practical use the convergence rate is decided as by how many iterations x the nodes in a graph has to share values with each other before it reaches a consensus average value. However, by mathematical operations, the

convergence rate can be seen as the second biggest value of the eigenvalues from the stochastic matrix A of adjacency matrix from the graph G [16].

Given in formulas, if $(\lambda_1, \lambda_2 \dots \lambda_n)$ are the sorted eigenvalues of A, then the function $\mu(G)$ that denotes the convergence rate of the graph G can be written as

$$\mu(G) = \lambda_{n-1},$$

where λ_{n-1} is the second biggest eigenvalue. The convergence rate $\mu(G)$ will continuously be referred to as λ.

2.5 Consensus energy

The consensus energy w is denoted as the total energy needed for a network to reach a consensus average. Since every iteration to normalize the values closer to an consensus average in the network requires every node in the network to read the values of its neighbours, the energy needed for one iteration for all the nodes in the network is then equal to the total edge cost $C(G)$.

Given that λ is the convergence rate of the graph G, then the graph converges to zero if:

$$\lim_{x \to \infty} \lambda^x = 0.$$

Where x can be seen as the number of iterations needed to reach a consensus [16]. However, unless λ is not actually zero, in practical use it will never reach zero. Instead, a very small threshold value ϵ close to zero can instead be set and used to determine when a convergence is reached, and let a convergence be reached when:

$$\lambda^x \leq \epsilon$$

Breaking out x from the equation, the number of iterations to reach a consensus can then be written as:

$$x = \frac{\log \epsilon}{\log \lambda}.$$

Given the number of iterations x to reach a consensus and the total edge cost is $C(G)$ in a graph G, the total energy needed to reach a consensus is then:

$$w = x \times C(G).$$

The consensus energy of a graph G will continuously be denoted as:

$$W(G) = w.$$

2.6 Time complexity

Time complexity describes the relation between the size of input data and the time it takes to complete an algorithm [17]. It is usually described in Big O-notation and is denoted with $\mathcal{O}(\alpha)$ where α is a factor representing the increase of computational power with respect to the input size [17].

2.7 Exhaustive search

Exhaustive search, also referred to as brute-force, is an approach to finding an optimal solution by searching the entire problem space for all possible solutions of a problem. When the algorithm has finished looking at all the solutions it is able to select the one (or ones) with the best outcome with a 100 % certainty [18, p. 32]. For more complex problems with a high time complexity this approach quickly becomes ineffective. For instance, if a problem that grows with the time complexity of $\mathcal{O}(2^n)$ it would take roughly 18 minutes to iterate over all solutions when the size of the problem (given as n) equals to 30. If n equals to 50 it takes around 36 years to find all possible outcomes. For a problem with time complexity of $\mathcal{O}(n!)$, even problems as small as $n = 30$ would take 10^{25} years to finish [18, p. 34].

2.8 Simulated annealing

Simulated Annealing is a random search technique that is applied to areas where there is a global optimization problem. An example of such a problem can usually be summarized as finding the global Maximum or Minimum of a function $f(x)$ [19]. The idea behind the algorithm is that an initial solution x is generated to the problem, the quality of x can be evaluated using some objective function $f(x)$, also known as the *energy* of the solution [19]. The algorithm makes random changes to the solution and compares the energy of the proposed solution with the previous solution using the *energy* function $f(x)$. Changes that provide an improvement are always accepted, while negative changes may be accepted with some probability $P(x)$ [19]. A negative change is kept if the result of an evaluation function $P(x)$ is larger than some randomly selected number r, in the range between: $0 \leq r \leq 1$. A typical function to use for evaluating if a negative change is accepted or not is [19]:

$$A(x) = r > P(x) \rightarrow P(x) = e^{-\frac{\Delta f}{T}},$$

where Δf denotes the change in energy, T the current temperature at a certain point in time t. The temperature can be described as a rate of acceptance. At higher temperatures the algorithm is more likely to accept larger negative changes, however the temperature value of T progressively being lowered over time. This means that as the algorithm progresses it is less likely that negative changes will be accepted [18]. A technique for adjusting the temperature is the following function [19]:

$$T = T_0 \alpha^t,$$

where T_0 denotes the original temperature, and α the rate of change for the algorithm. These functions make the algorithm able to initially explore a variety of different solutions, and over time restrict itself to finding an *optimal* one. Since the algorithm is a random search algorithm, it is at no point certain that it will find the optimal solution. The algorithm is however very powerful since it can avoid being trapped in local min/max points as seen in Figure 2.1, it is proven that

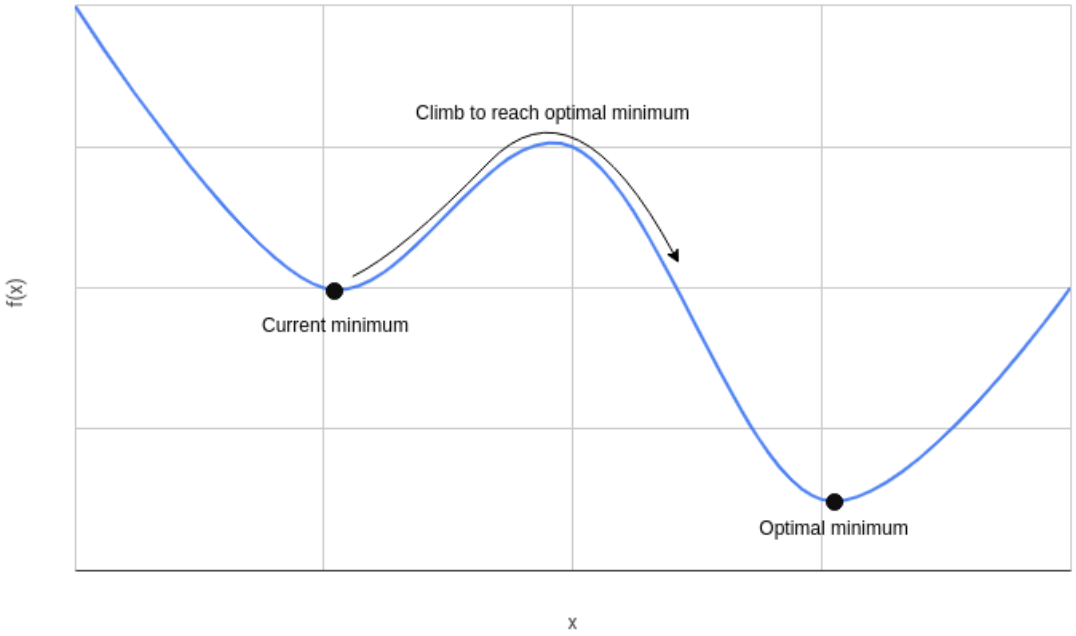

Figure 2.1: Visualization of hill climbing in an optimization problem.

given the right circumstances the algorithm will find the true global max/min [19].

3 Methodology

The following Chapter will describe the methods and techniques that were applied to fulfill the purpose and reach the goal of the thesis. The Chapter will also describe motivations as to why certain methods were used in favour of others.

3.1 Building Simulation Environments

In order to be able to run simulations with the purpose of both initial research, and to later be able to run simulation scenarios to help answering the problem statement, code libraries had to be built to help facilitate this. Three separate libraries were constructed:

1. *Graph Tools*, had to be implemented in order to provide the underlying foundation, such that we can: build future simulation environments, visualize graph-networks and perform graph operations on a given network. Building a separate library also proved helpful due to the modularity of the implementation, which allowed us to support future function installments without interfering with old implementations.
2. *Exhaustive Search*, initially used for smaller graph instances to explore optimal graph topologies. Beyond initial research exploration, this library was used in a limited fashion, due to runtime constraints mentioned in Section 2.7.
3. *Simulated Annealing*, this library was built to be able to find optimal graph edge topologies for a given network problem. This was especially helpful when the number of nodes reached quantities of $n \geq 9$, due to being able to find optimized solutions within reasonable time constraints unlike exhaustive search. This library provided the basis for all of the simulations that were ran past the initial studies.

3.2 Programming Tools

To be able to build the code libraries discussed in Section 3.1, choices had to be made as to which programming tools should be used to implement them. These programming tools were such as: which language and libraries would be appropriate to use in order to effectively be able to tackle the problem statement. The following tools were ultimately applied:

3.2.1 Python

Python is a programming language used for a wide range of software applications such as YouTube, Google [20]. Python was used as the foundation for all of the code written in relation to all the simulations. Other programming languages such as C were initially tried, however due to functions supported by Python such as: high-level functions and wide-range of different code libraries, the choice was made to use Python.

3.2.2 Numpy

Numpy is a library written in Python with its primary field of application being scientific computation [21]. The decision to use the Numpy library to help solve the problem statement, is the librarys support for a wide range high-level functions in the fields of data manipulation and Linear Algebra [21]. These functions were needed to efficiently be able to perform operations such as matrix multiplication or retrieving eigen-values from a matrix structure.

3.2.3 NetworkX

NetworkX is a Python library that was created with the intent of allowing users to: create, manipulate and study network structures [22]. This library offers a wide range of high level data structures, functions and standard graph algorithms [22]. This is the reason that this library was selected to form the underlying support for the graph structures in our simulation scenarios.

3.3 Optimization method

As previously mentioned, the method that was ultimately applied to help solve the problem was Simulated Annealing. The implementation that was used is described in the following section. The algorithm will try to optimize to a local minimum with respect to convergence rate combined with the total edge cost in the graph $G = (V, E, c)$ where c is the edge cost function for the graph.

3.3.1 Optimization value

In order to find a local minimum for a given optimization problem, an optimization value must be formulated that can be compared between different states of the given problem instance. In the given application it would make sense to optimize on either the convergence rate λ, total edge cost $C(G)$ or the consensus energy w.

In the chosen simulations, optimizing on either the convergence rate or the total edge cost would cause the consensus energy to increase to a limit where the battery capacity in the drones would die out in a very short time. By optimizing on the consensus energy however, the algorithm would focus on increasing the lifespan of the drones while doing their tasks. This would decrease both the total edge cost while also increasing the convergence rate in synthesis.

Given the formula from Section 2.5, we know that $w = C(G) \times (\log \epsilon / \log \lambda)$. Since ϵ is a small constant and it is only interesting to optimize on the formula, it is possible to omit $\log \epsilon$ completely from the graph and instead write the energy as:

$$w = \frac{C(G)}{-\log \lambda},$$

where λ is the convergence rate and $C(G)$ is the total edge cost in the graph.

3.3.2 Allowed changes

The implemented algorithm is allowed to perform one of three changes to the graph every iteration. Initially it randomly selects three nodes t, u and v from the graph G, then does one of three changes to the graph:

1. Given it exists an edge between the nodes (t, u), remove the edge.
2. Given it doesn't exists an edge between the nodes (t, u), add an edge.
3. Given it exists an edge between (t, u) and not an edge between (t, v), move the edge from (t, u) to (t, v)

These changes are selected randomly, but the algorithm ensures that one of them is always performed.

3.3.3 State evaluation

After every change the current state of the graph is evaluated based on two criterias:

- The graph is one component.
- The new energy is lower than the old energy, such as $w_{new} < w_{old}$ OR
- If $w_{old} < w_{new}$, accept the current state only if $x < e^{-(w_{new} - w_{old})/T}$ where x is a random real number such that $0 \leq x \leq 1$

If any of these conditions are unmet, the algorithm will revert to the previous state.

3.4 Data Collection

After constructing the simulation environments, data collection could begin. The primary method of data collection that was used in the thesis comes from different simulating different scenarios. These scenarios were meant to emulate some situation that a multi-agent network may find itself in. Each simulation scenario was constructed to ultimately help answer a question regarding the underlying

problem statement of the thesis. The reason that simulation was chosen as the primary method of data collection was to retrieve a larger set of statistical data from which analytical methods could be applied to be able to draw conclusions from.

To measure the effects and results between different networks with different number of nodes, a large amount of data for each graph size needed to be generated. For each graph size, 100 problems was optimized and collected. In total, 30 different graph sizes was generated and collected, spread between 5 to 200 nodes. The graph sizes was increments of 5 from 5 to 100, and increments of 10 from 100 to 200. In summary, the problem set for each simulation scenario was defined as $\{5, 10, 15 \dots 95, 100, 110, 120 \dots 190, 200\}$, accumulating to a total amount of 3 000 optimized networks per simulation scenario.

The reason increments of 10 was chosen for graph sizes between 100 and 200, was due to the time it takes to optimize larger graphs $|G| \geq 100$, as it required more computational power in order to evaluate the current state.

3.5 Simulation Scenarios

Two different simulation scenarios were drawn up to generate data on how a networks ability to detect events, convergence rate and energy cost is affected when the number of agents are adjusted in a system.

3.5.1 Static Area

The static area simulation was constructed to emulate a scenario in which a multi-agent distributed network is set up to observe some area such as a forest. As previously described in the problem statement from Section 1.2, there can exists some sort of event that needs to be detected such as a forest fire. The idea is to generate a number of different graphs with randomly positioned nodes, in which each node has a radius representing an area of observation for which they gather information. Simulated Annealing is then applied to the graph in order to optimize the communication within the network. After which information such

as: communication cost, detection speed, and detection rate is collected. The reason that this simulation is performed is to find out at which points it becomes inefficient from a communication cost perspective to make slight improvements in detection rate and detection speed.

3.5.2 Dynamic Area

The dynamic area simulation was designed to emulate a scenario where there's a need or desire to observe an increasingly larger area depending on how many agents are deployed. In order to perform this a number of nodes proportional to the observation are randomly placed within that observation area. Simulated Annealing is then applied to the graph structure to find an optimal edge structure in regards to the same specifications as in Section 3.5.1. The goal of the simulation is to be able to observe the networks ability to: Converge, communicate, and detect events changes as more agents are introduced in an increasingly large area.

3.6 Data Processing

After the simulations had been completed a large set of data had been produced. In order to be able to draw any meaningful analysis from this data it had to be processed. The data from the simulations were in the form of graphs that had been entered and solved by the simulated annealing algorithm, meaning that the graphs had an edge structure that had been somewhat optimized, but further information lacked. From the different graphs there was a need to answer three key questions for each of the individual graphs, formalized on the original research question from Section 1.4. These were:

1. How efficient is the network at communicating its results in terms of convergence energy.
2. How fast is the network at conveying its results.
3. How much of the total area is being covered by the agents?

In order to be able to do this we processed the data from the graphs to extract:

Coverage, Convergence Rate, convergence energy, average eccentricity, and convergence energy per node. Convergence energy is the simplest to extract, since it is essentially already calculated by the simulated annealing process for each graph that was solved.

To decide how fast the network was at conveying its results, the convergence rate was the natural measurement as it's a factor on how quick a network can reach a consensus average. Due to it being a measurement of how fast the network is at reaching an agreement. In order to extract this measurement, the function described in Section 2.4 was used.

Coverage was defined as the percentage of the total area that the nodes would be able to observe. In order to be able to extract that type of information, a decision had to be made on a range in which each node would be able to observe. This number was eventually set to 15×15 area units around the node, see Figure 3.1. In order to now extract the percentage of the total area that was being covered by the nodes, a grid matrix representing the total area was constructed in which each value in the matrix was initiated to 0. After which each node was iterated over and in its eccentricity was inserted in the 15×15 area around the node, if there was an overlap in coverage meaning that another node was already covering that position the lowest eccentricity of the two was chosen. When all of the nodes had been iterated over, extracting the coverage was done with the formula: Coverage $=$ area units covered/total area. This method of data processing also meant that the average eccentricity could be extracted for each graph, due to their values being stored in the grid matrix.

After that these values had been calculated for each individual graph, the different results were compiled together to form 5 separate graphs for both of the simulation scenarios. These graphs presented information on the key figures as functions of the total number of nodes in the network $|G|$, also referred to as Network Size.

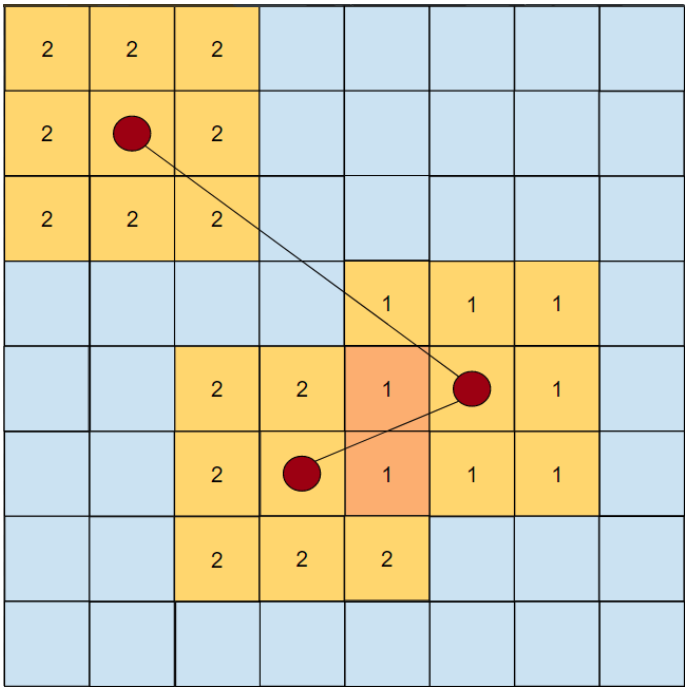

Figure 3.1: Grid Matrix showing three nodes with their ranges set to 3 x 3

3.7 Data Analysis

After that the data had been processed into the desired formats, it could be analyzed. The data was analyzed by trying to identify potential trends, fluctuations, and breaking points in how the different key figures behaved. This was done by observing the results for the different figures as the network size and/or area was increased. The goal of the data analysis was to tie these observations to the three questions mentioned in Section 3.6.

3.8 Delimitations

When examining the described problem, a few assumptions had been made:

1. There are no geographical restraints in the connectivity in the graph, such as all nodes are allowed to communicate directly with each other as long as there is an edge connecting them.
2. The edges in the graphs are bidirectional, such as if there is an edge $e = (v, u)$ it is then equal to $e = (u, v)$.

3. The energy cost of communication between two nodes are the same, meaning that the graph has symmetrical edges.

4 Results

In the following chapter the relevant results from the simulations are presented. From the presented data, the average, maximum and minimum value for each network size of each simulation scenario is recorded and presented.

4.1 Static area

In the following section, results from the simulation scenario that was based on a static area of 100×100 area units will be presented.

The convergence rate of the networks converges at around 0.4, even when the number of agents increases, as seen in Figure 4.1. With an exception for the initial networks with 5 and 10 agents, the average value stays between 0.3 and 0.4. The total edge cost represented in Figure 4.2 increases linearly with the network size. This trend is imitated by the consensus energy from Figure 4.3 which also linearly increased with the same rate. The consensus energy per node from Figure 4.4 shows that the energy w per agent flattens out when more agents are introduced to the network. From 4.5 it can be seen how the coverage of the network converges to 100 % as more agents are introduced, but the average value has a hard time reaching it, while the maximum coverage reaches 100 % coverage with 190 agents.

4.1.1 Convergence rate

Figure 4.1 shows how the convergence rate $\lambda = \mu(G)$ correlates to its network size $|G|$. A low convergence rate means that the network will reach a consensus average faster with fewer iterations. In the given simulation scenario a lower value is more beneficial than a higher value, but a lower convergence rate may require more edges and thus lead to a higher consensus energy. A table representation can be found in appendix A.1.

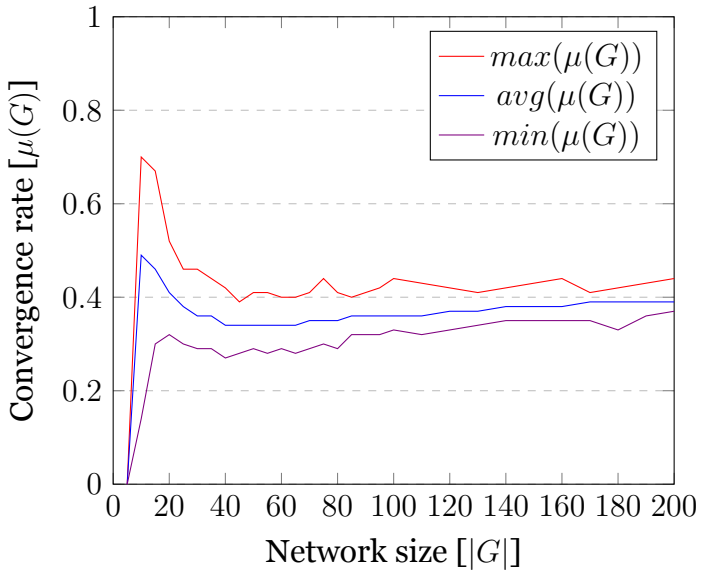

Figure 4.1: Convergence rate over network size in a fixed area (100×100)

4.1.2 Total edge cost

Figure 4.2 shows how the edge cost $C(G)$ correlates to its network size $|G|$. A higher edge cost could either mean more edges or longer distances between the edges, however, in the static scenario a higher edge cost is due to more edges being added since the area stays the same. A table representation can be found in appendix A.2.

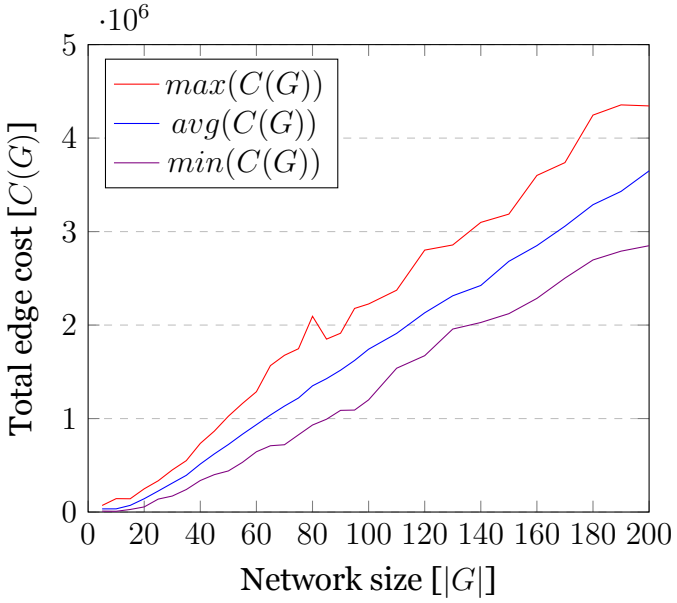

Figure 4.2: Total edge cost over network size in a fixed area (100×100)

4.1.3 Consensus energy

Figure 4.3 shows how the consensus energy $w = W(G)$ correlates to its network size $|G|$. A lower consensus energy means that the network requires an overall lower energy in order to reach an average consensus. In the given simulation scenario a lower value is more beneficial than a higher value. The consensus energy is dependent on both the convergence rate $\mu(G)$ and the total edge cost $C(G)$. A table representation can be found in appendix A.3.

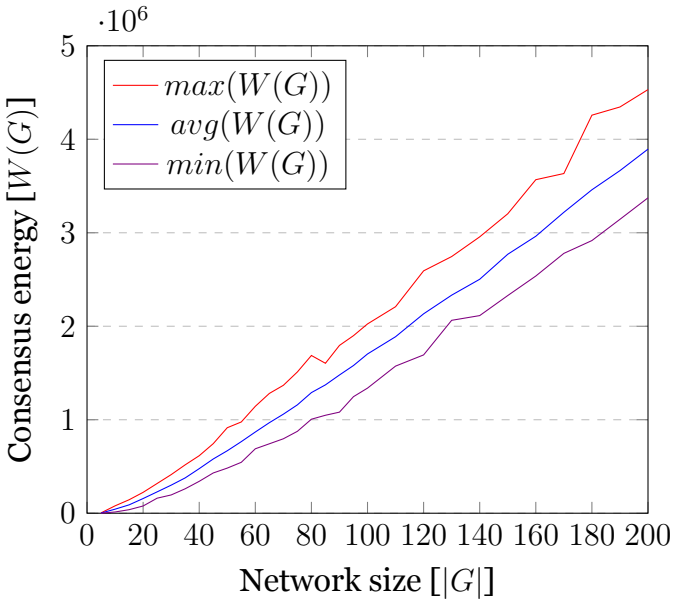

Figure 4.3: Consensus energy over network size in a fixed area (100×100)

4.1.4 Consensus energy per node

Figure 4.4 shows how much of the total energy w each drone represents in the network G. A table representation can be found in appendix A.4.

4.1.5 Coverage

Figure 4.5 shows how much percentage of the area the drones are covering in the given network G. In the given simulation scenario a higher coverage is more beneficial, since the drones can discover more anomalies in the area. A table representation can be found in appendix A.5.

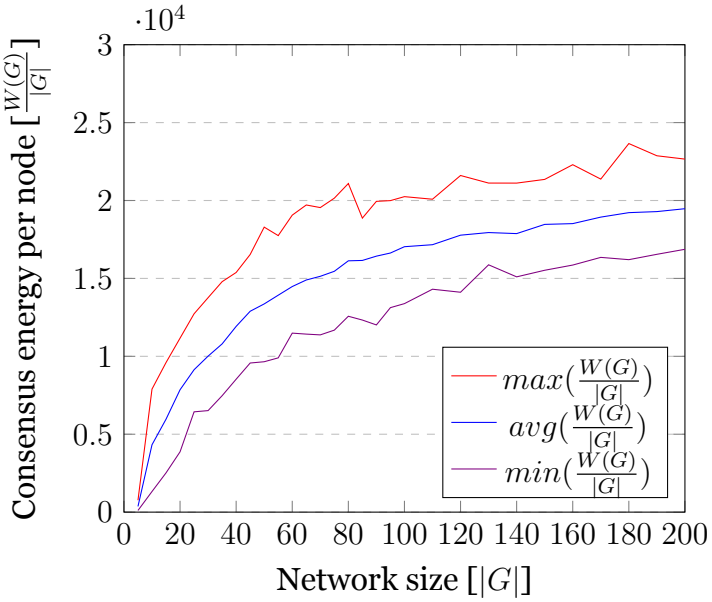

Figure 4.4: Consensus energy per node over network size in a fixed area (100×100)

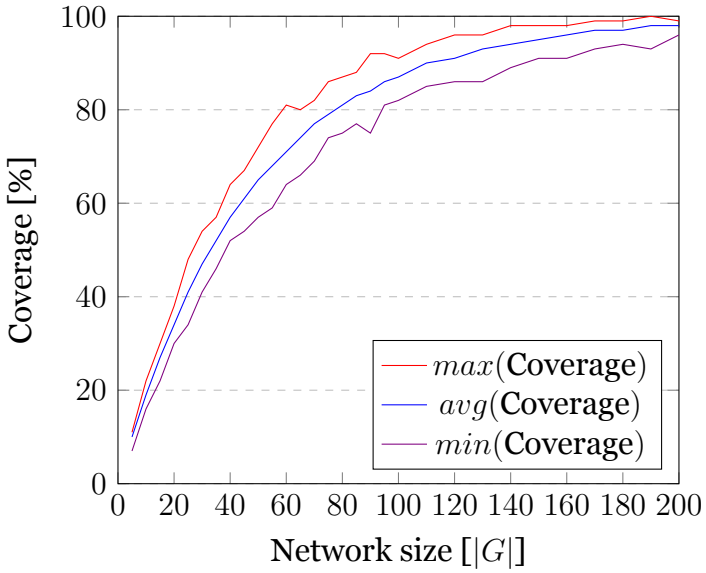

Figure 4.5: Coverage in a fixed area (100×100)

4.2 Dynamic area

In the following section, results from the simulation scenario that was based on a growing area of $|G| \times |G|$ area units will be presented.

The convergence rate in Figure 4.6 initially follows the feature as the static simulation scenario with a peak for smaller networks, to after that increase linearly with around 0.1 per 100 agents. The total edge cost in Figure 4.7 increases

polynomial as the network size grows, while the energy is relatively low for smaller networks. Figure 4.8 shows that the consensus energy follows the same trait with an polynomial increasing energy cost in order to reach consensus. The consensus energy per node in Figure 4.9 shows that the energy spread among the nodes increases as well as the network grows, and isn't converging as the static network tended to. The coverage in Figure 4.10 shows an decreasing coverage after its been bound to 100 % for networks under the size of 30 agents.

4.2.1 Convergence rate

Figure 4.6 shows how the convergence rate $\lambda = \mu(G)$ correlates to its network size $|G|$. A low convergence rate means that the network will reach a consensus average faster with fewer iterations. In the given simulation scenario a lower value is more beneficial than a higher value, but a lower convergence rate may require more edges and thus lead to a higher consensus energy. A table representation can be found in appendix B.1.

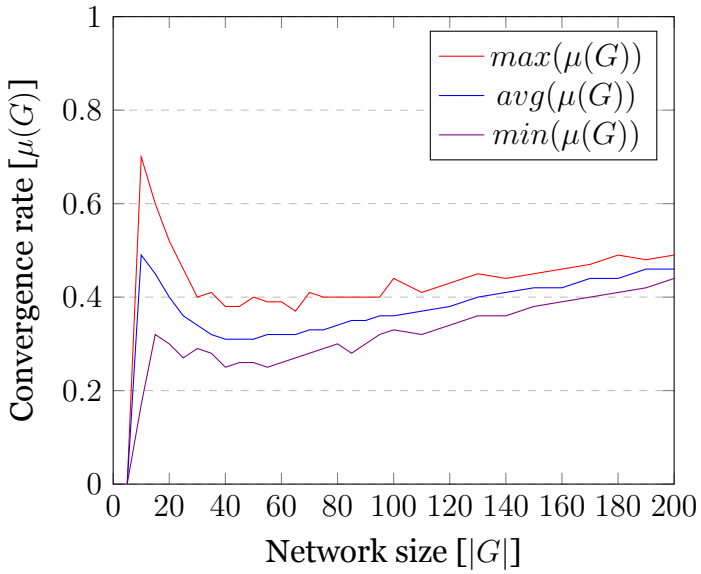

Figure 4.6: Convergence rate over network size in a dynamic area ($|G| \times |G|$)

4.2.2 Total edge cost

Figure 4.7 shows how the edge cost $C(G)$ correlates to its network size $|G|$. A higher edge cost could either mean more edges or longer distances between the edges. In the dynamic scenario a higher edge cost is a combination of both, since more edges is added and the distances are increasing simultaneously.A table representation can be found in appendix B.2.

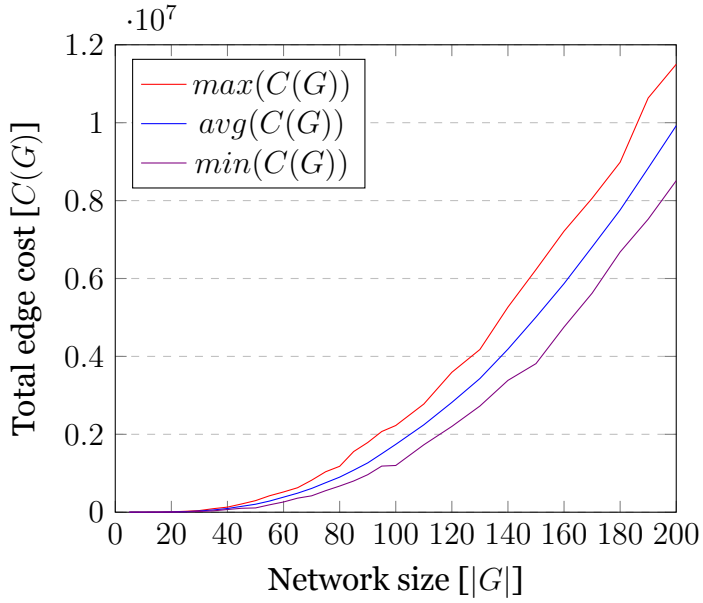

Figure 4.7: Total edge cost over network size in a dynamic area ($|G| \times |G|$)

4.2.3 Consensus energy

Figure 4.8 shows how the consensus energy $w = W(G)$ correlates to its network size $|G|$. A lower consensus energy means the the network requires an overall lower energy in order to reach an average consensus. In the given simulation scenario a lower value is more beneficial than a higher value. The consensus energy is dependent on both the convergence rate $\mu(G)$ and the total edge cost $C(G)$. A table representation can be found in appendix B.3.

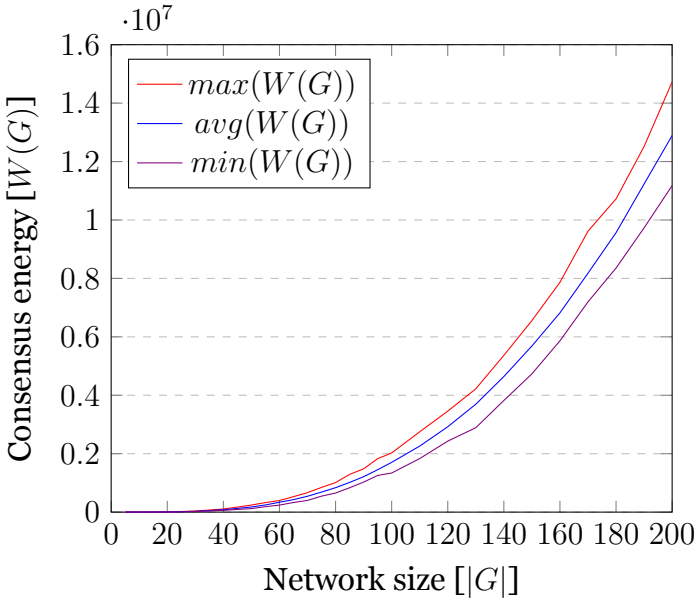

Figure 4.8: Consensus energy over network size in a dynamic area ($|G| \times |G|$)

4.2.4 Consensus energy per node

Figure 4.9 shows how much of the total energy w each drone represents in the network G. A table representation can be found in appendix B.4.

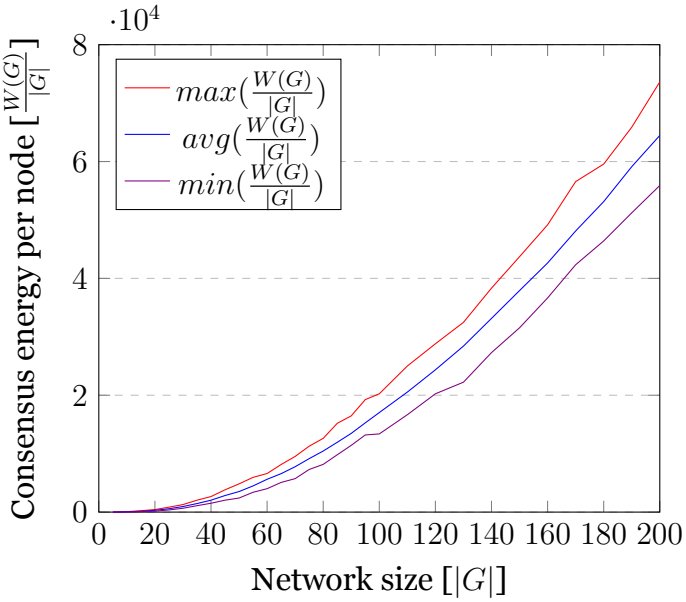

Figure 4.9: Consensus energy per node over network size in a dynamic area ($|G| \times |G|$)

4.2.5 Coverage

Figure 4.10 shows how much percentage of the area the drones are covering in the given network G. In the given simulation scenario a higher coverage is more beneficial, since the drones can discover more anomalies in the area. A table representation can be found in appendix B.5.

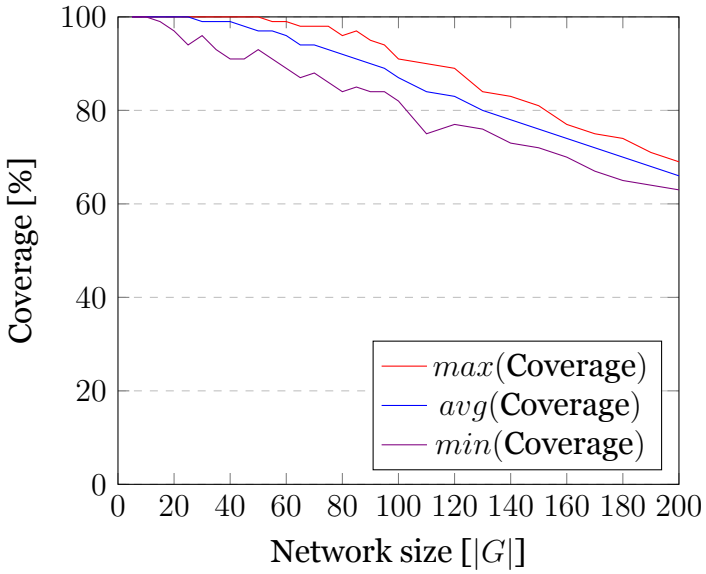

Figure 4.10: Coverage in a dynamic area ($|G| \times |G|$)

4.3 Additional observations

Figure 4.11 and 4.12 shows the average eccentricity $\mathcal{E}(G)$ of the graph G. A table representation can be found in appendix A.6 and B.6. The results in the figures follows the trend from the convergence rate with a peak for smaller networks and is later on converging to a constant value the larger the networks becomes.

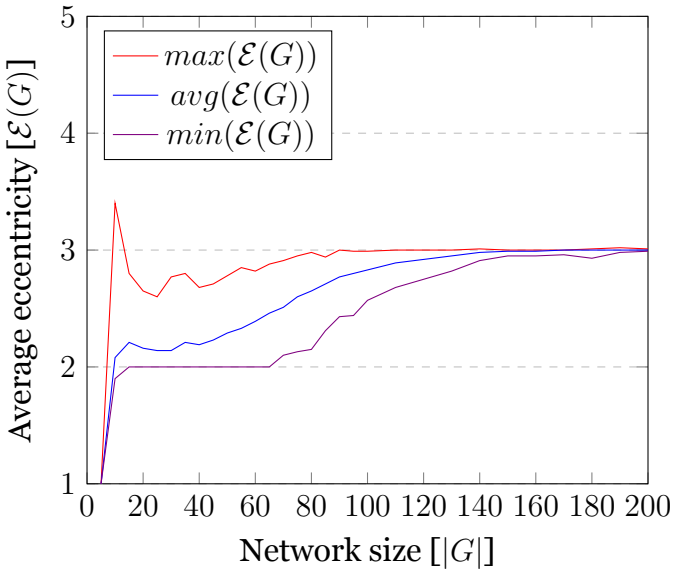

Figure 4.11: Average eccentricity over network size in a fixed area (100×100)

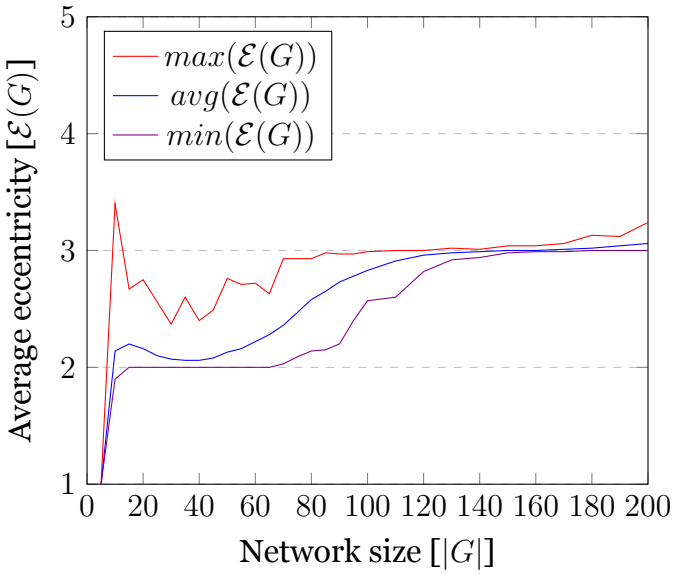

Figure 4.12: Average eccentricity over network size in a dynamic area ($|G| \times |G|$)

5 Discussion

In the discussion the results are discussed from a basis on the questions stated in Section 3.6 that is based on the research question presented in Section 1.4 of the report:

- How efficient is the network at communicating its results in terms of convergence energy?
- How fast is the network at conveying its results?
- How much of the total area is being covered by the agents?

5.1 Static area

From Figure 4.1 a trend was identified where the convergence rate stabilized at a value of around 0.4. This value remained fairly constant for networks with a size of $|G| \geq 40$ nodes when these nodes were deployed in a static sized area of 100×100 area units. This could be seen as indication that for a sufficiently large network in a given area, introducing more agents will neither have a positive nor negative effect on the general time it takes for a network to reach consensus. To refer back to the question of how fast the network could convey its result as the number of nodes increased mentioned in Section 3.6, the networks ability to perform this remained fairly constant. This perhaps suggests that the number of nodes necessarily doesn't have to affect how quickly information can spread throughout the network. We are unsure as to why there's a sudden spike in the value starting at 10 nodes before leveling out. A potential explanation for this is that this occurs as that the cost of reaching consensus is fairly low for smaller graphs. This in turn could lead to that the simulated annealing algorithm runs into problems in identifying poor decisions with respect to the convergence rate, as the difference between these results are too small.

The total consensus energy of the network increased in a fairly linear fashion as more nodes were introduced as seen in Figure 4.3. When contrasting this with the results from consensus energy per node from Figure 4.4, some interesting

observations could be made. For smaller networks, the cost of introducing more agents had a higher cost, when it comes to energy consumption per node. An argument could then be made that in a scenario of a smaller network of nodes, introducing more agents will perhaps more significantly increase the frequency in which individual nodes will have to be brought down for recharging. This in turn would lead to a potential increase in downtime for such a network. Depending on how detrimental this is to the general application, it may in fact not be beneficial to increase the number of agents.

However for larger networks where the number of nodes exceeded $|G| \geq 60$ the increased communication cost per node was significantly reduced. This suggests that in a scenario where energy consumption from communication is of lesser concern to a network. The negative effects on energy consumption per node can largely be ignored when introducing more agents to a network. This ocurrs after a certain threshold as the number converges to an almost constant value. To draw this back to the question of how efficiently the network was at communicating its results, the data showed very situational results where it really comes down to how energy efficient the individual drones need to be in their application.

The question of how important coverage is can vary from application to application, but for a network set up to detect and observe forest fire one could assume that the ability to detect such an event is of paramount importance. When it came to the question of how the coverage had been affected as the number of nodes increased in the static area simulation, the networks coverage increased with a diminishing factor per node. This can be seen in Figure 4.5, where the average network had a coverage of 90 % at around 110 agents. It however took around 200 agents before the network reached a coverage of $98 - 100$ %, making the last percentages extremely costly in comparison to the previous 90 %. From the perspective of network striving to increase its coverage from 90 % to $98-100$ %, the application wouldn't necessarily suffer in terms of energy consumption per agent as described in the previous paragraph. This is due to the consensus energy per node being fairly consistent when going from $90 - 100$ agents to 200, however the material cost of doubling the number of agents is most likely too large to warrant this type of action.

A better course of action would most likely be to try and optimize the positioning of each individual agent to avoid overlapping observation areas. This could however affect the overall energy consumption for each agent due to the moving of nodes. It is hard to gauge whether this would overall provide positive or negative effects without running further simulations for those cases.

5.2 Dynamically growing area

Regarding the question of how the convergence rate was affected in the case of the dynamically growing area, we could see that as the area grew the convergence rate of the network increased. The convergence rate increased by about 50 % when going from 40 agents to 200. One of the reasons that this could be is that as the simulation area increased, the average distance between the nodes became more spread out. This may have resulted in that the algorithm would choose to optimize around a network with fewer edges which affected the overall connectivity of the network and as a result the convergence rate. Further support for this conclusion can be seen in Figure 4.7 where the total edge cost of the network grew exponentially as more nodes were introduced.

The consensus energy grew exponentially both in terms of total energy and per node as the area grew. Similarly to the convergence rate this is most likely a result of the average distance of the nodes increasing as the area grew. This points to that as an area grows, if enough agents aren't introduced to help monitor that area. The energy cost of communication could quickly become too high, and directly impact the efficiency of the application.

As seen in Figure 4.10, up until 25 nodes the average coverage was bound at 100 %. This indicates that there is probably an overflow of nodes for a field area and graph size of these sizes, where a lot of the covering areas of the nodes are overlapping. As the graphs reached 90 nodes, the average coverage of the network had gone down to 90 %. When the total number had reached 200 the average coverage was only 65 %.

A more general statement could be made from these results, that not enough agents were introduced as the area grew and this had adverse effects on the

networks ability to: converge, communicate and detect events.

5.3 Additional observations

During the analysis of the results it was discovered that the eccentricity $\mathcal{E}(G)$ of the graphs G converged around 2 for graph sizes $|G| \leq 40$ and 3 for graph sizes $|G| \geq 100$, as seen in Figure 4.11 and 4.12 there seem to be some kind of correlation between the eccentricity and the convergence rate as they both peaks at graph sizes at 10 nodes. As this information provided little basis for the purpose of the report no analysis was performed on these results. However we feel that they could point to some interesting indications.

6 Conclusions

To tie back to the research question stated in Section 1.4: How does introducing more agents to a multi-agent network affect the energy cost, coverage and convergence speed of a network? Some interesting conclusions were drawn from the results.

The convergence rate didn't necessarily change as more agents were introduced to monitor a static area. The conclusions that was drawn from this, was that there exists network topologies in which the size of the network doesn't largely impact the speed of convergence. This can be helpful when deciding on if more agents should be introduced to the network to improve some other metric such as energy or coverage. If the network is rapidly changing in terms of number of agents, it can also be useful to know that it would be able to function at similar speeds, as not to have to re-calibrate the application to account for this.

As the increase in consensus energy per node leveled out when the network grew in a static area, it was concluded that the energy consumption progresses more steady for larger networks than smaller. When knowing the number of agents that can be deployed in a network beforehand it is then easier to calculate how often, in general, the agents needs to be recharged in order to keep the network operating. However for smaller networks the results showed that introducing more agents can have a large energy cost per agent. This points to that it can be important to consider the characteristics of the network when making adjustments to the number of agents.

From the results of the dynamic area simulations we conclude that for larger areas it is important to consider building more robust networks, rather than spreading the resources out thin. The simulations showed that as distances between the individual nodes became too large the networks speed, coverage, and energy per node was negatively impacted.

6.1 Retrospective

One of the delimitations stated for the simulation scenarios that could be criticized, is that there were no geographical limitations on the range of communication between the nodes. In real world applications, such limitations would most likely exist if the observation areas grew too large. If such a limitation had been introduced to the simulation scenarios we may have seen very different results, however this would have made for completely different simulations. This would also potentially introduce the factor of not being able to connect the entire network due to certain nodes being too far apart. This problem would have to somehow be dealt with in order to be able run the simulations. In Section 1.5 the two related works solved this by having the sensors/UAVs store information and communicate it when in range of each other, which would have been a possible solution to solve this problem. However in order for this to work the drones would have to be allowed to move around the network. This leads us to another delimitation that was made: the drones were placed in static positions. In both the works presented in Section 1.5 the decision was made to keep the position of the drones more dynamic. This is something to consider, since in a real-world application it could be useful to continuously move the drones to observe areas of interest.

Another factor that could have affected the results of the simulations is the optimization method; more specifically simulated annealing. From the works presented by Ho Dac-Tu, et al referred to in Section 1.5, we saw that depending on the optimization technique the results could be vastly improved [8]. There could therefore exist solutions that are drastic improvements to the suggested ones. We however feel that our data pool was significantly large enough to offset this type of error, but it is worth mentioning. One way that this could be investigated would be to tweak the iterations and temperature on the simulated annealing algorithm and record the results. Another way this would be to use one or several other optimization algorithms to examine the results from those simulations.

6.2 Future Work

Both simulation scenarios used in this report were done by randomly placing drones in a network and optimizing on the resulting empty graphs. Another scenario not covered by this report would be to strategically placing drones at fixed positions and optimizing on the resulting networks or having the drones move around in a more dynamic environment. This could result in other practical uses where the field of application is known in beforehand, and how the resulting energy would change if more drones were introduced into the network.

The average eccentricity of each of the graphs were recorded but did not end up being relevant to help answer the research question. One interesting observation that was drawn from this data however, was that the simulated annealing algorithm seemed to favour graphs in which the nodes had a fairly evenly distributed eccentricity. To the optimization algorithm it seemed as though there was a direct correlation between minimizing the longest shortest distance between any two nodes and an optimized graph. No further conclusions could be drawn from the data however. But it would be interesting to expand upon how the eccentricity of the nodes in the graph affects the performance of a network.

6.3 Ethical Concerns

The range of applications that distributed networks can be applied to fit is very wide. Some of the ethical concerns that we see as potential problems in the future as these type of networks and applications become more sophisticated and efficient, are in the field of government, military- and surveillance applications that could use it to violate human rights.

One could see a future in which networks such as these are used to track and survey populations, or to monitor military battlefields. These type of applications can often be placed in some form of moral grey scale, where a very thin blurry line separates ethical use from misuse. As mobile networks grow and are able to carry larger data communications and increase in availability, the possibility for individuals to employ these type of networks to perform more sophisticated

tasks will becomes more feasible. Therefore concerns surrounding this technology does not only revolve around government- and security agencies, but also private institutions such as corporations or political groups.

6.4 Final Words

We however see great potential in the field of application for distributed multi-agent networks such as the ones described in this report, to contribute in a positive manner to society. Today it is very unclear how a definitive optimal network topology looks. From the results of this report we can conclude that it can vary from application to application depending on a number of factors such as: observation range, area and energy constraints etc. We believe that in the future it will be a very important tool to be able to quickly mount these types of networks to perform tasks such as observing forest fires. From the results and discussion some interesting conclusions could be drawn to attempt and explain how different key figures and factors can vary depending on network topology. We therefore conclude that the research question, together with the purpose and goal of the report to be fulfilled.

Bibliography

[1] Alsheikh, Mohammad Abu et al. “Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications”. In: *IEEE Communications Surveys & Tutorials* 16 (2014), pp. 1996–2018.

[2] Olshevsky, Alex and Tsitsiklis, John N. “Convergence Speed in Distributed Consensus and Averaging”. In: *SIAM Review* 53 (2011), pp. 747–772.

[3] Ren, Wei, Beard, R.W., and Atkins, E.M. “A survey of consensus problems in multi-agent coordination”. In: *Proceedings of the 2005, American Control Conference* (2005), pp. 1859–1864.

[4] You, Keyou and Xie, Lihua. “Network Topology and Communication Data Rate for Consensusability of Discrete-Time Multi-Agent Systems”. In: *IEEE Transactions on Automatic Control* 56 (2011), pp. 2262–2275.

[5] Hoos, Holger H and Stützle, Thomas. *Stochastic local search : foundations and applications*. Morgan Kaufmann Publishers, Cop, 2005, pp. 15–16.

[6] Kang-Ping Wang et al. “Particle swarm optimization for traveling salesman problem”. In: (Nov. 2003).

[7] Lin, Zhongjie and Liu, Hugh H.T. “Topology-based distributed optimization for multi-UAV cooperative wildfire monitoring”. In: *Optimal Control Applications and Methods* 39 (2018), pp. 1530–1548.

[8] Ho, Dac-Tu et al. “Optimization of Wireless Sensor Network and UAV Data Acquisition”. In: *Journal of Intelligent & Robotic Systems* 78 (2015), pp. 159–179.

[9] Wang, Le Yi and Yin, George. “Weighted and Constrained Consensus Control with Performance Optimization for Robust Distributed UAV Deployments with Dynamic Information Networks”. In: *Recent Advances in Research on Unmanned Aerial Vehicles* 444 (2013), pp. 181–205.

[10] Rabinovich, Sharon, Curry, Renwick E., and Elkaim, Gabriel H. “Toward Dynamic Monitoring and Suppressing Uncertainty in Wildfire by Multiple Unmanned Air Vehicle System”. In: *Journal of Robotics* 2018 (2018), pp. 1–12.

[11] Annighoefer, B., Reif, C., and Thieleck, F. “Network topology optimization for distributed integrated modular avionics”. In: *2014 IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC)* (Oct. 2014), ISSN: 2155-7195. DOI: 10.1109/DASC.2014.6979461.

[12] Diestel, Reinhard. *Graph theory*. Berlin Springer, 2018, pp. 1–15, 147.

[13] Nettleton, David F. “Data mining of social networks represented as graphs”. In: *Computer Science Review* 7 (2013), pp. 1–34.

[14] Watkins, John J. *Combinatorics: ancient & modern*. Oxford University Press, 2015, pp. 332–350.

[15] Ozdaglar, Asu and Nedic, Angelia. *Consensus Problem in Multi-Agent Systems*. Joint EUROPT-OMS Conference. 2007. URL: http://web.mit.edu/asuman/Desktop/asuman/www/presentations_web/EUROPT_talk2.pdf.

[16] Olshevsky, Alex and Tsitsiklis, John N. “Convergence Speed in Distributed Consensus and Averaging”. In: *SIAM Journal on Control and Optimization* 48 (2009), pp. 33–55.

[17] Blum, E K and Aho, Alfred V. *Computer science : the hardware, software and heart of it*. Springer, 2014, pp. 246–247.

[18] Kleinberg, Jon and Tardos, Éva. *Algorithm Design*. 1st ed. Pearson/Addison-Wesley, 2006.

[19] Yang, Xin-She. “Simulated Annealing”. In: Engineering Optimization. John Wiley & Sons, Inc, 2010, pp. 181–188.

[20] Foundation, Python Software. *Quotes about Python*. 2019. URL: https://www.python.org/about/quotes/.

[21] Developers, NumPy. *Numpy*. 2019. URL: https://www.numpy.org/.

[22] developers, NetworkX. *NetworkX, Software for complex networks*. 2019. URL: https://networkx.github.io/.

Appendices

Appendix - Contents

A Tables (fixed area)

A.1 Convergence rate

Table A.1: Convergence rate in a fixed area (100×100)

$\lvert G \rvert$	Min	Avg	Max
5	0.00	0.00	0.00
10	0.14	0.49	0.70
15	0.30	0.46	0.67
20	0.32	0.41	0.52
25	0.30	0.38	0.46
30	0.29	0.36	0.46
35	0.29	0.36	0.44
40	0.27	0.34	0.42
45	0.28	0.34	0.39
50	0.29	0.34	0.41
55	0.28	0.34	0.41
60	0.29	0.34	0.40
65	0.28	0.34	0.40
70	0.29	0.35	0.41
75	0.30	0.35	0.44
80	0.29	0.35	0.41
85	0.32	0.36	0.40
90	0.32	0.36	0.41
95	0.32	0.36	0.42
100	0.33	0.36	0.44
110	0.32	0.36	0.43
120	0.33	0.37	0.42
130	0.34	0.37	0.41
140	0.35	0.38	0.42
150	0.35	0.38	0.43
160	0.35	0.38	0.44
170	0.35	0.39	0.41
180	0.33	0.39	0.42
190	0.36	0.39	0.43
200	0.37	0.39	0.44

A.2 Total edge cost

Table A.2: Total edge cost in a fixed area (100×100)

$\|G\|$	Min	Avg	Max
5	10	10	10
10	9	17	33
15	24	37	53
20	51	67	86
25	83	102	125
30	105	144	176
35	147	184	222
40	185	234	294
45	233	279	350
50	262	331	396
55	298	383	468
60	356	433	521
65	399	483	610
70	424	538	645
75	431	580	695
80	486	630	815
85	568	681	802
90	596	726	844
95	639	783	931
100	637	838	942
110	736	943	1 101
120	854	1 047	1 201
130	973	1 155	1 330
140	1 068	1 235	1 426
150	1 134	1 356	1 521
160	1 204	1 465	1 663
170	1 390	1 557	1 794
180	1 493	1 682	2 101
190	1 530	1 767	2 014
200	1 583	1 892	2 164

A.3 Consensus energy

Table A.3: Consensus energy in a fixed area (100×100)

$\|G\|$	Min	Avg	Max
5	9 916	33 695	70 326
10	8 185	34 013	143 314
15	27 282	71 174	141 923
20	54 844	142 480	248 488
25	138 502	224 911	334 511
30	171 003	309 306	450 409
35	240 722	392 449	549 643
40	337 269	513 537	734 216
45	400 058	621 986	865 484
50	439 635	722 240	1 024 606
55	531 188	831 767	1 160 778
60	644 704	934 597	1 286 987
65	709 146	1 037 948	1 566 372
70	719 862	1 133 469	1 676 557
75	825 951	1 219 751	1 746 931
80	930 779	1 350 008	2 094 126
85	992 914	1 426 602	1 849 543
90	1 087 621	1 517 226	1 912 701
95	1 090 714	1 621 339	2 177 965
100	1 199 718	1 741 657	2 225 724
110	1 538 416	1 911 017	2 372 988
120	1 672 467	2 131 079	2 801 852
130	1 957 365	2 313 134	2 856 926
140	2 027 458	2 423 630	3 098 277
150	2 121 916	2 681 750	3 186 829
160	2 285 215	2 849 857	3 600 522
170	2 502 000	3 057 379	3 737 748
180	2 696 499	3 287 558	4 246 090
190	2 790 597	3 429 771	4 355 176
200	2 848 930	3 649 385	4 344 271

A.4 Consensus energy per node

Table A.4: Consensus energy per node in a fixed area (100×100)

$\|G\|$	Min	Avg	Max
5	108	366	764
10	1 323	4 324	7 896
15	2 519	5 965	9 599
20	3 871	7 854	11 166
25	6 437	9 139	12 729
30	6 516	10 011	13 764
35	7 471	10 797	14 790
40	8 530	11 920	15 380
45	9 570	12 892	16 535
50	9 652	13 362	18 291
55	9 901	13 922	17 748
60	11 488	14 478	19 061
65	11 425	14 899	19 713
70	11 376	15 135	19 546
75	11 680	15 456	20 150
80	12 571	16 129	21 090
85	12 328	16 153	18 870
90	12 014	16 429	19 945
95	13 124	16 630	19 993
100	13 382	17 035	20 248
110	14 306	17 163	20 082
120	14 112	17 775	21 605
130	15 873	17 943	21 119
140	15 100	17 876	21 117
150	15 523	18 467	21 354
160	15 857	18 517	22 298
170	16 350	18 934	21 374
180	16 203	19 221	23 655
190	16 547	19 288	22 871
200	16 868	19 473	22 657

A.5 Coverage

Table A.5: Coverage in a fixed area (100×100)

$\|G\|$	Min	Avg	Max
5	7 %	10 %	11 %
10	16 %	19 %	22 %
15	22 %	27 %	30 %
20	30 %	34 %	38 %
25	34 %	41 %	48 %
30	41 %	47 %	54 %
35	46 %	52 %	57 %
40	52 %	57 %	64 %
45	54 %	61 %	67 %
50	57 %	65 %	72 %
55	59 %	68 %	77 %
60	64 %	71 %	81 %
65	66 %	74 %	80 %
70	69 %	77 %	82 %
75	74 %	79 %	86 %
80	75 %	81 %	87 %
85	77 %	83 %	88 %
90	75 %	84 %	92 %
95	81 %	86 %	92 %
100	82 %	87 %	91 %
110	85 %	90 %	94 %
120	86 %	91 %	96 %
130	86 %	93 %	96 %
140	89 %	94 %	98 %
150	91 %	95 %	98 %
160	91 %	96 %	98 %
170	93 %	97 %	99 %
180	94 %	97 %	99 %
190	93 %	98 %	100 %
200	96 %	98 %	99 %

A.6 Average eccentricity

Table A.6: Average eccentricity in a static area (100×100)

$\|G\|$	Min	Avg	Max
5	1.00	1.00	1.00
10	1.90	2.08	3.40
15	2.00	2.21	2.80
20	2.00	2.16	2.65
25	2.00	2.14	2.60
30	2.00	2.14	2.77
35	2.00	2.21	2.80
40	2.00	2.19	2.68
45	2.00	2.23	2.71
50	2.00	2.29	2.78
55	2.00	2.33	2.85
60	2.00	2.39	2.82
65	2.00	2.46	2.88
70	2.10	2.51	2.91
75	2.13	2.60	2.95
80	2.15	2.65	2.98
85	2.31	2.71	2.94
90	2.43	2.77	3.00
95	2.44	2.80	2.99
100	2.57	2.83	2.99
110	2.68	2.89	3.00
120	2.75	2.92	3.00
130	2.82	2.95	3.00
140	2.91	2.98	3.01
150	2.95	2.99	3.00
160	2.95	2.99	3.00
170	2.96	3.00	3.00
180	2.93	3.00	3.01
190	2.98	3.00	3.02
200	2.99	3.00	3.01

B Tables (dynamic area)

B.1 Convergence rate

Table B.1: Convergence rate in a dynamic area ($|G| \times |G|$)

$\|G\|$	Min	Avg	Max
5	0.00	0.00	0.00
10	0.17	0.49	0.70
15	0.32	0.45	0.60
20	0.30	0.40	0.52
25	0.27	0.36	0.46
30	0.29	0.34	0.40
35	0.28	0.32	0.41
40	0.25	0.31	0.38
45	0.26	0.31	0.38
50	0.26	0.31	0.40
55	0.25	0.32	0.39
60	0.26	0.32	0.39
65	0.27	0.32	0.37
70	0.28	0.33	0.41
75	0.29	0.33	0.40
80	0.30	0.34	0.40
85	0.28	0.35	0.40
90	0.30	0.35	0.40
95	0.32	0.36	0.40
100	0.33	0.36	0.44
110	0.32	0.37	0.41
120	0.34	0.38	0.43
130	0.36	0.40	0.45
140	0.36	0.41	0.44
150	0.38	0.42	0.45
160	0.39	0.42	0.46
170	0.40	0.44	0.47
180	0.41	0.44	0.49
190	0.42	0.46	0.48
200	0.44	0.46	0.49

B.2 Total edge cost

Table B.2: Total edge cost in a dynamic area ($|G| \times |G|$)

$\|G\|$	Min	Avg	Max
5	24	86	150
10	104	334	1 058
15	785	1 737	3 413
20	2 231	5 723	9 873
25	8 341	14 594	23 831
30	18 066	31 229	42 782
35	35 053	58 432	91 471
40	69 321	95 324	130 815
45	101 041	150 201	211 221
50	109 803	205 003	297 169
55	189 867	286 540	422 298
60	262 768	387 208	520 184
65	360 881	488 716	628 604
70	420 709	610 324	821 162
75	553 075	756 974	1 038 725
80	671 819	900 592	1 178 626
85	801 459	1 079 396	1 557 460
90	962 981	1 265 564	1 788 842
95	1 183 987	1 501 725	2 071 659
100	1 199 718	1 741 657	2 225 724
110	1 730 150	2 243 151	2 776 654
120	2 202 327	2 818 685	3 591 767
130	2 725 531	3 431 815	4 177 274
140	3 384 375	4 185 926	5 265 240
150	3 814 991	5 009 564	6 228 532
160	4 754 487	5 868 727	7 215 526
170	5 622 477	6 808 618	8 058 489
180	6 680 363	7 759 686	8 984 616
190	7 523 298	8 834 581	10 634 377
200	8 511 363	9 923 121	11 499 728

B.3 Consensus energy

Table B.3: Consensus energy dynamic area ($|G| \times |G|$)

$\|G\|$	Min	Avg	Max
5	0.6	4.1	8.1
10	182	439	768
15	1 143	2 122	3 516
20	3 026	6 137	8 707
25	9 225	14 298	21 022
30	19 695	28 596	38 578
35	38 404	51 175	71 304
40	60 642	81 532	106 479
45	91 763	127 590	171 631
50	120 417	175 766	241 719
55	186 571	246 630	326 994
60	240 460	337 050	396 300
65	330 017	427 624	529 544
70	400 979	543 498	666 792
75	547 048	686 339	846 162
80	653 390	836 251	1 009 734
85	833 774	1 015 308	1 291 146
90	1 028 036	1 212 632	1 482 258
95	1 254 344	1 451 048	1 828 840
100	1 338 154	1 703 462	2 024 760
110	1 833 157	2 259 515	2 749 884
120	2 429 727	2 922 156	3 456 737
130	2 890 430	3 693 288	4 221 469
140	3 821 556	4 641 390	5 366 010
150	4 731 720	5 686 687	6 556 576
160	5 864 089	6 821 347	7 866 773
170	7 196 556	8 178 491	9 616 235
180	8 353 873	9 564 811	10 721 395
190	9 730 692	11 232 027	12 515 532
200	11 179 010	12 892 470	14 724 695

B.4 Consensus energy per node

Table B.4: Consensus energy per node in a dynamic area ($|G| \times |G|$)

$\|G\|$	Min	Avg	Max
5	0.1	0.8	1.6
10	18	43	76
15	76	141	234
20	151	306	435
25	369	571	840
30	656	953	1 285
35	1 097	1 462	2 037
40	1 516	2 038	2 661
45	2 039	2 835	3 814
50	2 408	3 515	4 834
55	3 392	4 484	5 945
60	4 007	5 617	6 605
65	5 077	6 578	8 146
70	5 728	7 764	9 525
75	7 293	9 151	11 282
80	8 167	10 453	12 621
85	9 809	11 944	15 189
90	11 422	13 473	16 469
95	13 203	15 274	19 250
100	13 381	17 034	20 247
110	16 665	20 541	24 998
120	20 247	24 351	28 806
130	22 234	28 409	32 472
140	27 296	33 152	38 328
150	31 544	37 911	43 710
160	36 650	42 633	49 167
170	42 332	48 108	56 566
180	46 410	53 137	59 563
190	51 214	59 115	65 871
200	55 895	64 462	73 623

B.5 Coverage

Table B.5: Coverage in a dynamic area ($|G| \times |G|$)

$\|G\|$	Min	Avg	Max
5	100 %	100 %	100 %
10	100 %	100 %	100 %
15	99 %	100 %	100 %
20	97 %	100 %	100 %
25	94 %	100 %	100 %
30	96 %	99 %	100 %
35	93 %	99 %	100 %
40	91 %	99 %	100 %
45	91 %	98 %	100 %
50	93 %	97 %	100 %
55	91 %	97 %	99 %
60	89 %	96 %	99 %
65	87 %	94 %	98 %
70	88 %	94 %	98 %
75	86 %	93 %	98 %
80	84 %	92 %	96 %
85	85 %	91 %	97 %
90	84 %	90 %	95 %
95	84 %	89 %	94 %
100	82 %	87 %	91 %
110	75 %	84 %	90 %
120	77 %	83 %	89 %
130	76 %	80 %	84 %
140	73 %	78 %	83 %
150	72 %	76 %	81 %
160	70 %	74 %	77 %
170	67 %	72 %	75 %
180	65 %	70 %	74 %
190	64 %	68 %	71 %
200	63 %	66 %	69 %

B.6 Average eccentricity

Table B.6: Average eccentricity in a dynamic area ($|G| \times |G|$)

$\|G\|$	Min	Avg	Max
5	1.00	1.00	1.00
10	1.90	2.14	3.40
15	2.00	2.20	2.67
20	2.00	2.16	2.75
25	2.00	2.10	2.56
30	2.00	2.07	2.37
35	2.00	2.06	2.60
40	2.00	2.06	2.40
45	2.00	2.08	2.49
50	2.00	2.13	2.76
55	2.00	2.16	2.71
60	2.00	2.22	2.72
65	2.00	2.28	2.63
70	2.03	2.36	2.93
75	2.09	2.47	2.93
80	2.14	2.58	2.93
85	2.15	2.65	2.98
90	2.20	2.73	2.97
95	2.40	2.78	2.97
100	2.57	2.83	2.99
110	2.60	2.91	3.00
120	2.82	2.96	3.00
130	2.92	2.98	3.02
140	2.94	2.99	3.01
150	2.98	3.00	3.04
160	2.99	3.00	3.04
170	2.99	3.01	3.06
180	3.00	3.02	3.13
190	3.00	3.04	3.12
200	3.00	3.06	3.24

TRITA-EECS-EX-2019:393