ENHANCED BIO-INSPIRED ANT COLONY OPTIMIZATION ALGORITHM FOR FAULT-TOLERANT NETWORKS

BY

LUSWETI SAMUEL WAFULA

A Thesis submitted to the Board of Postgraduate Studies in Partial Fulfillment of the Requirements for the Award of the Degree of Master of Science in Information Technology of Masinde Muliro University of Science and Technology

October, 2024

DECLARATION AND APPROVAL

Declaration

This thesis is my original work and has not been presented for an award of a diploma or conferment of degree in any other university or institution.

Signature: …… Date: ……
Lusweti Samuel Wafula
SIT/G/15/15

Approval

Supervisor

This thesis has been submitted with our approval as the university supervisors.

Signature: …… Date: ……
Dr. Collins Odoyo
School of Computing and Informatics
Masinde Muliro University of Science and Technology

Signature: …… Date: ……
Dr. Dorothy A Rambim
School of Computing and Informatics
Masinde Muliro University of Science and Technology

ABSTRACT

This this presents a summary of introduction, literature review, methodology, research finding, conclusion and recommendations. Today, computer networks form a very important part of institutions and organizations helping them to run their operations swiftly and on time while keeping them up to date with the current changing technologies in the market. The networks are also used by many people to access resources on the internet and also for communication purposes. This has made the world to be indeed a global village. However, networks may be intentionally deterred by the introduction of forced physical loops in the switches, which is common in institutions of higher learning and organizations. As a result of these network loops, packets may be got up in cycles limiting communication process due to network convergence issues. Secondly, the research sought a way of addressing the convergence issues in trying to solve network problems in ACO. To do this, the study aimed to understand the number of artificial ants needed to find an optimal solution in the search space. MMAS, RankedAS and EliteAS were studied for optimal number of ants needed. It was found that the optimal number depended largely on various factors including 1) type of ACO algorithm in use, 2) The complexity of the problem under study, and 3) The ratio of the number of specialized ants to that of normal ants. Lastly, the study aimed at establishing the functionality of current network systems, evaluating network faults, and developing an enhanced model based on the existing ACO model to help solve these network issues. The new model developed suggests ways of solving packet looping and traffic problems in common computer networks. The study used simulation method to carry out research whereby an enhanced algorithm was developed and used to monitor and control the flow of packets over the computer network. The research employed experimental research design that involved the development of a computer model and data was collected from the model. Packet traffic was monitored by the Cisco Packet Tracer tool. In this tool, a network of four computers, a router and two switches was and used to simulate a real network system. Data collected from the simulated network was analyzed using the ping tool, direct observation of the movement of packets in simulation mode and message delivery status displayed by the Cisco Packet Tracer in the real time mode. In the experiment, a control was used to show the behavior of the network in ideal conditions without varying any parameters. Here, all the packets sent were completely and correctly received. Secondly, when a loop was introduced in the network it was found that the network was adversely affected because none of the packets sent by the computers on the network was delivered due to stagnation. In the third experiment, still, with the loops on, a new Enhanced ACO (EACO) model was introduced in the Cisco Packet Tracer used to simulate the network. In this experiment, all the packets sent were completely and correctly delivered just like in the control experiment. In summary, this research found that when conditions of networks are varied for instance introduction of forced physical loops in computer networks, the computers lose communication. This is a common situation in colleges and universities where some students forcefully introduce loops into networks affecting the communication process. However, with a new EACO model, it was found that the problem can be solved where the looping packets can still be rerouted to their destination hence no effect on the communication process. From this research, we can conclude that EACO model can be applied in computer network systems especially dynamic systems having many users to solve the common problem of network loops. However, this research has some challenges in that the algorithm must be run on all computers on the network for optimal results.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS AND ACRONYMS

ACO- Ant Colony Optimization Algorithm

ANN- Artificial Neural Networks

APO- Artificial Plant Optimization

LFA- Leaping Frog Algorithm

GA- Genetic Algorithm

ACR- Ant Colony Routing

OSI- Open System Interconnection

MAC- Media Access Control

RTT- Round Trip Time

TTL- Time To Live

OSPF- Open Shortest Path First

MACO- Multiple Ant Colony Optimization

EACO- Enhanced Ant Colony Optimization Algorithm

CHAPTER ONE

INTRODUCTION

1.1 Overview

This chapter is organized into the following headings and subheadings. First is the background of the study, which describes the usage of bio-inspired systems in problem-solving using simple local means while suggesting the necessity to enhance the functionality of the existing ACO system. Second is the statement of the problem which discusses the gaps in the already available bio-inspired models aimed at solving network problems. After the statement of problem are the objectives and formulated research questions entailing what the outcome of the research would achieve. The significance of the study shows how important this research would be to the society. Lastly, the scope, limitation of the study, and theoretical frameworks are also discussed in this chapter.

1.2 Background of the study

A computer network is a group of computers, printers, and other peripheral devices that are interconnected via a channel/link so that they can communicate or share resources [1]. One can rarely do anything with data that doesn't involve a computer network since these networks enable us to share information and other resources [2]. Nevertheless, these networks have to be maintained well to keep providing users with these functionalities. Computer networks can fail to work especially when a network device fails or the communication link malfunctions or is being overused against its capacity [3]. The existing networks are more complex than the conventional networks therefore it becomes hard to create, install, manage and keep them up and running efficiently all the time [4]. As such, new technologies are needed to be employed in managing these dynamic networks that are at the center of business transactions today.

Self-organization technologies are therefore needed to help overcome the technical limitations [4] of computer networks for instance link malfunction, device failure or looping. There exist similar problems in real-life situations and their biological solutions that are naturally evolved can also be applied in networking paradigms to help curb the drawbacks [4]. These are commonly referred to as Bio-inspired systems. A bio-inspired system depicts a strong relationship between a proposed algorithm aimed at solving a certain problem and a biological or natural system possessing similar capabilities [5]. There exists a necessity to employ bio-inspired systems in computer networks because living organisms like the ant colonies look better organized in their daily activities than the current internet [5]. This is because of the resilience to failure by biological systems to both internal and external environmental factors [6].

The biological systems have been in use for some time since the 1970s with the introduction of Neural Networks making the network adopt self-organization and self-healing features in case of network outages and fluctuations. Ever since bio-inspired systems have been applied in many different fields as algorithms to enhance their functionality. Examples of bio-inspired algorithms applied in various fields include Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), artificial neural networks, flower pollination, bat algorithm, bacterial foraging, cuckoo search, genetic algorithm, leaping frog, firefly algorithm, artificial plant optimization, and artificial bee colony [7]. Among these algorithms, ACO is the most popular and most successful algorithm that has ever been used in combinatorial optimization problems [8]. Combinatorial optimization problems are concerned with discrete variables for finding values so that an optimum solution is found with respect to a certain objective function [9]. In real life ant colony system, during foraging, the forward ants acting as agents deposit pheromone trails along the routes that they take

for others to follow to the source of food. These pheromone trails acting as stigmergy would be handy in helping the future ants to make a routing decision [10].

ACO has for many years been applied in various fields in tackling complex optimization problems using simple means. For instance, the algorithm has been successfully applied in gaming theory, data compression, dispatch problems, feature selection, congestion control, estimation of parameters in dynamic systems, job scheduling problems, medicine for decision making, control of satellites, mining of social graphs, tracking of targets among others [7]. ACO has also been applied in computer networks to help find the shortest route possible in sending data from source to destination just like what the ants do during foraging, [11] thereby becoming the minimum-cost route existing between any two linked nodes. This has improved the functionality of networks while reducing the latency that packets experience to reach their destination in absence of the ACO algorithm. An ant in this algorithm is a mobile agent having the additional capacity to solve different problems like congestion and packet routing. This is made possible by continuous and consistent modification of the routing tables by the agents in response to any congestion in the network [12].

Most computer network switches fail to handle broadcast storms formed by network loops [14] leading to flooding of the network. In Kenyan institutions of higher learning like colleges and universities, some students forcefully gain access to network switches and forcefully introduce physical loops by interconnecting active ports of the same switch leading to packet loss [86] and break down of communication.

Apart from this, the growing numbers of users of both the internet and other networks have seen even more serious problems like traffic overload cropping into the application of computer networks. ACO is being applied in computer networking to solve many problems due to its ability to dynamically adjust its routing tables, enhance network robustness, improve routing efficiency

and fault-tolerance [84]. Broadcast storms, traffic overload and other problems have adverse effects on functionality, uptime and fault-tolerance of computer networks. Therefore, the problem of traffic overload also needs to be addressed by self-inspired ACO model due to its dynamic routing capability if well optimized.

ACO models are made up of simulated ants whose work is to forage while looking for the shortest route possible between source and destination. When applied in computer networks, these ants are translated in packets and send to the destination from the host for communication purposes [85].

ACO has proved to handle the looping issues in computer networks via destruction of cycling packets leaving other packets in a consistent state. That is, if an ant (packet in networks) is stuck in a cycle (loop) and is forced to revisit an already visited node, it is destroyed [8] consequently, the nodes making up the cycle are also destroyed. This destruction solves the problem of broadcast storms but leaves the network with fewer nodes and loss of some important packets. As a result, further study to enhance the algorithm and apply it in computer networks for packet resilience formed the basis of this study.

1.3 Statement of the problem

Due to the importance of computer networks, the management of these networks is necessary for maintaining communication, reliability, and security [13]. One of the challenges facing computer networks is packet looping because it floods the network causing broadcast storms hence all connected nodes lose communication. This study focused on improving an existing Bio-inspired Ant Colony Optimization algorithm and applying the product in computer networks to address the network faults brought about by packet looping.

1.4 Objectives

1.4.1 General objective

To create an enhanced Ant Colony Optimization algorithm for fault-tolerant networks

1.4.2 Specific objectives

i. To establish the challenges facing the current computer network systems in Kenyan institutions of higher learning

ii. To evaluate the average number of artificial ants needed in ACO for optimal network convergence time

iii. To develop an improved ACO model for optimized resilience of computer networks

1.5 Research questions

i. What are the challenges facing the current computer networks in Kenyan institutions of higher learning?

ii. What is the average number of artificial ants needed in ACO for optimal network convergence time?

iii. Which type of ACO model can enhance resilience in computer networks?

1.6 Significance of the study

Computer networks are very important to the functionality of any institution; therefore, they must be up and running all the time regardless of any challenges they may face. As such, the technological changes in their maintenance are necessary. Some researchers have suggested the adoption of ACO to avoid the usage of infrastructure like nodes and switches which may malfunction failing the communication process [16]. However, in this current research, the researcher suggested how best the existing self-healing ACO algorithm can be enhanced and

applied in the networks to solve problems associated with malfunctioning of the infrastructure instead of completely doing away with them. The model which is a product of this study will help the institutions to get a solution to network issues like looping and congestion. Aside from this, the results of this research will consequently contribute to the body of knowledge.

1.7 Scope of the study

The study focused on the development of an Enhanced ACO (EACO) model using python programming language to make computer networks fault-tolerant. This study was carried out in MMUST computer lab where the model was developed. Cisco packet tracer network simulator was used to simulate the movement of data packets over the network developed. Some parameters were varied for instance, introduction of loops in presence of the developed EACO model, simulations were then done and results were collected. The study was confined to improving the tolerance of computer networks against packet routing problems and took approximately 5 months.

1.8 Limitation of the study

The researcher carried out this research using a network simulator and a real network. Nonetheless, it was applied on a small network segment, a complex network may give varied results especially on the average packet RTT. Secondly, the experiments were performed within short simulated periods of time, we are not sure if the results may be altered by long time durations. Lastly, the program must be run on every computer in the network to be effective, this means new computers added to the network may not work well if they have no algorithm installed thereby affecting the network.

1.9 Definition of Terms

Stigmergy: This is the mechanism of indirect and spontaneous coordination among agents leaving a trace in the environment stimulating or informing the performance of a subsequent action taken by a different or same agent.

Metaheuristic:

A class of stochastic algorithms that use a combination of random movement and local search. Metaheuristics are based on biological systems thus they learn from nature. Metaheuristic algorithms are usually designed for global optimization.

Pheromone trail: This is a chemical that a body of an organism secretes to impact the behavior of another individual receiving it. Many individuals secrete this pheromone and leave it as trails in the environment to lead other members of that species towards a source of food

Layer 2: Data Link Layer of the OSI Model

Forage: To search (a place) so as to obtain food.

Paradigm: A conventional framework, viewpoint, or collection of concepts. A paradigm represents a particular framework through which one interprets and understands a subject or phenomenon.

Algorithms: A systematic procedure or established guidelines to be adhered to in computational tasks or various problem-solving endeavors, particularly by a computer.

DoS: Denial of Service

***NP*-hard**: An issue is classified as NP-hard if a method for resolving it can be converted into a method for addressing any problem within the NP (nondeterministic polynomial time) category.

NP-hard signifies a level of difficulty that is at minimum equivalent to any NP-problem, though it may indeed present greater challenges.

Combinatorial Optimization Problems - These involve the application of mathematical techniques to identify optimal solutions among a limited array of potential options. The collection of potential solutions is typically delineated by a series of constraints, and this collection is excessively vast for a comprehensive examination.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

In contemporary society, the expansion of computer networks has reached a scale sufficient to accommodate the escalating requirements for resource sharing and communication. Nonetheless, they present a considerable array of challenges, both in the design of the networks and in their ongoing maintenance. In the realm of small networks, it is quite common to underestimate the significance of seemingly trivial issues that may arise from inadequate design and maintenance practices. However, it is rather complex to presume these challenges for extensive networks, as they could potentially bring the entire system to a standstill, obstructing the communication process. Network experts and security personnel are tasked with the management of extensive computer networks, addressing various issues that could potentially threaten the integrity of the network. At times, discerning transmission issues within an expansive network can prove challenging due to the multitude of potential problems that may influence its performance.

2.2 Introduction to computer networks

A computer network is a group of computers, printers, and other peripheral devices that are interconnected via a channel/link so that they can communicate or share resources [1]. These networks form the backbone for today's modern society as they enable seamless communication and information exchange [81]. There are a variety of computer networks available to day including Personal Area Networks (PANs), Local Area Networks (LANs), Metropolitan Area Networks (MANs) and Wide Area Networks (WANs).

2.3 Application of network in various areas

Application of computer networks promotes collaboration and information sharing thus eliminating geographical barriers [82]. Computer networks support organizations through activities such as online sales, and financial transactions, hence improving efficiency, convenience and security of these activities. Computer networks can also be applied in healthcare for managing medical information systems, electronic medical records, telemedicine etc [81]. These network applications accelerate the provision of medical services, improve the quality and efficiency of healthcare, and facilitate seamless communication between doctors and patients.

2.2 Challenges facing computer networks

Several obstacles confront computer networks in contemporary times. In our pursuit to enhance the human interactivity of computer networks, numerous challenges arise in the realms of development, operation, and maintenance of these systems. Some arise from deliberate actions, while others emerge as a consequence of expanding networks in response to a growing user base or heightened demand for requests. Network attacks such as Distributed Denial of Service (DDoS) attacks, SQL injection attacks, and Cross-Site Scripting (XSS) represent a significant threat to information systems [83]. Identifying and addressing diverse forms of network attacks promptly represents a significant challenge within the domain of network security.

2.2.1 Security threats

The security of computer networks is an essential issue, which generally may involve protecting the network from DOS attacks, maintaining the integrity of the networks, and preventing

unauthorized access. All of these problems facing the computer networks increase when the network is scaled up. Therefore, larger computer networks are susceptible to various kinds of attacks, because they contain more vulnerable attack points that malicious users can use to gain access into the networks [17]. The higher the number of users a network possesses, the more the passwords are used, as such, more hardware will be used increasing the points of weakness since attackers will always infiltrate through the hardware [17]. OSI was created in such a way that it allows the communication between the various layers to take place without the knowledge of what is happening at each layer. Among all the layers of OSI stack, Layer 2 forms the weakest point of attack from intruders [18]. The attacks may include MAC address attacks, Spanning tree attacks, and ARP attacks among others [18].

2.2.2 Traffic overload

Each computer network has its capacity of working. The capacity limits the bandwidth and traffic that the network [19] can support before it starts affecting its performance. Should an excessive number of nodes connect to the network beyond its capacity, the system becomes inundated with requests for data from these devices, resulting in a deterioration of the network's responsiveness.

2.2.3 Loops

Loops may arise within a network when there is more than one pathway (redundant links) at layer 2 connecting two endpoints, when multiple connections exist between any two switches, or when there is a physical link between two active ports on the same switch. Furthermore, the presence of numerous routes or pathways complicates the management of routing tables, thereby elevating the likelihood of encountering packet loops [15]. The loops creates broadcast storms during the forwarding of broadcasts and multicasts by switches. These switches repeatedly rebroadcast the message hence flooding the network [14]. This means that packets that are sent along a given path will eventually be stuck in that network forever making cycles, [20] unless some mechanisms are

invoked which will help in flushing such packets out of the cycle. Whenever you are using link-state protocols for instance OSPF, the forwarding loops may transiently occur if the routers in use adapt their own forwarding route tables in response to a change in topology [21]. The adverse effects of the loop on Ethernet switched networks include a reduction in bandwidth, memory clogging, and packet loss [14].

2.3 Bio-inspired systems

Biology is frequently being employed as an inspiration for research in computer science [3] and other fields like engineering, mathematics, energy [7], and business. Ant algorithms are in use today mainly to solve optimization problems instead of the problems and challenges that they were originally or initially developed to solve [22]. A direct approach to getting a solution to the combinatorial optimization problems is an exhaustive search [23], whereby the agents in these biological systems whose analogy is used to create bio-inspired systems, enumerate all possible solutions and choose the best one. The ant method, a bio-inspired system, has demonstrated superiority over other general-purpose algorithms for optimization, such as genetic algorithms. This becomes particularly clear when applied to combinatorial optimization challenges that necessitate the collaboration of multiple agents [22]. A considerable array of sophisticated algorithms is currently under development, aimed at addressing a multitude of intricate challenges. While certain studies endeavor to investigate the theoretical applications of bio-inspired algorithms, others are persistently engaged in enhancing the algorithms' functionality [7]. This serves as a catalyst for the advancement and evolution of artificial intelligence systems that emulate the foraging behaviors of ants. The algorithms encompass Neural Networks, Particle Swarm, Genetic Algorithm, and Ant Colony Optimization, among others [7].

2.3.1 Bio-inspired systems applied in other areas

The development of bio-inspired algorithms stems from the insights gained by researchers observing biological systems. Their introduction has since been utilized across diverse domains to address a range of combinatorial optimization challenges. Since their inception, they have exerted a beneficial influence across numerous domains in which they have been utilized. These systems have permeated various fields owing to their capacity for self-organization, healing, and adaptability. The subsequent enumeration presents various algorithms along with their respective domains of application.

2.3.1.1 Particle swarm

Particle swarm algorithms are algorithms that are either insect-based, bird-based, or animal-based approaches [24] that mimic the organization of these living things. Applications of the algorithms based on swarm intelligence can be deterministic optimization problems or constraint-based optimization problems among others. The areas where this Bio-inspired algorithm has been applied include chaotic systems, global optimization, location identification, oscillatory systems, resource allocation, path optimization, and adaptive learning [7].

2.3.1.2 Bee Colony Algorithm

The artificial bee colony algorithm has been in use for optimization of a single objective which is a numerical value [25]. The areas in which the bee colony optimization has been applied include bench-marking [7], feature selection, network routing, assignment, test suite optimization, and

probability distribution. The BCO algorithm is motivated by the intelligent behavior depicted by a colony of honey bees during the process of nectar collection [5]. BCO provides the population-based search approach whereby food sources are modified by artificial bees as time goes by intending to discover the food sources having a high amount of nectar and ultimately select the source that has the highest amount of nectar [5].

2.3.1.3 Bacterial Foraging Optimization algorithm

The bacterial foraging optimization algorithm [26] is used by researchers to explain how natural selection eliminates the organisms that are poor in searching, locating, and ingesting food as their own means of survival. The theory of foraging in this algorithm is based on an assumption that the individuals search for and get food in a way that maximizes their acquisition of energy per unit of time that they spent during the process of searching for food [7]. The organisms may deviate from their main objective of foraging as a result of reproduction, fighting, migration, or other environmental conditions [27]. This algorithm has been applied in many areas including gradient-based search, multi-optimal function optimization, linear combiners, forecasting models, numerical optimization, load forecasting and compensation, global optimization, minimization and maximization, portfolio value prediction, clustering, load dispatch and calibration [7].

2.3.1.4 Artificial Immune System (AIS)

These systems are computational systems that are inspired by the theoretical immunology alongside the observed immune functions, models, and principles [28], which are applied in complex problem optimization domains in systems that are supposed to be resilient to any kinds of faults, errors, or attacks. AIS systems are important since they possess a range of characteristics

that make them desirable to be applied in various areas. They are mainly used because they possess recognition, diversity, robustness, memory, distributed functionality, and elicit reinforcement learning [28]. Due to these characteristics, AIS has been successfully applied in the following fields: Agent-based systems, Fault and anomaly detection, Autonomous control and robotics, Scheduling, data mining (machine learning, pattern recognition) and Security of information systems [28].

2.3.2 Bio-inspired systems application in Networks

Numerous domains have witnessed the successful application of bio-inspired systems, several of which have been discussed in the preceding sections. Nevertheless, several of these concepts have been implemented within computer networks to augment their functionality and bolster their resilience against faults, with a primary focus on adaptive routing. An exemplary illustration is the ACO algorithm utilized for network analysis and adaptive routing [7]. Other examples also applied successfully in computer networks for adaptive routing include Artificial Neural Networks (ANN) for switching networks, Artificial Plant Optimization (APO) algorithm for the implementation of the telecom sensor network, Leaping Frog Algorithm (LFA) for network designing and scaling problems and Genetic Algorithm (GA) for network path routing [29].

ANN speeds up the computation of human brain, nevertheless implementing ANN in computer networks is very tedious and costly thereby making the algorithm implantation process very challenging [88]. GA is used for path routing because it is scalable, adaptive and is easy to implement in networking, nonetheless GA has a number of challenges too. Challenges facing GA include selection of encoding schemes, initial population, premature network convergence, selection of efficient fitness functions and degree of mutation and crossover [89].

Amongst these optimization algorithms, ACO has been most widely used and studied because it has shown better performance and has been the most successful algorithm [8] . Drawbacks of ACO are fewer than many other algorithms notably the most challenging issue affecting ACO is the stagnation of ants/packets which inhibits its adaptive routing nature [87]. This makes the model to be less tolerant to network faults and the only way ACO handles this is through destruction of stagnated packets. The destruction of stagnated packets is not a popular approach, as such, this study focused on the architecture and application of ACO in computer networks alongside its enhancement further for better network performance and fault-resilience.

2.4 ACO Architecture and its Application areas

Swarm intelligence [30], has been shown by biological swarms that have many powerful characteristics desirable in several engineering fields, for instance, network routing systems [31]. Consequently, new paradigms that can be used for coming up with autonomous systems may be a result of analytical comprehension of the operations displayed by the intelligent biological swarms leading to the development of more resilient bio-inspired systems. The ACO has been presented as the most successful swarm intelligence [32] algorithm as applied in many combinatorial optimization problems.

Swarm intelligence has a focus on the artificial recreation concept of the decentralized living organisms that are naturally intelligent and whose combined intelligence of the whole group is more vital than the sum of the intelligence of a single individual agent [7]. Based on this principle of swarm intelligence, some algorithms have been developed such as ACO, bacteria foraging, bird flocking, and fish schooling to solve the combinatorial optimization problems in complex systems [7].

2.4.1 ACO architecture and design

The early years of the 1990s saw the introduction of the ACO model proposed by Marco Dorigo and his companions as a metaheuristic optimization algorithm that is naturally inspired for solving hard combinatorial optimization problems [33]. ACO is in the category of metaheuristics which are probabilistic algorithms for obtaining good solutions to hard combinatorial optimization problems with a reasonable time of computation [34]. The other examples of metaheuristics include tabu-search [35], [36] evolutionary computation, and simulated annealing. The foraging behavior of real ants inspired the creation and deployment of ACO in various fields.

During the process of searching for food (foraging), the forward ants start by randomly exploring the environment that surrounds their nest before they locate the source of food and deposit pheromone trails [37]. ACO is a very popular and modern optimization paradigm that gets its motivation from the scenario of how ant colonies find the shortest routes between their nest and the food source [8]. Despite its well-known popularity and application, the theory of ACO is still under development and is in its infancy stages therefore a solid foundation of its theory is required [38]. The ACO model has undergone various modifications since its inception. Nevertheless, these forms exhibit a shared architectural framework. The subsequent illustrations depict a

representation of actual ants within a colony engaged in foraging, alongside a flowchart detailing the activities performed by ants in an Ant Colony Optimization algorithm during their foraging processes.

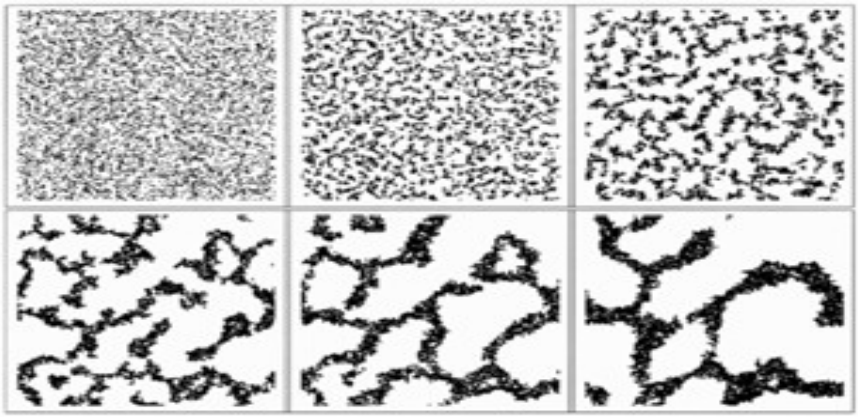


Figure 2. 1: Image of real ants during foraging [5]

Figure 2.1 above shows ants forming routes from random movements to clear trails that have been identified as routes to the source of food [5]. This foraging method is what the ants use to move to and from their nest to the sources of food.

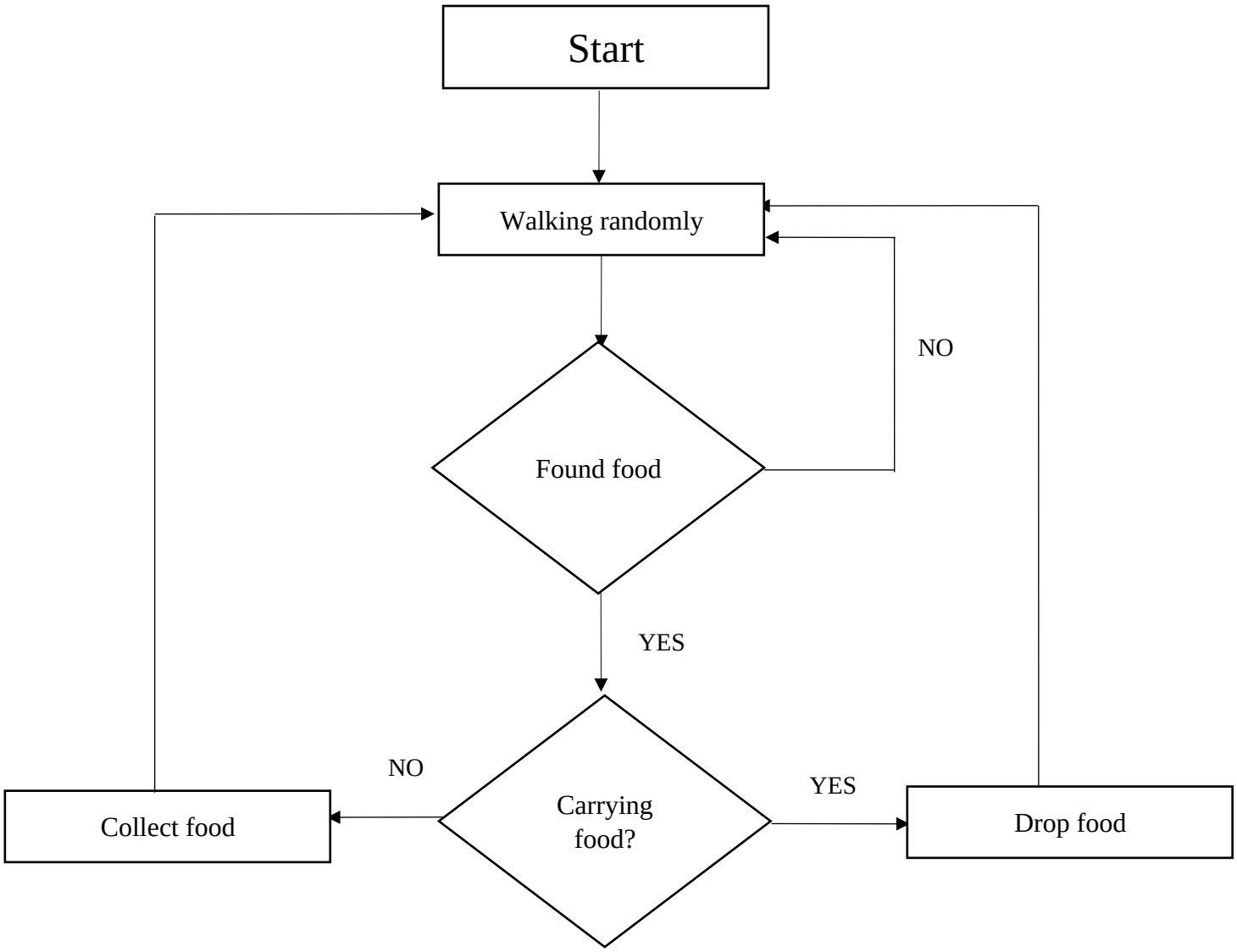


Figure 2. 2: Flow chart of foraging ants

Figure 2.2 shows a flow chart depicting the activities carried out by an ant during foraging as it moves from the nest randomly looking for food activities. Figure adopted from [5].

2.4.2 How ACO works

A typical ACO algorithm depicts the following characteristics: [39], [40]

i. Mimics the foraging behavior of ants

ii. Consists of a strategy for path finding heavily relying on the following conditions

a. Random movement and search to explore any alternative solutions to search space

b. The intensity of pheromone trails deposited by ants thus showing the quality of previous search operations

iii. The ants select the shortest paths that have high intensity of pheromone-based on the probability of transition for ant *k* to move from location *i* to *j* (p_{ij})

$$p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \times [\eta_{ij}]^{\beta}}{\sum_{l \in J_i^k} [\tau_{il}(t)]^{\alpha} \times [\eta_{il}]^{\beta}} & \text{if } j \in J_i^k \\ 0 & \text{otherwise} \end{cases}$$

Equation 1

$\boldsymbol{J_i^k}$: list of not yet visited nodes, ant k avoids visiting node *j* more than once using J_i^k (tabu list)

$\boldsymbol{\eta_{ij}}$: visibility of node *j* when ant *k* is in node *i* (inverse of distance)

$\boldsymbol{\tau_{ij}}$: pheromone level at edge (i,j) (learned desirability of choosing *j*

$\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$: adjustable parameters {control the relative weight of trail intensity(pheromone) and visibility(heuristic) respectively} [39], [40]

iv. After completing a tour, each ant *k* lays a quantity of pheromone $\Delta\tau_{ij}^{k}(t)$ on each edge (*i*, *j*) according to the following rule

$$\Delta \tau^{k}_{ij}(t) = \begin{cases} Q/L^{k}(t) & if\ (i,j) \in T^{k}(t) \\ 0 & otherwise \end{cases}$$

Equation 2

$T^k(t)$: The tour done by ant k at iteration t

$L^k(t)$: The length of the tour done by ant *k* at iteration *t*

Q: Is a parameter that only influences the final result (constant)

v. Pheromone slowly evaporates. In this case, the following pheromone update rule is needed

$$\tau_{ij}(t) \leftarrow (1-\rho)\ X\ \tau_{ij}(t) + \Delta\tau_{ij}(t) \qquad \text{where } \Delta\tau_{ij}(t)$$

Equation 3

m: total number of ants
ρ : decay coefficient {0(decay),1(no decay}

Ants are self-organized biological systems exhibiting three main principles of self-organization mechanisms which include interaction between individuals, feedback loops, and local state evaluation [41].

ACO algorithm works following an indirect communication among the simple agents of a colony, known as artificial ants, enabled by their artificial pheromone trails as their media of communication [42]. The foraging ants thus communicate by laying pheromone chemicals on the ground as they search the environment for food [38]. The other ants are consequently attracted by the laid pheromone trails and therefore tend to closely follow previous ants. In the case whereby the foraging ants discover different routes between the nest and a source of food, the shortest path typically gets filled with pheromone quicker than the longer path [38]. A fascinating aspect of ants is their ability to find the shortest routes to the source of food. This is made possible only by the ability of these ants to follow the laid down pheromone trails by the predecessor ants taking in mind

that most ants are almost blind meaning visual aspects are not in use [43]. The more the ants take the shortest route, the more pheromone gets deposited, until almost all ants follow the shortest path [38].

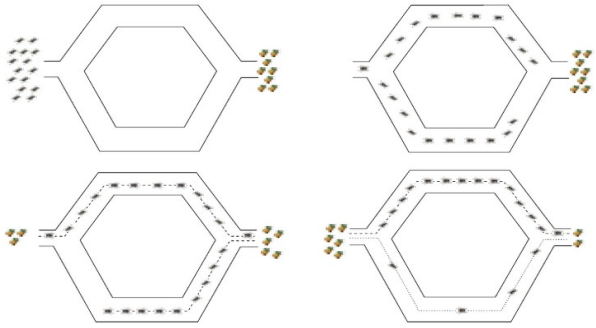


Figure 2. 3: Foraging ants [30]

Figure 2.3 shows foraging ants that finally end up with many of them selecting the shortest path between the source of food and their nest **[30]**.The behavior of the artificial ants is inspired by the real ants since they lay pheromone trails mathematically on the edges of the graph and select their path concerning the probabilities that rely on the pheromone trails [44]. These already laid down pheromone trails progressively decrease through evaporation.

2.4.3 Application of ACO

The ACO has been applied in two classes; the first one being *NP*-hard combinatorial optimization problems and the second being the search for shortest path problems [43]. The very first

combinatorial problem solved by the ACO was the Travelling Salesman Problem (TSP) which is the best-known example of *an NP*-hard problem also depicting a problem of the shortest path, as such the behavior of ants becomes the best solution [43].

The next problem to be solved by ACO in that order is the Quadratic Assignment Problem(QAP) [45], [43] and the Jo-Shop scheduling problem [46] [43] which responded well to the ACO model. These applications worked well and there was a need to advance and enhance the functionality of the then ACO model. The next works were the introduction of the network routing applications relying on the ACO [43]. Among these applications was the AntNet [47].

Beginning in 1997, there was a remarkable increase in the number of ACO applications across various domains. These applications included Sequential ordering [48], Flow shop [49], Group Coloring Problems [50], and Vehicle Routing [51].

Today, ACO has garnered significant scholarly attention and has been refined to address increasingly intricate challenges. The ACO meta-heuristic is presently being utilized in innovative real-world challenges and is demonstrating highly practical and encouraging outcomes, such as in the design of Combinatorial Logic Circuits [52]. There are also many routing algorithms based on ACO algorithms that are designed to be used for MANET enhancement [53]. However, the ACO shows some similarities with some other optimization, modeling, and learning algorithms like the Neural Networks and the Heuristic graph search [40].

2.4.4 ACO Application in Networks

Numerous challenges in networking are articulated as problems of multidimensional optimization. As the dimensions of networks expand, both in terms of the number of nodes and spatially, the feasibility of centralized control over communication within these networks diminishes significantly. When examining biological systems, the focus shifts from the individual to the

collective dynamics exhibited by larger groups, such as ant colonies, which present a more compelling subject of study [7]. As a result, Bio-inspired systems like ACO have been developed and successfully applied in network node research and design due to the appealing analogies existing between biological systems and large computer networks [3].

ACO has been applied in computer network routing problems in different formats such as AntNet, AntHocNet, ACR [8], and Ant-BasedControl(ABC) [53] among other algorithms. ABC was the first routing algorithm applied in circuit-switched networks for instance telephone wire networks [54]. This algorithm was deployed on a simulated version of the British Telecom network, which formed the basis [42] of more research in the area of network routing problems.

A highly successful application of ACO to the dynamic routing problems is the ANTNET algorithm, which was proposed by Marco D and Di Caro [55], [56], [47]. This ANTNET being successful was recommended and applied in packet-switched networks in this case the internet for adaptive routing [42]. Later on, the ANTNET became widely useful in mobile ad-hoc networks, to solve the routing problems and was used as ANTHOCNET posting exemplary results [57]. ANTNET [58] and ANTHOCNET [59] are two today very well-known ACO-based routing algorithms. ANTNET is a proactive routing algorithm while the ANTHOCNET is a reactive routing algorithm. They possess a very high delivery rate and find paths whose lengths are very close to the length of the shortest route [58], [60].

Two types of ants are applied in these algorithms which include the forward ants and backward ants [61]. ANTNET being a proactive ACO routing algorithm has been successfully applied in packet-switched networks [42], [62].In the ANTNET algorithm, a forward ant is launched from the nest or source node at regular intervals towards the food destination. An advanced ant meticulously records the nodes it has traversed in its memory. Upon arriving at its destination, it employs this

information to retrace its steps back to the nest or source, simultaneously updating the pheromone values along the paths or links it has taken.

The ACO has demonstrated several advantages, including favorable responses for swift solutions, adaptable applications that respond to alterations such as new distances, and intrinsic parallelism. However, it has also depicted some drawbacks which include probability distribution changes by iterations and convergence time [44]. More specifically, the drawback exhibited by ANTHOCNET is the actual number of routing messages that need to be sent in the network before the establishment of routes to the destination while [58] the draw-back of ANTNET is the time needed before a path system is established between any two nodes in the network. This time is referred to as the convergence time [44].

2.5 Number of ants needed for optimal functionality of ACO

According to [63] the number of ants is among the controllable parameters affecting the performance of ACO. This research examined 12 tests from four studies to determine how the number of ants in the ACO model affects its convergence time. Here, this study examined whether or not an optimal solution is found and how long it takes for the solution to be found as the number of ants is varied.

2.5.1 Experiment 1

This experiment was done by Aydin and Yilmaz [64]. The two presented an investigation into the number of ants used in ACO in relation to the number of iterations, penalized objective function, and optimization time. For the purposes of this study, the results obtained from the number of iterations as well as time of optimization versus the number of ants are taken into account.

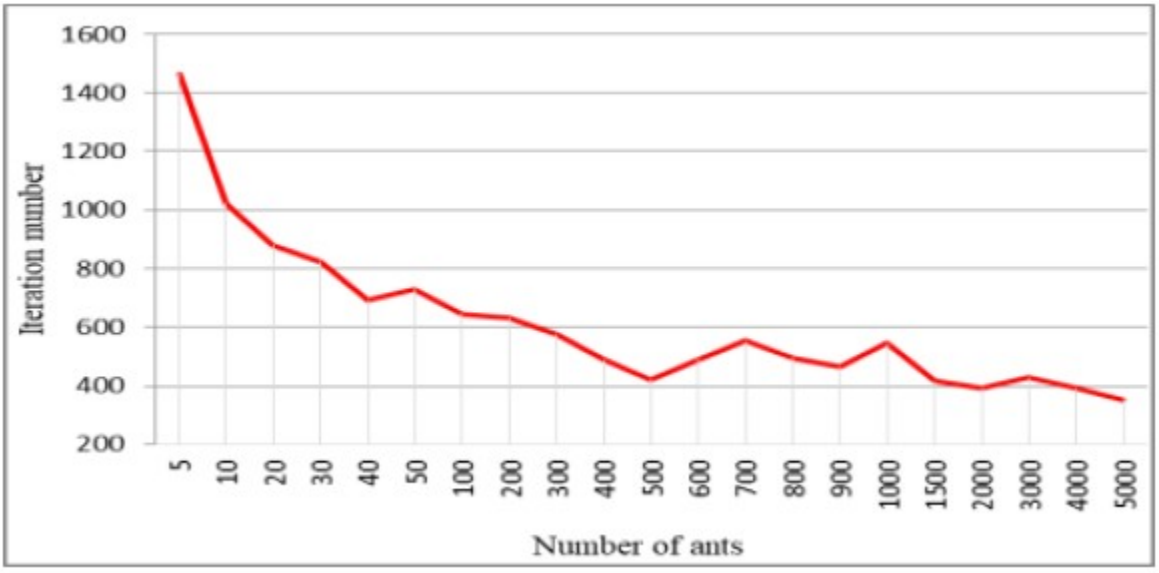


Figure 2. 4: Average Iteration Number versus Number of Ants [75]

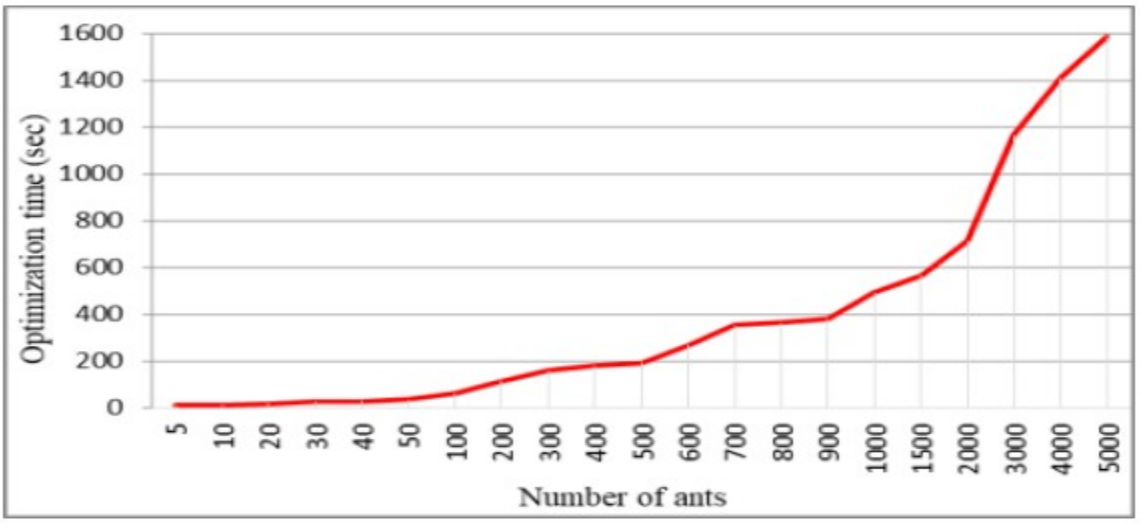


Figure 2. 5: Optimization Time versus Number of Ants [75]

From Figure 2.4 above, as the numbers of ants increase, the number of iterations reduces. In contrast to Figure 2.5, the optimization time increases quickly when the population of ants grows. In terms of the precise number of ants required for the optimum solution, this experiment did not give a direct answer, as it only suggests that it is critical to identify the appropriate number of ants

in order to get the best solution in the shortest amount of time [75]. Nonetheless, from the experimental results in figures 2.4 and 2.5, it is clear that the fewer the ants, the greater the number of iterations, and hence the shorter the optimization time. In other terms, when the number of ants rises, the number of iterations decreases yet the optimization time increases since numerous ants take longer to converge. However, this experiment was limited to small problems, and the exact number of optimal ants could not be established clearly [75].

2.5.2 Experiment 2

This study also examined the experiment done by [65], [66]. In this experiment, the researchers categorized optimization problems into small and medium scales using data sets of 50 and 100 cities respectively. They were able to measure the execution time, best solution and total number of new solutions obtained among other metrics. Figure 2.6 below shows the results obtained in terms of execution time against the number of ants for small scale problem of 50 cities. Figure 2.7 shows that increasing the number of ants causes an increase in the total execution time.

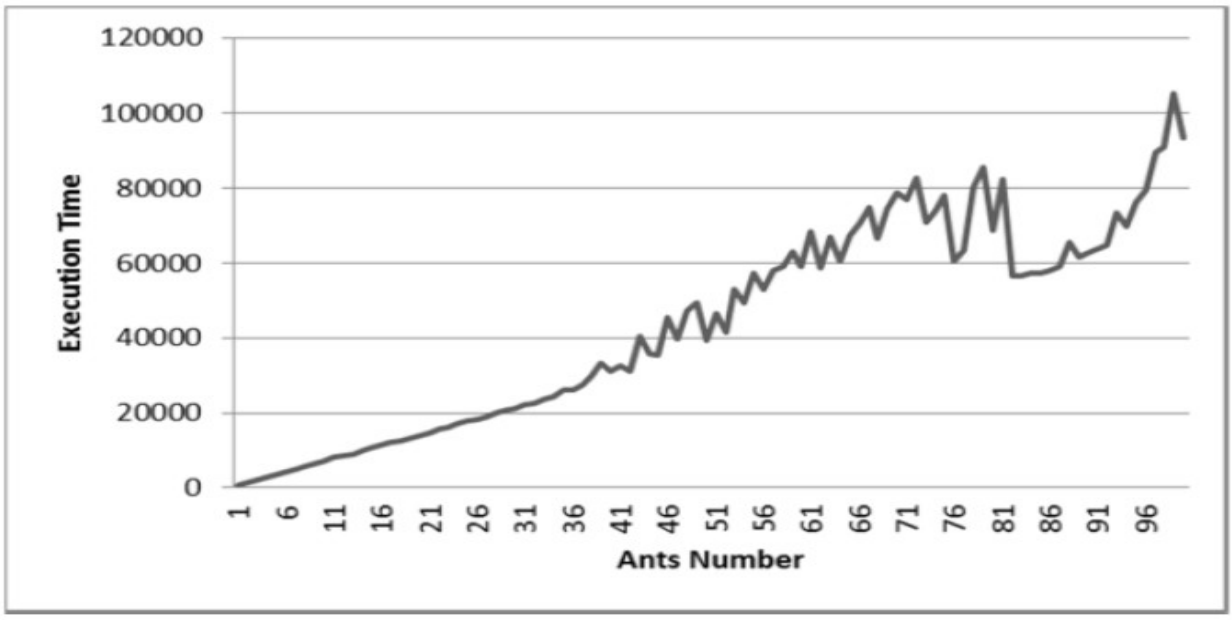


Figure 2. 6: Execution Time against the Number of Ants (50 Cities) [66]

When the experiment with 100 ants (medium) was run as shown below in Figure 2.7, a similar pattern was noted. However, due to the complexity of this issue, the execution time of the model increased in figure 2.7 as shown.

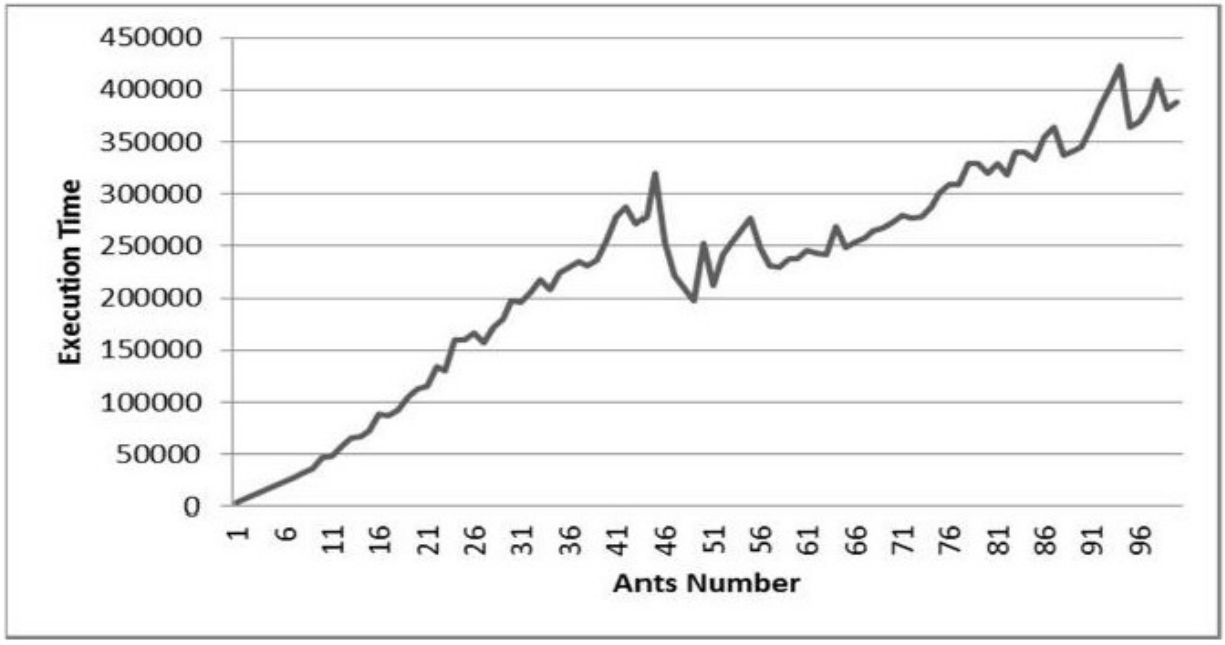


Figure 2. 7: Execution Time against the Number of Ants (100 Cities) [77].

This research reveals that increasing the number of ants did not improve the algorithm's performance, but if the number is low, the performance was enhanced. Nevertheless, the experiment was performed under two problem variations that is small and medium sized problems. For small sized problem the execution time was small and difficult to determine the exact number of ants needed to optimize the solution. Whereas on the medium sized problem, the execution time doubled and the number of optimum ants for that solution was found to be 16 out of 100 [77].

2.5.3 Experiment 3

In this experiment, Christoffer and Lars [76], [67] carried out comparisons on three ACO model variations i.e RankedAS, EliteAS and MMAS. The EliteAS relies on specialist ants to work. These secondary ants, or specialized ants, are utilized to impose the elitist theory/approach. The other ants

called the normal ants work differently. The specialist ants multiply the pheromone on the best solution found by normal ants making it stay longer without decomposing unlike the normal routes in other ant systems. The RankedAS on the other hand also use specialist ants but these ants deposit pheromones on many good paths found instead of depositing all pheromones on the best solution found. Every route is graded by length, with the best-ranked route getting the most pheromones and the worst- ranked route receiving the fewest. Lastly, the MMAS has no specialist ants which means it only uses the normal ants. Here, the pheromone deposited on a given path can never exit a maximum value or get lower than a given minimum value. This ensures that the pheromone level on a path does not get too low that the path is rendered unusable or the path should never be filled with the pheromone so much that it overshadows all the other routes [68]. This happens through smoothing of the edges whenever pheromone concentration levels are going below or above the extremes (See Figures 2.8-2.12).

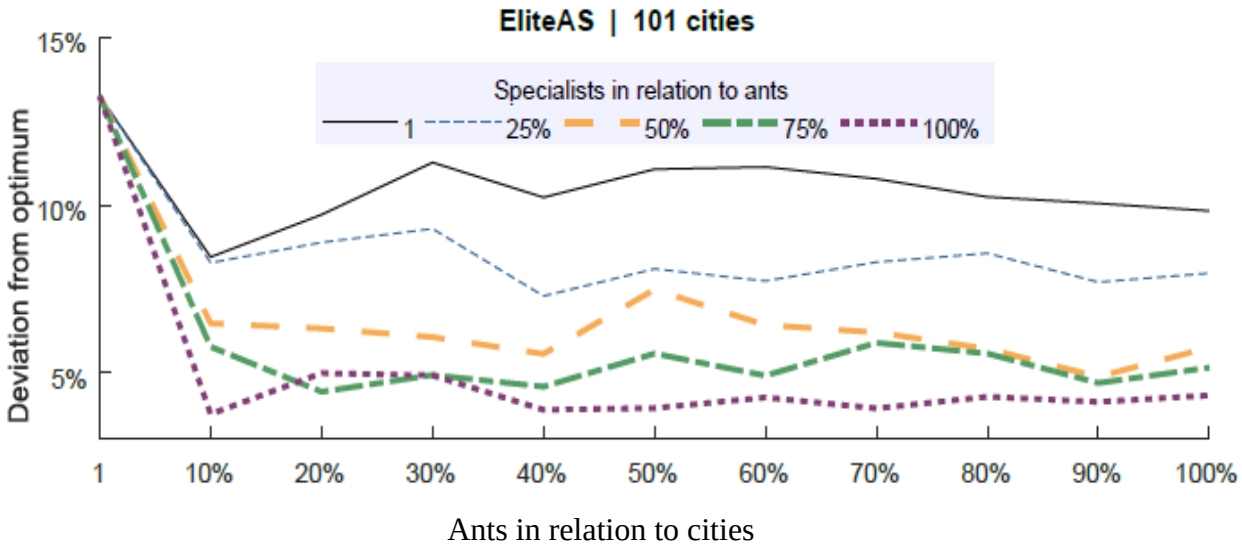


Figure 2. 8: Average Deviation from Optimum Ants in relation to 101 Cities [76]

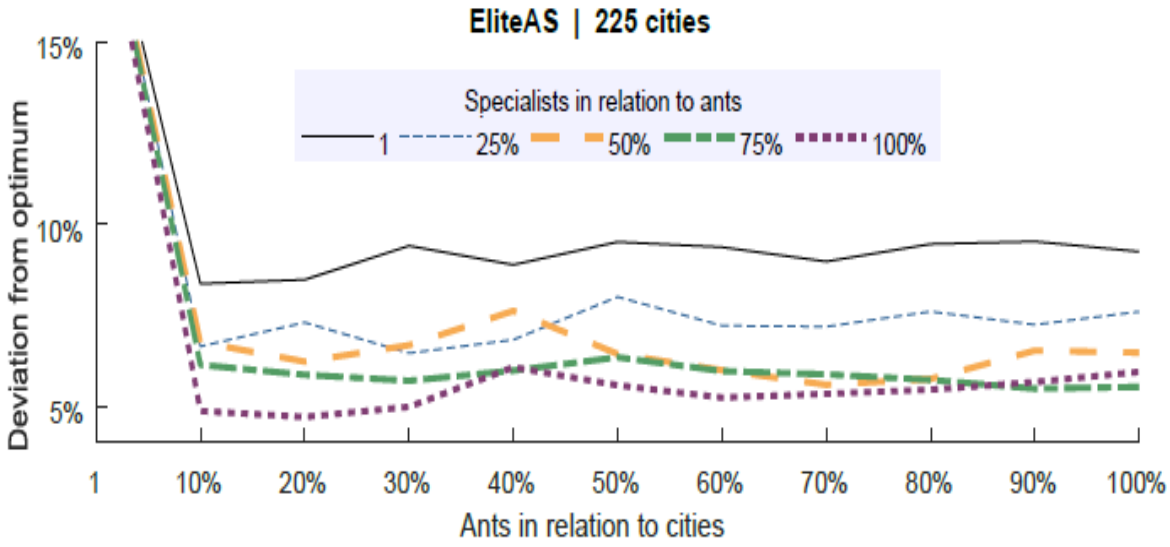


Figure 2. 9: Average Deviation from Optimum Ants in relation to 225 Cities [76]

In EliteAS as shown in figures 2.8 and 2.9, only between 10-30% of the normal ants in relation to cities toured showed a better performance when using 100% of the specialist ants with less than 5% deviation from optimum solution.

2.5.3.1 RankedAS

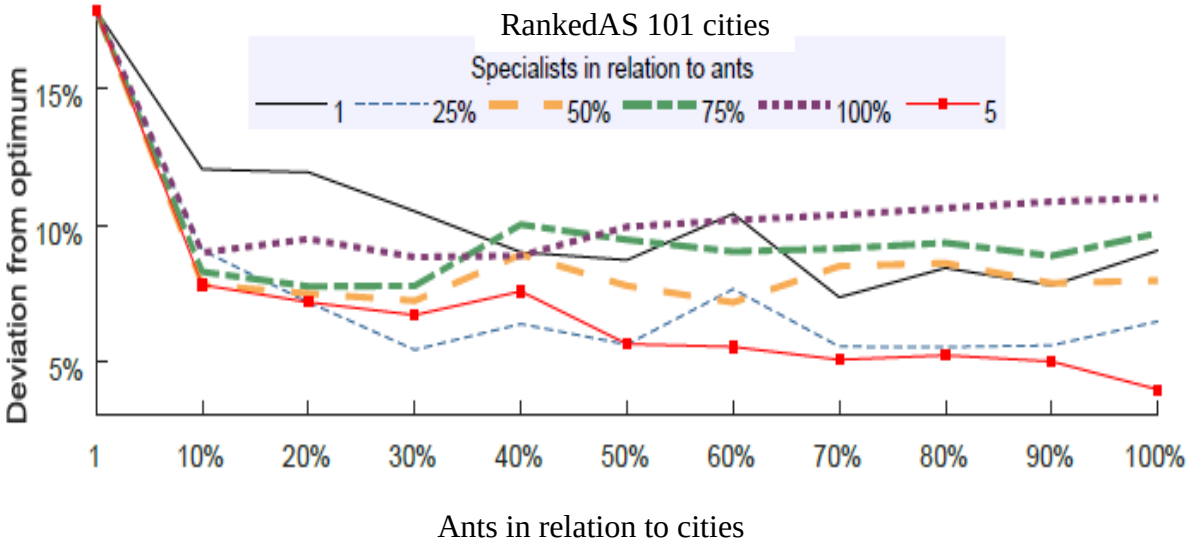


Figure 2. 10: Average Deviation from Optimum Ants in relation to 101 Cities [76]

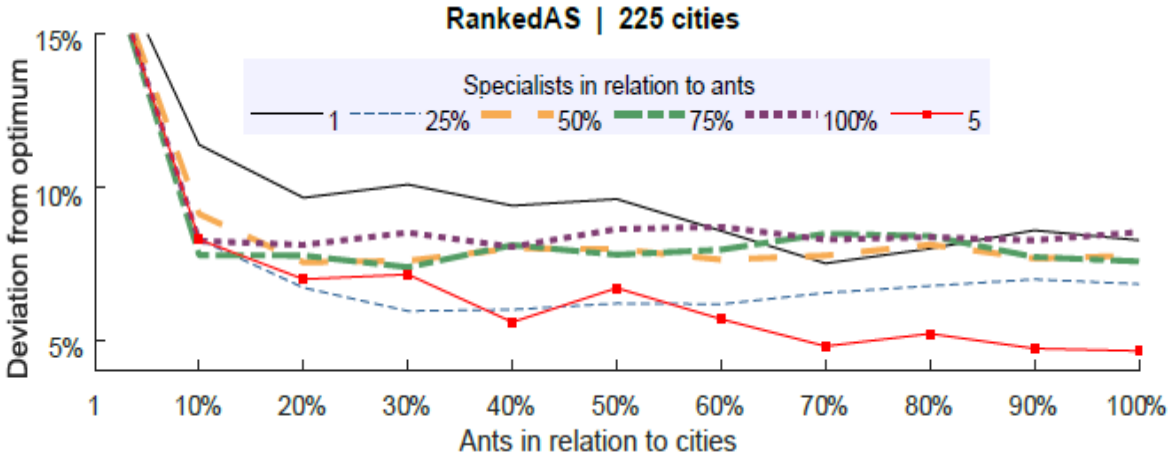


Figure 2. 11: Average Deviation from Optimum Ants in relation to 225 Cities [76]

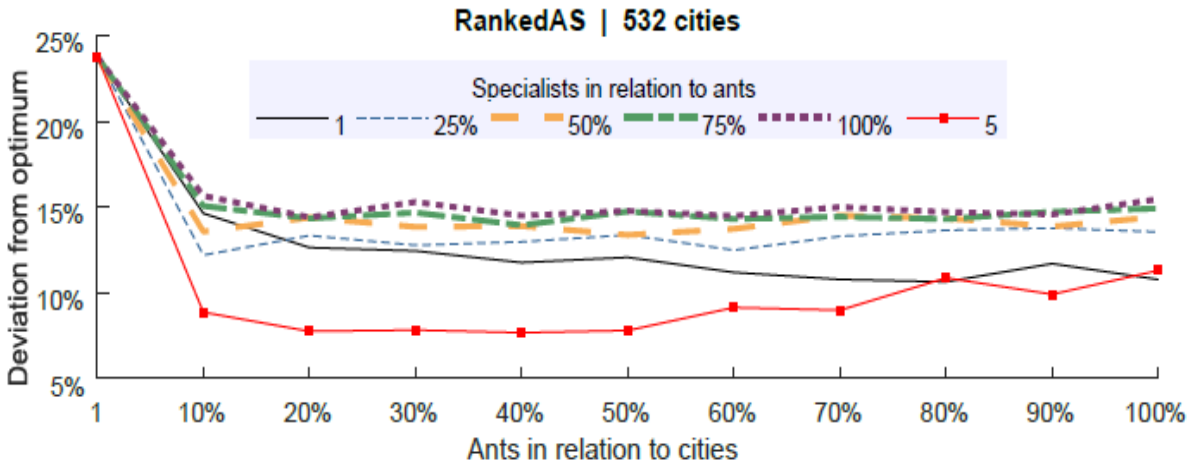


Figure 2. 12: Average Deviation from Optimum Ants in relation to 532 Cities [76]

As shown from the results in figures 2.8, 2.9, 2.10, 2.11 and 2.12 above, the results of RankedAS and EliteAS are contrasting each other. A bigger number of specialized ants in RankedAS improves the performance of the algorithm more than a low proportion of specialized ants to a certain level, while the opposite is true for EliteAS. In RankedAS, the number of normal ants has an effect of deviation from optimum solution when using 5 specialized ants than when using 1 specialized ant. In this case, the optimal solution is obtained when 5 specialized ants and 25% specialized ants are used in relation to 10% of the cities toured respectively [76]. This is proved by the blue line in figures 2.10, 2.11 and 2.12 with lower deviation from optimal solution. Furthermore, in RankedAS, 30% of cities toured by 25% of total ants showed better results with less deviation from optimum solution as displayed by figures 2.10 and 2.11 having 101 and 225 cities respectively. Lastly, in EliteAS, the highest number of normal ants which is 100% of ants showed better performance in terms of convergence showing the least deviation from optimal solution (See Figures 2.8 and 2.9).

2.5.3.2 Min-Max Ant System (MMAS)

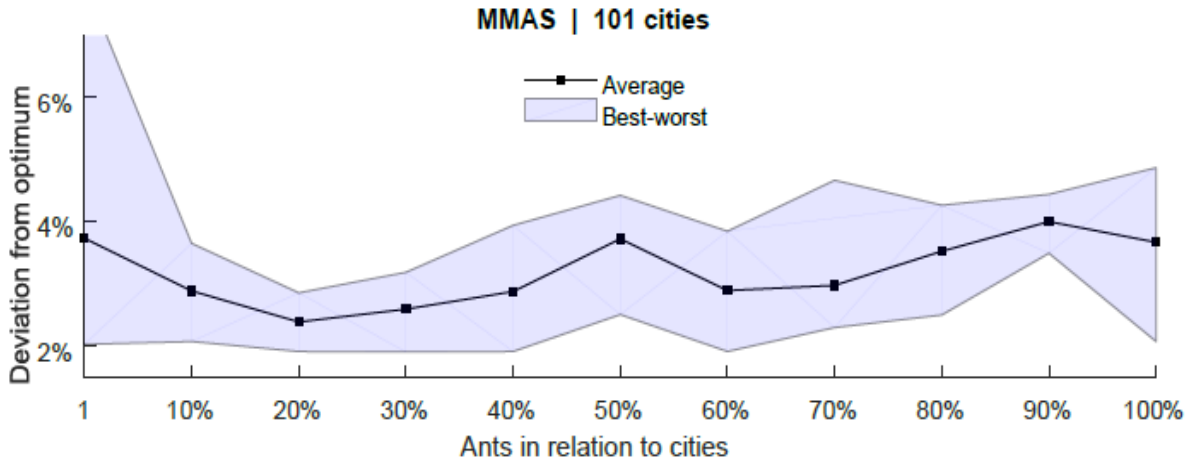


Figure 2. 13: Average Deviation from Optimum against Ants in Relation To 101 Cities [76]

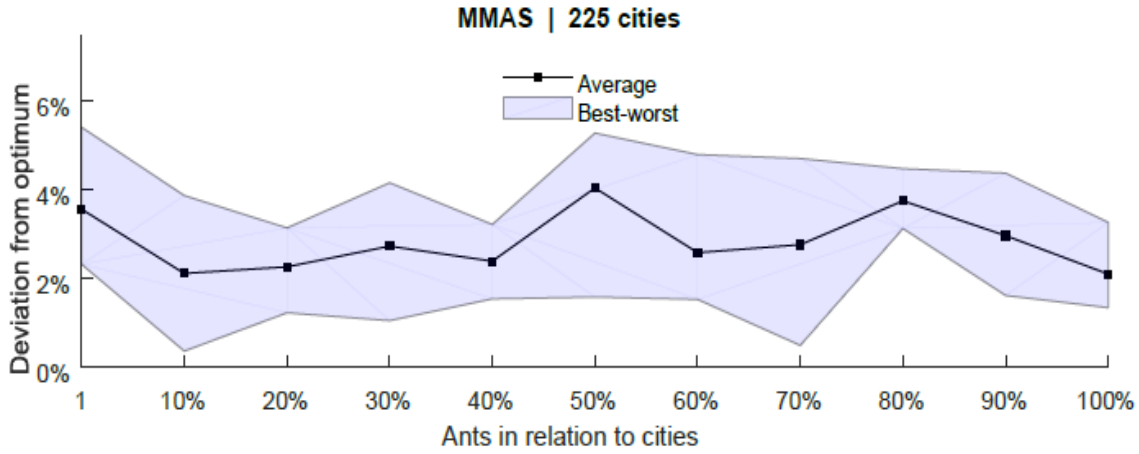


Figure 2. 14: Average Deviation from Optimum Against Ants in Relation To 225 Cities [76]

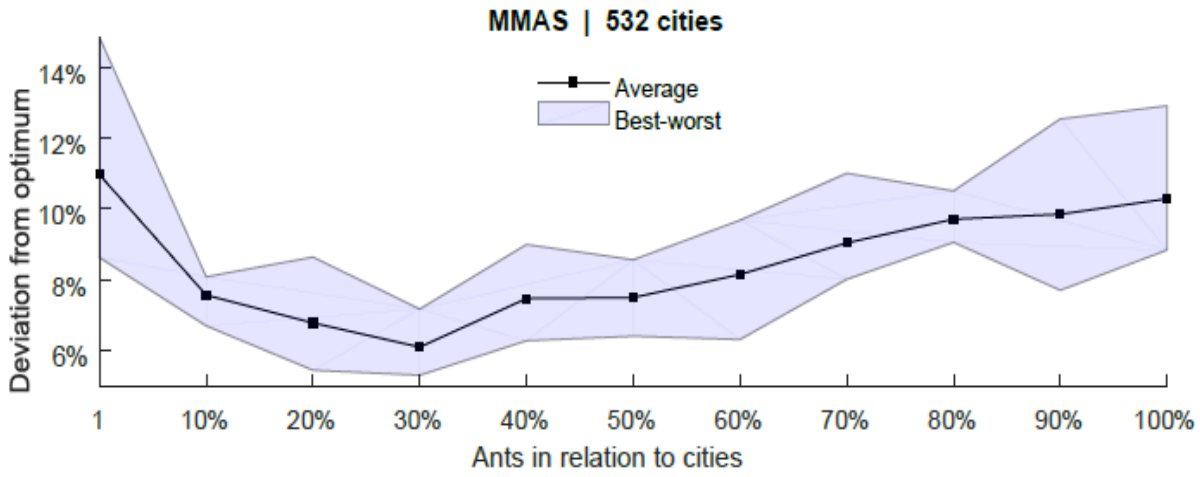


Figure 2. 15: Average Deviation from Optimum against Ants in Relation To 532 Cities [76]

Figures 2.13 and 2.14 show that increasing the number of ants in MMAS has no effect on the deviation from optimum results. Except for figure 2.15 which shows some variation, overall performance of MMAS is not largely affected by a large number of ants [76].

Discussion

While some experiments (see figures 2.4-2.7) in the data collected show that an increase in the number of ants degrades the optimization of various ACO versions, some of them (see figures 2.8 and 2.9) indicate that it actually improves the optimization of the algorithm. However figures 2.13 and 2.14 reveal that increasing the number of ants has no impact on the results of the algorithm. To help understand the variation in RankedAS, using one specialized ant has poor results than 5 specialized ants as shown by black and red lines respectively in figures 2.10-2.12. First, we look at the number of specialized ants kept at 5 in all the figures 2.10, 2.11 and 2.12. For optimal performance of RankedAS, for every 5 specialized ants, there needs to be 25% of normal ants and 30% of 101 and 225 cities toured and 10% of 532 cities. Simplifying this ratio, we get 30, 68 and 53 cities to be toured. Calculating the ratio between the number of specialized ants to that of cities toured in all the three experiments when the algorithm is at its optimum performance we find; 5:30,

5:68 and 5:53 for figures 2.10, 2.11 and 2.12 respectively. When we simplify the ratio, we get 1:6, 1:14 and 1:10 respectively. From this data we can see that for the optimal functioning of the algorithm the ratio of specialized ants and that of cities toured has to be considered. For instance, for every 1 specialized ant, there needs to be about 6 to 14 normal ants to optimize the solution as shown by RankedAS. This means that the complexity of the problem is determined by the number of cities or nodes toured by both the specialized ants and the normal ants. This is evident as shown by ACO, EliteAS, RankedAS and MMAS in figures 2.4-2.15 above. This calls for optimization of the algorithms in order to determine the shortest convergence time. When the ACO algorithms are optimized, they can then be applied in various fields of study. In this case when it is applied in computer networks, the packets would be translated into ants and made to be as intelligent as the ants of the algorithm, hence prevent packet looping. This reduces the convergence time and enhances the science of obtaining the solution space quickly and accurately. This helps improve on the network convergence time without making the time too short to cause premature convergence or too long to bring a lot of latencies in the network. It can also handle many other problems associated with un-optimized computer networks especially under dynamic situations.

2.4.5 Challenges faced by ACO

In the context of network routing, the application of ACO results in packets that emulate the behavior of actual ants. Nonetheless, it continues to face certain challenges. In this scenario, an issue emerges when an ant (packet) becomes ensnared in a cycle (loop) and is compelled to return to a previously traversed node, resulting in its destruction [8]. Destroying the packet breaks down the communication process and hence brings a limitation to this ACO. Although an improved ACO was developed known as MACO, to address this problem whereby, the ants are made to deposit

pheromones of different colors and where ants select routes with similar colors. However, MACO cannot fully eradicate stagnation of ants [12]. Consequently, too many ants that quickly reinforce suboptimal tracks [69] lead to early convergence to non-optimal solutions. Finally, too few ants would not really produce enough pheromone trail to make other ants select optimal paths, therefore, system of ants fails to achieve the desired cooperative behavior [69]. Lastly, according to [87], ACO is also faced by the problem of getting the correct values of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ parameters which if not well optimized lead to stagnation of ants, change in convergence speed as the number of iterations increases and exploitation rate.

Therefore ACO algorithm can be analyzed for future enhancement in such a way as to produce a better solution by enhancing its effectiveness and reducing the limitations mentioned above [44]. The limitations observed in ACO contributed significantly to this research, aiming to explore how enhanced solutions can be derived from this algorithm to ensure both responsiveness and fault tolerance in the context of computer networks.

2.5 Theoretical framework

A theoretical framework can be constituted by a researcher to explore, explain and interpret the behavior of events under study [70]. The theoretical framework makes connections between the the statement of the problem, research questions, and data to be collected together with the process of interpretation of the research findings [71]. This research study used a theoretical framework to help in creating a relationship between the objectives research questions and the data collected. The following is a machine learning graph showing the theoretical framework adopted from Barton [72] and modified to explain the theory of network loops.

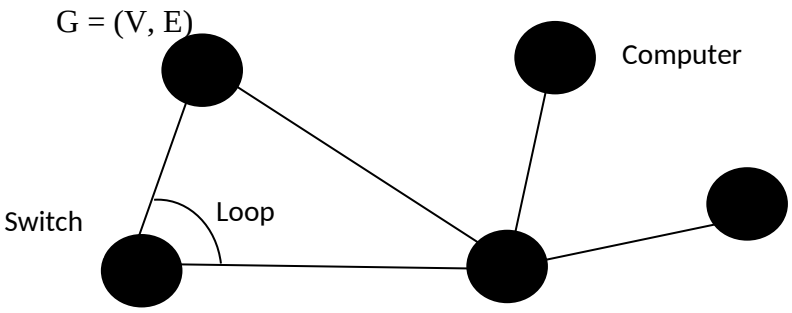


Figure 2. 16: Theoretical framework

Figure 2.16 above shows the theoretical framework of a network diagram. The graph has two main elements which include nodes and edges [72]. The edges illustrate the connections among the nodes; in this context, they may represent various network communication channels, such as Ethernet cables or Wi-Fi radio waves, facilitating the linkage. The entities in this scenario encompass computers, laptops, phones, or switches that function as terminals within computer networks. The initial node, which serves as the switch, introduces a network loop. This has implications for the network's stability and the flow of packets throughout the system. The graph utilizes Ramsey theory, illustrating that a graph G = (V, E) consists of a collection of edges and vertices, where V represents the vertices and E denotes the edges of the graph G [72].

This research adopted Graph Theory concepts to address the objectives of the study as follows:

Objective 1: Challenges facing the current network systems in Kenyan institutions of higher learning. This study used graph theory to develop a structure of four connected nodes and a loop attached on the switch to test the connectivity of the hosts as shown in figure 2.16

Objective 2: Optimal number of ants. The study used the graph theory's Dijkstra's algorithm to find the shortest route between any two nodes on the network as well as Ford-Fulkerson algorithm to find the number of routing messages required in the network for optimal convergence time.

Objective 3: Resilience of networks. This study adopted Depth-First (DFS) Cycle Detection to detect cycles in nodes interconnected to form the graph. This problem was addressed using EACO model that was developed in the study.

Within the aforementioned framework, the switch serves the purpose of directing packets towards their intended destination computers within the network. Nevertheless, when the packets become ensnared in a loop, it presents significant challenges for their rerouting to the intended destination. While implementing ACO proposed by [8] such packets will be destroyed. The research aimed to investigate how best the packets can be rerouted to their intended destination without being destroyed.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

This chapter delineates the chosen research design, the SDLC methodology employed, considerations regarding ethical issues, the processes of data collection, and the assessment of reliability and validity pertaining to Cisco Packet Tracer, alongside the analysis of the data gathered. The study was conducted within institutions of higher education, with MMUST serving as the focal point for the experiments. The researcher chose this institution because of its closeness to computer networks that experience high levels of simultaneous user access, as well as the numerous instances reported regarding the deliberate introduction of physical loops into the institution's network by unidentified individuals.

3.2 Research design

This study examined the importance of a research design in the process of data collection and selected experimental research design which is relevant to this type of research. According to [90], a research design provides a blue print for integration of all elements of a study to make the results unbiased, credible and generalizable. Research design is like a structure or a plan for one's research and is divided into five primary structures [91], out of which, this study adopted one-

experimental research design. Experimental research design is a framework of procedures designed to carry out experimental research with a scientific view while making use of two variables [92], the independent and dependent variables. Experimental research design is important since it is the backbone of a good research and provides a roadmap to readers. Experimental research design directs the experiment through data collection, definition of statistical analysis of the collected data while guiding result interpretation and presentation [93]. In this study, experiments were done and the simulated data that was collected was used to develop an enhanced ACO model for fault-tolerant networks.

3.3 System Development Methodology

A software development methodology serves as a structured framework utilized for the planning, management, and oversight of the entire process involved in creating an information system [73]. Software development methods are vital tools used by the project development team to plan, structure, and control the development of IT projects [74]. Most IT projects or software are characterized by delays. In a study carried out by the Standish Group in the year 2009, it was concluded that 32% of all IT projects were delivered on budget and on time [75]. The study also found out that 44% of the projects were delivered late and over-budgeted while 24% of the projects completely failed to be delivered or were never used at all after delivery. In most of the cases, the project failures resulted from either not using a methodology of software development or using the wrong methodology [76]. As a result of these failures, therefore almost three-quarters of IT projects face either one or more problems including time overruns, total failure, not according to user requirements, or cost overruns [77]. Software development methods are concerned with all the activities that take place throughout the System Development Life Cycle, without which it is very hard to develop a good system due to omissions [73].

There are many software development methods that system developers use during the coding of the software. All these methodologies follow some kinds of phases in the development of software which includes problem definition, requirement analysis, design, coding, testing, implementation, and maintenance [78]. The following are some of the methodologies used in the software development cycle: waterfall model, agile, incremental, iterative, and Joint Application Development among others [74]. The choice of a methodology depends on the nature and size of the software under development.

3.3.1 Agile family methodology

This study used the agile family since these methods are meant to quickly adapt to changing requirements, and minimize costs of production while upholding the quality of the software under development [74]. Agile is a combination of both incremental and iterative types of SDLC [78]. Here the user requirements can easily be changed according to the needs of the user. It helps the software developers in planning and coordinating their work well involving all the stakeholders from within and external stakeholders [79]. The agile methodology of software development adheres to the conventional Software Development Life Cycle (SDLC), encompassing stages such as requirement elicitation, analysis, design, coding, testing, and implementation. This approach places significant emphasis on customer satisfaction [78]. It necessitates minimal planning and segments tasks into manageable increments, adopting an iterative approach that allows for modifications based on user satisfaction. This entity possesses the subsequent attributes [78]. The initial aspect is iterative, indicating that the primary aim of the agile methodology is to concentrate on a singular requirement through multiple iterations. The second aspect is modularity, in which the agile process facilitates the decomposition of the system into smaller, more manageable

modules. The third characteristic is incremental, permitting a system to be developed in discrete increments. Each increment stands alone, yet by the conclusion of the cycle, all increments are synthesized into a cohesive whole. The algorithm formulated in this study underwent multiple iterations prior to achieving stability and producing consistent results. Every iteration represented a refinement over its predecessor, resulting in enhanced convergence time, increased randomization, and optimized pheromone evaporation.

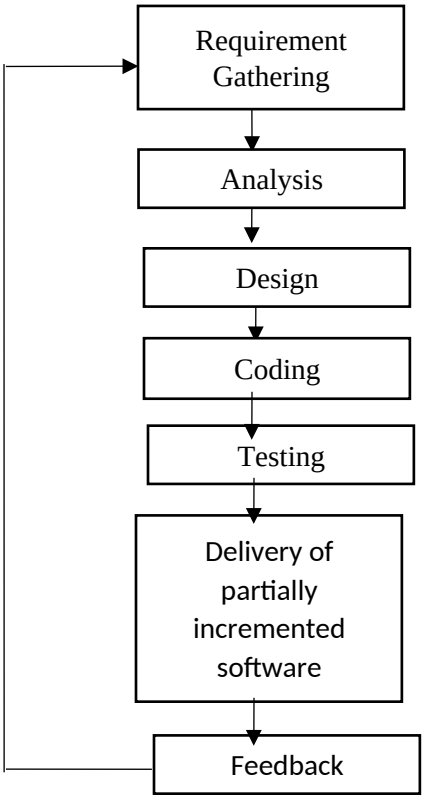


Figure 3. 1: Agile phases

3.4 Research instruments

According to [94], there are six sources of evidence in resesrch. These source include, documentation, direct observation, interviews, physical artifacts, archival records and participant observation. This study employed simulations for collecting data and direct observation used to collect and document data. The simulators used in this research are the Cisco Packet Tracer and

Python Pygame simulator. In Cisco Packet Tracer, various parameters were varied in the network infrastructure according to the objectives of this study and observations made. Similarly it was observed that the variations in the network infrastructure had an effect in the movements of artificial ants on the Pygame simulator.

3.6 Ethical Issues

The researcher filled research permit form approved by both the supervisors and the Directorate of Post Graduate Students. The researcher then got a research permit from National Commission for Science, Technology, and Innovation (NACOSTI) to carry out the research in MMUST.

3.7 Data Collection

The study used simulated experiments to help in testing and manipulating some variables which were then tested against their effect on the behavior of the computer network. In this case, a model was designed and coded using python programming language, and python pygame simulator was used to simulate the foraging behavior of real ants with specific packages like scikit-learn (sklearn), pillow, NumPy, pandas, TensorFlow, matplotlib, scapy, openCV among others. This model was based on the existing ACO models for network routing problems. It was thus modified and enhanced for better results of the delivery of packets in the network through self-healing mechanisms. First, from the cisco packet tracer, data was collected through direct observation from the simulation mode, real time mode and command line interface. From the simulation mode, the play controls were pressed and observation was made on the packets being sent and received between the four computer devices on the developed network. On the real time mode, the message status was observed as simulation continues. The message delivery is either successful, failed of in progress (see figs 4.1-4.12 in chapter four). From the command line interface, ping tool was used to test reachability of hosts on the network and results were visualized. Lastly, from the Pygame simulator, the movement of artificial ants is directly observed under various conditions mimicking

the real ants as shown by figures 4.13-4.17 in chapter four. As the researcher introduced loops in the network, the artificial ants formed the loops on the pygame simulator and test data was generated on excel sheet showing time taken for network convergence.

3.8 Data Analysis

The analysis of collected data for this study was done as follows;

Objective 1: Data was collected on the challenges facing the current computer network systems in Kenyan institutions of higher learning through the use of documents as secondary method of data collection and the collected data was analyzed using thematic analysis.

Objective 2: Data was collected to evaluate the average number of artificial ants needed in ACO for optimal network convergence time based on documented data. The various experiments done by various researchers was analyzed using descriptive analysis and a conclusion drawn on the optimal number of artificial ants needed.

Objective 3: Data collected aimed at developing an improved ACO model for optimized resilience of computer networks was analyzed using predictive analysis. The data collected in presence of network loops can be used to predict the future behavior of networks when EACO model is used to enhance tolerance of computer networks to faults.

3.9. Validity and Reliability of Research instrument

Validity is the ability of a data collection instrument to measure the intended measurement [95] whereas reliability is the measure of consistence and accuracy of results. To ensure validity of Cisco Packet Tracer, checked the connectivity of the nodes on the network using ping command found on the Command Line Interface of the simulator by sending ICMP packets. Secondly, the researcher was able to visualize the movement of packets through the network in simulation mode.

For reliability, the researcher used internal-consistency to test if the simulator is able to produce similar results in different experiments. Various experiments were run packets RTT was observed for all the experiments. The RTT was found to be similar for every experiment run. Besides, CISCO packet tracer developed by Cisco Net Academy is a validated tool and used for simulating small and large networks as well as for teaching and learning purposes.

3.10 Original ACO model

3.10.1 Simple-ACO

The old model called Simple-ACO was applied in various fields including computer networks. S-ACO ants were used to implement loop elimination which in turn improve the performance of the system [23]. While moving backwards through the path that is in their memory, the ants update the pheromone concentration on the paths they pass through. The following example shows how ants in S-ACO eliminate loops using backward pass as presented by Marco Dorigo [23].

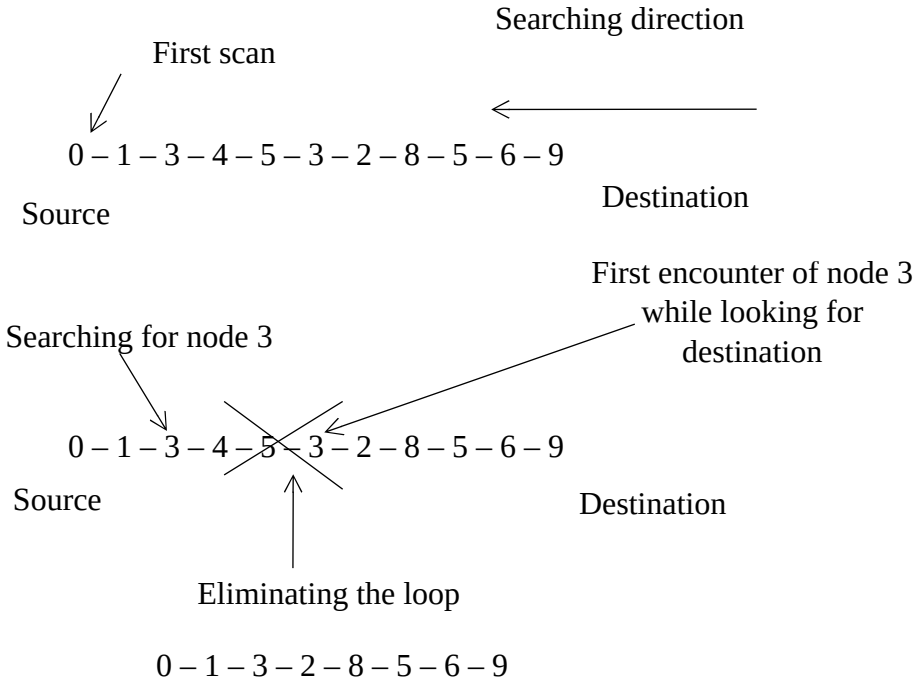
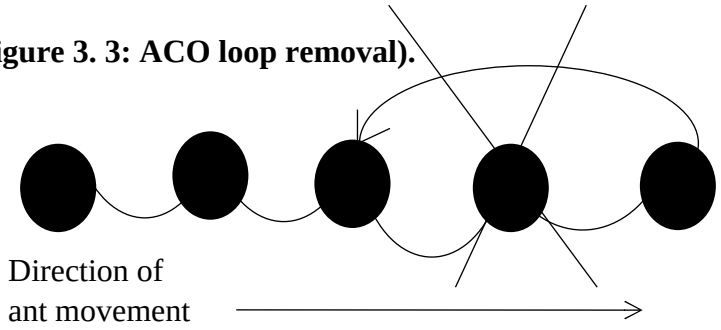


Figure 3. 2: S-ACO loop removal

3.10.2 Original ACO

In ACO, the private memory of ant is used to guarantee the probability of an ant building a feasible solution. However during the process of finding these feasible routes, loops may form making an ant to rotate in a cycle hence wasting time and resources [8]. When trying to avoid these loops, if an ant is forced to go back to an already toured node, the nodes having that cycle are removed from the private memory of the ant and information about the nodes is completely destroyed. If an ant moves in a loop for a time which is greater than half its age (Time To Live),the ant is consequently destroyed [8].

Figure 3. 3: ACO loop removal).



In ACO example above, loops are removed by detruction of node 4 ash shown by figure 3.3. The two examples of loops removal shown by figures 3.2 and 3.3 above, can have a general diagram of the ACO model as shown in flow chart in figure 3.4 below:

Original ACO model

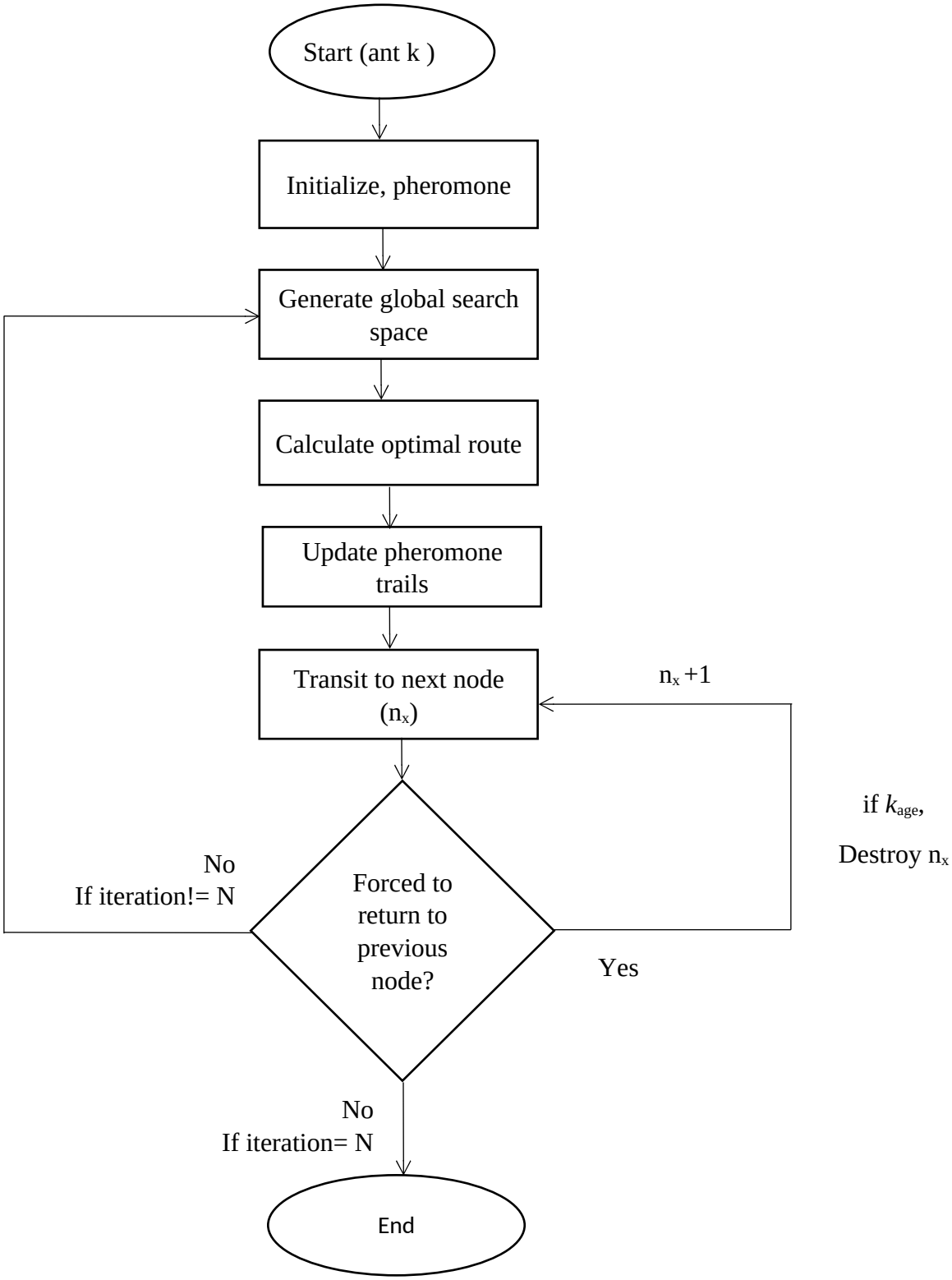


Figure 3. 4: Original ACO flowchart showing how to eliminate loops

flow chart, if an ant is forced to return to an already visisted node, the node of ant will be destroyed. This saves the system from staying in a loop forever but the killed ant never reached its destination. If the node is detryoed, then it is removed from the network making the node lose network connection.

Pseudo-code

```
procedure AntNet_ForwardAnt(source_node, destination_node)
   k ← source_node;
   hops_fw ← 0;
   V [hops_fw] ← k; /∗  V = LIST OF VISITED NODES ∗/
   T'[hops_fw] ← 0;        /∗  T' = LIST OF NODE-TO-NODE TRAVELING TIMES
   ∗/
   while (k != destination_node)
   t_arrival ← get_current_time();
     n ← select_ next_hope_node(V, destination node, T^k, L^k);/∗  L^k = LINK QUEUES
     ∗/
     wait_on_ data_ link_ queue(k, n);
     cross_the_link (k, n);
     T_k→n ← get_current_time() − t_arrival;
     k ← n;
     if (k ∈ V ) /∗  CHECK IF THE ANT IS IN A LOOP AND REMOVE IT ∗/
       hops_cycle ←get_cycle_ length(k, V );
       hops_fw ← hops_fw − hops_cycle;
      else
       hops_fw ← hops_fw + 1;
       V [hops_fw] ← k;
  T'[hops_fw] ← T_k→n;
    end if
     end
    while
   become_a_backward_ant(V, T');
end procedure
```

Algorithm 3.0 Pseudo-code of original ACO model showing how to remove loops [8]

From the above algorithm, when a forward ant is caught up in a loop, the loop is removed. When read together with the flow chart in figure 3.4 above, this means the ant is either destroyed or the nodes forming the loop are destroyed.

3.11 Enhanced ACO model Physical movement of ants

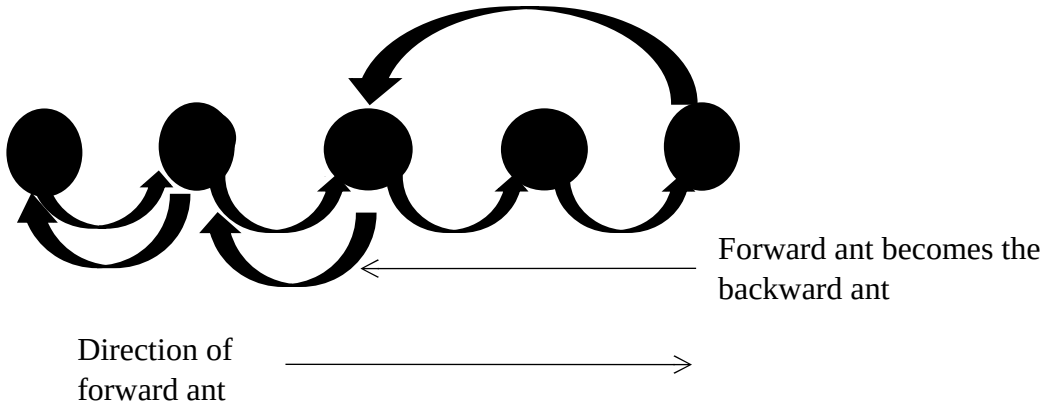


Figure 3. 5: Loop avoidance.

As shown in figure 3.5, if a loop is formed in the modified ACO model, it will be assumed by the ants as the ants will still be rerouted out of the loop if they cycle for more than half of their TTL. In this case no ant will be killed and no node will be destroyed. The ants are made to be a little more intelligent in that, when an ant is caught up in a cycle, its assigned time to live will be used to determine its position in the network. If the ant takes rotates in a loop for a time interval which is greater than half of its TTL, then it is rerouted out of the loop to its intended destination using a different route.

Enhanced ACO model

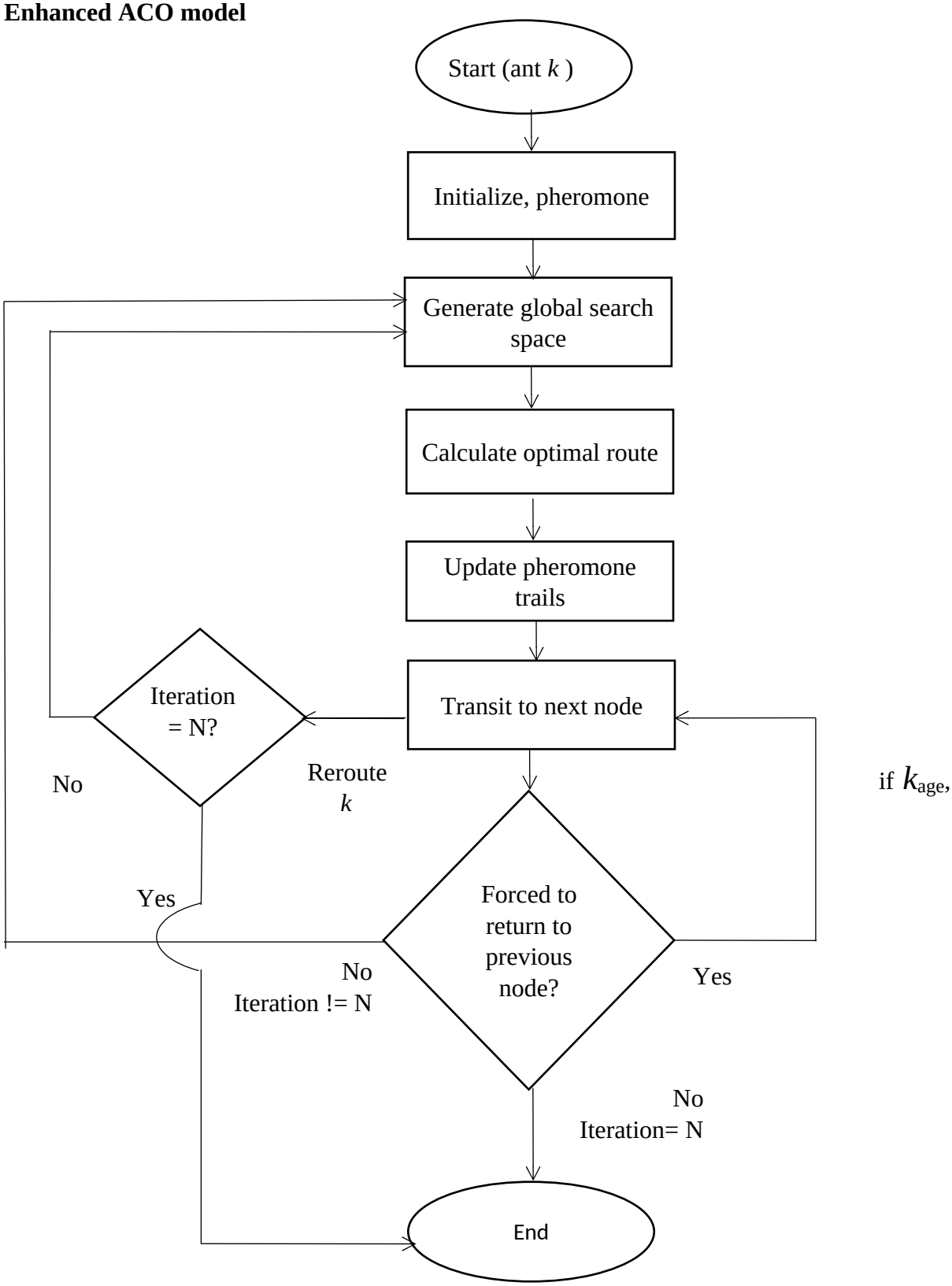


Figure 3. 6: EACO Flow Chart EACO model showing how to eliminate loops

Pseudo-code of enhanced ACO model

```
procedure AntNet_ForwardAnt(source_node, destination_node)
  k ← source_node;
 hops_fw ← 0;
 V [hops_fw] ← k; /∗  V = LIST OF VISITED NODES ∗/
 T ’[hops_fw] ← 0;        /∗  T ’ = LIST OF NODE-TO-NODE TRAVELING TIMES
 ∗/
 while (k != destination_node)
 t_arrival ← get_current_time();
   n ← select_ next_hope_node(V, destination node, T k, Lk);/∗  Lk = LINK QUEUES
   ∗/
   wait_on_ data_ link_ queue(k, n);
   cross_the_link (k, n);
   T_k→n ← get_current_time() — t_arrival;
   k ← n;
   if (k ∈ V ) /∗  CHECK IF THE ANT IS IN A LOOP AND REMOVE IT ∗/
         hops_cycle ← get_cycle_length (k, V);
         if (hops_cycle >TTL / 2) // Checking if cycle_length is greater than TTL/2
            hops_fw ← TTL / 2; // Assigning TTL/2 to hops_fw
         else
           hops_fw ← hops_cycle; // Assigning cycle_length to hops_fw if it's less than TTL/2
     end if
        hops_fw ← hops_fw + 1;
        V [hops_fw] ← k;
        T ’[hops_fw] ← T_k→n;
    end
   while
  become_a_backward_ant(V, T ’);
end procedure
```

Algorithm 3.1 Pseudo-code of EACO model showing how to remove loops

In the enhanced algorithm above, when an ant is got up in a cycle for a time greater than its half-life, *hops_cycle* > , then the ant is assigned to *hops_fw* function. This in turn increments the forward hops counter *(hops_fw* + 1) to get the ant out of the loop unconditionally. The ant randomly selects the next node described in its foraging instructions. This enhanced algorithm was applied in the

simulated computer networks to test if it can help reroute packets intentionally forced to return to selected nodes through introduction of loops.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Introduction

This chapter presents the results of the research. This addresses objective ii and iii. First the optimal number of artificial ants needed for generate a quick optimal solution in EACO is studied. Then finally, results from simulation experiments performed and analysis of tolerance of computer networks to packet looping in presence of bio-inspired systems are discussed. In the experiments, the Cisco Packet Tracer simulator to was used to test functionality of various configurations of the network as shown by figure 4.1 to figure 4.17 in this chapter.

First of all, the packet tracer is launched and a simple network configuration of four computer devices (two desktop computers and two laptops) are created and configured in the network. A router is used to bridge between two different network classes each having two computers. These classes include B having a default gateway of 172.16.0.1 and class C with a default gateway of 192.168.0.1. These computers are interconnected on the network using two 24-port swiches. The so-developed network of computers is then subjected to different conditions and tested as shown by various screenshots in this chapter.

4.2 Results for objective 1

4.2.1 Challenges facing the current computer networks

After a careful analysis of the challenges facing computer networks as found from secondary published data, it was determined that the current computer networks are constantly facing a number of bottlenecks afftecting their uptime, relibility and functionality. This challenges include

security issues, packet loops, traffic overload, and network resilience. The computer networks in Kenyan institutions of higher learning are also facing the same challenges stated above.

4.3 Results for objective 2

4.3.1 Optimum number of ants

After identifying the main challenges faced by ACO which include stagnation of ants, actual number of routing messages that are needed, and convergence time and their main causes which are associated with parameter setting in the algorithm, this research singled out one of the parameters which is the convergence time influenced by the number of ants in the solution space. It is difficult to quantify the number of ants necessary to solve an issue. First, some problems are less complex than others, meaning they need a smaller number of ants to be solved, and secondly, we have different types of ACO algorithms working differently. A well optimized algorithm will have a short convergence time to get an optimized solution. For EACO developed in this research, the optimal number of ants needed for the problem under study was 100 ants. The research therefore considered the results from the experiments done by various researchers as shown in the figures 2.4-2.15 above and came up with the following conclusions that can help determine the optimum number of ants needed. 1) The type of ACO algorithm in use, 2) The complexity of the problem under study, and 3) The ratio of the number of specialized ants to that of normal ants. Further

research needs to be carried out to determine how best we can calculate and determine the optimal number of ants needed.

4.4 Experimental Results for objective 3 (Fault-tolerant networks)

After determining the challenges faced by computer networks today as stated by results in objective 1, the research singled out one of the challenges which is network loops. After singling out this challenge, study specified it further and studied how to avoid packet loops thereby solving the problem of broadcast storms associated with presence of these loops as discussed in the sections below. Broadcast storms flood the network making all or most of the computer devices connected to the network to lose communication due to link failure. While testing the connectivity one needs to send an arbitrary ICMP echo and wait for the reply. In this experiment, the study used both the control group and the experimental groups to determine the effect of the loops on network communication while providing a possible solution to the problem of loops.

4.4.1 Enhanced ACO (EACO) Control experiment

In this experiment no conditions in the ideal network are varied. The computers are simply connected and assigned IP adresses on the network as shown in figure 4.1 below.

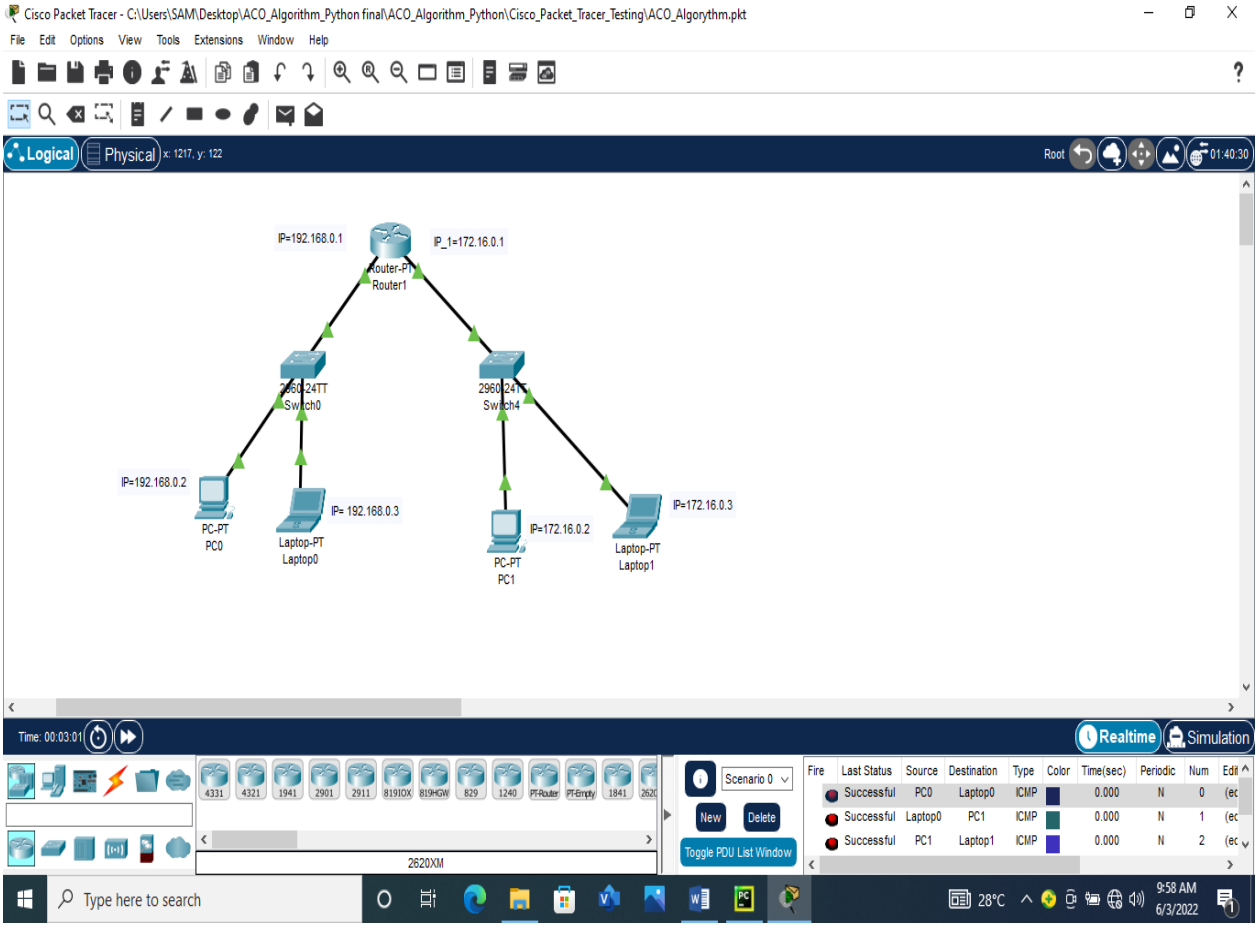


Figure 4. 1: Control Experiment

The study then simulated the control experiment to test if the packets that are forwarded by computers through the switches reach the intended destination. The ICMP messages sent are received successfully as shown in figure 4.2. The green tick signifies receipt of the message. In the

same figure, at the bottom right, the message status also confirms it was received successfully.

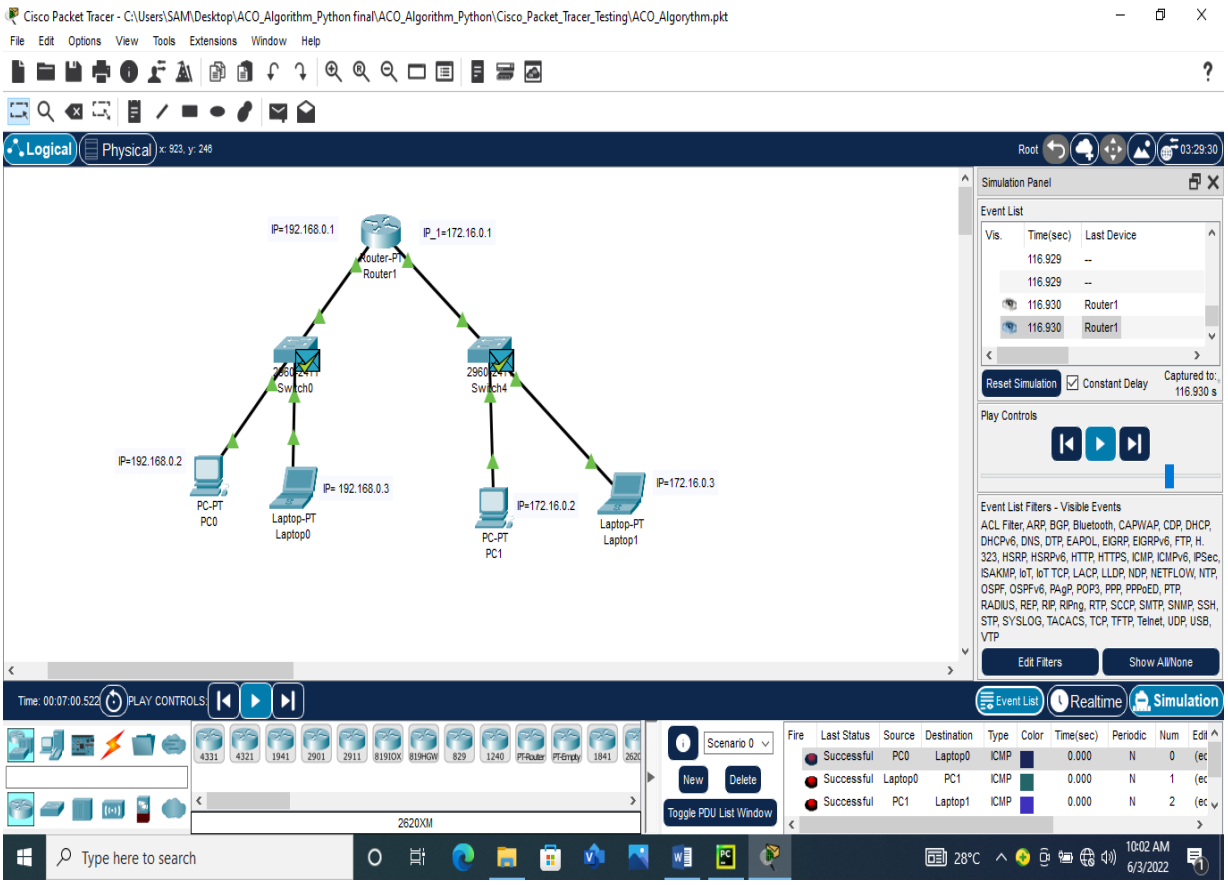


Figure 4. 2: Control results 1

Pinging the devices in the control experiment for instance from laptop 1 having IP address 172.16.0.3 to laptop 0 and PC 0 having IP addresses 192.168.0.3 and 192.168.0.2 respectively, still shows communication taking place as shown by the command prompt screen in figure 4.3 below.

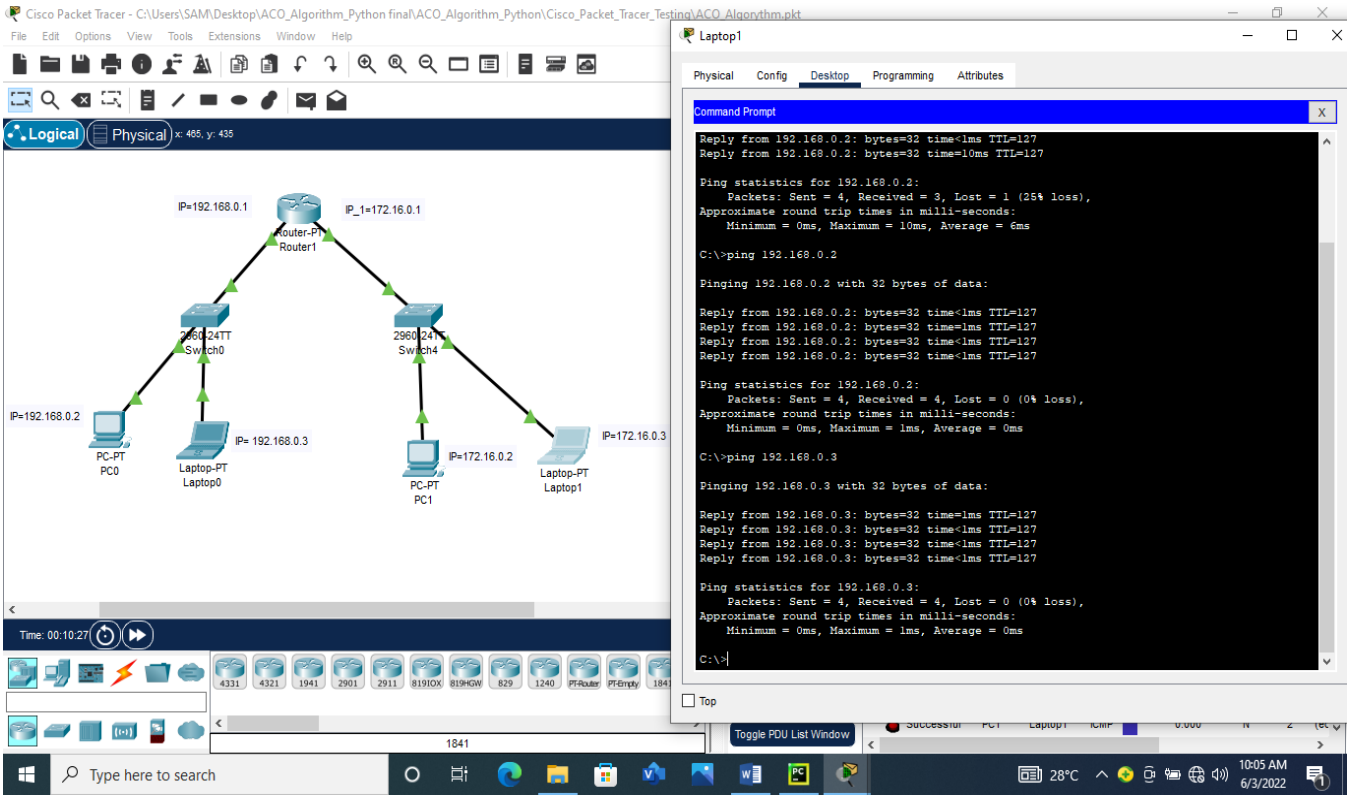


Figure 4. 3: Control results 2

Ping from laptop 0 with IP address 192.168.0.3 to laptop 1 with IP address 172.16.0.3 and PC 1 with IP address172.16.0.2 shows the same results. See figure 4.4. Communication is successful because the CMD screen shows percentage packet loss as 0 %. That means that all the four ICMP arbitrary packets send were all well received and echeod a reply.

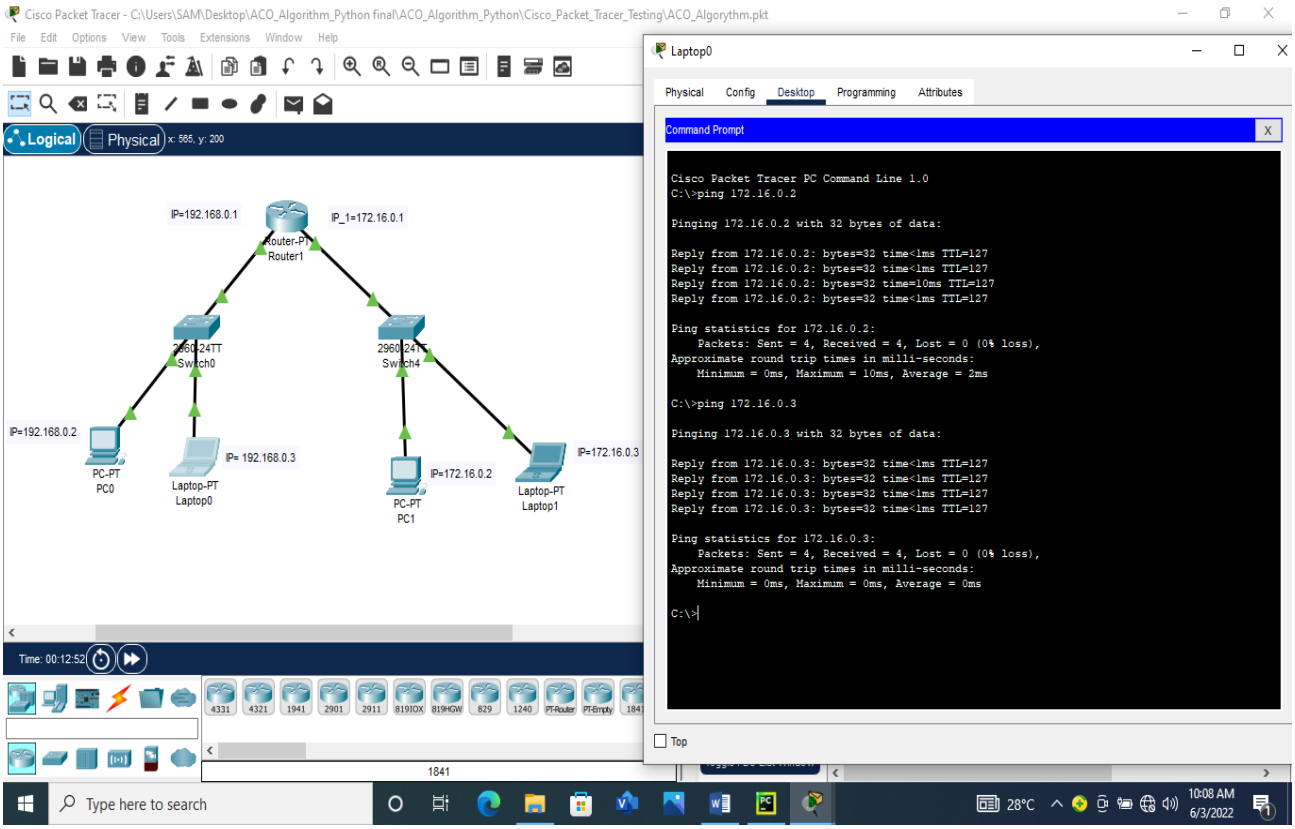


Figure 4. 4: Control results 3

4.4.2 Simulator with loops but without the algorithm

The simulated network is configured as shown below. There is a modification from the control experiment by the introduction of physical loops on the network. When we sent 4 ICMP packets from laptop 0 to PC 1 , all the packets fail to be delivered and is displayed in the real time mode of the packet tracer. Figure 4.5 shows the configuration.

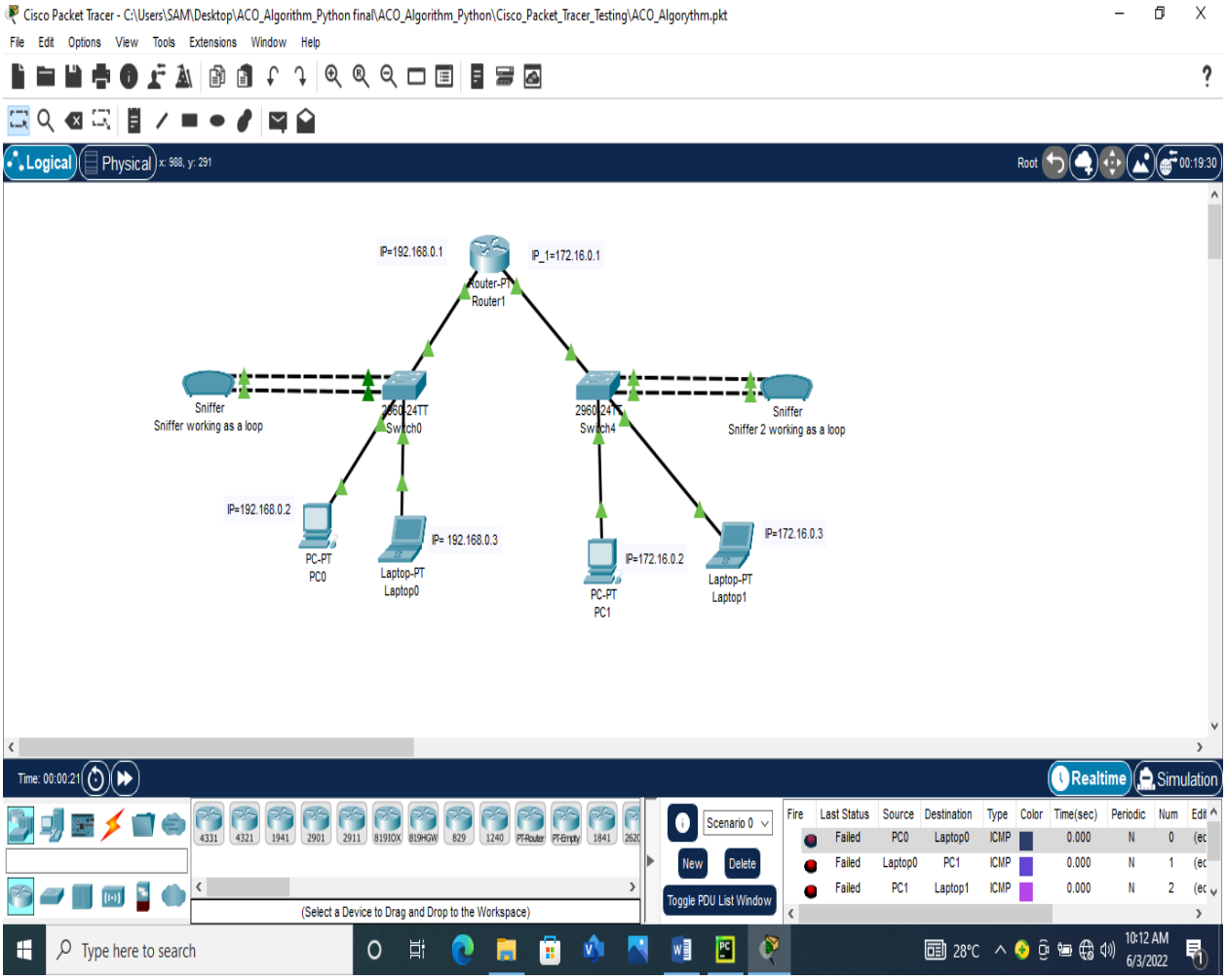


Figure 4. 5: Loops no ACO

After simulating packets in this network, the message remained in progress forever as shown in the bottom right corner of figure 4.6 below. Secondly, the messages in the network simulator kept on rotating between the loops created and the two switches 0 and 4.

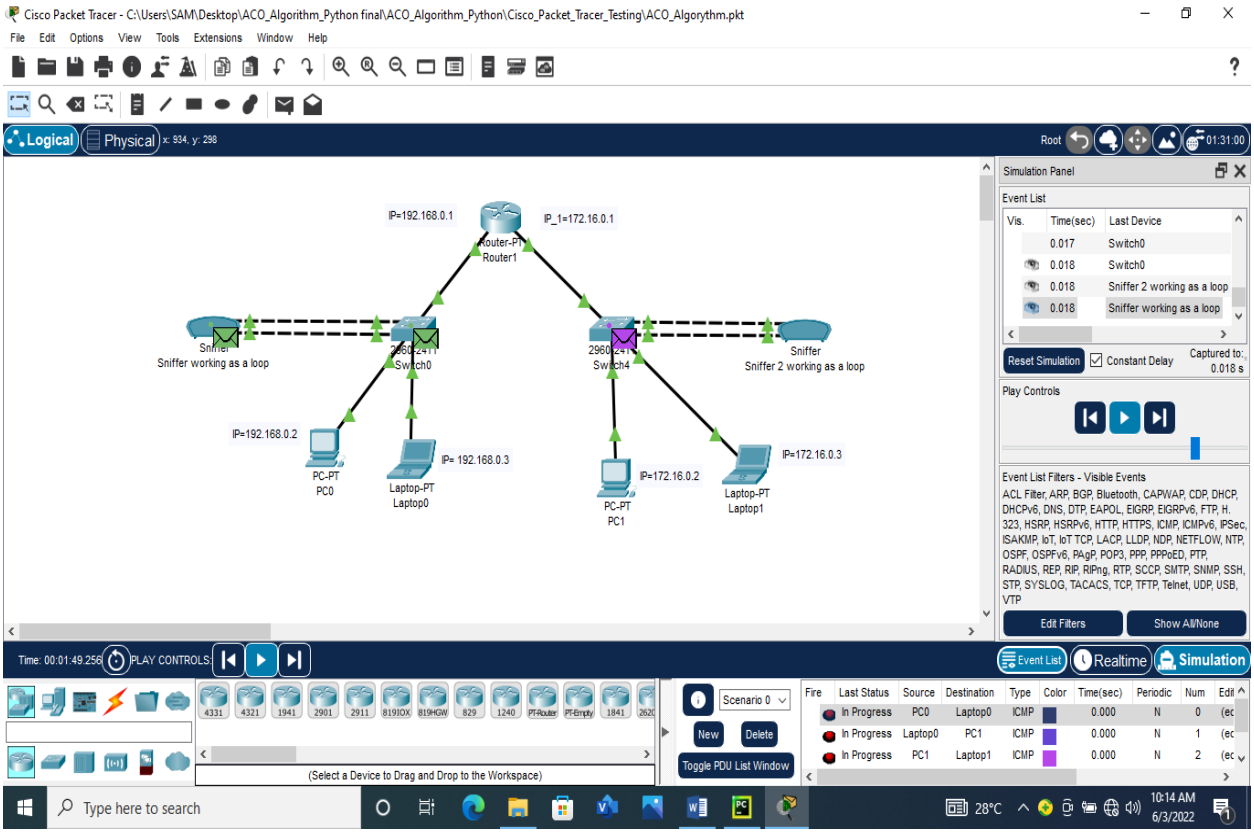


Figure 4. 6: Loops no ACO result 1

When we open the command prompt and ping from laptop 1 having IP address 172.16.0.3 to laptop 0 and PC 0 having IP addresses 192.168.0.3 and 192.168.0.2 respectively, the message fails to be delivered and shows time out on the command prompt and failed on packet tracer as shown in figure 4.7 below.

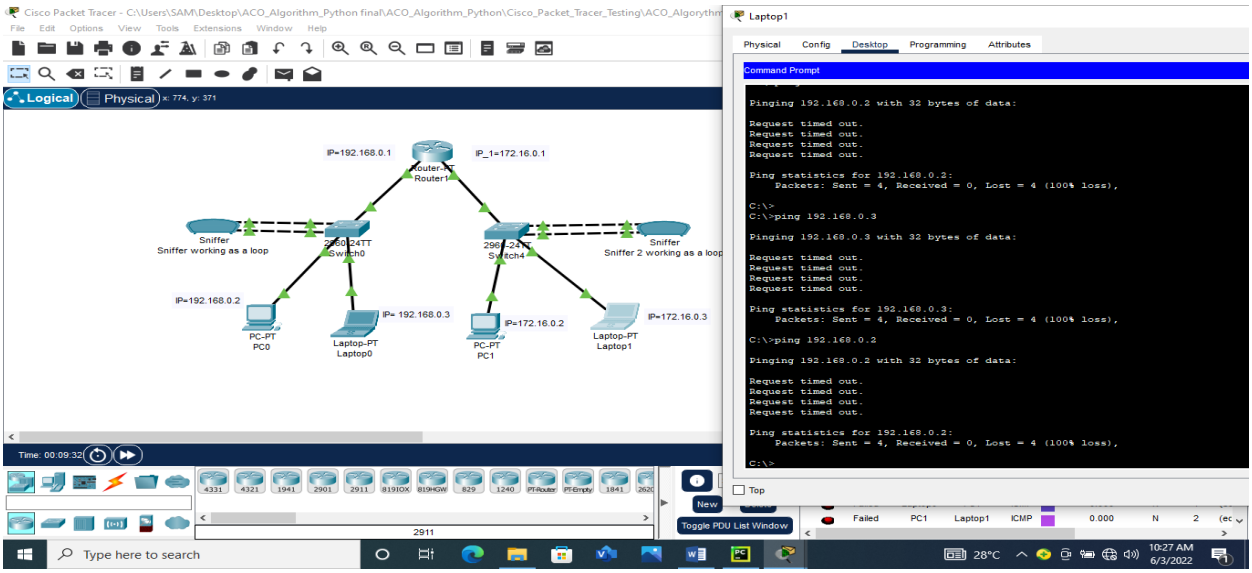


Figure 4. 7: Loops no ACO result 2

Consequently, pinging from laptop 0 with IP address 192.168.0.3 to laptop 1 with IP address 172.16.0.3 and PC 1 with IP address172.16.0.2 shows the same results. See figure 4.8.

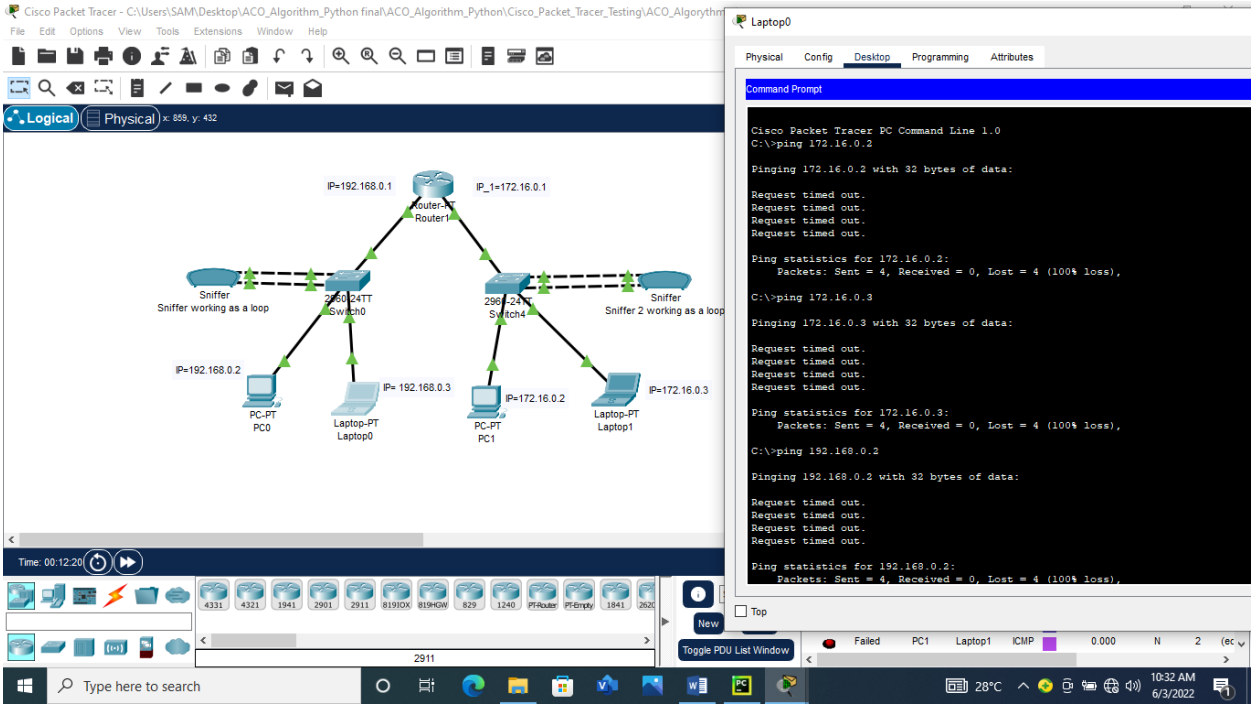


Figure 4. 8: Loops no ACO result 3

4.4.3 Simulator with loops and with the algorithm

The researcher was able to run the algorithm and then simulate the network having the loops. Secondly, the main algorithm is imported in the programming mode under the TCP Python package of each computer device running on the network. The algorithm displays the activity of random movement of ants on the pygame simulator.

The simulation mode of the Cisco Packet Tracer shows message delivery in the network with green ticks as shown in figures 4.9 and 4.10 below. At the bottom right corner of the simulator, there is an indication of the successful delivery of messages.

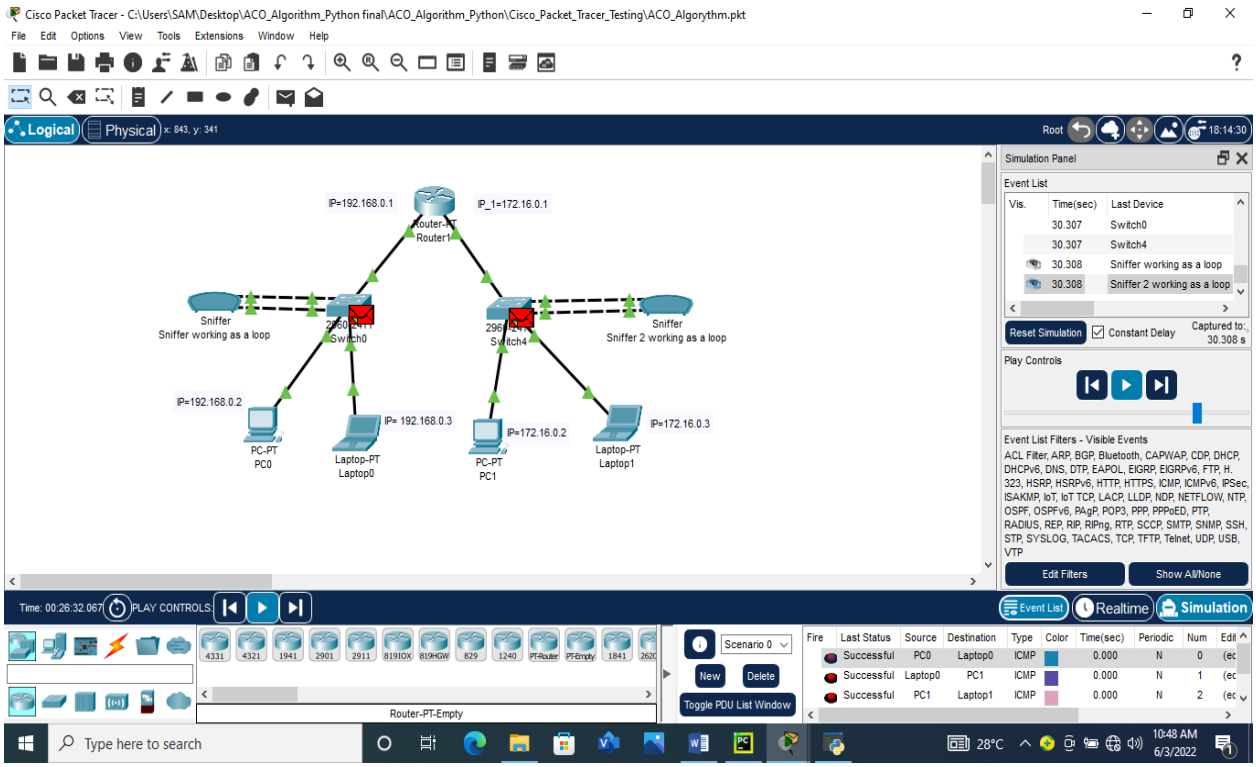


Figure 4. 9: Loops with ACO result 1

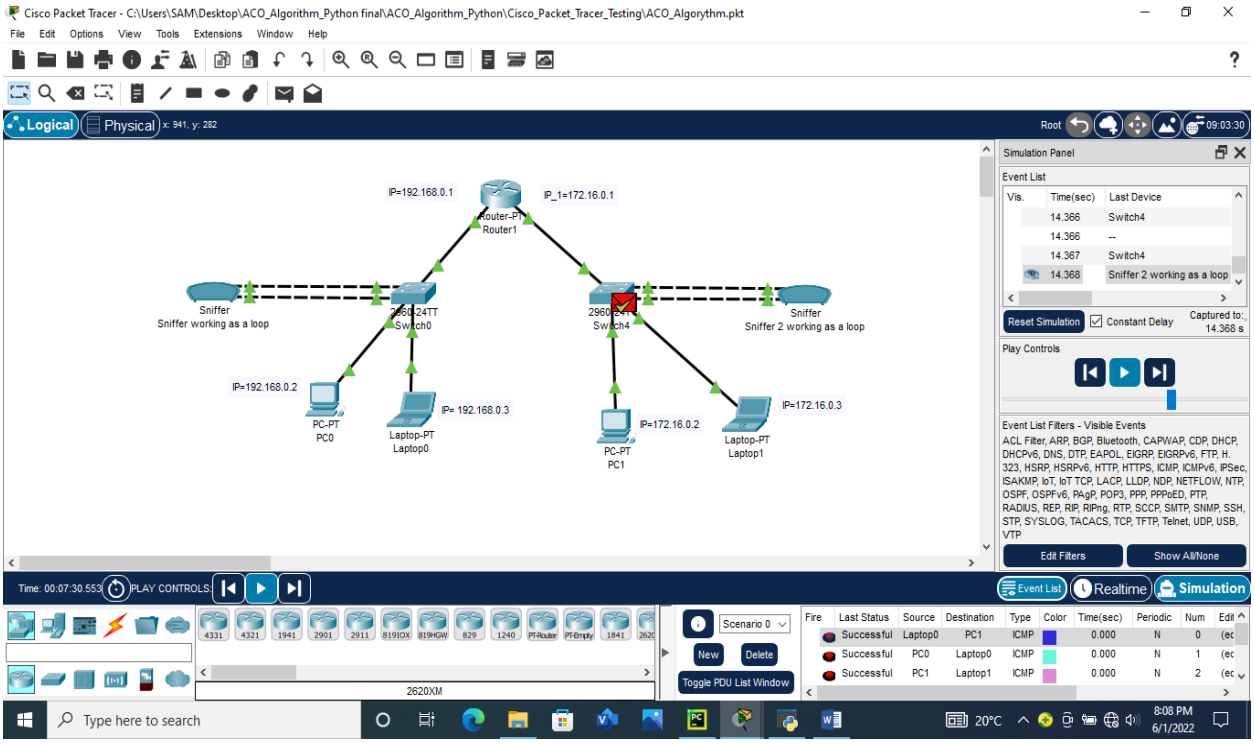


Figure 4. 10: Loops with ACO result 2

Pinging the devices for instance from laptop 1 having IP address 172.16.0.3 to laptop 0 and PC 0 having IP addresses 192.168.0.3 and 192.168.0.2 respectively, indicates communication taking place as shown by the command prompt screen in figure 4.11 below.

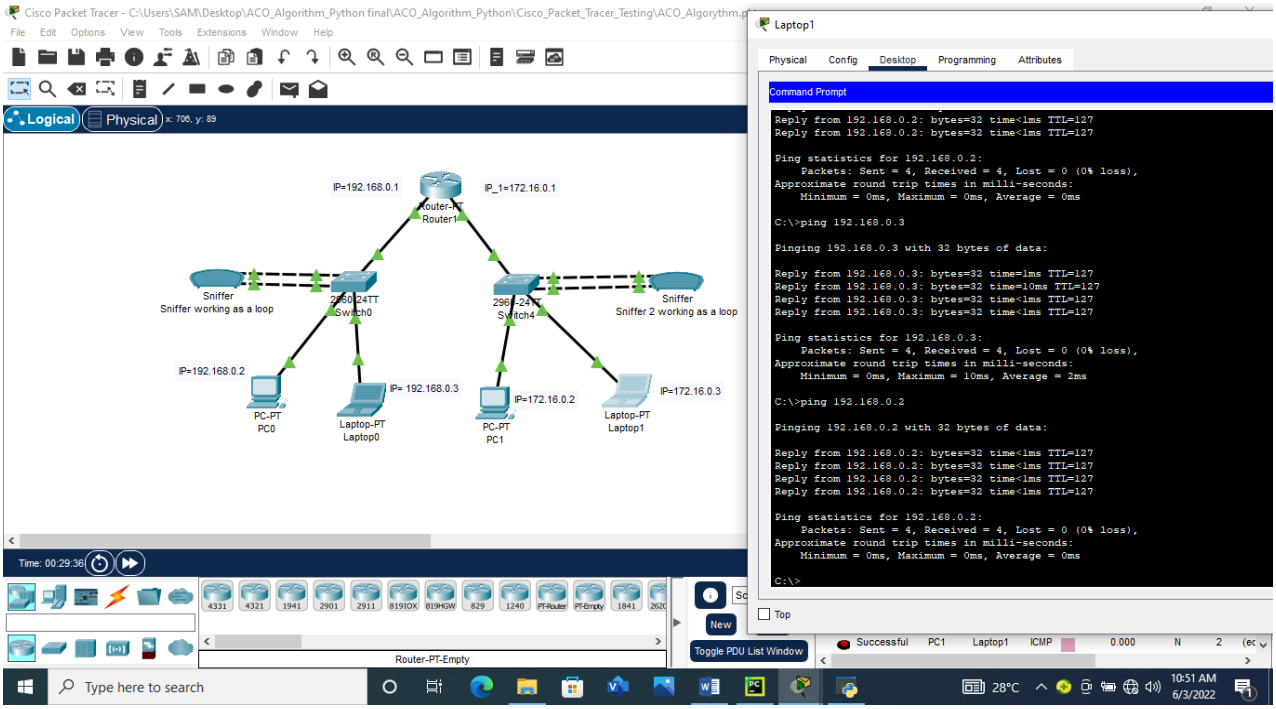


Figure 4. 11: Loops with ACO result 3

Ping from laptop 0 with IP address 192.168.0.3 to laptop 1 with IP address 172.16.0.3 and PC 1 with IP address172.16.0.2 shows the same successful replies too. See figures 4.11 and 4.12

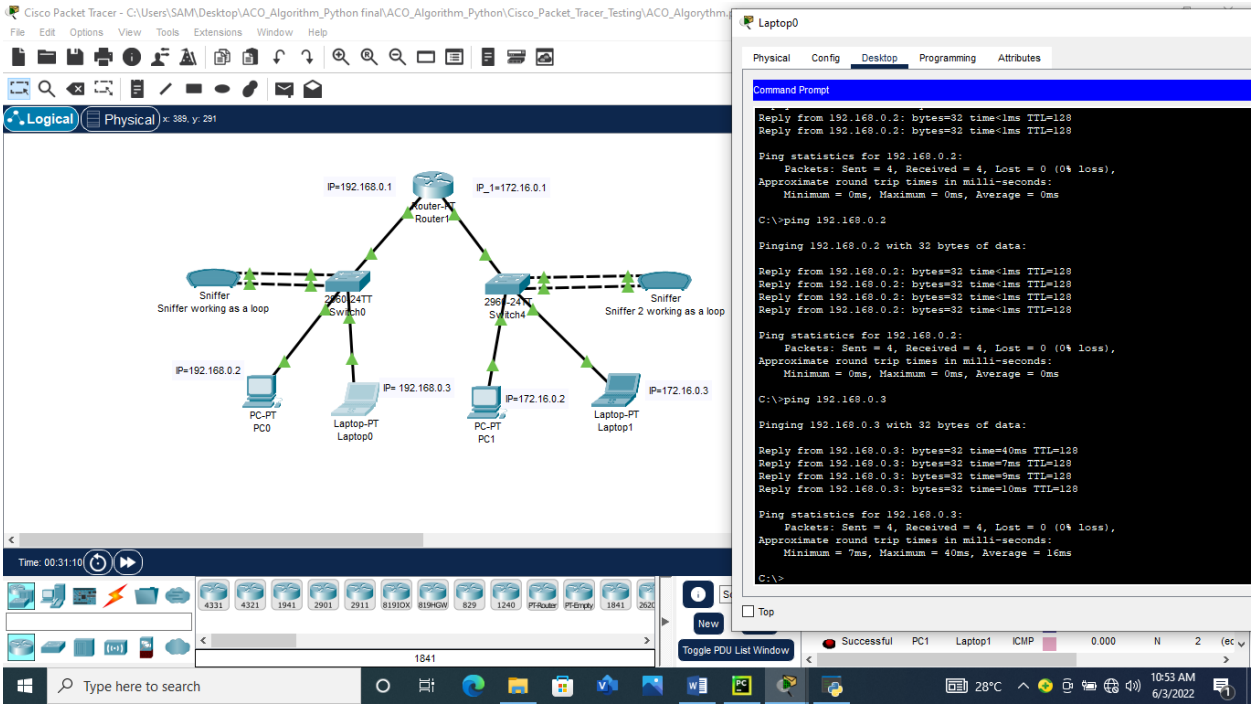


Figure 4. 12: Loops with ACO result 4

4.5 Interpretation of results

In the above simulation, the researcher used three experiments to test and validate the data collected. The first which is a control experiment provides an ideal network with no variations in the network parameters working under normal conditions. It was discovered that the network messages were delivered properly between the devices on the network without any problem showing 100% success for the 4 messages sent over the network. In the second experiment, the researcher subjected the simulated network to forced loops as it normally happens in colleges and universities (mostly done by students). In this experiment, it was discovered that the network went completely down and could not recover from the looping packets which completely failed to be delivered to the intended destination. Pinging the network showed that the 4 messages send had a 100% failure rate. In the third experiment, the ACO Algorithm was simulated alongside the network with the loops still on. This experiment showed packets being delivered normally in the simulation mode by showing success to all ICMP packets sent. The ping tool also showed a 100% success rate for all the 4 test packets sent by various devices on the network. From the above experiments, five average Round Trip Times (RTT) were collected from 5 control experiments and 5 from enhanced experiment. The control experiment had the following RTT as shown in figure 4.3 and figure 4.4 (6,0,0,2,0) averaging to 8/5 =1.6ms. The experiment of enhanced algorithm showed the following results for RTT as witnessed from figure 4.12 and figure 4.11 (0,0,2,0,16) aveараging to 18/5=3.6ms. Although the algorithm's results show that the packets get to destination on average 2 ms later than in the case of the control, it is still witin the allowed RTT. The 2ms delay may have been brought about by the cycling packets before getting out of the loop. For optimal network performance , studies have shown that RTT<500ms [80]. This shows that the algorithm intervened to reroute the packets that would have otherwise gotten locked up in the loop

as shown in experiment of figure 4.6 where the packets kept rotating in the loop thereby breaking communication between the computers on the network. The algorithm helped the network in rerouting any packets that would be caught up in the loop to reach their destination.

4.5.1 How the enhanced ACO algorithm works

The ACO model was first developed and applied in computer networks to help the network packets reach their destination very fast using the shortest route possible by mimicking the real ants through their foraging behavior and intelligence. However, the model could not solve the problem of stagnation of ants thus the new model developed in this research helps to get the stagnated ants out of the loop. The algorithm is then imported into the Cisco Packet Tracer and the packet tracer translates the ants into real packets of the network. When the packets are got up in a loop, they use their random intelligent movement to get out of the loop once they realize they are talking too long to reach their destination. The movement of the packets (similar to the ants after importing into packet tracer) is as shown in figure 4.13 below.

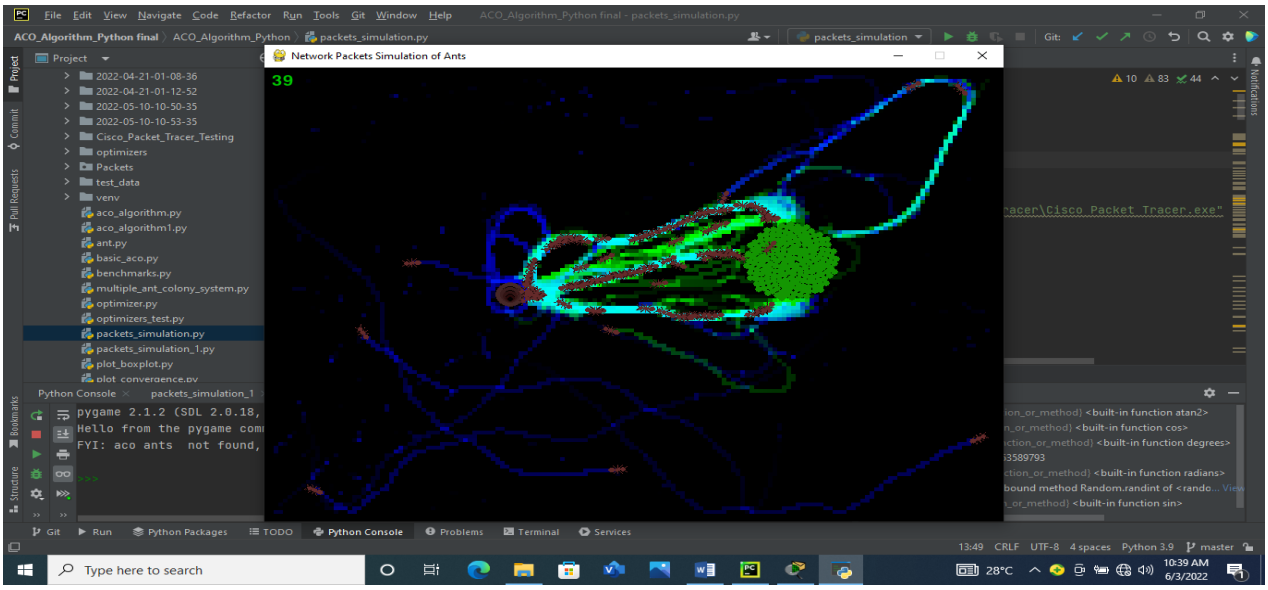


Figure 4. 13: Ants in model foraging
The figure above shows packets finding the routes from the source of food to the nest (dest).

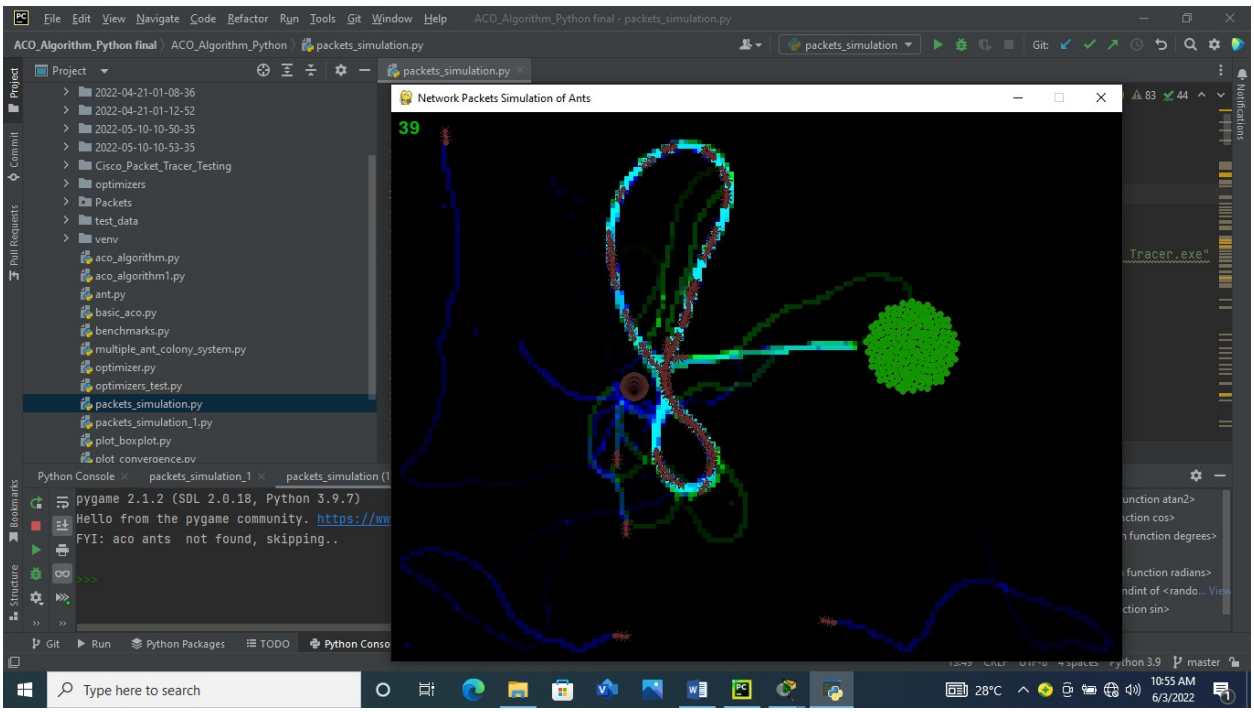


Figure 4. 14: Ants in model caught up in a loop
The figure above shows packets (ants) caught up in a loop for sometime

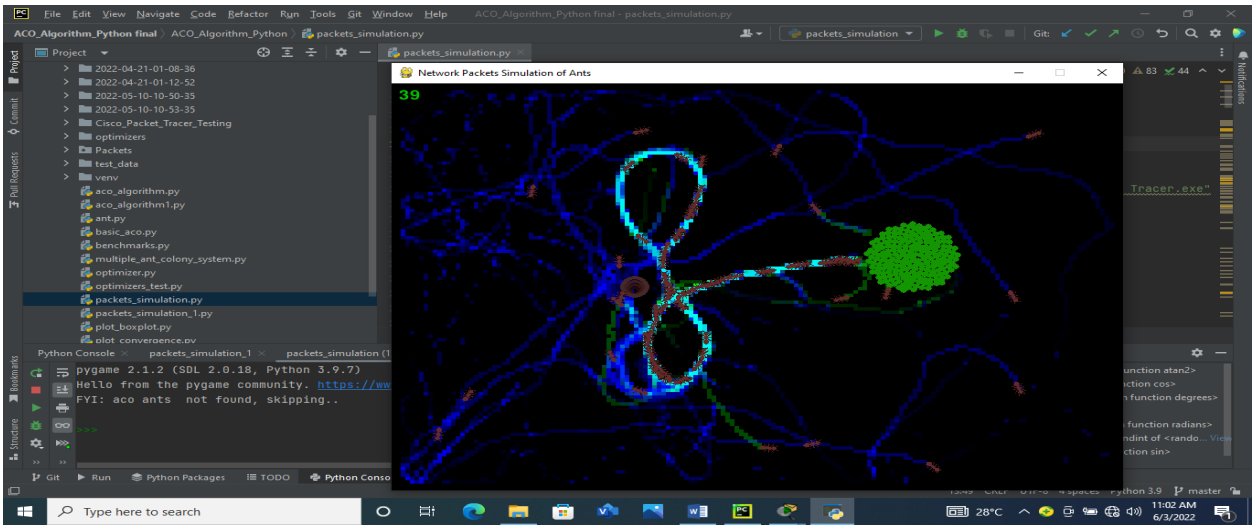


Figure 4. 15: Ants getting out of the loop

This figure shows (packets) ants getting out of the loop after rotating in the loop for a few moments. The loops will be unnoticeable in the real network since it happens at a very high speed.

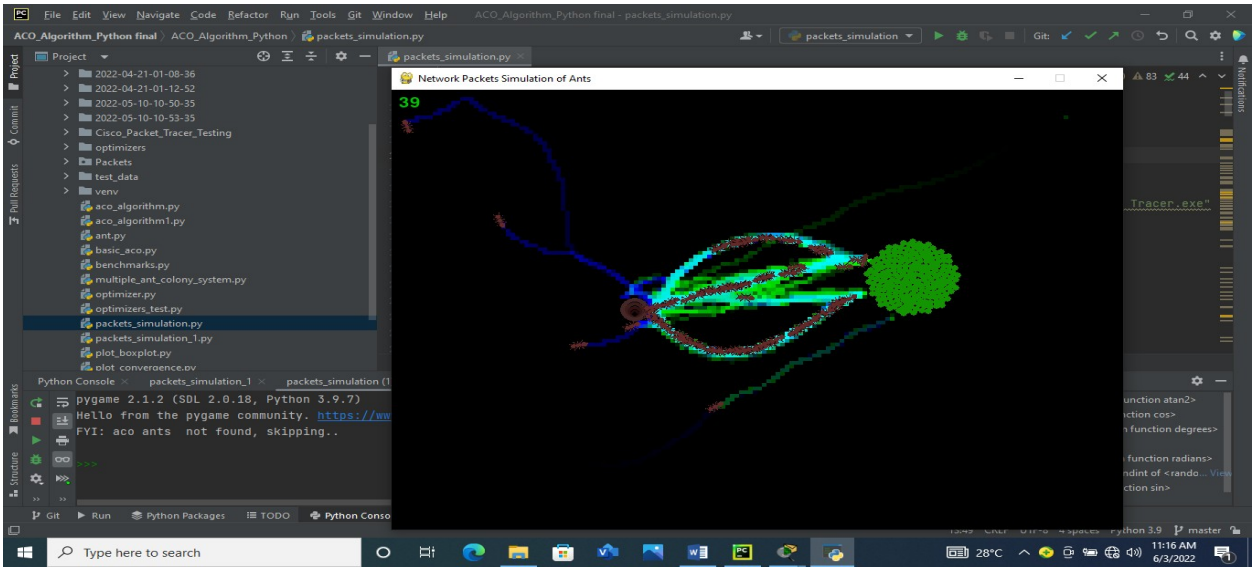


Figure 4. 16: Ants out of the loop

Ants (packets) are now completely out of the loop and looking for the shortest distance between their nest(source) and destination (computer).

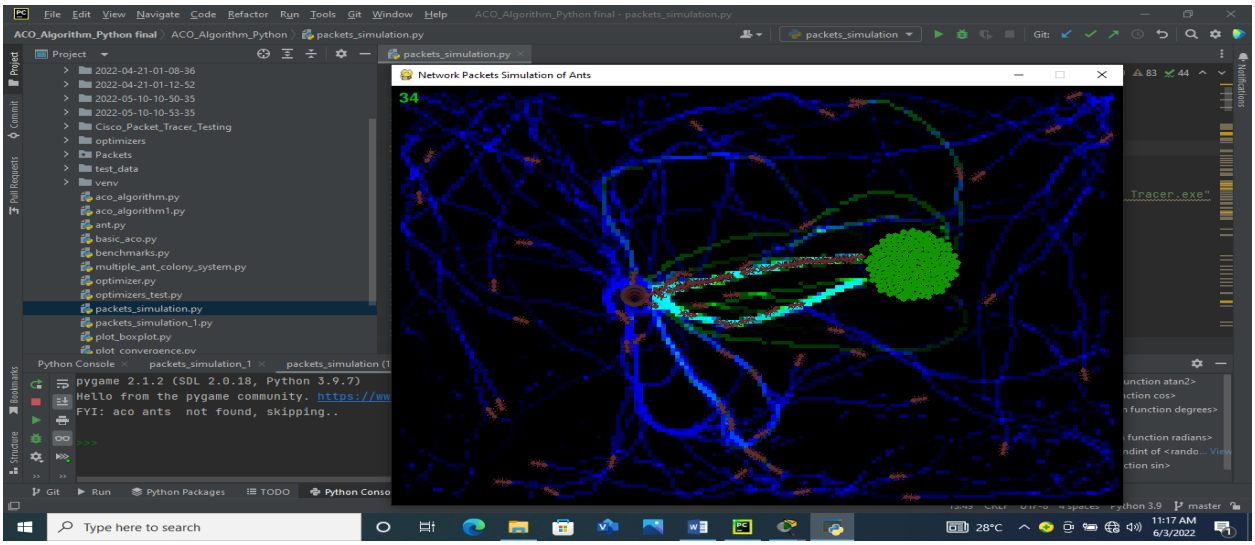


Figure 4. 17: Ants selecting the shortest route

Finally, ants (packets) select the shortest route to their destination.

4.5.2 EACO

According to research done by Kwang Hong Sim and Weng Hong Sun [12], when implementing ACO the following issues arise: stagnation and poor adaptiveness of ants. Stagnation will occur if a network reaches its convergence too early (prematurely). This leads to the following problems: 1) congestion of packets (ants) on the optimal path, 2) reduction of probability of ants choosing other routes. The research further indicated that congestion and low probability of choosing other paths by ants will consequently lead to the following problems in dynamic computer networks: a) The selected path may become non-optimal if congested b) The congested path may be disconnected due to network failure. Due to these issues, Sim and Sun noted that the network would eventually have a low degree of sensitivity due to changes in topology or link failure. In their research, Sim and Sun [12] proposed a new improved model called MACO which they noted that it enhanced the adaptiveness of the network and reduced the chances of stagnation of ants. In MACO, the ants are

made to deposit pheromones of different colors and observed. It was noted that ants follow routes with similar colors reducing congestion, nevertheless, they concluded that their research does not guarantee that MACO would fully eradicate stagnation of ants as it only reduced the chances of stagnation. Secondly, another research was done by Gianni Di Caro on the application of ACO to adaptive routing in telecommunications networks [8]. The research showed that loops that lead to stagnation of ants should be avoided as much as possible because they cause very high packet latencies. To implement loop avoidance, Di Caro explained using an equation for avoiding loops in networks. For instance, if an ant is forced to return to a node that it had previously visited, the nodes that composed the loop are taken out of the memory of the ant, and all information describing those nodes is destroyed. Consequently, if an ant moves on a cycle for a time that is greater than its half-life, the ant is destroyed. This destruction solves the problem of loops, however, another problem arises whereby the affected nodes and ants(packets) will be destroyed which means they will not communicate over the network as intended. Considering the finding of the above two kinds of research, the researcher carried out this study to help strike a balance between stagnation and the destruction of ants. From the results above, the problem of stagnation of ants is more highly reduced than that of MACO suggested by Sim and Sun [12] by making sure that ants that are got up in a loop simply get out of the loop using a new optimal route as shown in **figure 4.15 (pygame)**. This will help in making sure that all the packets in the network reach their intended destination as opposed to destroying stagnated ants as found out in the research by Di Caro [8]. **Figure 4.12 (packet tracer)** proves that the packets that are got up in the loop can still be rerouted to their destination.

4.6 EACO Enhancements

ACO models can learn from other algorithms to optimize their performance. For the purposes of this study, EACO model learned some lessons from FireFly Algorithm (FFA) and Particle Swarm Optimization (PSO) algorithm.

4.6.1 Lessons learned from FFA

a) Attraction mechanism

Fireflies in FFA are always attracted to brightest flies during mating. The brightest flies are usually nearer than less bright flies which are far from the best solution. Nearer fireflies are proportional to the best solution found. EACO incorporated a similar attraction whereby the ants were more strongly attracted to solutions having the highest pheromone concentration through dynamic adjustment of pheromone update rules.

b) Randomization and diversity

The movement equation of FFA model is composed of a randomization term which helps the algorithm maintain diversity and prevent premature convergence. EACO adopted this feature by introducing more controlled randomization in the ant movement and pheromone updating process to avoid ant stagnation.

4.6.1 Lessons learned from PSO

PSO algorithm relies on chaos theory as it is made up of particles that move with high velocity towards an optimal solution. EACO learned from PSO how to handle premature convergence issues through dynamic adjustment of speed of ants and introduction of randomness in pheromone updates as well as periodically resetting pheromone levels.

4.7 Implication of the results

These results helped the study answer the research questions developed in chapter one. The first research question sought to understand the challenges facing the current computer networks. It was determined that regular computer networks are not very reliable since network issues like forced loops can bring them down due to broadcast storms generated by the loops. This was practically done through the introduction of loops on the two switches in the experiments discused, which is a true picture of what happens in Kenyan instituions of higher learning . The second research question sought to evaluate the optimum number of artificial ants needed in ACO model to generate an optimal solution within the shortest time possible. It was found that the optimal number of ants needed for quick convergence time largely depended on a number of factors like problem complexity etc. The third research question sought to find out how we can enhance and use the ACO model to make computer networks tolerant to such problems as device failure caused by forced loops. From the research, the results from the enhanced ACO model confirm that indeed in the case of forced network loops, we can reroute the packets by getting them out of the loop to reach their intended destination. This will help in solving the problems of forced loops introduced in the networks especially as witnessed in institutions of higher learning like Masinde Muliro University.

4.8 Challenges of the study

The study incurred several challenges among them being the time frame of completion of the research. The research took a little more time than the expected finish time. The project also ended up taking a little more money than the budgeted amount to complete.

CHAPTER FIVE

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

5.1 SUMMARY

The research thesis is made up of the following sections; Introduction, literature review, methodology, results, and discussion. In the introduction chapter, the researcher presented the statement of the problem, research objectives, and research questions that guided the research. The research generally studied faults and challenges facing computer networks and presented research questions which aimed at finding a solution to the problems. In the literature review, the researcher discussed the various bio-inspired algorithms and how they have been applied in various fields including networking to help enhance the self-organization nature of computer networks. ACO algorithms have been successfully applied in computer networks to help find the shortest route possible from source to destination using their enhanced randomized movements. Some challenges of ACO were also identified in this chapter. This study aimed at looking at how we could enhance the algorithm to make the computer networks fault-tolerant especially to forced loops. In methodology, the researcher used agile as a software development life cycle model. This was important since prototypes were produced and tested in iterations until a final algorithm was produced. The results featured three experiments done using Cisco Packet Tracer network simulator using a control experiment and two other experiments under varied network conditions. The control experiment performed under ideal conditions showed the network working perfectly well. Loops were then introduced on two switches in the simulated network and its performance was monitored using packet delivery mechanisms both graphical and use of command prompt (ping). In the testing of the second experiment after the introduction of loops and without using the algorithm, there was a total failure of packet delivery due to broadcast storms. The new EACO

model developed was then subjected to the second experiment and tested to ascertain if it would influence the results from the simulator. The model was found to influence the results positively by rerouting the looped packets, thereby stopping broadcast storms in the network hence helping in achieving the first and last objective of the research. The results show that with the enhanced ACO model, we can make our computer networks tolerant to broadcast storms that are most commonly caused by loops in the network.

5.2 CONCLUSIONS AND RECOMMENDATIONS

The graph theory in figure 2.16 guided this research in helping determine the shortest route from one node to the other while keeping note of all vertices within the search space. The Ramsey graph theory was then modified for the purposes of this research to shown how loops may form within physical computer networks. A simple network simulation was created and used to simulate the modified Ramsey graph theory given, G = (V, E) as shown in figures 4.1-4.12. The results of this simulation show that this research can add some contribution to the body of knowledge. When well adopted in schools, colleges, homes, and offices, the algorithm developed under this research study can go a long way in improving the reliability of the computer networks that are constantly under attack by forced packet loops. However, the algorithm does not completely guarantee that it will solve all the problems faced by computer networks. Further research needs to be done by other interested researchers to improve EACO for optimal results.

6.0 References

[1] T. Bakardjieva, "Introduction to computers," Varna Free University “Chernorizec Hrabar” Institute of Technology,2012

[2] M. MEYERS, Introducing Basic Network Concepts.

[3] R. Hylsberg Jacobsen, Q. Zhang, and T. Skjødeberg Toftegaard, “Bioinspired principles for large-scale networked sensor systems: An overview,” *Sensors*, vol. 11, no. 4, pp. 4137–4151, Apr. 2011. doi:10.3390/s110404137

[4] S. Kumar and P. J., “Bio-inspired computing and Robots,” *Bio-Inspired Computing and Networking*, pp. 41–41, Mar. 2011. doi:10.1201/b10781-5 .

[5] A. Jamalipour, Bio-Inspired Networking, The University of Sydney: IEEE, 2009.

[6] F. Dressler, "Self-Organization in autonomous Sensor/Actuator Networks [SelfOrg]," 2010.

[7] A. K. Kumar, “Bio inspired computing – a review of algorithms and scope of applications,” *Expert Systems with Applications*, vol. 59, pp. 20–32, Oct. 2016. doi:10.1016/j.eswa.2016.04.018

[8] G. D. Caro, Ant Colony Optimization and its Application to Adaptive Routing in Telecommunication Networks, 2004.

[9] J. S. B. Gurpreet Singh, "A Review of Ant Colony System Algorithm and its Models," *International Journal of Advanced Research in Computer Science,* Vols. Volume 8,, 2017.

[10] J. O. Shahab Kamali, "A Position Based Ant Colony Routing Algorithm for Mobile Ad-hoc Networks," *JOURNAL OF NETWORKS,*, vol. VOL 3, no. NO.4, APRIL 2008.

[11] D. S. D. H. D. S. Vijayalaxmi, "Ant Colony Optimization Technique in Network Routing Problem-A Simulation Study," in *International Conference on Innovations in Engineering and Technology (ICIET'2013) Dec. 25-26, 2013* , Bangkok (Thailand), 2013.

[12] Kwang Mong Sim and Weng Hong Sun, “Multiple ant-colony optimization for network routing,” *First International Symposium on Cyber Worlds, 2002. Proceedings.* doi:10.1109/cw.2002.1180890

[13] A. H. Hamamoto, L. F. Carvalho, and M. L. Proenca, “Aco and GA metaheuristics for anomaly detection,” *2015 34th International Conference of the Chilean Computer Science Society (SCCC)*, Nov. 2015. doi:10.1109/sccc.2015.7416569

[14] O. C. .. O. C. IUfoaroh S.U, "Tracking The Effects of Loops in A Switched Network Using Rapid Spanning Tree Network," *International Journal of Research in Electronics and Communication Technology (IJRECT 2015)*, vol. 2, no. 3, 2015.

[15] L. Maniezzo,M. Gambardella. F.D. Luigi, V. Maniezzo, Ant Colony Optimization, 2001. 10.1007/978-3-642-18965-4_21

[16] F. Dressler and O. B. Akan, “A survey on bio-inspired networking,” *Computer Networks*, vol. 54, no. 6, pp. 881–900, Apr. 2010. doi:10.1016/j.comnet.2009.10.024

[17] Noel Inc, "The 5 Issues that Are Affecting Large Computer Networks Today," 2017.

[18] S. Convery, "Hacking Layer 2: Fun with Ethernet Switches," 2002.

[19] D. Hein, " *NEW** 2020 Network Monitoring Buyer’s Guide – GET IT HERE!," 20 March 2019. [Online]. Available: https://solutionsreview.com/network-monitoring/4-causes-of-network-congestion-and-how-to-prevent-them/. [Accessed 9 April 2020].

[20] H. Balakrishnan, "Network Routing - II Failures, Recovery, and Change," 2009.

[21] P. Francois and O. Bonaventure, “Avoiding transient loops during the convergence of Link-State Routing Protocols,” *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1280–1292, Dec. 2007. doi:10.1109/tnet.2007.902686

[22] M. H. AFSHAR, "A new transition rule for ant colony optimization algorithms: application to pipe network optimization problems," *Engineering Optimization*, vol. VOL.37, no. 5, pp. 525-540, 2005.

[23] T. S. Marco Dorigo, Ant Colony Optimization, London: Bradford Book, 2004.

[24] A. Chakraborty and A. K. Kar, “Swarm intelligence: A review of algorithms,” *Nature-Inspired Computing and Optimization*, pp. 475–494, 2017. doi:10.1007/978-3-319-50920-4_19

[25] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," *Technical report-tr06, Erciyes university, Computer engineering department.*, vol. 200, 2005.

[26] A. Biswas, S. Dasgupta, S. Das, and A. Abraham, “Synergy of PSO and bacterial foraging optimization — a comparative study on numerical benchmarks,” *Advances in Soft Computing*, pp. 255–263, 2007. doi:10.1007/978-3-540-74972-1_34

[27] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *Control Systems, IEEE,*, vol. 22, no. 3, pp. 52-67, 2002.

[28] D. Falko, “Self-Organization in Autonomous Sensor/Actuator Networks., 2010.

[29] A. K. Kar, "Bio Inspired Computing – A Review of Algorithms and Scope of Applications," *Expert Systems with Applications*, p. 54, 2016.

[30] E. Bonabeau, M. Dorigo, and G. Theraulaz, "Swarm Intelligence: From Natural to Artificial Systems.," in *New York, Oxford University Press*, 1999. doi:10.1093/oso/9780195131581.001.0001

[31] I. Kassabalidis, M. A. El-Sharkawi, R. J. Marks, P. Arabshahi, and A. A. Gray, “Swarm intelligence for routing in Communication Networks,” *GLOBECOM’01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*. 2002 doi:10.1109/glocom.2001.966355

[32] R. Eberhart, J. Kennedy, "Swarm Intelligence.," Morgano Kaufmann, 2001.

[33] V. Maniezzo. A. M. Dorigo, "Ant system: optimization by a colony of cooperating agents," *Trans. Systems, Man, Cybernet.-,* vol. 26, no. 1, pp. 29-41, 1996.

[34] G. Kochenberger, F. Glover, "Handbook of Metaheuristics,," Kluwer Academic Publishers, Norwell, 2002.

[35] F. Glover, "Tabu search—Part II,," *ORSAJ. Comput.,* vol. 2, no. 1, pp. 4-32, 1990.

[36] M. Laguna, F. Glover, "Tabu Search," Kluwer Academic Publishers, Dordrecht, 1997.

[37] C. Blumb, Marco Dorigo, "Ant colony optimization theory: A survey," *Theoretical Computer Science,* vol. 344, pp. 243-278, 2005.

[38] C. Thyssen, Dirk Sudholt, "Running time analysis of Ant Colony Optimization for shortest path problems," *Journal of Discrete Algorithms,* vol. 10, pp. 165-180, 2012.

[39] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: Optimization by a colony of cooperating agents,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, Feb. 1996. doi:10.1109/3477.484436

[40] M. Dorigo, G. D. Caro, and L. M. Gambardella, “ANT algorithms for discrete optimization,” *Artificial Life*, vol. 5, no. 2, pp. 137–172, Apr. 1999. doi:10.1162/106454699568728

[41] F. Dressler, Benefits of Bio-inspired Technologies for Networked Embedded Systems: Overview, 2006.

[42] M. Dorigo and T. Stützle, “Ant Colony Optimization: Overview and recent advances,” *International Series in Operations Research & Management Science*, pp. 227–263, 2010. doi:10.1007/978-1-4419-1665-5_8

[43] C. Oscar, H. Francisco and S. Thomas, "A Review of Ant Colony Optimization Metaheuristic: Basis, Models and New trends," 2002.

[44] V. Selvi and R.Umarani, "Comparative Analysis of Ant Colony and Particle Swarm Optimization Techniques," *International Journal of Computer Applications* , vol. 5, no. 4, p. 0975 – 8887, 2010.

[45] E. Morales, E Mariano, "Ant-Q Algorithm for Multiple Objective Optimization Problems," *Genetic and Evolutionary Computing Conference (GECCO-99),* vol. 1, pp. 894-901, 1999.

[46] M. Dorigo, V. Maniezzo, Turbian. A. Colorni, "Ant System for job-shop scheduling," *Belgian Journal of Operationa Research, Statistics and Computer Science,* vol. 34, no. 1, pp. 39-53, 1994.

[47] M. Dorigo, G.D. Caro, " AntNet: Distributed stigmergic Control for Communication Networks," *Journal of Artificial Intelligence Research,* vol. 9, pp. 317-365, 1998.

[48] M. Dorigo, L. Garmbardella, "Ant System Hybridized with New Local Search for the

Sequential Ordering problem," *INFORMS Journal on Computing*, vol. 12, no. 3, pp. 237-255, 2000, 10.1287/ijoc.12.3.237.12636

[49] T. Stuzzle, "An Ant approach to flow shop problem," in *Proceedings of 6th European Congress on Intelligent Techniques & Soft Computing*, Aachen Germany, 1998.

[50] D. Costa and A. Hertz, "Ants can colour graphs," *Journal of the Operational Research Society*, vol. 48, no. 3, pp. 295–305, 1997. doi:10.1038/sj.jors.2600357

[51] R. Hartl, C. Strauss, B Bullnheimer, "An improved Ant System Algorithm for vehicle routing problem," *Annals of Operatind Research*, vol. 89, pp. 319-328, 1999.

[52] C. Coello, B. Mendoza, "An ant approach on the use of Ant system to design combinatorial logic circuits," *Mathware & Soft Computing*, 2002.

[53] A. Sharma, Mandeep Kaur, "A review on Ant Colony Optimization in MANET," *International Journal for Science and Emerging*, vol. 19, no. 1, pp. 1-6, 2014.

[54] O. Holland, J. Bruten, L. Rothkrantz, R. Schoonderwoerd, "Ant-based load balancing in telecommunications networks.," *Adaptive Behaviour*, vol. 5, no. 2, pp. 169-207, 1996.

[55] M. Dorigo, G. D. Caro, "Ant colonies for adaptive routing in packet-switched communications," in *Fifth International Conference on Parallel Problem Solving from Nature*, 1998.

[56] M. Dorigo, G.D. Caro, "Mobile agents for adaptive routing.," in *Proceedings of the 31st International Conference on System Sciences*, 1998.

[57] G. D. Caro, L. M. Gambardella, F. Ducatelle, "Using ant agents to combine reactive and proactive strategies for routing in mobile ad hoc networks," *International Journal of Computational Intelligence and Applications*, vol. 5, no. 2, p. 169–184, 2005..

[58] M. Dorigo, G.D. Caro, "Ant colonies for adaptive routing in packet-switched communications networks.," *In Proceedings of the 5th ACM International Conference on Parallel Problem Solving from Nature*, p. 673–682, 1998.

[59] F. Ducatelle, L. M. Gambardella, G. D. Caro, "Anthocnet: An adaptive nature-inspired algorithm for routing in mobile ad hoc Networks," *Special Issue on Self-Organisation in Mobile Networking*, vol. 16:, p. 443–455, 2005.

[60] U. Sorges, I. Bouazizi, M. Gunes, "Ara - the ant-colony based routing algorithm for manets.," *In Proceedings of the 2002 International Conference on Parallel Processing Workshops, pages*, pp. 79-89, 2002.

[61] Q. Zhang, Thomas, R.H. Jacobsen, "Bioinspired Principles for Large-Scale Networked Sensor", Aarhus C: Aarhus School of Engineering, 2011.

[62] M. Dorigo, G.D Caro, "Ant colonies for adaptive routing in packet-switched Networks," *In Proceedings of the 5th ACM International Conference on Parallel Problem Solving from*

Nature, pp. 673-682, 1998.

[63] F. Dressler, "Self-Organization in Autonomous Sensor/Actuator Networks [SelfOrg]," 2010.

[64] Imenda, "Is there a conceptual difference between theoretical and conceptual frameworks?, 2014.," *Journal of ssocial science,* vol. 2, no. 38, pp. 185-195,, 2014.

[65] Elinas, "Knowing Your Neighbours: Machine Learning on Graphs," 2019.

[66] L. Barton, "Ramsey Theory," 2016.

[67] A. I. Wasserman, "Soetzre development methodologies and the user sofimare engineering methodology," 2014..

[68] D. Young, "Sofware Development Methodologies," 2013.

[69] STANDISH, "CHAOS SUMMARY (THE 10 LAWS OF CHAOS)," 2009.

[70] S. Jenner, "Why do projects "fail" and more to the point what can we do about it? the case for disciplined,'fast and frugal' decision-making," 2015.

[71] T. Wood-Hrper, John McManus, "A study in project failure," 2008.

[72] D. Sarkar, D. Gupta, S. Sharma, "Agile Processes and Methodologies: A Conceptual Study," 2012.

[73] T. Dyba, N. B. Moe, Torgeir Dingsøyr, "Agile Software Development: An Introduction and Overview," 2010.

[74] D. Marco. &. V. Maniezzo, A. Colorni, "Distributed Optimization by Ant Colonies.," in *In proceedings of ECAL91–European Conference on Artificial Life*, 1991.

[75] R. K. Ramakrishnan, S. Sivagaminathan, "A hybrid approach for feature subset selection using neural networks and ant colony optimization," *Expert systems with applications,,* vol. 33, no. 1, pp. 49-60, 2007.

[76] M. Khalaf, Muraina, I. D. Alobaedy, "Analysis of the number of ants in ant colony system algorithm.," *In 2017 5th International Conference on Information and Communication Technology (ICoIC7),* pp. 1-5, 2017.

[77] H. Aydin, Z. Yilmaz, "Examination of parameters used in ant colony algorithm over truss optimization," *Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi,,* vol. 24, no. 1, pp. 263-280., 2022.

[78] H. Hoos, T. Stützle, "MAX–MIN ant system," *Future generation computer systems, ,* vol. 16, no. 8, pp. 889-914., 2000.

[79] B. Hartl, R. F. &.C. Strauss, B. Bullnheimer, "A new rank based version of the Ant System. A computational study.," 1997.

[81] S. Zhang, "The Importance of Computer Network Applications and Security," *Journal of Computing and Electronic Information Management* , vol. 12, no. 3, pp. 2413-1660, 2024.

[82] C. Hua, Z. Juan. “Research on Computer Network Security and Countermeasures”, *Acta Physica Sinica*, 1287(4): 042027, 2019

[83] W. Xiaolin, Z. Dongmei, “Challenges and Countermeasures of Network Security.” *Science and Technology Review*, 40(1): 12-17, 2022.

[84] H. Hongzhang, J. Tang & Z. Jing, “Wireless sensor network routing optimization based on improved ant colony algorithm in the Internet of Things”, Vol. 10, no. 1, 2024, DOI: https://doi.org/10.1016/j.heliyon.2023.e23577

[85] D. Gordon, “The 'anternet' can teach us about digital interaction”, WIRED https://www.wired.com/story/deborah-gordon/ JUL 14, 2014

[86] K. Njoroge & Ayienga, “An Analysis of a campus LAN infrastructure: Case study for Kimathi University College” UON repository, 2012

[87] M. Mahesh & V.L. Desai, “Design and Implementation Issues in Ant Colony Optimization”, International Journal of Applied Engineering Research, ISSN 0973-4562 Volume 13, Number 16 pp. 12877-12882, 2018.

[88] B. Soni, P. Mathur & A. Bora“In Depth Analysis, Applications and Future Issues of Artificial Neural Network” DOI: 10.1007/978-3-030-52067-0_7, 2020.

[89] S. Katoch, S. Singh & S. Vijay , “A review on genetic algorithm: past, present, and future”, Springer Science+Business Media, LLC, part of Springer Nature. Volume 80, pages 8091–8126, 2021

[90] S. Dannels, "Research Design", *The Reviewer's Guide to Quantitative Methods in the Social Sciences*, 2nd Edition, 15, 2018.

[91] P. Leavy, "Research Design", *THE GUILFORD PRESS New York London,* 2017

[92] S. Sirisila, "Experimental Research Design", https://www.enago.com/academy/experimental-research-design/#: Jul 21, 2023

[93] K.L Knight. "Study/Experimental/Research Design: Much More Than Statistics", *Brigham Young University, Provo, UT,* doi: 10.4085/1062-6050-45.1.98, 2010

[94] R. Yin, "Collecting case study evidence", *Sage Publications London, UK.* 2009.

[95] Kothari , "Research Methodology; Methods and Techniques", *New age international (P) Ltd.* 2010

7.0 APPENDICES

7.1 APPENDIX A: TRAIN DATA (TEST DATA)

Optimizer	objfname	ExecutionT	Iter1	Iter2	Iter3	Iter4	Iter5
SSA	F3	0.57	0	50549.66	25297.42	20766.22	17017.96
SSA	F4	0.47	0	51.61	39.61	34.68	30.95
SSA	F5	0.68	0	29148290	9215553	5293100.05	3426865.68
SSA	F6	0.75	0	21472.21	12392.18	9403.04	8006.42
PSO	F3	0.36	282360.1	275875.8	259171	235053.47	214391.2
PSO	F4	0.13	91.94	88.29	83.36	78.19	73.19
PSO	F5	0.13	2.9E+08	1.52E+08	63539652	25318923.2	12370911.81
PSO	F6	0.13	74611.61	70424.45	63589.55	57237.77	50827.25
GA	F3	0.07	263465.1	209477	173669.9	150960.72	140239.46

GA	F4	0.02	90.35	89.55	88.73	87.98	87.82
GA	F5	0.03	3.03E+08	2.89E+08	2.65E+08	251083380.9	242612831.9
GA	F6	0.02	79764.86	77150.35	74689.5	70513.02	69412.9
BAT	F3	0.36	160103.8	153536.8	151951.1	148360.07	146428.3
BAT	F4	0.22	78.03	73.07	72.87	72.49	72.26
BAT	F5	0.2	1.52E+08	1.35E+08	1.34E+08	133523656.9	133396934.4
BAT	F6	0.16	50095.74	46443.7	45842.36	45761.3	45759.9
FFA	F3	0.12	246757.7	50725.7	32588.55	28349.13	26391.05
FFA	F4	0.06	90.79	43.22	40.5	38.64	37.9
FFA	F5	0.05	3.11E+08	10693115	8773491	8047995.48	7766313.74
FFA	F6	0.06	74434.91	12813.5	11675.75	11225.23	11055.36
GWO	F3	0.39	215852.5	175448.8	120851	82171.5	61106.36
GWO	F4	0.27	91.33	89.71	82.84	71.61	63.06
GWO	F5	0.29	3.08E+08	2.39E+08	1.43E+08	71393978.36	45318339.17
GWO	F6	0.23	76882.94	60737.95	43811.95	31316.99	25594.28

7.2 APPENDIX B: WORK PLAN

ACTIVITY	MAR - MAY	JUNE -OCT	NOVE -JAN	FEB-APRIL	MAY - JULY	AUGUST - OCT	NOVE -JANU	JAN-MAR	MAR - MAY	MAY -SEP	SEP - DEC
Developing Proposal Document											
Literature Review											
Proposal submission and defence (at Departmental Level)											

Proposal Submission and defence (at Faculty Level)											
Data Collection											
Data Analysis											
Thesis Writing											
Thesis Submission and Defense											
Final Thesis Submission											

7.3 APPENDIX C: BUDGET

ACTIVITIES	QUANTITY	RATE	TOTAL
PROPOSAL WRITING			
i. Stationery-print	4 reams	450.00	1800.00
papers	1 dozen	20 x 12	240.00
-Pens	2 GB	960	960.00
-Flash	3 pieces	50.00	150.00
disk	200 copies	30.00	6,000.00
Spring files	600 copies	3.00	1,800.00
Typesetting and	8	50.00	400.00
printing		100.00	100.00
		800.00	8000.00

Photocopying Binding (Loosely) Transport (Local)			4,000.00
Subtotal			**12,250.00**
DATA COLLECTION i. Producing questionnaires ii. Photocopying questionnaires iii. Subsistence iv. Transport (Local)	5 copies 50 20 days (Kakamega) 20 days (MMUST, TSNP)	30.00 50.00 200.00 200	150.00 2,500.00 4,000.00 4000.00
Subtotal			**10,650.00**
THESIS PREPARATION i. Typesetting and printing ii. Photocopying iii. Binding iv. Transport (Local) v. Subsistence	180 copies 7 copies 7 copies 4 days (Kakamega) 4 days (Kakamega)	30.00 360.00 400.00 1000.00 1000.00	5,400.00 3,600.00 2,800.00 4,000.00 4,000.00
Subtotal			**15, 800.00**
MISCELLENEOUS (10%)			**8,900.00**
GRAND TOTAL			**KSH. 50,100.00**

7.4 APPENDIX D: UNIVERSITY RESEARCH PERMIT

MASINDE MULIRO UNIVERSITY OF SCIENCE AND TECHNOLOGY (MMUST)

Tel: 056-30870
Fax: 056-30153
E-mail: directordps@mmust.ac.ke
Website: www.mmust.ac.ke

P.O Box 190
Kakamega – 50100
Kenya

Directorate of Postgraduate Studies

Ref: MMU/COR: 509099

Date: 7th July, 2021

Lusweti Samuel Wafula,
SIT/G/15/15
P.O. Box 190-50100,
KAKAMEGA.

Dear Mr. Wafula,

RE: APPROVAL OF PROPOSAL

I am pleased to inform you that the Directorate of Postgraduate Studies has considered and approved your Masters proposal entitled: ***"An Enhanced Bio-Inspired Aco Model for Fault Tolerant Networks"*** and appointed the following as supervisors:

1. Dr. Odoyo Collins - School of Computing and Informatics, MMUST
2. Ms. Dorothy A Rambim - School of Computing and Informatics, MMUST

You are required to submit through your supervisor(s) progress reports every three months to the Director Postgraduate Studies. Such reports should be copied to the following: Chairman, School of Computing and Informatics Graduate Studies Committee and Chairman, Computer Science Department. Kindly adhere to research ethics consideration in conducting research.

It is the policy and regulations of the University that you observe a deadline of two years from the date of registration to complete your Master's thesis. Do not hesitate to consult this office in case of any problem encountered in the course of your work.

We wish you the best in your research and hope the study will make original contribution to knowledge.

Yours Sincerely,

Dr. Consolata Ngala
DEPUTY DIRECTOR, DIRECTORATE OF POSTGRADUATE STUDIES

7.5 APPENDIX E: NACOSTI RESEARCH PERMIT

REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: 726104

Date of Issue: 04/August/2021

RESEARCH LICENSE

This is to Certify that Mr.. SAMUEL WAFULA LUSWETI of Masinde Muliro University of Science and Technology, has been licensed to conduct research in Kakamega on the topic: AN ENHANCED BIO-INSPIRED ACO MODEL FOR FAULT TOLERANT NETWORKS for the period ending : 04/August/2022.

License No: NACOSTI/P/21/12076

726104

Applicant Identification Number

Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.