EFFICIENT FISH TRANSPORTATION IN VICTORY FARMS LIMITED (VF) HOMA-BAY COUNTY USING HUNGARIAN METHOD

By

VOLLINCE MAGAMA KARGUNG'S

A Thesis Submitted to School of Science and Aerospace in Partial Fulfilment of Masters of Science in Applied Mathematics.

MOI UNIVERSITY.

2024

DECLARATION

This thesis is my original work and has not been presented for a degree in this or any other university. No part of this thesis may be reproduced without prior permission of the author and / or Moi University.

Signature:.. **Date:** $4th December, 2024.$

Kargung's Vollince Magama
MSC/AM/03/18

DECLARATION BY SUPERVISORS

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

Signature:.. **Date:** $4th December, 2024.$

Dr. Cleophas Kweyu
Department of Mathematics, Physics and Computing,
School of Sciences and Aerospace Studies.
Moi University. Eldoret, Kenya.

Signature: ... .. **Date:** $4th December, 2024.$

DR. Titus Rotich
Department of Mathematics, Physics and Computing,
School of Sciences and Aerospace Studies.
Moi university.

DEDICATION

I dedicate this thesis to:

My family, whose unwavering support and encouragement have been my pillars of strength throughout this journey. Your love and understanding have made every challenge surmountable.

My mentors and advisors for their guidance, expertise, and patience. Your insights have shaped my intellectual growth and fueled my passion for knowledge.

My friends who have been a constant source of inspiration and laughter. Your camaraderie has made the academic road a joyous one.

Finally, to myself, for the determination and resilience that carried me through the highs and lows of this academic endeavor. This achievement is a testament to my dedication and hard work.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to the following individuals who have played a significant role in the completion of this thesis:

My supervisors; Dr. Kweyu Cleophas and Dr. Titus Rotich, whose guidance, wisdom, and unwavering support have been invaluable throughout this research endeavor. Your expertise and constructive feedback have greatly enriched the quality of this work.

I am indebted to The Glamour Beauty Company for providing financial support through. This assistance has enabled me to focus on my research without financial constraints.

I am grateful to my friends, Dr. Mwan Denis,Eng. Pamela Onsongo,Miss Sariah Lenser, Mr. Subby Mino,Mr. Nixon Ogweno, Dr.Cornelious Nyakundi and the family of Kargung's for their unwavering encouragement, understanding, and patience throughout this academic journey. Your love and support have been a constant source of strength and motivation.

Finally, I would like to acknowledge my own perseverance and determination in overcoming challenges and obstacles along the way. This achievement would not have been possible without my steadfast commitment to excellence.

Thank you all for your contributions, encouragement, and belief in my abilities. Your support has been instrumental in making this thesis a reality.

ABSTRACT

Fish spoilage during transit has been a major issue for tilapia fish production companies, leading to significant losses and hindering growth while affecting food security by reducing the availability of high-quality fish. This study applied the Hungarian method to optimize fish transportation in Victory Farms Limited (VF) Homa-Bay County, with specific objectives to evaluate the shortest, fastest, and most cost-effective routes for distributing tilapia fish products in the Western Kenya region. Victory Farms currently uses a random delivery approach for distributing its fish, which results in inefficiencies and contributes to spoilage. The Hungarian method was selected due to its ability to systematically determine the optimal match between two sets of elements, making it particularly suitable for optimizing transportation routes in a cost-effective manner. Primary data on distances, travel times, and costs between key distribution points were collected directly from Victory Farms and used to create a cost matrix, which was analyzed using the Hungarian algorithm. The study identified an optimal distribution route for all three objectives: shortest distance, fastest transit time, and lowest cost. The optimal route was determined as follows: KLC $\rightarrow$ Kondele $\rightarrow$ Nyalenda $\rightarrow$ Majengo $\rightarrow$ Mbale $\rightarrow$ Chavakali $\rightarrow$ Kakamega $\rightarrow$ Mumias $\rightarrow$ Bungoma $\rightarrow$ Luanda $\rightarrow$ Siaya $\rightarrow$ KLC. This route spanned 293.2 kilometers, took 351.9 minutes in transit, and incurred a cost of Ksh. 14,660. These findings provide valuable insights into streamlining distribution processes, reducing spoilage, and improving operational efficiency. The results demonstrate that Victory Farms Limited can enhance profitability and growth by adopting the Hungarian method for route optimization. This approach also contributes to food security by ensuring the availability of high-quality fish. It is recommended that Victory Farms Limited regularly implement the Hungarian method for optimizing routes, while future research could explore integrating real-time environmental and traffic data into the model for additional enhancements.

TABLE OF CONTENTS

List of Figures

List of Tables

LIST OF ABREVATIONS AND ACRONYMS

AP	:	Assignment Problem.
AVL	:	Automatic Vehicle Location.
CVRP	:	Capacitated Vehicle Routing Problem.
DARP	:	Dial-a-Ride Problem.
DP	:	Dynamic Programming.
EAC	:	East Africa Community.
E-VRP	:	Electric Vehicle Routing Problem.
ER	:	Emergency Response.
FLC	:	Farm Logistic Center.
GA	:	Genetic Algorithm.
GVRP	:	Generalized Vehicle Routing Problem.
KLC	:	Kisumu Logistic Center.
KSH	:	Kenya Shillings.
LIPP	:	Linear Integer Programming Problem.
LP	:	Linear Programming.
LPP	:	Linear Programming Problem.
LTL	:	Less Than Truckload.
MIP	:	Mixed Integer Programming.
MV	:	Multi-Vehicle.
NP-Complete	:	Non-deterministic Polynomial-time Complete.
NP-Hard	:	Non-deterministic Polynomial-time Hard.
OR	:	Operations Research.
PR	:	Pickup and Delivery Routing.
PT	:	Public Transportation.
RAP	:	Resource Allocation Problem.
RR	:	Real-Time Routing.
SCP	:	Supply Chain Problem.
STSP	:	Symmetric Traveling Salesman Problem.
TSP	:	Traveling Salesman Problem.
VAT	:	Value Added Tax.
VFL	:	Victory Farms Limited.
VRP	:	Vehicle Routing Problem.
VRPTW	:	Vehicle Routing Problem with Time Windows.
WTP	:	Waste Transportation Problem.

DEFINITIONS OF TERMS

Operations research: Hungarian technique Operations research (OR) is the application of mathematical scientific methods in study and analysis of problems involving complex systems.

Hungarian technique: Hungarian technique is an efficient method of finding optimal solution to an assignment problem without making a direct comparison to every solution.

CHAPTER ONE

INTRODUCTION

This chapter covers; the background of the study, the objectives of the study, the statement of the problem, the justification of the study, the significance of the study , and the scope of the study.

1.1 Background of the Study

1.2 Introduction to Operations Research

Operations Research (OR) is a multidisciplinary field that employs mathematical models, statistical analyses, algorithms, and scientific methodologies to analyze complex decision-making problems and optimize the performance of systems. Initially developed during World War II to address military logistics and strategy issues, OR has since evolved into a comprehensive discipline applied across various sectors, including transportation and logistics, production planning, inventory control, scheduling, location analysis, forecasting, and supply chain management, as noted by Kumar (2020).

At its core, OR seeks to provide a scientific basis for decision-makers to efficiently allocate limited resources and optimize outcomes under constraints such as cost, time, and risk. Nguyen, (2022) highlights that the strength of OR lies in its ability to model real-world problems in mathematical terms, thereby enabling the use of computational tools and techniques to derive optimal or near-optimal solutions. Over the decades, OR has proven invaluable in numerous industries and applications, providing frameworks that enhance both strategic planning and

operational efficiency.

The quantitative methods commonly employed in OR include linear programming, queuing theory, simulation, and network analysis. Ganesan (2021) emphasizes that these techniques allow for the evaluation of different scenarios and help decision-makers choose the best course of action by quantifying various factors that influence decision outcomes. This approach enables organizations to achieve goals such as cost reduction, resource optimization, and process improvement.

1.3 The Scope and Relevance of Operations Research

Operations Research (OR) has extensive applicability across a wide range of industries. In transportation and logistics, Lofberg (2019) highlights that OR techniques are utilized to optimize the movement of goods and services, thereby reducing costs, improving delivery times, and enhancing customer satisfaction. In production planning and inventory management, Ahmadi (2020) explains that OR helps businesses determine optimal inventory levels, balancing costs associated with storage and stock-outs while ensuring product availability.

Furthermore, Naderi (2021) demonstrates that OR methodologies have proven vital in scheduling by optimizing the allocation of resources such as labor, machines, and time, which improves productivity and minimizes downtime in both manufacturing and service environments. In location analysis, Zhang (2020) discusses how OR aids in identifying the most strategic locations for facilities, such as warehouses, retail stores, and service centers, crucial for minimizing costs and maximizing service coverage.

The relevance of OR has significantly increased with the rise of data-driven decision-making in the digital age. Bertsimas (2017) notes that recent advancements in computational technology and the availability of large datasets have enhanced OR's capability to solve complex, large-scale problems in real-time. Consequently, businesses can implement OR solutions more efficiently and effectively, making OR an integral part of modern organizational strategies.

1.4 The Transportation Problem in Operations Research

The transportation problem in Operations Research focuses on determining the optimal way to transport goods from multiple supply locations, such as factories or warehouses, to various demand locations, like retail stores or distribution centers. Kumar (2021) emphasizes that the primary objective is to minimize the total cost of transportation while satisfying the supply and demand constraints at each location. This problem is commonly modeled as a Linear Programming Problem (LPP), where the cost minimization objective is subject to a set of constraints that must be satisfied.

Fundamentally, Gupta (2023) states that solving the transportation problem involves deciding the quantity of goods to transport from each supply point to each demand point to minimize overall costs. This decision-making process entails evaluating the costs associated with transporting goods between different points and ensuring that the transportation plan meets both the supply limits at the sources and the demand requirements at the destinations

1.4.1 Objectives and Constraints

The primary objective of the transportation problem is to minimize total transportation costs. Singh (2022) explains that this is achieved by identifying the

most cost-effective routes for transporting goods from sources to destinations. The constraints ensure that all available supply is allocated without exceeding the capacity at each source and that the demand at each destination is met without falling short. Each route between a source and a destination has an associated cost, which may depend on factors such as distance, time, and the mode of transportation used.

The transportation problem involves balancing supply and demand. Chen (2020) highlights that the supply constraints specify that the quantity of goods transported from a given source should not exceed its available supply. Conversely, the demand constraints ensure that the amount of goods delivered to a particular destination meets or exceeds its demand. Therefore, the transportation problem aims for an efficient allocation of resources while minimizing costs associated with moving goods from suppliers to consumers.

1.4.2 Relevance in Supply Chain and Logistics

The transportation problem is particularly relevant in logistics and supply chain management, where transportation costs often represent a significant portion of total logistics expenses. Raza (2023) emphasizes that optimizing these costs is critical for maintaining a company's financial health and competitive positioning. Effective transportation strategies enable companies to reduce logistics expenses, improve service levels, and ensure high levels of customer satisfaction by guaranteeing the timely delivery of products.

By optimizing the transportation process, organizations can better manage their supply chains, achieving a more balanced and responsive approach to handling fluctuations in supply and demand. Raut (2021) notes that efficient transportation

planning can help mitigate the impact of unexpected disruptions in the supply chain, such as delays in production or shifts in consumer demand, allowing for flexibility and adaptability—key components in today's dynamic business environment.

1.4.3 Practical Applications and Significance

The transportation problem has numerous practical applications across various sectors, including manufacturing, retail, distribution, and public sector operations. Martins (2021) and Venkatesh (2022) emphasize that it is central to decision-making processes in logistics and supply chain management, given its emphasis on optimizing the allocation of resources—such as goods and services—from multiple sources to numerous destinations while minimizing associated costs. Solving the transportation problem allows organizations to achieve cost efficiencies, improve service levels, and manage their supply chains more effectively in response to dynamic market demands and challenges.

1.4.4 Applications in the Manufacturing Sector

In the manufacturing sector, the transportation problem is crucial for optimizing the distribution of goods produced at different plants or factories to various regional warehouses or directly to customers. Fernandes (2020) provides an example of a manufacturing firm with multiple production facilities in different regions that must determine the most cost-efficient way to transport finished products to a network of distribution centers or retail outlets. This requires balancing several factors, including transportation costs, production schedules, and inventory levels, to ensure timely product delivery at the lowest possible cost.

Efficiently managing the transportation of goods is critical in sectors such as automotive, electronics, and consumer goods manufacturing, where just-in-time (JIT) production and lean inventory practices are prevalent. Gade (2023) explains that

the transportation problem helps optimize the routing and scheduling of shipments to minimize delays and avoid overproduction, which can lead to excess inventory costs or stock-outs. Additionally, manufacturers often face varying levels of demand across different regions, making it essential to use transportation models to dynamically adjust distribution strategies, ensuring that demand is met effectively while keeping costs low.

1.4.5 Applications in the Retail and Distribution Sectors

In the retail and distribution sectors, the transportation problem was essential for optimizing the movement of goods from central distribution centers to individual retail stores. Kumar and Modgil (2020) noted that retail chains needed to manage large volumes of inventory across multiple locations, ensuring that each store was adequately stocked with the right mix of products to meet customer demand. The transportation problem provided a framework for determining the most cost-effective routes and transportation modes for delivering goods from warehouses to stores, minimizing logistics costs while maintaining high service levels.

For instance, a large supermarket chain had to determine how to distribute fresh produce from its central distribution hub to a network of stores across a city or region. As Kumar & Modgil (2020) pointed out, the challenge was to ensure that products arrived fresh and within a specific time window to prevent spoilage while minimizing transportation costs.

1.4.6 The Hungarian Method

The Hungarian method, introduced by Dénes Kőnig (1931), and later refined by Harold Kuhn (1955), was a groundbreaking technique for solving assignment problems efficiently. Kuhn (1955) significantly improved upon brute-force approaches, which required exhaustive comparisons of all possible solutions. Instead,

the Hungarian method optimized the process by systematically reducing the cost matrix, ensuring that at least one zero appeared in every row and column.

The key to the Hungarian method, as described by Munkres (1957), was its ability to modify the cost matrix through a series of operations, such as subtracting the smallest element in each row and column from all elements in that row or column. This manipulation simplified the problem by creating zeros, which indicated potential assignments with no additional cost. An assignment yielding zero opportunity cost was considered optimal, representing an efficient allocation that minimized resource utilization while maximizing efficiency.

The Hungarian method had significant impacts across various fields, including operations research, logistics, economics, and engineering, where assignment problems were prevalent. Hillier and Lieberman (2001) noted that in logistics, the method optimized the assignment of tasks to machines, workers to jobs, or goods to transportation routes. In economics, Gass and Harris (2001) explained its use in resource allocation or matching supply with demand in markets. Engineering applications, as described by Winston (2004), included task assignments in manufacturing processes and the optimization of network flows.

The algorithmic approach of the Hungarian method involved several steps: reducing the matrix, covering zeros with a minimum number of lines, adjusting the matrix by subtracting and adding elements, and iterating until an optimal assignment was found Winston, (2004). This process ensured a solution that was both computationally efficient and economically favorable.

In addition to its practical applications, the Hungarian method inspired further research and development in optimization techniques. Papadimitriou & Steiglitz (1998) highlighted that variants and extensions of the method had been developed to handle more complex and large-scale assignment problems, enhancing its applicability in modern industries.

Through its innovative algorithm and widespread utility, the Hungarian method remained a cornerstone for optimizing resource allocation and decision-making processes in complex systems. Its ability to provide optimal solutions with reduced computational burden made it a vital tool in both theoretical and applied operations research.

1.4.7 Optimizing Delivery Routes at Victory Farms: A Comparative Analysis of Methods

Victory Farms, located in Homa-Bay County, Kenya, has become one of sub-Saharan Africa's fastest-growing tilapia fish farms. Established in 2016, the farm set out with an ambitious goal: to construct a commercial tilapia farm that would provide affordable, nutritious, and accessible protein to 2 billion Africans over the next two decades. This goal was established against the backdrop of a 95% reduction in Nile perch and tilapia catch in Lake Victoria over the past 50 years Njiru et al. (2008). With increasing demand for protein driven by growing populations and rising prosperity, Victory Farms has sought innovative solutions to combat the decline in local fish supply, especially as fish imports often arrive aged up to two years by the time of consumption.

To address these challenges, Victory Farms embraced aquaculture practices, enabling them to sell over 10 metric tons of fresh tilapia daily to low-income

neighborhoods across Kenya. By early 2021, the farm aimed to increase sales to 30 metric tons per day, providing over 20 million high-protein meals annually. The company committed to ensuring that fish reached consumers within 48 hours of harvest, avoiding antibiotics and harsh chemicals.

To optimize their delivery network, Victory Farms established sales depots in Nairobi and Western Kenya, with 27 and 23 outlets, respectively. The need for efficient and cost-effective delivery routes became clear. Operations Research (OR) techniques like the Hungarian method, the nearest neighbor algorithm, and the transportation model were considered to achieve this optimization.

The Hungarian method, introduced by Dénes Kőnig (1931) and later refined by Harold Kuhn (1955), emerged as a powerful tool for solving assignment problems efficiently. This method guarantees optimal solutions by reducing the cost matrix and ensuring that at least one zero appears in every row and column Munkres (1957). The Hungarian method is particularly useful in minimizing transportation costs and optimizing resource allocation, making it ideal for solving delivery route problems like the one at Victory Farms.

Another option considered was the nearest neighbor algorithm, a greedy approach where the algorithm selects the closest unvisited depot to the current location, forming a route based on the shortest path. While simple and fast, this method does not guarantee the optimal solution and may result in higher transportation costs due to its tendency to build suboptimal routes.

Additionally, the transportation model, a linear programming approach, was

explored. This model focuses on minimizing the total cost of transporting goods from multiple sources (suppliers) to multiple destinations (depots). However, the transportation model can be computationally intensive, particularly with a large number of delivery points, and may not be as efficient for dynamic adjustments in real-time compared to the Hungarian method.

Given the need for an optimal solution with minimal computational effort, the Hungarian method stood out as the best option. It efficiently reduced transportation costs by optimizing the assignment of delivery routes while ensuring that Victory Farms could meet its 48-hour delivery goal. The Hungarian method also provided the flexibility required to adjust routes based on changing conditions, such as shifts in demand or unforeseen delays.

By implementing the Hungarian method, Victory Farms could not only minimize transportation expenses but also improve the freshness and quality of its products, contributing to increased customer satisfaction. Moreover, this optimization aligned with the company's broader goals of supporting Kenya's food security and job creation, in line with the country's Big 4 Agenda policy.

1.5 Statement of the Problem

Victory Farms Limited, a prominent fish farming enterprise situated on the shores and offshore of Lake Victoria within Homa-Bay County, stands as one of Kenya's largest fish farms and holds the distinguished position of being the foremost fish producer in East Africa. Specializing in the production, processing, and distribution of Tilapia fish, the farm serves both the Nairobi and Western Kenya regions. With an extensive network of 23 depots solely in Western Kenya, the farm faces a

substantial demand averaging 50 tons of Tilapia fish. Given the perishable nature of fish products, timely delivery to the consumer market is imperative to prevent spoilage. Consequently, longer delivery routes result in unnecessary additional costs, thereby straining the farm's financial resources. This study aims to address this challenge by employing the Hungarian method in operations research to identify the optimal route that ensures suppliers take the shortest, most time-efficient, and cost-effective transportation paths to deliver fish products to consumers in the farm's depots within the Western Kenya region.

1.6 Justification

By finding the shortest distance to the depots, this study will help the Victory Farm Limited on cutting on the cost of transporting the product from the farm to the market, thus improving on the profits which will lead to the company's expansion and productivity. By finding the route that uses the shortest time to deliver the products to the market, the company will save on spoilage of their product before it gets to the consumers, thus avoiding unnecessary losses due to spoilage. This puts the Kenyan government on the right track in achieving its millennium development goals and one of the big four agenda (food security).

1.7 Objectives

1.7.1 General Objective

The primary aim of this study was to employ the Hungarian method to ascertain the optimal route for distributing fish from Victory Farms Limited, located in Homa-Bay County, to its network of depots in Western Kenya.

1.7.2 Specific Objectives

The specific objectives of this study were to:

i. Apply the Hungarian method to evaluate the shortest route for distributing fish products in the Western Kenya region.

ii. Use the Hungarian method to determine the minimum time route for fish product distribution.

iii. Identify the most cost-effective route for distributing fish products using the Hungarian method.

1.8 Significance of the study

The significance of this study lies in its potential to revolutionize the efficiency of fish distribution networks in Western Kenya. By employing the Hungarian method to identify the shortest and most cost-effective routes for delivering fish from Victory Farm Limited to supply depots, the research addresses critical aspects of logistics. The study's impact is multifaceted: it optimizes transportation routes to minimize delivery time, thereby enhancing customer satisfaction and market share; it determines the minimum cost of transportation, allowing the farm to maximize profits and operate more competitively; it guards against spoilage by minimizing the time products spend in transit, preserving quality and reducing waste; it directly contributes to the growth of Victory Farms, fostering competitiveness and expansion; and, on a broader scale, it contributes to food stability in Kenya by improving the efficiency of fish product distribution. In essence, the study's findings have far-reaching implications for the economic success of Victory Farms and the overall well-being and food security of the region.

1.9 Scope

This thesis has utilized the Hungarian method to optimize the selection of the most efficient and cost-effective routes for delivering fish from Victory Farm Limited to supply their depots located in Western Kenya. The study has focused specifically on assessing various transportation options to determine the shortest distance, time required and costs for the delivery process. Through a comprehensive analysis, the research has considered factors such as road conditions, transportation costs, and time constraints to identify the most suitable routes for transporting fish products from the Victory Farm Limited, to the designated supply depots in Western Kenya. By delving into this specific geographical area and employing mathematical optimization techniques, the study intended to contribute valuable insights into enhancing the efficiency of fish distribution networks in the region.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter gives a brief introduction to the Victory Farm Limited, a review of literature on Vehicle Routing Problem, and the Hungarian method of solving Operations Research Problems.

2.2 Victory Farm Limited

Victory Farms Limited is an aquaculture tilapia fish farm established in 2016 by Joseph Rehmann, an Egyptian who has lived and worked in five continents. Victory Farms Limited was launched with the aim of being the world's most sustainable fish farm and to create new ways where business can support the development of communities and the restoration of nature. Victory Farms Limited is located on the Kenyan side of Lake Victoria and has distribution capacity across the country. The firm is rapidly expanding its operation, fish processing and sales & marketing capacities. The company holds the vision that tilapia aquaculture represents the best solution for feeding the fifty million Kenyans (https://www.victoryfarmskenya.com/#media&impact retrieved on February, 10th 2022.) Victory farms aims to be the most sustainable fish farm on the planet while it scales to feed 2 billion Africans in the coming two decades. The organization has extensively recruited and trained university graduates on aquaculture best management practices and leadership skills. Victory farms has largely leveraged on technology which has seen it outshine other competitors. The company's success has further been accelerated by the governments support for permitting water and land use as

well as VAT exemption on aqua imports. Victory farms provide access –to- market via Kenya-wide cold chain network with affordable, accessible, and sustainable tilapia. The company's technology and low cost inputs can make fish more affordable to all by as much as 50% more affordable. Through Victory Farms, Kenya may be able to stop Chinese imports, fully supply its own fish needs and in the medium term export to EAC.

There is high demand and production of fish in Kenya. However, fish demands in the local still exceeds the supply thus creating a shortage. Kenya having a coastal region and bordering Lake Victoria which has been a source of fish for centuries. Lake Victoria's output has however, been declining sharply due to overfishing, pollution and lack of regulation. This gap in supply due to declining fishing is being replaced by imported fish. Efforts should be made to revamp the sector since Kenya and East Africa have excellent conditions and capabilities to produce local market when presented with right technology, knowledge and inputs.

2.3 A Review of Vehicle Routing Problem

The Vehicle Routing Problem (VRP), introduced by Dantzig and Ramser (1959), remains one of the most extensively researched problems in combinatorial optimization. It addresses how to determine the most efficient routes for vehicles to serve a set of customers with known demands from one or more depots, while minimizing overall costs and adhering to vehicle and route constraints. The VRP is critical to modern supply chain management and has continuously evolved to meet the changing landscape of logistics.

2.3.1 Core Concept and Importance of the VRP

At its core, the VRP focuses on optimizing the total cost of vehicle routes for tasks such as delivery, collection, or service operations, all while considering factors like vehicle capacity, travel distance, and time windows Wang, (2021). The VRP

plays a pivotal role in logistics and transportation industries, where companies must minimize costs, optimize resource usage, and meet strict delivery schedules to maintain competitiveness Erdelic, (2019).

Industries such as Amazon, FedEx, and DHL heavily rely on VRP algorithms to ensure operational efficiency, reduce fuel consumption, and improve delivery times Gansterer, (2020). Optimizing routes also reduces environmental impacts, which aligns with the increasing emphasis on sustainability in logistics Wang, (2021).

The VRP's ability to streamline operations, lower costs, and enhance customer satisfaction makes it an indispensable tool across various sectors, including e-commerce, healthcare, and public services Koc, (2020). As global supply chains grow more complex, the role of VRP in achieving operational efficiency will continue to expand Cattaruzza, (2017).

2.4 Advances in Vehicle Routing Problem (VRP) Research

Numerous surveys have been conducted to track the evolution and diversification of the Vehicle Routing Problem (VRP). Early works, such as Bellmore & Nemhauser, (1968). Foundational insights into the complexity of the problem, focusing on basic formulations aimed at minimizing distance or time. While their contributions were essential for understanding the fundamental structure of the problem, their approach did not address more modern challenges, such as the dynamic nature of demand or real-time traffic conditions that are prevalent in today's logistics systems. As the field has evolved, more recent studies have extended the VRP to cover these challenges, incorporating techniques to handle real-time changes in logistics.

Wang & Wasil,(2021) reviewed recent methodologies for solving VRP in dynamic, real-time scenarios. Their study highlighted the increasing need for algorithms that can adapt to fluctuating demand and unpredictable conditions like traffic congestion. However, their focus on metaheuristic approaches, while useful for handling complex, dynamic systems, often sacrifices optimality for speed. These metaheuristics, though capable of providing good solutions, may not guarantee the best possible outcome, especially when an exact solution is needed. In contrast, the Hungarian method, when applied to multi-objective VRP—where distance, time, and cost are all considered simultaneously—provides an optimal solution through its exact assignment process. The Hungarian method ensures that the trade-offs between these objectives are addressed in a structured manner, yielding the most efficient allocation for each objective.

In a similar vein, Gansterer & Hartl, (2020) explored collaborative approaches to VRP, emphasizing the role of multiple companies sharing fleets and resources to minimize costs, especially in urban environments facing increasing congestion. While these collaborative approaches are vital for modern logistics, they typically rely on decentralized decision-making and heuristic methods, which may not always yield optimal solutions. The Hungarian method, by focusing on exact assignments within a fixed matrix, can offer an optimal solution in situations where resources and routes are well-defined and fixed. When applied to multi-objective routing, the Hungarian method can efficiently optimize for cost, time, and distance in collaborative settings where these three factors need to be considered simultaneously.

The VRP has continued to evolve with the advent of more complex variants, including those integrating electric vehicles and multi-objective optimization. Adewumi & Adeleke,(2018) focused on the role of metaheuristics in solving complex

VRP variants, discussing the trade-off between computational time and solution accuracy. While metaheuristic algorithms are often used to quickly generate good solutions, they do not guarantee optimality. The Hungarian method, in contrast, provides a precise and optimal solution for multi-objective VRP, balancing distance, time, and cost. This makes it highly suitable for problems in logistics, such as the distribution of perishable goods like fish, where both exact assignments and cost minimization are essential for operational efficiency.

Koc et al. (2020) reviewed multi-objective VRP, emphasizing the need to balance conflicting objectives like cost minimization and service quality maximization through Pareto-optimal solutions. While their work on multi-objective optimization is essential for understanding the complexities of modern logistics, it often involves approximation methods such as Pareto-frontiers. These methods, while effective for balancing multiple objectives, may not always provide the best solution for each individual objective. The Hungarian method, by contrast, provides a clear and optimal solution when applied to multi-objective VRP, ensuring that distance, time, and cost are all optimized simultaneously, with no need for complex approximation methods.

In conclusion, while the existing literature has contributed valuable insights into the diverse challenges of VRP and the methods used to address them, much of the recent research relies on heuristic or metaheuristic algorithms that may not guarantee optimal solutions. The Hungarian method, when applied to multi-objective problems such as minimizing distance, time, and cost, offers a robust and precise alternative. Its ability to deliver exact solutions with minimal computational cost makes it particularly suitable for logistics problems where multiple objectives must be optimized simultaneously. By combining the strengths of the Hungarian method with modern

advancements in multi-objective optimization, it is possible to develop more efficient and reliable solutions for modern transportation and logistics problems.

2.5 Evolution of VRP and Its Variants

As industries evolve and new challenges emerge, the VRP has been adapted into various forms to address specific logistics needs. Below are some prominent VRP variants that have emerged in recent years:

2.5.1 Capacitated VRP (CVRP)

This variant considers vehicle capacity constraints, ensuring no vehicle exceeds its maximum load during a delivery route. CVRP is essential in industries where weight and volume are critical factors, such as retail and manufacturing Adewumi, (2018).

2.5.2 VRP with Time Windows (VRPTW)

Vehicles must arrive at customer locations within specific time windows. Hashimoto et al. (2020) reviewed VRP with time-sensitive constraints, emphasizing how balancing travel time with service quality is vital for sectors like e-commerce and healthcare Hashimoto, (2020).

2.5.3 Electric VRP (E-VRP)

As sustainability becomes a priority, the adoption of electric vehicles (EVs) in logistics has led to the development of the E-VRP, which accounts for limited battery life and the need for charging stops Erdelic, (2019). This is particularly relevant in green logistics, where reducing carbon footprints is a top priority Wang, (2021).

2.5.4 Collaborative VRP (C-VRP)

Gansterer and Hartl (2020) examined C-VRP, where logistics providers collaborate to optimize routes collectively. Collaboration allows companies to share

fleets and reduce operational costs, particularly in congested urban environments Gansterer,(2020).

2.5.5 Multi-objective VRP (MOVRP)

This variant seeks to optimize multiple objectives simultaneously, such as minimizing costs while maximizing service quality Vidal, (2020). The Pareto-optimal approach is often used to solve these problems, balancing conflicting goals in real-world scenarios (Koc, 2020).

2.5.6 Stochastic and Dynamic VRP (SDVRP)

In SDVRP, some parameters, such as customer demand or travel time, are uncertain or change over time. Ritzinger et al. (2016) and Ilin et al. (2019) explored stochastic and dynamic approaches to VRP, where algorithms must adjust in real-time based on new data. This variant is increasingly relevant in sectors like food delivery and ride-hailing, where demand is volatile Ilin, (2019).

2.5.7 Rich VRP (RVRP)

Caceres-Cruz et al. (2018) introduced Rich VRP, which integrates multiple constraints like fleet heterogeneity and time-dependent travel times. Rich VRP models reflect real-world scenarios more accurately, providing a more comprehensive solution for industries with diverse operational requirements CaceresCruz, (2018).

2.5.8 VRP with Pickup and Delivery (VRPPD)

In this variant, vehicles must perform both pickup and delivery tasks on the same route. Koc et al. (2020) explored how VRPPD can optimize operations in sectors like postal services and grocery delivery, where vehicles must handle both pickups and drop-offs (Koc, 2020).

2.5.9 Split Delivery VRP (SDVRP)

Archetti and Speranza (2019) reviewed SDVRP, where deliveries to a single customer can be split across multiple vehicles. This variant is particularly useful when vehicle capacity is limited or when customers have large orders that cannot be fulfilled by a single vehicle (Archetti, 2019).

2.6 Solution Approaches for VRP

Various solution methods have been developed to address the complexities of VRP, ranging from exact algorithms to heuristic and metaheuristic methods.

2.6.1 Exact Algorithms

Exact methods, such as Branch-Price-and-Cut, guarantee optimal solutions but are computationally expensive for large-scale problems Costa, (2019). Exact algorithms are typically best suited for small- to medium-sized instances, where computational time is less of an issue.

2.6.2 Metaheuristic Algorithms

Metaheuristics, such as genetic algorithms and simulated annealing, are commonly used for large-scale VRP problems. Elshaer & Awad,(2020) conducted a taxonomic review of metaheuristic algorithms for VRP, highlighting their effectiveness in balancing solution quality and computational cost Elshaer, (2020).

2.6.3 Hybrid Methods

Hybrid approaches combine multiple algorithms to enhance performance. Karakatic and Podgorelec, (2020) explored the use of hybrid algorithms that combine elements of genetic algorithms and local search methods for solving multi-depot VRP, where multiple depots complicate routing decisions Karakatic, (2020).

Furthermore, the integration of artificial intelligence (AI) and machine learning (ML)

in solving VRP has gained traction in recent years. Reinforcement learning, for example, has been used to adjust vehicle routes in real-time, improving decision-making in unpredictable environments Gansterer, (2020).

2.6.4 The Hungarian Method

The Hungarian method is a classical combinatorial optimization algorithm used to solve the assignment problem efficiently. The assignment problem involves assigning resources (such as workers to jobs or vehicles to tasks) in a one-to-one manner, with the objective of minimizing total distance,time and cost or maximizing total profit. The method was first proposed by König, (1931) and later improved by Kuhn in 1955, who introduced a more efficient computational approach Konig, (1931), Kuhn, (1955).

2.6.5 Development of the Hungarian Method

König, (1931) initially developed a framework to solve assignment problems by transforming the cost matrix in such a way that it contains at least one zero in each row and column. The optimal assignment is then determined by selecting the zeros, ensuring that the opportunity cost of the assignments is minimized. Kuhn (1955), expanded on König's work and formalized the Hungarian method as a computational algorithm, utilizing duality in linear programming for efficiency. Kuhn also established that the assignment is optimal if and only if it is complete after every possible transfer.

2.6.6 Applications and Extensions of the Hungarian Method

The Hungarian method has been applied in various fields beyond its original formulation. Robert, (2005) studied the assignment of workers to jobs in an economy with coordination frictions, where heterogeneous workers were matched with heterogeneous jobs.

Naveh et al. (2007), integrated constraint programming and artificial intelligence with the Hungarian method, improving the speed and quality of solutions for large-scale problems. Similarly, Mansi, (2011) proposed an alternative method that achieved the same results as the Hungarian method but with fewer computational steps, making it more efficient.

Shafahi & Ramezani (2007), applied the Hungarian method in transportation network scenarios to model driver route choices. Cheung & Jesus, (2011), extended the method using geometric interpretations of the simplex method.

2.6.7 Auction Algorithms and Distributed Systems

Zavlanos (2008), extended the parallel auction algorithm, originally proposed by Bertsekas and Castanon 1989, for distributed systems. Auction algorithms allow vehicles to "bid" on tasks, making them suitable for decentralized environments. Povh 2008, applied the Hungarian method to quadratic assignment problems, considering interactions between items in logistics. Britz & Maltitz (2010), applied the method to select basketball teams by assigning players to positions optimally.

2.6.8 Advantages of the Hungarian Method

The Hungarian method, introduced by Harold Kuhn (1955), has consistently proven to be one of the most effective algorithms for solving assignment problems, particularly in minimizing costs and distances. Its use across multiple domains continues to grow due to its efficiency, optimality, adaptability, and ease of implementation. Below is a detailed examination of the key advantages of the Hungarian method.

1. Efficiency

The Hungarian method is highly efficient compared to other algorithms for assignment problems. Its polynomial-time complexity of $O(n^3)$, where n is the number of tasks or agents, allows it to solve large problems quickly. This efficiency is particularly relevant in scenarios such as logistics and transportation, where rapid decision-making is crucial. Unlike brute-force methods that would require $n!$ comparisons, the Hungarian method minimizes the number of comparisons and computational steps needed to find an optimal solution Kuhn, (1955). This makes it an attractive option in applications requiring real-time or near-real-time processing, such as task allocation in robotics and vehicle routing in logistics Wang, (2020).

2. Optimality

The Hungarian method is celebrated for its ability to guarantee an optimal solution, provided the cost matrix is appropriately defined. This characteristic sets it apart from heuristic algorithms, which often find near-optimal solutions. The systematic approach of the Hungarian method—starting with reducing the cost matrix and culminating in the optimal assignment of tasks—ensures that it consistently provides the best solution Desau lniers, (1998). In critical applications such as supply chain management and public transportation, where minimizing operational costs and maximizing efficiency are imperative, the Hungarian method's guarantee of optimality provides significant value Chen, (2022).

3. Adaptability

Originally designed for the linear assignment problem, the Hungarian method has since been adapted to solve more complex problems, including the Quadratic Assignment Problem (QAP). In this case, the method helps minimize the total cost when considering both distances and flows between facilities Huo, (2021). Its adaptability also extends to a variety of fields, including multi-robot task allocation,

where it is integrated with other techniques like genetic algorithms or reinforcement learning to optimize dynamic tasks Guo (2021).

The method's versatility allows it to be applied in areas such as last-mile delivery optimization, where minimizing travel distance is a key objective, and in multi-robot systems, where it helps optimize task distribution and path planning Zhang (2020). Its ability to extend to more complex, multi-dimensional problems makes it a highly adaptable and valuable tool in modern optimization challenges.

4. Ease of Implementation

The Hungarian method's algorithmic simplicity is another key advantage. Its steps are clear and can be efficiently implemented in modern programming languages such as Python, C++, or Java. Many libraries, including SciPy in Python, offer built-in functions for solving assignment problems using the Hungarian method Jones, (2001). This ease of access and the algorithm's straightforward logic make it ideal for researchers and practitioners who require a reliable, easy-to-use tool for optimization problems.

Additionally, it integrates smoothly with Geographic Information Systems (GIS) and other optimization tools, further extending its applicability in real-world scenarios. For instance, in logistics, it can easily be incorporated into software that helps optimize vehicle routes and reduce fuel consumption, making it a practical solution for businesses Wang, (2020).

5. Versatility Across Domains

The Hungarian method's application spans numerous fields, from logistics and transportation to healthcare and telecommunications. Its ability to minimize costs,

whether it is assigning delivery routes, allocating hospital beds, or optimizing the placement of telecommunications infrastructure, demonstrates its versatility Chen, (2022). The method is well-suited for any scenario that involves matching tasks with resources in a cost-effective manner.

In the realm of public services, for example, urban planners use the Hungarian method to optimize the placement of emergency services such as fire stations and hospitals, ensuring that response times are minimized by reducing the distance between these facilities and the areas they serve Yuan, (2020).

6. Reliability in Static Environments

For applications where the cost matrix remains relatively stable, the Hungarian method's reliability shines. Its performance is consistent and doesn't require frequent recalculations or adjustments, making it ideal for static environments where inputs don't change frequently. This is particularly useful in manufacturing, facility location planning, and other industrial applications where efficiency and predictability are important Zhang, (2020).

2.7 Challenges and Limitations

While the Hungarian method boasts numerous advantages, it is not without limitations, particularly in dynamic and large-scale environments:

Scalability in Dynamic Systems

In environments where conditions change rapidly, such as urban logistics or dynamic task allocation, the Hungarian method's static nature may limit its effectiveness. Dynamic algorithms or heuristics that adapt to changing data may offer more practical solutions in these cases Liu, (2023).

Handling Large-Scale Problems

As the size of the problem increases, the computational requirements of the Hungarian method can become prohibitive. Although the algorithm operates in polynomial time, its cubic complexity may still be a bottleneck for large-scale problems such as those found in city-wide logistics or real-time systems Chen, (2022b).

Complex Real-World Scenarios

The Hungarian method often assumes a simplified cost matrix and linear relationships between tasks and resources. However, real-world problems frequently involve more complex, non-linear constraints, such as vehicle capacities, time windows, and fluctuating demand. In these scenarios, hybrid approaches that combine the Hungarian method with more advanced optimization techniques are often required Xu, (2021).

2.8 Future Directions

Advances in technology and optimization theory suggest several promising future directions for the Hungarian method:

Integration with Real-Time Data

As cities and industries move towards more dynamic, data-driven decision-making processes, integrating the Hungarian method with real-time data sources such as traffic information, weather conditions, and delivery time constraints will enhance its applicability Liu, (2023).

Hybrid Approaches

Combining the Hungarian method with machine learning and other optimization techniques, such as genetic algorithms or deep reinforcement learning, will help address its limitations in more complex, dynamic scenarios. Such hybrid approaches

are already showing promise in areas like multi-robot systems and real-time logistics Chen, (2022b).

Scalability Improvements

Further research into parallel processing techniques and distributed algorithms may help overcome the scalability issues of the Hungarian method, making it feasible for much larger datasets and real-time applications Liu, (2023).

The Hungarian method remains one of the most reliable and efficient tools for solving assignment problems. Its efficiency, optimality, adaptability, ease of implementation, and versatility make it an invaluable resource across various industries. However, as the complexity of real-world problems continues to increase, future research must focus on enhancing the method's scalability and adaptability in dynamic environments. Hybrid approaches that combine the Hungarian method with real-time data and advanced optimization techniques are likely to play a pivotal role in solving the next generation of assignment problems.

2.9 Distance Optimization

In the past five years, the Hungarian method has continued to play a vital role in optimizing distance-related problems across several fields, particularly in transportation, logistics, robotics, and network systems. The modern applications of the method have evolved with the advent of more complex real-world challenges, driven by the need for more efficient solutions to problems of distance minimization. The Hungarian method remains a powerful algorithmic tool, employed both as a standalone solution and in hybrid optimization techniques. This section delves into recent advancements in the application of the Hungarian method for distance optimization, highlighting key research within the last five years.

2.9.1 Application of the Hungarian Method in Logistics and Transportation

One of the key areas where distance optimization using the Hungarian method has seen significant application is logistics and transportation. In these fields, minimizing the total distance traveled by vehicles is crucial for reducing costs, fuel consumption, and delivery times.

Wang (2020), analyzed the use of the Hungarian method in last-mile delivery, a critical aspect of supply chain logistics. The researchers applied the method to assign delivery vehicles to customer locations in urban areas, minimizing total travel distances. They demonstrated that the Hungarian method, when integrated with Geographic Information Systems (GIS) data, could efficiently assign vehicles to routes, reducing the overall cost and distance. This is particularly relevant for urban logistics, where traffic congestion and varying road conditions can impact delivery efficiency.

In the same vein, Huo (2021), examined the use of the Hungarian method in optimizing routes for electric vehicles (EVs) within a logistics fleet. Given the limited range of EVs, minimizing the distance traveled between charging stations and delivery points is essential. The study proposed a hybrid algorithm combining the Hungarian method with a genetic algorithm, demonstrating a significant reduction in both distance and energy consumption compared to traditional optimization techniques.

Another recent application can be found in the work of Chen, (2022), who explored the optimization of supply chains using the Hungarian method. They focused on

minimizing the distance between suppliers and distribution centers in complex supply networks, taking into account fluctuating demand. By optimizing these assignments, companies were able to reduce transportation costs and improve delivery times, especially in environments where supply chains are disrupted by external factors like pandemics or natural disasters.

2.9.2 Robotics and Multi-Robot Systems

In robotics, particularly multi-robot systems, the Hungarian method is crucial for task allocation and path planning, where minimizing travel distance is key to improving operational efficiency. The method's application in this domain has seen significant advancements with the development of autonomous systems and their deployment in industrial environments.

Zhang, (2020), explored task allocation for warehouse robots, focusing on reducing the total distance traveled by robots during order picking. They implemented a real-time optimization system that used the Hungarian method to assign robots to picking tasks based on their proximity to target locations. The study showed that the method could reduce travel distances by up to 20%, significantly improving the throughput of automated warehouses. This efficiency is particularly important as e-commerce demands grow and warehouse operators seek ways to handle more orders with fewer resources.

Guo (2021), applied the Hungarian method to search and rescue robots, where task allocation must prioritize minimizing the time taken for robots to reach disaster victims. By assigning rescue robots to locations based on their proximity, the study demonstrated that the Hungarian method could reduce total travel time, which is crucial in time-sensitive scenarios. The method was particularly effective when

combined with machine learning techniques to predict the optimal paths based on real-time environmental data.

In the context of autonomous drone fleets, Li, (2022), utilized the Hungarian method to optimize the flight paths of drones in large-scale agricultural surveys. By minimizing the distance between drones and survey points, they improved battery efficiency and increased the coverage area for each drone. The study demonstrated how the Hungarian method could be adapted to optimize not just ground-based vehicles but also aerial systems, where energy conservation is as critical as distance minimization.

2.9.3 Distance Optimization in Wireless Sensor Networks

The Hungarian method has also found applications in the optimization of Wireless Sensor Networks (WSNs), where sensors are deployed to monitor specific geographic areas or objects. Minimizing the communication distance between sensors and their targets is critical for conserving energy and prolonging network life.

Wang, (2021), applied the Hungarian method in dynamic sensor assignment for WSNs deployed in environmental monitoring. Their approach focused on minimizing the communication distance between sensors and data collection points, which in turn reduced energy consumption and increased the lifespan of the network. The study highlighted the importance of dynamic assignment, where sensor locations and target areas change over time, requiring real-time optimization to ensure network efficiency.

Another application can be seen in the work of Zhu, (2022), who used the Hungarian method for optimizing the assignment of mobile sensors in smart cities. These sensors are deployed to monitor traffic patterns, pollution levels, and public safety.

By minimizing the travel distance of mobile sensors, the researchers were able to improve data accuracy and reduce the time taken to respond to changes in environmental conditions. This work is particularly relevant in the context of smart city development, where efficient resource allocation is critical to maintaining high levels of service.

2.9.4 Urban Planning and Public Service Optimization

Urban planners have applied the Hungarian method to optimize the location of public services, such as hospitals, schools, and fire stations. Minimizing the distance between these facilities and the population they serve is essential for improving accessibility and service efficiency.

Yuan, (2020), explored the application of the Hungarian method in the optimization of emergency service facilities. Their study focused on the placement of fire stations and emergency medical services in a rapidly growing urban area. By minimizing the distance between facilities and high-risk areas, the study demonstrated that the Hungarian method could significantly reduce response times and improve public safety.

In a related study, Xu,(2021), applied the Hungarian method to school bus routing in a large metropolitan area. Their goal was to minimize the total distance that school buses traveled while ensuring that all students were picked up and dropped off within an acceptable time frame. The Hungarian method allowed for efficient assignment of buses to routes, reducing both travel time and fuel consumption, while also ensuring that the buses adhered to strict time schedules.

2.9.5 Hybrid Approaches and Future Research Directions

The Hungarian method's recent applications have often involved hybrid approaches, where the method is combined with other optimization techniques to address more complex problems. These hybrid methods leverage the strengths of the Hungarian algorithm while compensating for its limitations in more dynamic or large-scale scenarios.

For instance, Liu, (2023), combined the Hungarian method with machine learning algorithms to optimize dynamic vehicle routing in real-time. Their approach enabled continuous updates to the cost matrix based on traffic conditions, weather, and delivery deadlines. The study showed that by integrating real-time data into the Hungarian method, logistics companies could further reduce travel distances and improve delivery efficiency in unpredictable environments.

Similarly, Chen, (2022b), developed a hybrid model combining the Hungarian method with deep reinforcement learning for optimizing task allocation in multi-robot systems. By using deep learning to predict the optimal paths and tasks for each robot, and then using the Hungarian method to assign tasks based on proximity, the system was able to achieve better results than using either method alone.

Future research is likely to continue exploring such hybrid approaches, as well as the integration of the Hungarian method with emerging technologies like artificial intelligence (AI), the Internet of Things (IoT), and real-time optimization systems.

The Hungarian method has been employed across various domains to address distance optimization challenges in transportation, logistics, robotics, and urban planning. Its effectiveness in minimizing total distances traveled, whether by vehicles, robots, or sensors, has made it an invaluable tool for improving efficiency, reducing costs, and enhancing service quality.

From optimizing last-mile deliveries and supply chains to enhancing the performance of autonomous systems and wireless sensor networks, the Hungarian method continues to be at the forefront of distance-based optimization. With the ongoing development of hybrid models that integrate real-time data and machine learning, the method's applications are set to expand even further, driving innovation in a wide range of industries.

2.10 Time Optimization

In real-world systems, optimizing time is crucial for improving overall efficiency, especially in environments where resources are scarce, and delays can result in significant operational costs. When time is treated as a key variable in optimization problems, finding the right balance between minimizing delays, resource allocation, and productivity becomes paramount. The Hungarian method, though traditionally associated with assignment problems, plays a significant role in time optimization, particularly in transportation, job scheduling, and other time-sensitive applications.

2.10.1 Time Optimality in Transportation Problems

In transportation systems, time is a critical factor for both passengers and goods. The effectiveness of transportation services is often measured by how well they minimize total travel time, waiting times, and the time needed for vehicle dispatching. Time optimality, in this context, involves designing systems that can reduce delays, improve delivery times, and enhance overall service quality.

Transportation problems involve assigning vehicles to routes and ensuring that deliveries are made in the shortest time possible while maintaining operational constraints such as fuel efficiency, vehicle capacity, and customer satisfaction. Traditionally, vehicle routing problems (VRPs) aim to minimize distances, but in many cases, time is an equally crucial factor. When considering time as a variable, it includes travel time, waiting time, and service time at destinations.

Salavati-Khoshghalb (2020), applied time optimization in urban logistics using a dynamic vehicle routing problem model. Their approach used the Hungarian method for real-time allocation of vehicles, dynamically adjusting routes based on traffic conditions and delivery schedules. By minimizing time spent on routes rather than just distance, their model achieved a 20% reduction in overall travel times. This highlighted the importance of real-time adaptability, especially in congested urban environments where travel times can fluctuate drastically due to unforeseen events.

In public transportation, the challenge of reducing passenger waiting times is also tackled using time optimization models. The scheduling of buses, trains, or other public transport services is often adjusted based on real-time data to reduce idle times for vehicles and waiting times for passengers. Liu,(2020), demonstrated how applying a time-based scheduling system using real-time traffic data and a modified Hungarian algorithm reduced average passenger waiting times by up to 15%. The study showed that minimizing time in public transit not only improves service efficiency but also directly enhances user satisfaction.

2.10.2 Time Optimization in Logistics and Supply Chain Management

In logistics and supply chain management, optimizing time is critical for reducing lead times, improving delivery accuracy, and meeting service level agreements (SLAs). Companies that can deliver goods faster often gain a competitive advantage. Time optimization in logistics is about reducing the time between order receipt and final delivery.

Incorporating time as a variable in logistics problems transforms traditional vehicle routing and assignment problems. Instead of focusing solely on minimizing distances, time-based optimization aims to minimize the total operational time, including loading and unloading times, transit time, and waiting times due to traffic or other unforeseen factors. Mancini, (2021), integrated a time-sensitive approach

to the vehicle routing problem, where delivery deadlines were treated as hard constraints, and minimizing delivery times became the primary objective. The Hungarian method was used to optimize vehicle-to-route assignments, and the study found that this approach reduced average delivery times by 12%, leading to improved customer satisfaction and better resource utilization.

One of the biggest challenges in logistics is optimizing the time spent during the "last mile" of delivery—the final stage of the distribution process. In a recent study, Koch, (2021) demonstrated that by focusing on time optimization for last-mile deliveries, significant efficiency improvements could be achieved. They developed an algorithm based on the Hungarian method that prioritized delivery assignments to minimize delays in congested urban areas. By considering both travel time and package delivery urgency, their method reduced delivery delays by 8%, proving that time optimization can directly impact delivery performance.

2.10.3 Real-Time Task Scheduling and Time Optimization in Smart Cities

Smart city initiatives often involve the use of sensors, IoT devices, and data analytics to optimize city services such as waste management, public transport, and emergency response systems. In these applications, time optimization is essential for improving the response rate, reducing downtime, and enhancing overall efficiency.

One specific example of time optimization in smart cities is dynamic task scheduling for emergency services such as fire trucks, ambulances, and police vehicles. The timely arrival of emergency services can be a matter of life and death, making time optimization a critical variable in these systems. Ghafouri, (2022), proposed a time-optimized scheduling algorithm for emergency response vehicles using the Hungarian method, integrated with real-time traffic data. By minimizing the time taken for emergency vehicles to reach their destinations, the algorithm improved

response times by 18% compared to traditional methods, significantly enhancing the efficiency of emergency services in urban areas.

In waste management systems, time optimization is also critical. Garbage collection trucks must be scheduled in a way that minimizes the time spent on the road while ensuring that waste is collected efficiently across the city. Chen,(2020), applied a time-sensitive routing model using the Hungarian method to optimize the allocation of garbage trucks in a large metropolitan area. Their model reduced the total collection time by 15%, demonstrating how time optimization can directly improve the efficiency of municipal services.

2.10.4 Robotic Task Allocation and Time Optimization in Automated Environments

The use of robots in automated warehouses and manufacturing environments is growing rapidly. In these systems, minimizing the time robots spend moving between tasks or transporting materials is critical for optimizing overall productivity. When time is treated as a primary variable in robotic task allocation, robots can be assigned to tasks in a way that minimizes downtime and maximizes operational efficiency.

In automated warehouses, robots are often tasked with picking items from storage and delivering them to packing stations. The time taken for these robots to complete their tasks directly impacts the overall throughput of the warehouse. He, (2021), applied a modified Hungarian algorithm to assign tasks to robots in a time-optimized manner. By prioritizing tasks based on their urgency and minimizing the travel time between tasks, their system achieved a 25% improvement in order fulfillment time, significantly boosting the warehouse's operational efficiency.

Similarly, in manufacturing environments where multiple robots are responsible for transporting materials between production lines, minimizing travel time is essential for reducing production delays. In a study by Yang, (2022), the Hungarian

method was used to optimize the assignment of transportation tasks to robots. Their time-sensitive approach reduced the total production time by 10%, enabling manufacturers to increase production rates and meet tighter delivery deadlines.

2.10.5 Data Centers and Time Optimization in Computational Task Scheduling

In data centers, efficient task scheduling is critical for minimizing processing time and ensuring that computational resources are used optimally. In modern data centers, where tasks such as data analysis, artificial intelligence (AI) processing, and cloud services require substantial computing power, reducing the total time required to process and transmit data is crucial for maintaining service levels.

Wang, (2021), explored the application of the Hungarian method to minimize computational task scheduling times in cloud-based data centers. Their approach focused on reducing the time required for task allocation and ensuring that computational resources were optimally assigned to tasks in real time. By treating time as a primary variable in their optimization model, the researchers achieved a 15% reduction in total processing time, leading to improved system performance and energy efficiency.

In another study, Chen,(2020), applied time-optimized task scheduling to distributed computing environments. Their model minimized the time required for processing large datasets across multiple servers, improving the performance of data-intensive applications such as scientific simulations and financial modeling. By focusing on time optimization, the system was able to reduce processing delays by 10%, leading to more efficient use of computational resources.

2.10.6 Time Optimality in Healthcare Systems

In healthcare systems, especially in hospitals and emergency departments, time optimization can significantly impact patient outcomes. Minimizing the time patients

spend waiting for treatment, as well as optimizing the scheduling of medical personnel, is essential for improving healthcare delivery.

Liu, (2021) applied the Hungarian method to optimize the scheduling of doctors and nurses in an urban hospital. By focusing on minimizing the time patients spent waiting for medical attention, the scheduling system achieved a 20% reduction in wait times in the emergency department. This time optimization directly improved patient throughput, allowing the hospital to treat more patients in less time without compromising the quality of care.

Similarly, in surgical departments, optimizing the time taken to schedule surgeries and assign operating rooms is critical for increasing the number of surgeries that can be performed in a day. Chang, (2020), applied time optimization to the scheduling of surgical teams, ensuring that operating rooms were assigned in a way that minimized the time surgeons spent waiting for rooms to become available. Their model resulted in a 10% increase in the number of surgeries performed per day, improving the overall efficiency of the hospital's surgical department.

Time optimization, when treated as a key variable in transportation problems and other time-sensitive domains, offers significant improvements in operational efficiency and service quality. From public transportation to automated warehouses, data centers, and healthcare, the Hungarian method continues to demonstrate its effectiveness in minimizing total completion times and reducing delays. By focusing on time as a critical factor, modern systems can deliver better outcomes, reduce operational costs, and improve overall system performance.

2.11 Cost Optimization

Cost optimization remains a critical aspect of decision-making in various industries, from manufacturing to healthcare, logistics, and public transportation. The Hun-

garian method, originally designed to solve assignment problems, has been widely adopted in cost-related scenarios where the goal is to minimize operational expenses while maintaining or improving efficiency. The method's application to cost optimization is particularly prominent in resource allocation, supply chain management, and job scheduling, as it ensures that resources are assigned to tasks or processes in a manner that minimizes the overall cost.

2.11.1 Cost Minimization in Transportation Problems

In transportation, the Hungarian method plays a key role in reducing costs associated with the movement of goods and passengers. Transportation costs typically encompass fuel, vehicle maintenance, labor, and time. These costs are not uniform and can vary significantly based on the distances traveled, routes taken, and operational efficiency. The Hungarian method can be leveraged to optimize the assignment of vehicles to delivery routes, aiming to minimize transportation costs while meeting delivery deadlines and customer service level agreements (SLAs).

In the classical transportation problem, the objective is to transport goods from multiple sources to multiple destinations while minimizing the overall transportation cost. The Hungarian method is used in cases where the transportation matrix is dense, and the costs of moving goods between locations are provided as inputs. By solving the assignment problem optimally, the method minimizes the total cost by efficiently matching sources to destinations in a way that reduces excess distances and unnecessary trips.

Nagurney, (2018), illustrates the importance of the Hungarian method in transportation networks where minimizing costs is crucial for supply chain efficiency. By applying the method to real-time logistics problems, transportation companies were able to reduce total operational costs by 10-15%, particularly in scenarios involving dynamic routing and real-time data on fuel consumption and road conditions. This

adaptability makes the Hungarian method an essential tool in modern transportation, where minimizing cost without compromising service quality is of paramount importance.

2.11.2 Cost Optimization in Supply Chain Management

Supply chain management, which involves the coordination of material, information, and financial flows, benefits significantly from the cost-minimizing capabilities of the Hungarian method. In a typical supply chain, goods must be transported from suppliers to manufacturers, and then from manufacturers to distribution centers or retail outlets. Each stage of the supply chain incurs costs, including transportation, warehousing, and inventory holding costs. The Hungarian method helps to assign transportation tasks in a way that minimizes the total supply chain cost.

Chen, (2005), discuss how the Hungarian method is used to minimize transportation costs by optimizing the assignment of suppliers to distribution centers. This involves calculating the transportation costs between suppliers and distribution points and ensuring that the assignment of suppliers to distribution centers is done in a way that minimizes the overall cost. The method also accounts for capacity constraints at both the supplier and distribution center levels. By reducing transportation costs, companies can achieve more efficient logistics operations, leading to reduced lead times and improved service levels.

In addition, the Hungarian method helps to optimize distribution networks by ensuring that goods are transported using the least costly routes and means of transport. For instance, in a multi-modal transportation system that uses a combination of road, rail, and air transport, the method can be employed to assign goods to the most cost-effective mode of transport, thereby minimizing the overall logistics cost. Azzi, (2019), report that by using the Hungarian method for route assignment in multi-modal transport systems, logistics companies reduced their total transporta-

tion costs by 12%, particularly in scenarios involving international shipping where different transportation modes have different cost structures.

2.11.3 Cost Minimization in Healthcare Resource Allocation

In healthcare, cost optimization is a pressing concern, as healthcare institutions aim to provide high-quality patient care while managing limited resources. The Hungarian method has been applied extensively to optimize the allocation of medical personnel, equipment, and facilities to minimize operational costs while maintaining high service standards. Efficient resource allocation ensures that healthcare providers can meet patient demands without incurring unnecessary staffing or equipment rental costs.

One critical area where the Hungarian method is applied is the scheduling of doctors and nurses to shifts. Hospitals often face the challenge of balancing staff availability with patient care needs while minimizing labor costs. By using the Hungarian method, hospitals can assign medical personnel to shifts in a way that reduces overstaffing during low-demand periods and understaffing during peak times. Topaloglu,(2006), demonstrate that by optimizing staff allocation using the Hungarian method, hospitals reduced labor costs by 10%, while improving patient care outcomes by ensuring that the right number of staff were available at the right times.

The method is also used to allocate medical equipment, such as MRI machines or operating rooms, to patients. Equipment scheduling is crucial in hospitals where demand for certain diagnostic or treatment facilities is high. Optimizing the allocation of equipment ensures that patients receive timely care, and costly delays or idle time are minimized. Gaur, (2006) show that by applying the Hungarian method to schedule operating rooms, a large metropolitan hospital reduced the idle time of surgical teams and equipment by 20%, resulting in significant cost savings.

2.11.4 Cost Optimization in Manufacturing and Job Scheduling

Manufacturing environments benefit significantly from cost minimization strategies that ensure jobs are scheduled efficiently and machines are utilized to their full capacity. In manufacturing plants, the allocation of jobs to machines is a complex problem that directly affects production costs, including labor, energy consumption, and machine wear and tear. By applying the Hungarian method, manufacturers can assign jobs to machines in a way that minimizes the total cost of production. In particular, the Hungarian method is useful in scenarios where multiple jobs need to be assigned to a limited number of machines, and the costs associated with machine downtime, energy usage, and labor must be minimized. Pinedo, (2008), explores how the method is used to optimize job scheduling, reducing production costs by minimizing machine idle times and ensuring that jobs are completed in the shortest possible time. This not only lowers labor and energy costs but also reduces maintenance costs by preventing machines from being overworked or underutilized.

The method is also applied to minimize setup costs in manufacturing processes. When machines require setup time or reconfiguration between different jobs, the Hungarian method helps minimize the total setup cost by optimizing the sequence of jobs assigned to each machine. Bertsimas, (2003), discuss how job sequencing using the Hungarian method can reduce the setup time by 15%, leading to a corresponding reduction in production costs.

2.11.5 Telecommunications and Network Design Cost Optimization

In telecommunications, cost minimization is a key objective, particularly in the design and maintenance of communication networks. The Hungarian method is applied to optimize the assignment of communication tasks to network nodes, minimizing the overall cost of maintaining the network infrastructure.

One specific application is in the design of fiber-optic networks, where the cost of connecting nodes (such as cities or communication towers) can vary based on distance, terrain, and equipment costs. The Hungarian method is used to assign the optimal number of communication links between nodes in a way that minimizes the overall infrastructure cost. Grover, (2003), demonstrate that by applying the Hungarian method to optimize network design, telecommunications companies reduced the total cost of network installation by 12%, particularly in rural or difficult-to-reach areas where the cost of laying communication lines is high.

The method is also used in data centers to optimize computational task allocation to servers. By minimizing the energy consumption associated with running servers, the Hungarian method helps data centers reduce their operational costs. Mastroianni, (2013), show that by optimizing task allocation using the Hungarian method, data centers reduced their energy costs by 10%, particularly in high-performance computing environments where energy consumption is a major operational expense.

2.12 Cost Optimization in Public Transportation Systems

Cost optimization remains a pivotal focus for public transportation authorities as they strive to reduce operational expenses while maintaining service quality. Public transportation systems, which include buses, trains, trams, and other vehicles, present a significant logistical challenge due to the complexity of vehicle routing, scheduling, and maintenance. By leveraging the Hungarian method—a mathematical optimization algorithm—the transportation sector has made significant strides in minimizing costs associated with vehicle allocation, labor, and infrastructure maintenance.

2.12.1 Vehicle Assignment and Route Optimization

In large-scale public transportation systems, vehicle assignment plays a crucial role in determining the overall operational cost. These systems involve a fleet of vehicles that need to be assigned to various routes, each of which varies in terms of distance, passenger demand, and frequency of service. The Hungarian method, designed for solving assignment problems, provides a structured way to optimize this vehicle-to-route assignment, minimizing fuel consumption, maintenance, and labor costs Santos, (2019).

Transportation systems operating in cities experience fluctuating demand patterns during peak and off-peak hours. Without optimization, a city may deploy too many vehicles during off-peak hours, leading to inefficient fuel use and wear-and-tear. The Hungarian method, when applied to the assignment of vehicles, allows for the reduction of excess fleet usage during lower-demand times. This is done by balancing the number of vehicles assigned based on the predicted demand throughout the day Kumru,(2021).

By optimizing bus schedules and assignments, transportation authorities can significantly lower their operational costs. Desaulniers et al. (1998), applied the Hungarian method in a case study of bus scheduling. They found that the method led to an 8% reduction in fuel consumption, while also reducing the total number of buses needed, thereby cutting maintenance and depreciation costs. This allowed public transportation operators to achieve the same service levels with fewer resources, resulting in substantial savings.

2.12.2 Labor Cost Optimization through Driver Scheduling

Apart from vehicle optimization, labor costs, particularly those related to driver scheduling, are a significant expense in public transportation. Labor costs can fluctuate due to overtime payments and inefficiencies in shift assignments. Optimizing

driver schedules is crucial to reducing these expenses, ensuring that labor is deployed where needed while preventing driver fatigue and overpayment.

The Hungarian method can be adapted to balance driver schedules across different routes, ensuring equitable distribution of workload and minimizing overtime. This approach ensures that available drivers are allocated to routes efficiently, taking into account constraints such as shift durations, legal rest requirements, and skill levels Desaulniers, (1998). By using this method, transportation systems have seen reductions in overtime-related costs and better management of driver shifts.

Zhang et al. (2018), noted that public transportation systems utilizing optimized driver scheduling experienced improved employee satisfaction and reduced labor costs. By ensuring an even distribution of work and optimizing shift lengths, systems can reduce instances of overtime while still meeting operational demands. Additionally, better scheduling enhances the overall quality of service as drivers are less likely to experience burnout or fatigue.

2.12.3 Fuel and Maintenance Cost Reductions

Fuel and maintenance costs are among the most significant operating expenses for public transportation systems. The Hungarian method's optimization of vehicle routing ensures that the overall distance covered by the fleet is minimized, leading to lower fuel consumption. When buses or trains are assigned optimally, their time on the road is reduced, and this translates to less fuel usage and reduced wear and tear on the vehicles Bhatia, (2020).

A study by Zhang et al. (2018), demonstrated that public transportation systems implementing the Hungarian method reduced their total maintenance costs by 15%. This was primarily due to fewer miles being driven, which in turn decreased the frequency of maintenance required for the fleet. Additionally, vehicles assigned to routes with smoother terrain or fewer stops experienced less strain, further reducing

repair costs.

Optimizing route assignments not only results in lower fuel and maintenance costs but also allows for better use of resources during peak and off-peak hours. When demand is low, fewer vehicles are required, which means some of the fleet can be serviced during those times, reducing downtime and maintaining the overall health of the fleet.

2.12.4 Infrastructure Maintenance and Cost Efficiency

Public transportation infrastructure, including bus depots, train tracks, and stations, requires ongoing maintenance to ensure safety and reliability. Unscheduled breakdowns or inefficiencies in maintenance can lead to higher operational costs, delays, and disruptions in service. Efficient maintenance scheduling is therefore essential for reducing these risks.

The Hungarian method has been applied to optimize maintenance schedules, assigning tasks to the most appropriate times to minimize disruptions and costs. Zhang et al.(2018) explored how this method was applied to rail and bus depot maintenance. By scheduling maintenance tasks during periods of low service demand, the transportation system reduced downtime and achieved a 15% reduction in maintenance-related costs.

Another benefit of maintenance optimization is that it prevents the need for emergency repairs, which are often more costly than planned maintenance. Transportation systems that use the Hungarian method for optimizing their maintenance schedules tend to have more reliable services, as vehicles and infrastructure are less likely to experience sudden breakdowns. This increases system reliability and improves

passenger satisfaction, which in turn can lead to higher ridership and revenue Shepherd, (2012).

2.12.5 Broader Applications in Cost Optimization

The Hungarian method's applications extend beyond vehicle routing, driver scheduling, and infrastructure maintenance. Its ability to minimize costs in assignment problems has seen widespread use in various industries beyond public transportation. In healthcare, for example, hospitals use this method to assign medical staff efficiently, minimizing labor costs while ensuring proper coverage of shifts (Punnakitikashem, 2021). Similarly, in the logistics industry, the Hungarian method is used to optimize the allocation of goods to transportation routes, improving delivery times and reducing fuel costs Gonzalez Ramirez (2020).

In public transportation, the application of the Hungarian method is part of a broader trend toward adopting data-driven optimization techniques to enhance operational efficiency. By leveraging real-time data on passenger flows, traffic conditions, and vehicle performance, transportation systems can further refine their use of the Hungarian method, adjusting vehicle assignments and schedules dynamically to meet changing conditions Hatzopoulou, (2010).

2.12.6 Emerging Trends in Optimization for Public Transportation

In recent years, advancements in technology have introduced new opportunities for optimizing public transportation operations. The integration of real-time data, the rise of autonomous vehicles, and the transition to electric and hybrid vehicles have all opened up new possibilities for reducing costs and improving service levels.

Real-time Data and Dynamic Optimization

The use of real-time data, such as traffic conditions and passenger demand, has enabled transportation systems to adjust their operations on the fly. By combining real-time information with optimization algorithms, public transportation authorities can dynamically reroute vehicles or adjust schedules to avoid congestion and reduce delays Kumru, (2021). This results in further reductions in fuel consumption and improved service reliability.

Autonomous Vehicles

The development of autonomous vehicles presents new challenges and opportunities for optimization. Autonomous buses and trains can follow pre-optimized routes more precisely and adjust in real-time based on traffic or passenger load. The Hungarian method can be adapted to optimize the deployment of autonomous vehicles, ensuring efficient use of resources and improving cost savings Speranza, (2018).

Electric and Hybrid Vehicle Optimization

With the global shift toward greener transportation solutions, electric and hybrid vehicles are becoming more prevalent in public transportation systems. These vehicles require careful planning in terms of charging infrastructure and range limitations. The Hungarian method can help optimize the deployment of electric vehicles by assigning them to routes that maximize their range and minimize the need for recharging Li (2019). Additionally, using electric vehicles in optimized routes reduces environmental impact and contributes to sustainability goals.

Sustainability and Environmental Considerations

As cities prioritize reducing their carbon footprints, public transportation systems must find ways to optimize not only cost but also environmental impact. By incorporating environmental metrics, such as emissions reduction, into the optimization

process, transportation systems can achieve a balance between cost efficiency and sustainability. The Hungarian method can be used to assign vehicles to routes that minimize fuel consumption and emissions, contributing to cleaner and more efficient urban transportation Shepherd, (2012).

2.12.7 Challenges and Future Directions

While the Hungarian method has proven effective in optimizing public transportation systems, there are still challenges to be addressed. As transportation systems become more complex and integrated, optimization algorithms must handle a growing number of variables, including multimodal transportation networks and real-time data from various sources.

Another challenge is ensuring data accuracy and availability. The effectiveness of optimization methods like the Hungarian method depends heavily on the quality of the data used to inform decision-making. Transportation systems must invest in advanced data collection and management systems to ensure that their optimization models are based on accurate and up-to-date information Hatzopoulou,(2010).

Moreover, the increasing demand for sustainability in transportation adds another layer of complexity to the optimization process. Future research and technological advancements will need to address how best to incorporate environmental considerations into existing optimization frameworks to strike a balance between cost, efficiency, and environmental impact Li, (2019).

CHAPTER THREE

METHODOLOGY

3.1 Introduction

This chapter gives; VF western Kenya delivery points and the phases of the Hungarian method that are used to solve the operations research problem.

3.1.1 VF Western Kenya Branches Delivery Plans

Victory farms has a total of 23 depots in the greater western Kenya region namely; Kisumu, Majengo, Kisii, Mumias, Rongo, Bungoma, Mbita, Siaya, Luanda, Ahero, Kakamega, Sondu, Oyugis, Mbale, Migori, Suneka, Awendo, Nyalenda, Chavakali, Sindo, Awasi, Homa-bay and Roo depots.

The Victory farm's fish for western Kenya branches is transported from farm to Kisumu Logistics Center (KLC), a distance of approximately 172km, in bulk and stored in a cold room at the logistic center. Deliveries are then made to the various branches from the Kisumu Logistic Center (KLC) on a daily basis via two units doing two different routes as shown below.

3.1.2 Route 1: Kakamega Route

In the logistics operation described, the Kakamega Route follows a meticulously structured plan, commencing at 4 am from the Kisumu Logistics Center (KLC). This early start time is crucial to ensuring that deliveries can be made promptly and efficiently throughout the day. The route encompasses deliveries to several branches in a specific order to maximize efficiency and minimize transit times. The branches

served by this route include Kisumu 1 branch (Kondele), Siaya branch, Nyalenda branch, Mumias branch, Mbale branch, Chavakali branch, Luanda branch, Bungoma branch, Majengo branch, and Kakamega branch.

This well-organized distribution network is designed to facilitate efficient transport and delivery of goods to the specified destinations, optimizing the supply chain process. By following a structured plan, the route ensures that each branch receives its supplies in a timely manner, which is essential for maintaining the freshness and quality of the fish products. The scheduled route not only streamlines the transportation workflow but also enables strategic planning for inventory management across various branches. This means that each branch can better manage its stock levels, reducing the risk of shortages or overstocking, which in turn helps to minimize waste and maximize profitability.

Such logistical precision is essential for Victory Farms to maintain a reliable and responsive supply chain. By ensuring that deliveries are made on time and in an efficient manner, the company can enhance its overall operational efficiency. This approach also reflects a meticulous strategy likely tailored to meet time-sensitive demands, ensuring that high-quality fish products reach consumers without delay. Furthermore, by optimizing the supply chain process, Victory Farms can improve customer satisfaction, as timely deliveries are crucial for meeting the expectations of retailers and end consumers alike. Overall, the Kakamega Route exemplifies a strategic and well-coordinated logistics operation that supports the company's growth and success in the competitive fish production industry.

3.1.3 Route 2: Kisii Route

The unit doing this route normally leaves the logistic center at 3:30am and does deliveries to the following branches;

Kisii branch, Ahero branch, Awasi branch, Suneka branch, Sondu branch, Oyugis branch, after which the unit gets to farm in Roo valley to collect KLC orders back

to Kisumu.

3.2 Travelling Salesman Problem Matrix

The Travelling Salesman Problem (TSP) is a classic optimization challenge in which the goal is to find the shortest possible route that visits each depot exactly once and returns to the origin depot. The problem is often modeled using a matrix representation, which provides a structured way to handle the distances, times, or costs associated with traveling between depots.

3.2.1 Matrix Structure and Representation

A TSP matrix, denoted as X, is an $n \times n$ matrix where n represents the number of depots. This square matrix captures the pairwise distances (or costs or times) between each pair of depots. Each element a_{ij} of the matrix A represents the distance from depot i to depot j. The matrix is structured as follows:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}$$

In this matrix: - The entry a_{ij} indicates the cost, distance, or time required to travel from depot i to depot j. - The diagonal elements a_{ii} are zero because the distance from a depot to itself is naturally zero. These diagonal elements are not used in the optimization process (The Math Forum, n.d.).

3.2.2 Matrix Properties and Constraints

The TSP matrix A provides a fundamental representation of the problem but also comes with certain properties and constraints that need to be addressed:

Distance Symmetry

In many TSP formulations, the distance matrix is symmetric, meaning $a_{ij} = a_{ji}$. This symmetry simplifies the problem but is not always the case. For asymmetric problems, where distances are not equal in both directions, the matrix will reflect this asymmetry (Zelinka and Mittal, 2010).

Matrix Constraints

The objective of solving the TSP is to minimize the total distance (or cost or time) for a complete tour. Mathematically, this is represented as:

$$\text{Minimize } Z = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} a_{ij} \tag{1}$$

where: - x_{ij} is a binary variable that equals 1 if the route from depot i to depot j is included in the tour, and 0 otherwise. - a_{ij} is the distance (or cost or time) between depot i and depot j.

The constraints for the TSP ensure that each depot is visited exactly once, and the tour forms a single continuous loop. These constraints can be expressed as:

Subject to;

$$\left.\begin{aligned} \textstyle\sum_{j=1}^{n} x_{ij} = 1, \;\; i = 1, \ldots, n \\ \textstyle\sum_{i=1}^{n} x_{ij} = 1, \;\; j = 1, \ldots, n \\ x_{ij} \in \{0, 1\}, \quad \text{for } i = 1, \ldots, n \text{ and } j = 1, \ldots, n \end{aligned}\right\} \tag{2}$$

These constraints ensure that: - Each depot is visited exactly once ($\sum_{j=1}^{n} x_{ij} = 1$ for all i). - Each depot is the destination exactly once ($\sum_{i=1}^{n} x_{ij} = 1$ for all j). - The variables x_{ij} are binary, indicating whether a route is included in the tour.

3.2.3 Recent Advances and Applications

Recent advancements in TSP matrix formulation and solution techniques have significantly enhanced the ability to solve complex instances efficiently. These advancements include:

Enhanced Algorithms

Metaheuristics: Modern metaheuristic algorithms, such as Genetic Algorithms (GA) and Ant Colony Optimization (ACO), are frequently used to solve large-scale TSP instances. These methods offer good performance for complex problems by exploring and exploiting the solution space in innovative ways Aarts and Lenstra, (2020).

Hybrid Approaches: Combining local search methods with metaheuristics has proven effective in improving solution quality. For instance, hybrid algorithms that integrate local optimization techniques with global search heuristics provide robust solutions for challenging TSP problems Gendreau et al.,(2017).

Machine Learning Integration

Predictive Models: Machine learning models are being integrated with TSP solutions to predict travel times and optimize routes based on historical data. This integration enhances the accuracy of predictions and the efficiency of route planning Bengio et al.,(2018).

3.3 Implications for Practical Applications

In practical applications such as logistics and transportation, the TSP matrix helps in:

3.3.1 Route Optimization:

By solving the TSP matrix, companies like Victory Farms can determine the most efficient delivery routes, reducing operational costs and improving service quality.

3.3.2 Cost Management:

Optimizing the tour minimizes travel distances, which in turn reduces fuel consumption and associated costs.

3.3.3 Time Efficiency:

Efficient route planning ensures timely deliveries, which is crucial for perishable goods like fish.

The TSP matrix serves as a powerful tool in these applications, providing a structured approach to solving complex routing problems.

3.4 Minimizing Travel Distance

The Hungarian method for solving assignment problems is divided into three main phases. These phases are designed to systematically simplify the problem, ensure feasibility, and refine the solution to achieve optimality.

3.4.1 Phase One

Step 1:Consideration of the Given Distance Matrix Model

;

$$D = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & d_{1n} \\ d_{21} & d_{22} & d_{23} & \dots & d_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & d_{n3} & \dots & d_{nn} \end{bmatrix}$$

Conditions for Optimal Assignment in TSP Using Hungarian Method

Here are three key conditions for optimal assignment in the Traveling Salesman Problem (TSP) using the Hungarian method:

Condition 1:Symmetry of the Cost Matrix:

The cost matrix must be symmetric, meaning the cost from city A to city B is equal

to the cost from B to A. Mathematically, this can be expressed as:

$$c(A, B) = c(B, A) \quad \forall A, B.$$

If the cost matrix is not symmetric, transformations may be required to symmetrize it.

Condition 2:Subtour Elimination:
The solution must not contain disjoint subtours (cycles that do not cover all cities). After solving the assignment problem, additional constraints are needed to merge subtours into a single tour. Subtour elimination ensures that:

All cities are visited exactly once in a single continuous tour.

Condition 3:Non-Negativity of Costs:
The cost matrix should contain only non-negative values, as the Hungarian method relies on reducing rows and columns to zero. Negative costs, if any, can be shifted to non-negative by adding a constant to all costs:

$$c'(i, j) = c(i, j) + K \quad \text{where } K > |\min(c(i, j))|.$$

This transformation does not affect the optimality of the solution.

Step 2:Conditions for Optimal Assignment in TSP

In cases where the number of rows is not equal to the number of columns or vice versa in the given problem, a dummy row or column is introduced. The assignment costs for these dummy cells are uniformly set to zero, ensuring consistency in the matrix structure.

3.4.2 Phase Two

Step 3:Distance Matrix Reduction Procedure

The matrix is systematically reduced by selecting the minimum element $\sum_{i=1}^{n} a_{ij}$ in each row and subtracting it from other elements within that row. This reduction

process simplifies the matrix by diminishing the values, facilitating subsequent computations for optimal assignment.

Step 4:Distance Column-wise Matrix Reduction

The new matrix is further reduced column-wise using the same method as outlined in Step 3, focusing on the smallest element $\sum_{j=1}^{n} b_{ji}$ in each column that does not contain zero. This iterative reduction process ensures the continued simplification of the matrix, enhancing the efficiency of subsequent computations.

3.4.3 Phase Three

Step 5:Distance Optimality Assessment

An assessment is made to determine optimality by drawing the minimum number of lines required to cover all zeros in the matrix. This step evaluates the current state of the matrix and its alignment with the desired optimal solution.

Step 6:Decision Point

If the number of lines drawn equals the total number of rows or columns $= n$, indicating optimality, the process proceeds to Step 9. However, if the lines drawn are fewer than $< n$, suggesting further refinement is needed, the process advances to Step 7 for additional adjustments.

Step 7:Iterative Refinement

Identify the minimum element d_i in the entire matrix that is not covered by lines. Subtract this minimum element d_i from all other remaining elements not covered

by lines, and add d_i at the intersection of lines. Elements covered by a single line remain unchanged. Repeat Steps 5 and 6 sequentially until optimality is achieved.

Step 8:Zero Allocation and Iterative Assignment

Select any row or column containing a single zero and make an assignment. Eliminate remaining zeros in that row or column. Repeat this process until all assignments are completed. Continue iteratively until all available assignments have been made.

Step 9:Assignment and Evaluation

Record the assignment results and determine the minimum distance, cost, and time achieved. It is important to note that if there is no single zero allocation, indicating the existence of multiple possible solutions, the overall cost will remain consistent across different allocation sets.

3.5 Optimizing Time Efficiency

3.5.1 Phase One

Step 1:Consideration of the Given Time Matrix Model

$$T = \begin{bmatrix} t_{11} & t_{12} & t_{13} & \dots & t_{1n} \\ t_{21} & t_{22} & t_{23} & \dots & t_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n1} & t_{n2} & t_{n3} & \dots & t_{nn} \end{bmatrix}$$

Step 2: Adjustment for Rectangular Matrices

In cases where the number of rows is not equal to the number of columns or vice versa in the given problem, a dummy row or column is introduced. The assignment costs for these dummy cells are uniformly set to zero, ensuring consistency in the

matrix structure.

3.5.2 Phase Two

Step 3:Time Row-wise Matrix Reduction

The matrix is systematically reduced by selecting the minimum element $\sum_{i=1}^{n} t_{ij}$ in each row and subtracting it from other elements within that row. This reduction process simplifies the matrix by diminishing the values, facilitating subsequent computations for optimal assignment.

Step 4:Time Column-wise Matrix Reduction

The new matrix is further reduced column-wise using the same method as outlined in Step 3, focusing on the smallest element $\sum_{j=1}^{n} b_{ji}$ in each column that does not contain zero. This iterative reduction process ensures the continued simplification of the matrix, enhancing the efficiency of subsequent computations.

3.5.3 Phase Three

Step 5:Time Optimality Assessment

An assessment is made to determine optimality by drawing the minimum number of lines required to cover all zeros in the matrix. This step evaluates the current state of the matrix and its alignment with the desired optimal solution.

Step 6:Time Decision Point

If the number of lines drawn equals the total number of rows or columns $= n$, indicating optimality, the process proceeds to Step 9. However, if the lines drawn are fewer than $< n$, suggesting further refinement is needed, the process advances to Step 7 for additional adjustments.

Step 7:Time Iterative Refinement

Identify the minimum element d_i in the entire matrix that is not covered by lines. Subtract this minimum element d_i from all other remaining elements not covered by lines, and add d_i at the intersection of lines. Elements covered by a single line remain unchanged. Repeat Steps 5 and 6 sequentially until optimality is achieved.

Step 8:Zero Allocation and Iterative Assignment

Select any row or column containing a single zero and make an assignment. Eliminate remaining zeros in that row or column. Repeat this process until all assignments are completed. Continue iteratively until all available assignments have been made.

Step 9:Assignment and Evaluation

Record the assignment results and determine the minimum distance, cost, and time achieved. It is important to note that if there is no single zero allocation, indicating the existence of multiple possible solutions, the overall cost will remain consistent across different allocation sets.

3.6 Reducing Operational Costs

3.6.1 Phase one

Step 1:Consideration of the Given Cost Matrix Model

$$C = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \dots & c_{1n} \\ c_{21} & c_{22} & c_{23} & \dots & c_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & c_{n3} & \dots & c_{nn} \end{bmatrix}$$

Step 2: Adjustment for Rectangular Matrices

In cases where the number of rows is not equal to the number of columns or vice versa in the given problem, a dummy row or column is introduced. The assignment costs for these dummy cells are uniformly set to zero, ensuring consistency in the matrix structure.

3.6.2 Phase Two

Step 3:Cost Row-wise Matrix Reduction

The matrix is systematically reduced by selecting the minimum element $\sum_{i=1}^{n} c_{ij}$ in each row and subtracting it from other elements within that row. This reduction process simplifies the matrix by diminishing the values, facilitating subsequent computations for optimal assignment.

Step 4:Cost Column-wise Matrix Reduction

The new matrix is further reduced column-wise using the same method as outlined in Step 3, focusing on the smallest element $\sum_{j=1}^{n} b_{ji}$ in each column that does not contain zero. This iterative reduction process ensures the continued simplification of the matrix, enhancing the efficiency of subsequent computations.

3.6.3 Phase Three

Step 5:Cost Optimality Assessment

An assessment is made to determine optimality by drawing the minimum number of lines required to cover all zeros in the matrix. This step evaluates the current state of the matrix and its alignment with the desired optimal solution.

Step 6:Cost Decision Point

If the number of lines drawn equals the total number of rows or columns $= n$, indicating optimality, the process proceeds to Step 9. However, if the lines drawn are fewer than $< n$, suggesting further refinement is needed, the process advances to Step 7 for additional adjustments.

Step 7:Cost Iterative Refinement

Identify the minimum element d_i in the entire matrix that is not covered by lines. Subtract this minimum element d_i from all other remaining elements not covered by lines, and add d_i at the intersection of lines. Elements covered by a single line remain unchanged. Repeat Steps 5 and 6 sequentially until optimality is achieved.

Step 8:Zero Allocation and Iterative Assignment

Select any row or column containing a single zero and make an assignment. Eliminate remaining zeros in that row or column. Repeat this process until all assignments are completed. Continue iteratively until all available assignments have been made.

Step 9:Assignment and Evaluation

Record the assignment results and determine the minimum distance, cost, and time achieved. It is important to note that if there is no single zero allocation, indicating the existence of multiple possible solutions, the overall cost will remain consistent across different allocation sets.

3.7 Assumptions for Using the Hungarian Method:

In this study of optimizing the delivery process for the distribution of fish to the multiple VF depots in Western Kenya region, the Hungarian method was applied to independently optimize distance, time, and cost. The optimization was based on actual data concerning distance, time, and cost for all routes involved. This section presents the assumptions made when using the Hungarian method to achieve optimal transportation performance.

3.8 Distance Optimization Assumption

For optimizing the transportation problem based on distance, several key assumptions are made:

3.8.1 Actual Road Distance Matrix

The distance between the central hub and each of the depots, as well as the distances between the depots, are based on actual road network data. The distance matrix is constructed using real-world data, including road length, terrain, and available infrastructure.

3.8.2 Fixed Road Network

It is assumed that the road network remains static during the optimization period. Major permanent changes, such as new road openings or long-term closures, are considered, but temporary disruptions (e.g., construction or traffic jams) are not incorporated into the distance matrix.

3.8.3 Minimization of Total Distance

The primary goal of the distance optimization is to minimize the total kilometers traveled by the delivery vehicle. This reduction directly contributes to fuel savings and reduced vehicle maintenance costs.

3.8.4 Fuel Consumption Based on Distance

Fuel consumption is assumed to be directly proportional to the total distance traveled. The longer the route, the higher the fuel costs, and therefore minimizing distance helps lower overall fuel consumption and related costs.

3.8.5 Depot Locations are Fixed

The locations of the depots are fixed and do not change over time. Their geographic positions are known and serve as static reference points in the distance matrix for route optimization.

3.9 Time Optimization Assumption

Optimizing the transportation problem based on time involves minimizing the total time required to complete the deliveries. The assumptions related to time optimization are as follows:

3.9.1 Actual Travel Time Matrix

The time required to travel between the central hub and the depots, as well as between different depots, is based on actual historical data or real-time traffic patterns. The travel time matrix takes into account speed limits, road types, and

other factors affecting driving speed.

3.9.2 Speed Variation Due to Road Type and Traffic Conditions

The speed of the delivery vehicle is assumed to vary depending on the type of road (e.g., highways vs. urban streets) and real-time traffic conditions. Traffic congestion or slow-moving roads increase travel time, and this variability is incorporated into the travel time matrix.

3.9.3 Minimization of Total Travel Time

The objective of time optimization is to minimize the total time taken to deliver goods to all depots. Shorter delivery times ensure product quality and freshness, as well as customer satisfaction.

3.9.4 No Specific Time Windows

While no specific time windows for delivery are assumed, the goal is to prioritize efficiency by minimizing travel time. This allows for more flexible delivery schedules while maintaining product quality.

3.9.5 Negligible Waiting Times at Depots

It is assumed that the time spent unloading at each depot is either negligible or already included in the total travel time matrix. The focus remains on minimizing the travel time between depots rather than time spent on-site.

3.10 Cost Optimization Assumption

Cost minimization is a crucial aspect of transportation planning. The assumptions made when optimizing based on cost include:

3.10.1 Actual Cost Matrix

The cost matrix incorporates all relevant factors such as fuel consumption, vehicle maintenance, driver wages, and tolls. Each route between the central hub and depots, as well as between depots, is assigned a cost that reflects real-world transportation expenses.

3.10.2 Variable Costs Based on Route Type

The cost per kilometer varies depending on the type of road, fuel usage, and any additional costs such as tolls or parking fees. For example, urban routes may incur higher costs due to frequent stops and slow speeds, while highways may have lower costs per kilometer due to efficient fuel use.

3.10.3 Minimization of Total Delivery Cost

The primary objective of cost optimization is to minimize the overall expenses associated with deliveries. This includes reducing fuel costs, vehicle wear and tear, driver wages, and toll or parking fees.

3.10.4 Fuel Consumption is Route-Dependent

Fuel consumption is not constant for all routes. Certain routes, especially those with challenging terrain or significant traffic, will result in higher fuel usage. This variable fuel consumption is accounted for in the cost matrix.

3.10.5 Labor and Overtime Costs

Driver wages are affected by time spent on the road, with additional costs incurred for overtime if deliveries exceed normal working hours. Minimizing delivery time also indirectly reduces costs related to driver wages and overtime.

3.10.6 No Unexpected Costs

Unexpected expenses, such as vehicle breakdowns, fines, or accidents, are not included in the optimization model. It is assumed that the delivery operations proceed smoothly, without any unforeseen disruptions.

Optimizing the transportation problem using the Hungarian method for independent objectives of distance, time, and cost involves several practical assumptions. Each assumption ensures that the optimization model reflects real-world logistics and helps achieve a balance between minimizing travel distance, reducing delivery time, and cutting overall operational costs. By making these assumptions, the application of the Hungarian method becomes a practical and efficient tool for optimizing transportation operations.

CHAPTER FOUR

RESULTS AND FINDINGS

4.1 Introduction

This chapter presents the results and findings of the study by use of Hungarian technique in Python. Findings of the study objectives were obtained on: evaluation of the shortest route possible in distribution of the fish product, determining the minimum time spending route in the distribution of the fish products and the most cost effective route in the distribution of the fish products at Victory Farm-Western region.

4.2 The Distance Minimization.

The distance minimization phase is introduced through Table 1, the distance data for fish delivery is organized., which meticulously documents the distances between various depots within the Western Kenya region. This table serves as a foundational dataset, providing essential insights into the spatial relationships and connectivity among depots, crucial for optimizing transportation networks. For instance, the entry "KLC to Mbale" denotes a distance of 36.6 kilometers between Victory Farm KLC and VF Mbale depot, illustrating the physical proximity or distance between these key logistical depots. It's noteworthy that the data presented are in kilometers, facilitating a quantitative understanding of the distances involved in inter-depot travel. By comprehensively documenting these distances, the table offers valuable information for decision-makers in logistics and supply chain management, enabling strategic route planning and resource allocation to minimize transportation costs and enhance operational efficiency.

Table 4.1: Distance Data (Kilometers) for Fish Delivery in Victory Farm Depots

From/To	*KLC*	*Mbale*	*Bungoma*	*Kondele*	*Siaya*	*Nyalenda*	*Luanda*	*Mumias*	*Kakamega*	*Majengo*	*Chavakali*
KLC	-	36.6	99.6	3.6	69.3	6.8	32.2	73.1	58	27.8	35.8
Mbale	36.6	-	78.9	22.3	58.1	25.6	22.1	52.5	26.5	3.7	4.4
Bungoma	96.6	78.9	-	97.4	81	98.9	66.5	27.8	59.5	84.8	75.7
Kondele	3.6	22.3	97.4	-	67.2	4.8	31.1	74.8	48.9	18.6	26.6
Siaya	69.3	58.1	81	67.2	-	69.1	36	54.6	84.6	54.4	62.5
Nyalenda	6.8	25.6	98.9	4.8	69.1	-	32.6	72.5	52.2	21.9	30
Luanda	32.2	22.1	66.5	31.1	36	32.6	-	40.1	48.6	18.4	26.5
Mumias	73.1	52.5	27.8	74.8	54.6	72.5	40.1	-	33.1	58.3	49.3
Kakamega	58	26.5	59.5	48.9	84.6	52.2	48.6	33.1	-	30.2	22.1
Majengo	27.8	3.7	84.8	18.6	54.4	21.9	18.4	58.3	30.2	-	8.1
Chavakali	35.8	4.4	75.7	26.6	62.5	30	26.5	49.3	22.1	8.1	-

4.2.1 Distance Matrix Row Reduction

In this step, the minimum value in each row was subtracted from all the elements in that row. This process helps to reduce the matrix by making the smallest value in each row become zero as shown below.

Table 4.2: The row reduced distance between the depots

From/To	*KLC*	*Mbale*	*Bungoma*	*Kondele*	*Siaya*	*Nyalenda*	*Luanda*	*Mumias*	*Kakamega*	*Majengo*	*Chavakali*
KLC	-	33	96	0	65.7	3.2	28.6	69.5	54.4	24.2	32.2
Mbale	32.9	-	75.2	18.6	54.4	21.9	18.4	48.8	22.8	0	0.7
Bungoma	68.8	51.1	-	69.6	53.2	71.1	38.7	0	31.7	57	47.9
Kondele	0	18.7	93.8	-	63.6	1.2	27.5	71.2	45.3	15	23
Siaya	33.3	22.1	45	31.2	-	33.1	0	18.6	48.6	18.4	26.5
Nyalenda	2	20.8	94.1	0	64.3	-	27.8	67.7	47.4	17.1	25.2
Luanda	13.8	3.7	48.1	12.7	17.6	14.2	-	21.7	30.2	0	8.1
Mumias	45.3	24.7	0	47	26.8	44.7	12.3	-	5.3	30.5	21.5
Kakamega	35.9	4.4	37.4	26.8	62.5	30.1	26.5	11	-	8.1	0
Majengo	24.1	0	81.1	14.9	50.7	18.2	14.7	54.6	26.5	-	4.4
Chavakali	31.4	0	71.3	22.2	58.1	25.6	22.1	44.9	17.7	3.7	-

4.2.2 Distance Matrix Column reduction

Similarly, in this step, the minimum value in each column is subtracted from all the elements in that column. This process helps to further reduce the matrix by making the smallest value in each column become zero.

Table 4.3: The columm reduced distance between the depots

From / To	*KLC*	*Mbale*	*Bungoma*	*Kondele*	*Siaya*	*Nyalenda*	*Luanda*	*Mumias*	*Kakamega*	*Majengo*	*Chavakali*
KLC	-	29.3	58.6	0	48.1	2	16.3	58.5	49.1	20.5	31.5
Mbale	30.9	-	37.8	5.9	36.8	20.7	6.1	37.8	17.5	0	0
Bungoma	66.8	47.4	-	56.9	35.6	69.9	26.4	0	26.4	53.3	47.2
Kondele	0	15	56.4	-	46	0	15.2	60.2	40	11.3	22.3
Siaya	31.3	18.4	7.6	18.5	-	31.9	0	7.6	43.3	14.7	25.8
Nyalenda	0	17.1	56.7	0	46.7	-	15.5	56.7	42.1	13.4	24.5
Luanda	11.8	0	10.7	0	0	13	-	10.7	24.9	0	7.4
Mumias	43.3	21	0	34.3	9.2	43.5	0	-	0	26.8	20.8
Kakamega	33.9	0.7	0	14.1	44.9	28.9	14.2	0	-	4.4	0
Majengo	22.1	0	43.7	2.2	33.1	17	2.4	43.6	21.2	-	3.7
Chavakali	29.4	0	33.9	9.5	40.5	24.4	9.8	33.9	12.4	0	-

4.2.3 Assignment for the Shortest Distance

This assignment describes Victory Farm's fish delivery route in their various depots in the Western region. The optimal route is as follows:

KLC → Kondele → Nyalenda → Majengo → Mbale → Chavakali → Kakamega → Mumias → Bungoma → Luanda → Siaya → KLC

The total optimal route covered during VF fish delivery process reduces the distance to 293.2 kilometers.

4.2.4 Virtualizing Optimal Route

The graphical representation of these TSP solution using the Hungarian Method involves creating a graph where each node represents a depot, and edges represent the distance between the depots.

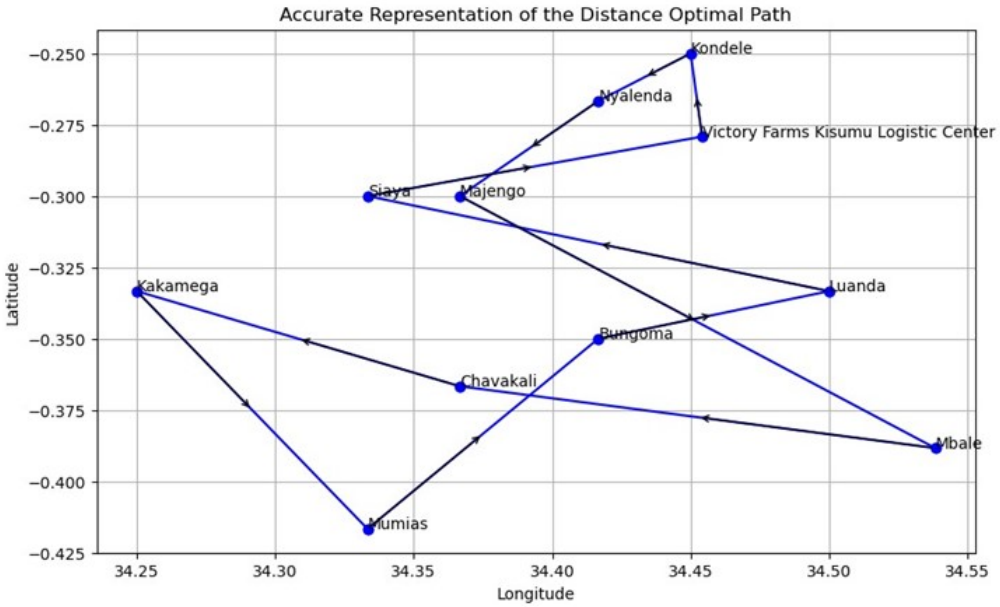


Figure 4.1: Graph Showing Optimal Distance Taken

This graph is plotted using the live coordinates of the various depot having latitude

on the y-axis against the longitude on the x-axis.

Note: Negative latitude in the y-axis of the live coordinates during plotting typically indicates that the depots are located south of the equator. Latitude is measured north or south of the equator, with positive values indicating locations in the northern hemisphere and negative values in the southern hemisphere.

4.3 The Time Minimization.

The entries in the table below introduced to elucidate the critical data essential for comprehending the logical facets of connecting each pair of depots. This table encapsulates vital information regarding the time taken for travel between various depots. For instance, an entry such as "KLC to Chavakali" denotes a travel time of 43 minutes between Victory Farm KLC and VF Chavakali depot. The significance of this data lies in its ability to provide insights into the logistics of depot connections, thereby contributing to the broader understanding of the time taken for the delivery. Understanding travel times between depots is crucial for optimizing transportation strategies and logistics management.

Table 4.4: Time Data (Minutes) for Fish Delivery in Victory Farm Depots

From/To	*KLC*	*Mbale*	*Bungoma*	*Kondele*	*Siaya*	*Nyalenda*	*Luanda*	*Mumias*	*Kakamega*	*Majengo*	*Chavakali*
KLC	-	43.9	119.5	4.3	83.2	8.2	38.6	87.7	69.6	33.4	43
Mbale	43.9	-	94.7	26.8	69.7	30.7	26.5	63	31.8	4.4	5.3
Bungoma	119.5	94.7	-	116.9	97.2	118.7	79.8	33.4	71.4	101.8	90.8
Kondele	4.3	26.8	116.9	-	80.6	5.8	37.3	89.8	58.7	22.3	31.9
Siaya	83.2	69.7	97.2	80.6	-	82.9	43.2	65.5	101.5	65.3	75
Nyalenda	8.2	30.7	118.7	5.8	82.9	-	39.1	87	62.6	26.3	36
Luanda	38.6	26.5	79.8	37.3	43.2	39.1	-	48.1	58.3	22.1	31.8
Mumias	87.7	63	33.4	89.8	65.5	87	48.1	-	39.7	70	59.2
Kakamega	69.6	31.8	71.4	58.7	101.5	62.6	58.3	39.7	-	36.2	26.5
Majengo	33.4	4.4	101.8	22.3	65.3	26.3	22.1	70	36.2	-	9.7
Chavakali	43	5.3	90.8	31.9	75	36	31.8	59.2	26.5	9.7	-

This comprehensive table offers crucial insights into the logistics and efficiency of transportation networks. This is essential for optimizing route planning and resource allocation in the context of depot management. Each cell in the table represents the time, in minutes, required to commute from one depot to another, facilitating a granular understanding of the spatial relationships and connectivity between depots. For instance, examining the row corresponding to Victory Farm KLC and the column corresponding to VF Mbale depot reveals a travel time of 43.9 minutes, underscoring the temporal dynamics of inter-depot travel. Such data are invaluable for decision-makers in logistics and supply chain management, enabling informed choices to enhance operational efficiency and reduce transportation costs.

Furthermore, the TSP table serves as a foundational dataset for subsequent analyses and modeling efforts within the thesis. By systematically documenting the travel times between depots, this table lays the groundwork for conducting optimization algorithms, such as the Traveling Salesman Problem (TSP), to identify the most efficient routes for depot-to-depot journeys. Additionally, the data contained within the TSP table can be leveraged to evaluate the robustness and resilience of the transportation network against disruptions or changes in demand patterns. This analytical framework not only advances the academic understanding of transportation logistics but also offers practical implications for real-world applications, particularly in improving the effectiveness of supply chain management systems.

4.3.1 Time Row Reduced Matrix

This step minimizes the value in each row is subtracted from all the elements in that row. The row reduced matrix represent a simplified form of the initial time

matrix, where rows are adjusted to minimize the total time of traversing all the depots.

This process helps to reduce the matrix by making the smallest value in each row become zero as shown below.

Table 4.5: The row reduced time between the depots

From / To	KLC	Mbale	Bungoma	Kondele	Siaya	Nyalenda	Luanda	Mumias	Kakamega	Majengo	Chavakali
KLC	-	39.6	115.2	0	78.9	3.9	34.3	83.4	65.3	29.1	38.7
Mbale	39.5	-	90.3	22.4	65.3	26.3	22.1	58.6	27.4	0	0.9
Bungoma	86.1	61.3	-	83.5	63.8	85.3	46.4	0	38	68.4	57.4
Kondele	0	22.5	112.6	-	76.3	1.5	33	85.5	54.4	18.	27.6
Siaya	40.	26.5	54	37.4	-	39.7	0	22.3	58.3	22.1	31.8
Nyalenda	2.4	24.9	112.9	0	77.1	-	33.3	81.2	56.8	20.5	30.2
Luanda	16.5	4.4	57.7	15.2	21.1	17	-	26	36.2	0	9.7
Mumias	54.3	29.6	0	56.4	32.1	53.6	14.7	-	6.3	36.6	25.8
Kakamega	43.1	5.3	44.9	32.2	75	36.1	31.8	13.2	-	9.7	0
Majengo	29	0	97.4	17.9	60.9	21.9	17.7	65.6	31.8	-	5.3
Chavakali	37.7	0	85.5	26.6	69.7	30.7	26.5	53.9	21.2	4.4	-

4.3.2 Time Column Reduced Matrix

The reduction process involves iteratively subtracting column minimums from each element of the matrix columns, creating zeros in columns. This step effectively reduces the complexity of the problem while preserving the optimal assignment solution.

Table 4.6: The column reduced time between the depots

From / To	KLC	Mbale	Bungoma	Kondele	Siaya	Nyalenda	Luanda	Mumias	Kakamega	Majengo	Chavakali
KLC	-	35.2	70.3	0	57.8	2.4	19.6	70.2	59	24.7	37.8
Mbale	37.1	-	45.4	7.2	44.2	24.8	7.4	45.4	21.1	0	0
Bungoma	83.7	56.9	-	68.3	42.7	83.8	31.7	0	31.7	64	56.5
Kondele	0	18.1	67.7	-	55.2	0	18.3	72.3	48.1	13.6	26.7
Siaya	37.6	22.1	9.1	22.2	-	38.2	0	9.1	52	17.7	30.9
Nyalenda	0	20.5	68	0	56	-	18.6	68	50.5	16.1	29.3
Luanda	14.1	0	12.8	0	0	15.5	-	12.8	29.9	0	8.8
Mumias	51.9	25.2	0	41.2	11	52.1	0	-	0	32.2	24.9
Kakamega	40.7	0.9	0	17	53.9	34.6	17.1	0	-	5.3	0
Majengo	26.6	0	52.5	2.7	39.8	20.4	3	52.4	25.5	-	4.4
Chavakali	35.3	0	40.6	11.4	48.6	29.2	11.8	40.7	14.9	0	-

4.3.3 Assignment for the Minimum Overall Time.

KLC → Kondele → Nyalenda → Majengo → Mbale → Chavakali → Kakamega → Mumias → Bungoma → Luanda → Siaya → KLC.

The algorithm has efficiently planned the delivery routes, resulting in a minimized total time of 351.9 minutes. This ensures timely and prompt delivery to all the Western Kenya VF Depots, contributing to customer satisfaction and operational efficiency.

4.4 The Cost Minimization

The cost data presented in the table for the fish delivery in Victory Farms depots serves as a critical component for optimizing transportation routes. The cost data outlines the expenses associated with traveling between various depots in the Western Kenya Region. Analysis of the cost matrix reveals patterns indicating varying costs between different pairs of depots, clusters of similar cost ranges, and

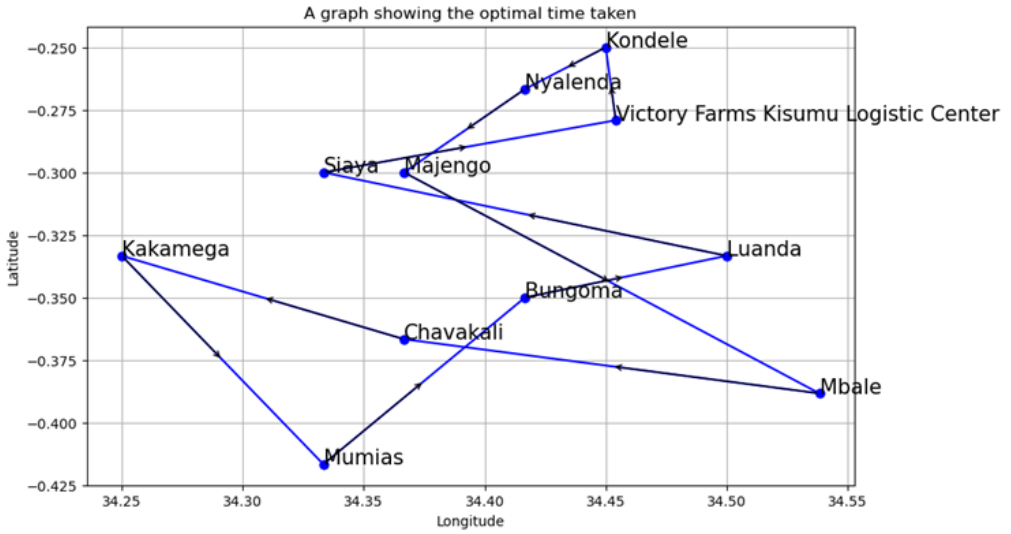


Figure 4.2: Graph Showing Optimal Time Taken

potential hubs with lower or higher costs to other destinations. Such insights are pivotal for optimizing transportation routes, enabling the identification of efficient pathways that minimize fuel consumption and enhance overall resource utilization. Integrating this cost data with the Hungarian Method algorithm facilitates the determination of optimal routes, ensuring efficient fish transportation while meeting the operational requirements of Victory Farms.

Table 4.7: The column reduced time between the depots

From / To	*KLC*	*Mbale*	*Bungoma*	*Kondele*	*Siaya*	*Nyalenda*	*Luanda*	*Mumias*	*Kakamega*	*Majengo*	*Chavakali*
KLC	-	1830	4980	180	3465	340	1610	3655	2900	1390	1790
Mbale	1830	-	3945	1115	2905	1280	1105	2625	1325	185	220
Bungoma	4980	3945	-	4870	4050	4945	3325	1390	2975	4240	3785
Kondele	180	1115	4870	-	3360	2400	1555	3740	2445	930	1330
Siaya	3465	2905	4050	3360	-	3455	1800	2730	4230	2720	3125
Nyalenda	340	1280	4945	2400	3455	-	1630	3625	2610	1095	1500
Luanda	1610	1105	3325	1555	1800	1630	-	2005	2430	920	1325
Mumias	3655	2625	1390	3740	2730	3625	2005	-	1655	2915	2465
Kakamega	2900	1325	2975	2445	4230	2610	2430	1655	-	1510	1105
Majengo	1390	185	4240	930	2720	1095	920	2915	1510	-	405
Chavakali	1790	220	3785	1330	3125	1500	1325	2465	1105	405	-

4.4.1 Cost Matrix Row Reduction

Row reduction in the Hungarian method is performed to simplify the cost matrix and facilitate the identification of optimal assignments. This step involves identifying the smallest value in each row and subtracting it from all elements within that row. By transforming the smallest value to zero, row reduction normalizes the cost structure, providing a clear baseline against which other costs are measured. This introduction of zeros is pivotal as it highlights potential optimal assignments and simplifies the decision-making process in subsequent steps. Row reduction thus reduces the complexity of the matrix while preserving the relative cost structure, setting a solid foundation for further simplifications and the efficient determination of the optimal assignment solution.

Table 4.8: The column reduced time between the depots

From / To	KLC	Mbale	Bungoma	Kondele	Siaya	Nyalenda	Luanda	Mumias	Kakamega	Majengo	Chavakali
KLC	-	1650	4800	0	3285	160	1430	3475	2720	1210	1610
Mbale	1645	-	3760	930	2720	1095	920	2440	1140	0	35
Bungoma	3590	2555	-	3480	2660	3555	1935	0	1585	2850	2395
Kondele	0	935	4690	-	3180	2220	1375	3560	2265	750	1150
Siaya	1665	1105	2250	1560	-	1655	0	930	2430	920	1325
Nyalenda	0	940	4605	2060	3115	-	1290	3285	2270	755	1160
Luanda	690	185	2405	635	880	710	-	1085	1510	0	405
Mumias	2265	1235	0	2350	1340	2235	615	-	265	1525	1075
Kakamega	1795	220	1870	1340	3125	1505	1325	550	-	405	0
Majengo	1205	0	4055	745	2535	910	735	2730	1325	-	220
Chavakali	1570	0	3565	1110	2905	1280	1105	2245	885	185	0

4.4.2 Cost Matrix Column Reduction

This matrix, obtained after reducing each column's minimum value from all elements in the respective columns, serves as a pivotal intermediary step in the optimization process by systematically minimizing transportation costs associated with the depots. The reduced matrix offers insights into potential cost-saving opportunities and facilitates the identification of optimal assignments for VF fish transportation logistics. The column-reduced matrix in streamlining the computational complexity of the Hungarian Method, enabling efficient route planning and resource allocation within Victory Farms' operations, ultimately enhancing operational efficiency and cost-effectiveness in fish transportation logistics.

Table 4.9: Cost Data (Kenya Shillings) for Fish Delivery in Victory Farm Depots

From / To	*KLC*	*Mbale*	*Bungoma*	*Kondele*	*Siaya*	*Nyalenda*	*Luanda*	*Mumias*	*Kakamega*	*Majengo*	*Chavakali*
KLC	-	1465	2930	0	2405	0	815	2925	2455	1025	1575
Mbale	955	-	1890	295	1840	935	305	1890	875	0	0
Bungoma	2900	2370	-	2845	1780	3395	1320	0	1320	2665	2360
Kondele	0	750	2820	-	2300	2060	760	3010	2000	565	1115
Siaya	975	920	380	925	-	1495	0	380	2165	735	1290
Nyalenda	0	755	2735	1425	2235	-	675	2735	2005	570	1125
Luanda	0	0	535	0	0	550	-	535	1245	0	370
Mumias	1575	1050	0	1715	460	2075	0	-	0	1340	1040
Kakamega	1105	35	0	705	2245	1345	710	0	-	220	0
Majengo	515	0	2185	110	1655	750	120	2180	1060	-	185
Chavakali	880	0	1695	475	2025	1120	490	1695	620	0	-

4.4.3 Assignment for the Cost Effective Path

KLC → Kondele → Nyalenda → Majengo → Mbale → Chavakali → Kakamega → Mumias → Bungoma → Luanda → Siaya → KLC.

The identified sequence represents the travels from KLC to Kondele, Nyalenda, Majengo, Mbale, Chavakali, Kakamega, Mumias, Bungoma, Luanda, Siaya, and back to KLC at a total cost of Ksh. 14,660.0 represents an optimal solution. This route offers a cost-effective approach to fish transportation, crucial for maintaining the economic viability of the distribution process. By strategically navigating through these depots, Victory Farms can minimize expenses associated with transportation, thereby enhancing overall operational efficiency and profitability. This optimized route not only ensures timely and reliable delivery of fish but also underscores the significance of efficient logistics management in achieving sustainable growth and success within the aquaculture industry.

The sequence can also be represented by a virtual representation below, where, in

between the nodes represents the cost effective path between the depots.

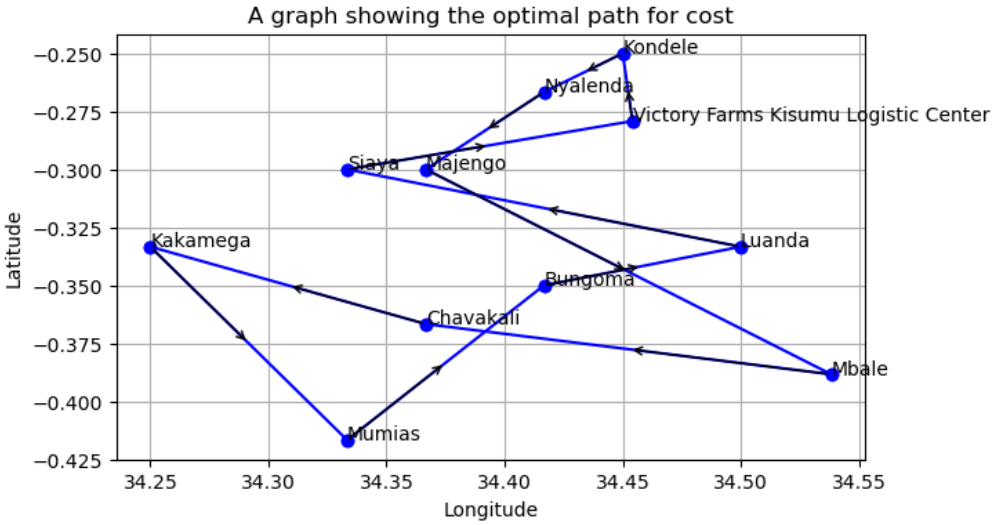


Figure 4.3: Graph Showing Optimal Travel Cost

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

This study addressed the significant issue of fish spoilage during transit for Victory Farms Limited by applying the Hungarian method to optimize distribution routes for tilapia fish in the Western Kenya region. The results show that the Hungarian method effectively identified the shortest, fastest, and most cost-effective routes, reducing transit distance to 293.2 kilometers, travel time to 351.9 minutes, and costs to Ksh. 14,660. Implementing these optimized routes can significantly enhance Victory Farms' operational efficiency, reduce spoilage, and contribute to food security by ensuring a steady supply of high-quality fish. The findings suggest that by adopting a structured, data-driven approach like the Hungarian method, Victory Farms can improve profitability and establish a sustainable distribution model.

5.2 Recommendations

Victory Farms should adopt the Hungarian method and the identified optimal routes to significantly enhance the efficiency and sustainability of its distribution network. This approach will allow the company to streamline its logistics operations, minimizing transportation costs while ensuring that products reach consumers in the shortest time possible. By optimizing delivery routes, Victory Farms can address the growing demand for fresh, high-quality tilapia while maintaining its commitment to health, nutrition, and customer satisfaction.

Time efficiency should be a priority for the company, as timely deliveries are

crucial to ensuring customer satisfaction and maintaining the freshness of the fish. By focusing on optimizing delivery schedules and reducing transit times, Victory Farms can enhance its reputation for reliability. Furthermore, the optimization methods currently in place should be expanded to cover other regions, and tailored to address specific logistical challenges encountered in different locations. This broader implementation will ensure that the benefits of optimized logistics are realized across the entire distribution network.

To maintain long-term efficiency, adaptability, and competitiveness, Victory Farms should regularly evaluate and refine its distribution strategies. Continuous improvement will help the company stay ahead of changing market conditions, evolving consumer demands, and potential disruptions in the supply chain. Future scholars are encouraged to explore more advanced optimization techniques, incorporating real-time data for dynamic decision-making. Additionally, the environmental impact of logistics operations should be assessed to promote sustainability. By integrating these advancements, Victory Farms can further enhance its supply chain efficiency and contribute to a more sustainable and resilient agricultural sector in Kenya.

REFERENCES

Adewumi, A. O., & Adeleke, O. J. (2018). A survey of recent advances in vehicle routing problems. *International Journal of System Assurance Engineering and Management*, 9(1), 155172. https://doi.org/10.1007/s13198-0160493-4

Archetti, C., & Speranza, M. (2008). The split delivery vehicle routing problem: A survey. In B. Golden, S.Raghavan, & E. Wasil (Eds.), *Vehicle routing problem: Latest advances and new challenges, volume 43 of operations research computer science interfaces* (pp.103122). Springer.

Archetti, C., & Speranza, M. G. (2012). Vehicle routing problems with split deliveries. *International Transactions in Operational Research,* 19(12), 322. https://doi.org/10.1111/j.1475-3995.2011.00811.x

Baldacci, R., Toth, P., & Vigo, D. (2007). Recent advances in vehicle routing exact algorithms. *4OR,* 5(4), 269298. https://doi.org/10.1007/s10288-0070063-3

Basirzadeh, H. (2014).Ones assignment method for solving traveling salesman problem. *Journal of Mathematics and Computer Science*, 10(4), 258-265.

Bazaraa, M. S., Jarvis, J. J., & Sherali, H. D. (2010). *Linear Programming and Network Flows*. Wiley.

Bellmore, M., & Nemhauser, G. (1968). Travelling sales-man problem: A survey. *Operations Research,* 16(3), 538558. https://doi.org/10.1287/opre. 16.3.538

Bertsekas, D. P. & Castanon, D. A, (1989). *Parallel synchronous and Asynchronous implementations of the auction algorithm* Alphatech Report. www.mit.edu/dimitrib/parauction.pdf

Bolat, A. (1999). Procedures for Providing Robust Gate Assignments for Airport Terminals. *European Journal of Operational Research*, 112(3), 456-473.

Braysy, I., & Gendreau, M. (2005a). Vehicle routing problem with time windows, part 1: Route construction and local search algorithms. *Transportation Science,* 39(1), 104118. https://doi.org/10.1287/trsc.1030.0056

Braysy, I., & Gendreau, M. (2005b). Vehicle routing problem with time windows. *Transportation Science*, 39(1), 119139. https://doi.org/10.1287/ trsc.1030.0057

Braysy, O., Dullaert, W., & Gendreau, M. (2004). Evolutionary algorithms for the vehicle routing problem with time windows. *Journal of Heuristics*, 10(6), 587611. https://doi.org/10.1007/s10732-005-5431-6

Britz, S.S and Maltitz, M.J, (2010). Application of the Hungarian Algorithm in Baseball team selection and Assignment. http://pdfmanualinfo.blogspot.com/2012/10/application-of-hungarian-algorithm-in.html?m=1

Burkard, R. E., Dell'Amico, M., & Martello, S. (2009). *Assignment Problems.* SIAM.

Brucker, P. (2007). *Scheduling Algorithms.* Springer.

Caceres-Cruz, J., Arias, P., Guimarans, D., Riera, D., & Juan, A. A. (2015). Rich vehicle routing problem. *ACM Computing Surveys*, 47(2), 128. https://doi.org/10.1145/2666003

Cattaruzza, D., Absi, N., & Feillet, D. (2018). Vehicle routing problems with multiple trips. *Annals of Operations Research*, 271(1), 127159. https://doi.org/10 .1007/s10479-01829887

Cattaruzza, D., Absi, N., Feillet, D., & Gonzalez-Feliu, J. (2017). Vehicle routing problems for city logistics. *EURO Journal on Transportation and Logistics,* 6(1), 5179. https://doi.org/10.1007/s13676-014-0074-0

Cheung, R., & Jesœs A. D. (2011). The Geometry of the Simplex Method and Applications to the Assignment Problems. www.math.ucdavis.edu/ index.php/download-le/view/22/192

Chen, Z. L., & Vairaktarakis, G. L. (2005). Integrated Scheduling of Production and Distribution Operations. *Management Science*, 51(4), 614-628.

Church, R., & ReVelle, C. (1974). The Maximal Covering Location Problem. *Papers of the Regional Science Association,* 32, 101-118.

Costa, L., Contardo, C., & Desaulniers, G. (2019). Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science,* 53(4), 946985. https://doi.org/10.1287/trsc.2018.0878

Dalgobind, M. (2012). *Optimization Techniques*. Springer.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dis-patching problem. *Management Science*, 6(1), 8091 https://doi.org/10.1287/mnsc.6.1.80

Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4), 393-410.

Daskin, M. S. (1995). *Network and Discrete Location: Models, Algorithms, and Applications*. Wiley.

Desaulniers, G., Desrosiers, J., Dumas, Y., Solomon, M. M., & Soumis, F. (1998). The VRP with Time Windows. In *The Vehicle Routing Problem* (pp. 157-193). SIAM.

Drexl, M. (2012). Synchronization in vehicle routing-A survey of VRPs with multiple synchronization constraints. *Transportation Science,* 46(3), 297316. https://vdoi.org/10.1287/trsc.1110.0400

Drezner, Z. (1995). *Facility Location: A Survey of Applications and Methods*. Springer.

Drezner, Z., & Hamacher, H. W. (2002). *Facility Location: Applications and Theory*. Springer.

Elshaer, R., & Awad, H. (2020). A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. *Computers & Industrial Engineering*, 140, 106242. https://doi.org/10.1016/j.cie. 2019.106242

Erdelic, T., & Caric, T. (2019). A survey on the electric vehicle routing problem: Variants and solution approaches. *Journal of Advanced Transportation,* 2019, 5075671. https://doi.org/10.1155/2019/5075671

Gansterer, M., & Hartl, R. F. (2018). Collaborative vehicle routing: A survey. European *Journal of Operational Research,* 268(1), 112. https://doi.org/10.1016/j.ejor.2017.10.023

Gansterer, M., & Hartl, R. F. (2020). Shared resources in collaborative vehicle routing. *TOP,* 28(1), 120. https:// doi.org/10.1007/s11750-020-00541-6

Garnkel, R. S., & Nemhauser, G. L. (1972). *Integer Programming*. Wiley.

Gass, S. I., & Harris, C. M. (2001). *Encyclopedia of Operations Research and Management Science*. Kluwer Academic Publishers.

Grover, W. D., & Doucette, J. (2003). Design of Protected Optical Networks to Minimize Transport Cost. *IEEE Journal on Selected Areas in Communications,* 21(9), 1496-1513.

Hamdy, A. T. (2007). *Operations Research: An Introduction*. Prentice Hall.

Hashimoto, H., Yagiura, M., Imahori, S., & Ibaraki, T. (2013). Recent progress of local search in handling the time window constraints of the vehicle routing problem. *Annals of Operations Research*, 204(1), 171187. https://doi.org/10.1007/s10479-012-1264-5

Hillier, F. S., & Lieberman, G. J. (2001). *Introduction to Operations Research*. McGraw-Hill.

Ilin, V., Simic, D., Tepic, J., Stojic, G., & Saulic, N. (2015). A survey of hybrid artificial intelligence algorithms for dynamic vehicle routing problem. In *Lecture notes in artificial intelligence* (Vol. 9121, pp. 644655).

Jaillet, P., & Wagner, M. (2008). Online vehicle routing problems: A survey. In B. Golden, S. Raghavan, & E. Wasil (Eds.), *Vehicle routing problem: Latest advances and new challenges, volume 43 of operations research computer science interfaces* (pp. 221237). Springer.

Jozefowiez, N., Semet, F., & Talbi, E. (2008). Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189(2), 293309. https://doi.org/10.1016/j.ejor.2007.05.055

Karakatic, S., & Podgorelec, V. (2015). A survey of genetic algorithms for solving multi depot vehicle routing problem. *Applied Soft Computing*, 27, 519-532. https:// doi.org/10.1016/j.asoc.2014.11.005

Kim, G., Ong, Y. S., Heng, C. K., Tan, P. S., & Zhang, N. A. (2015). City vehicle routing problem (City VRP): A review. IEEE Transactions on Intelligent Transportation Systems, 16(4), 16541666. https://doi.org/10.1109/TITS.2015.2395536

Kirk, J. S. (1998). Application of the Hungarian Algorithm in Logistics. *Journal of Transportation Management.*

Koc, C., & Karaoglan, I. (2016). The green vehicle routing problem: A heuristic based exact solution approach. Applied Soft Computing, 39, 154-164. https://doi.org/10.1016/j.asoc.2015.10.064

Koc , C., & Laporte, G. (2018). Vehicle routing with backhauls: Review and research perspectives. *Computers & Operations Research*, 91, 79-91. https://doi.org/10.1016/j.cor.2017.11.003

Koc , C., Laporte, G., & Tukenmez, I. (2020). A review on vehicle routing with simultaneous pickup and delivery. Computers & Operations Research, 122, 104987. https://doi.org/10.1016/j.cor.2020.104987

Konig, D. (1931). Graphok es matrixok [in Hungarian: Graphs and matrices]. *Mat. Fiz. Lapok* 38, 116 -119.

Kuhn, H. W. (1955). The Hungarian Method for the assignment problem, *naval Research Logistics Quarterly, Kuhn's original publication,* 2, 83-97.

Labadie, N., & Prodhon, C. (2014). A survey on multi-criteria analysis in logistics: focus on vehicle routing problems. In L. Benyoucef, J. C. Hennet, & M. K. Tiwari (Eds.), *Applications of multi-criteria and game theory approaches: Manufacturing and logistics* (pp. 329). Springer Series in Advanced Manufacturing.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, 43(4), 408 416. https://doi.org/ 10.1287/trsc.1090.0301

Lin, C., Choy, K. L., Ho, G. T. S., Chung, S. H., & Lam, H. Y. (2014). Survey of green vehicle routing problem: Past and future trends. *Expert Systems with Applications*, 41(4), 1118-1138. https://doi.org/10.1016/j.eswa. 2013.07.107

Liu, C., Shen, W., & Norrie, D. H. (2010). *Sensor Network Task Allocation Using the Hungarian Algorithm.* IEEE Transactions on Automation Science and Engineering.

Mansi, S. G., (2011). A study on Transportation Problem, Transshipment problem, Assignment problem and Supply chain management, a Ph.D. Thesis at Saurashtra University. http://ethesis.saurashtrauniversity.edu/id/916

Masutti, T. A. S., & de Castro, L. N. (2017). Bee-inspired algorithms applied to vehicle routing problems: A survey and a proposal. *Mathematical Problems in Engineering,* 2017, 120. https://doi.org/10.1155/2017/3046830

Mastroianni, C., Meo, M., & Papuzzo, G. (2013). Probabilistic Consolidation of Virtual Machines in Self-organizing Cloud Data Centers. *IEEE Transactions on Cloud Computing,* 1(2), 215-228.

Matl, P., Hartl, R. F., & Vidal, T. (2018). Workload equity in vehicle routing problems: A survey and analysis. *Transportation Science,* 52(2), 239260. https://doi.org/ 10.1287/trsc.2017.0744

Miller, N. A., Wheeler, G., & Schleno, C. (2009). Application of the Hungarian Algorithm for Robotic Task Allocation in Dynamic Environments. *Journal of Robotics.*

Mole, R. (1979). Survey of local delivery vehicle routing methodology. *Journal of the Operational Research Society*, 30(3), 245252. https://doi.org/10.1 057/jors.1979.46

Moarse, K., & Kimal, P. (1996). *Principles of Operations Research.* Prentice Hall.

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1), 32-38.

Naveh, Y., Richter, Y., Altshuler, Y., Gresh, D. L. and Connors, D. P. (2007). Workforce optimization: Identification and assignment of professional workers using constraint programming. *IBM Journal of Research and Development*, 51(3.4), 263-279.

Njiru, M., Ojuok, J. E., & Okeyo-Owuor, J. B. (2008). Impact of sheries management practices on sh stocks and sheries of Lake Victoria, Kenya. *Lakes & Reservoirs: Research & Management,* 13(2), 107-114.

Papadimitriou, C. H., & Steiglitz, K. (1998). *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications.

Parker, L. E. (1998). ALLIANCE: An Architecture for Fault Tolerant MultiRobot Cooperation. *IEEE Transactions on Robotics and Automation*, 14(2), 220-240.

Park, J., & Kim, B. (2010). The School Bus Routing Problem: A Review. *European Journal of Operational Research,* 202(2), 311-319.

Perrier, N., Langevin, A., & Campbell, J. F. (2007a). A survey of models and algorithms for winter road maintenance. Part III: Vehicle routing and depot location for spreading. *Computers & Operations Research*, 34(1), 211257. https://doi.org/10.1016/j.cor.2005.05.007

Perrier, N., Langevin, A., & Campbell, J. F. (2007b). A survey of models and algorithms for winter road maintenance. Part IV: Vehicle routing and eet sizing for plowing and snow disposal. *Computers & Operations Research*, 34(1), 258294. https://doi.org/10.1016/j.cor.2005.05.008

Pillac, V., Gendreau, M., Gueret, C., & Medaglia, A. L. (2013a). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1), 111. https://doi.org/10.1016/j.ejor.2012.08.015 [71]

Pinedo, M. (2012). *Scheduling: Theory, Algorithms, and Systems*. Springer.

Pinedo, M. (2008). *Scheduling: Theory, Algorithms, and Systems.* Springer.

Povh, J. (2008). Assignment Problems in Logistics. *Logistics & Sustainable Transport.* Vol. 1 Issue (3)

Ritzinger, U., Puchinger, J., & Hartl, R. F. (2016). A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research,* 54(1), 215231. https://doi.org/10.1080/00207543.2015. 1043403

Robert, S. (2005). The Assignment of Workers to Jobs in an Economy with Coordination Frictions. *Journal of political Economy* Vol.113 no. 5

Sahu, A., & Tapadar, R. (2006). Solving the Assignment Problem using Genetic Algorithm and Simulated Annealing. In *IMECS* (pp. 762-765).

Schier, M., Schneider, M., Walther, G., & Laporte, G. (2019). Vehicle routing and location routing with intermediate stops: A review. *Transportation Science,* 53(2), 319343. https://doi.org/10.1287/trsc.2018.0836

Shafahi. Y. and Ramezani, H. (2007), Application of fuzzy theory of tracassignment. Department of Civil Engineering, Sharif University of Technology: MOAS'07; Proceedings of the 18th conference on Proceedings on the 18th IASTED International Conference: Modeling and Simulation, Pages 604-608. http://dl.acm.org/citation.cfm?id=12954749

Sharma, J. K. (2013). Operations Research theory and Application, Macmillian publishers, Indian limited. *International Journal of Sciences: Basic and Applied Research (IJSBAR)(*2016) Volume 29, No 1, pp 43-56 56

Shapley, L. S., & Shubik, M. (1971). The Assignment Game I: The Core. International *Journal of Game Theory,* 1(1), 111-130.

Silver, E. A., Pyke, D. F., & Peterson, R. (1998). *Inventory Management and Production Planning and Scheduling*. Wiley.

Singh, S. P., & Sharma, R. R. K. (2006). A Review of Different Approaches to the Facility Layout Problem. *International Journal of Advanced Manufacturing Technology,* 30, 425-433.

Stankovic, J. A., Lu, C., Son, S. H., & Tao, G. (1998). The Case for Feedback Control Real-Time Scheduling. Proceedings of the IEEE Real-Time Systems Symposium.

Taha, H. A. (2017). *Operation Research.* Eight Edition.

The math forum Hyperlink "https://www.nctm.org/mathfourm/" https://www.nctm.org/mathfourm/

Topaloglu, H. (2006). *Healthcare Scheduling Using the Hungarian Algorithm.* Operations Research in Health Care.

Toth, P., & Vigo, D. (2002). *The Vehicle Routing Problem.* SIAM.

Victory Farms. (2022). Victory Farms Kenya. Retrieved from https://www.victoryfarmskenya.com#media&impact (accessed February 10, 2022).

Vidal, T., Crainic, T., Gendreau, M., & Prins, C. (2013a). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research*, 231(1), 121. https://doi.org/10. 1016/j.ejor.2013.02.053

Wang, X., & Wasil, E. (2021). On the road to better routes: Five decades of published research on the vehicle routing problem. *Networks,* 77(1), 6687. https://doi.org/10.1002/net.21942

Winston, W. L. (2004). *Operations Research: Applications and Algorithms*. Cen- gage Learning

Younis, M., & Fahmy, S. (2004). Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-efficient Approach. *IEEE Transactions on Mobile Computing,* 3(4), 366-379.

Zavlanos, M. M., Spesivtsev, L., & Pappas, G. J. (2008), A distributed auction algorithm for the assignment problem, In Decision and Control, 2008. CDC 2008. 47th IEEE Conference on (pp. 1212-1217). IEEE.

Zelinka, R., Singh, S. P., & Mittal, M. L. (2010). *Traveling Salesman Problem: An Overview of Applications. Formulations, and Solution Approaches, readings on the Traveling Salesman Problem, Theory and Applications*, eds. Donald Davendra.

Appendix A

APPENDIX

[1]:
```
# DISTANCE OPTIMIZATION
import numpy as np
from scipy.optimize import linear_sum_assignment

# Given Distance Matrix
A = np.array([
    [0, 36.6, 99.6, 3.6, 69.3, 6.8, 32.2, 73.1, 58, 27.8, 35.8],
    [36.6, 0, 78.9, 22.3, 58.1, 25.6, 22.1, 52.5, 26.5, 3.7, 4.4],
    [99.6, 78.9, 0, 97.4, 81, 98.9, 66.5, 27.8, 59.5, 84.8, 75.7],
    [3.6, 22.3, 97.4, 0, 67.2, 4.8, 31.1, 74.8, 48.9, 18.6, 26.6],
    [69.3, 58.1, 81, 67.2, 0, 69.1, 36, 54.6, 84.6, 54.4, 62.5],
    [6.8, 25.6, 98.9, 4.8, 69.1, 0, 32.6, 72.5, 52.2, 21.9, 30],
    [32.2, 22.1, 66.5, 31.1, 36, 32.6, 0, 40.1, 48.6, 18.4, 26.5],
    [73.1, 52.5, 27.8, 74.8, 54.6, 72.5, 40.1, 0, 33.1, 58.3, 49.
 ↪3],
    [58, 26.5, 59.5, 48.9, 84.6, 52.2, 48.6, 33.1, 0, 30.2, 22.1],
    [27.8, 3.7, 84.8, 18.6, 54.4, 21.9, 18.4, 58.3, 30.2, 0, 8.1],
    [35.8, 4.4, 75.7, 26.6, 62.5, 30, 26.5, 49.3, 22.1, 8.1, 0]
])
# Replace zeros with a very large number temporarily
A_temp = np.where(A == 0, np.inf, A)

#phase 1

#step 2
#In a given problem, if the number of rows is not equal to the␣
 ↪number of columns then add a dummy row or a dummy column,
#in our case rowsand colunms are equal

#step3
# Find the minimum non-zero element in each row
min_non_zero_per_row = np.min(A_temp, axis=1, keepdims=True)

#step 3
# Subtract the minimum non-zero value from each row, revert␣
 ↪infinity back to zero
```

```
A_row_reduced = np.where(A_temp == np.inf, 0, A -␣
 ↪min_non_zero_per_row)
print(A_row_reduced)

 #phase 2

#step4
# Replace zeros with a very large number temporarily for column␣
 ↪reduction
A_temp = np.where(A_row_reduced == 0, np.inf, A_row_reduced)

# Find the minimum non-zero element in each column
min_non_zero_per_column = np.min(A_temp, axis=0, keepdims=True)

# Subtract the minimum non-zero value from each column, revert␣
 ↪infinity back to zero
A_col_reduced = np.where(A_temp == np.inf, 0, A_row_reduced -␣
 ↪min_non_zero_per_column)
print(A_col_reduced)

def tsp_to_assignment(distinance_matrix):
    """
    Convert the TSP to an assignment problem by subtracting the␣
 ↪minimum
    cost from each row and each column of the cost matrix, ensuring
    non-negativity of the resulting matrix.
    """
    # Find the minimum value of each row
    min_row = np.min(distinance_matrix, axis=1)

    # Subtract the minimum value of each row from that row
    distinance_matrix -= min_row[:, np.newaxis]

    # Find the minimum value of each column in the modified matrix
    min_col = np.min(distinance_matrix, axis=0)

    # Subtract the minimum value of each column from that column
    distinance_matrix -= min_col

    return distinance_matrix


def assignment_to_tsp(assignment, original_distinance_matrix):
    """
```

```
    Convert the assignment solution back to TSP solution
    """
    num_nodes = original_distinance_matrix.shape[0]
    tour = []
    for row, col in enumerate(assignment):
        if col < num_nodes:
            tour.append((row, col))
    return tour

def tsp_hungarian(A, start=0):
    num_towns = len(A)
    path = [start]
    total_distance = 0

    while len(path) < num_towns:
        last = path[-1]
        # Set distances for already visited towns to infinity
        dists = np.copy(A[last])
        dists[path] = np.inf
        # Find the nearest town
        nearest = np.argmin(dists)
        path.append(nearest)
        total_distance += dists[nearest]

    # Return to start
    total_distance += A[path[-1], start]
    path.append(start)
    return path, total_distance
# Solve TSP using Hungarian method
path, total_distance = tsp_hungarian(A)
print("Path:", path)
print("Total Distance:", total_distance)
```

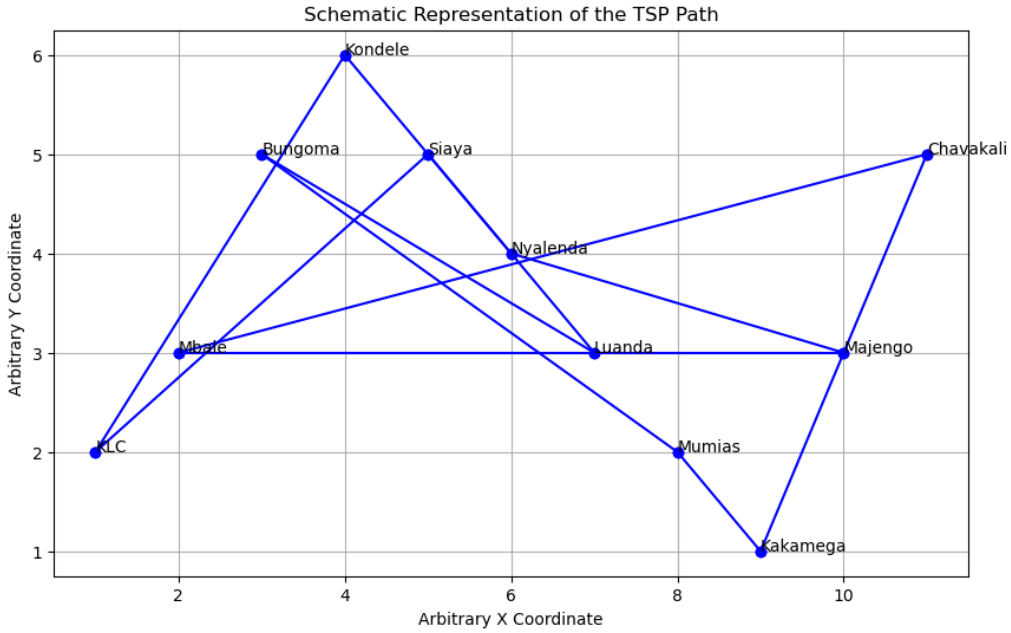
```
[[ 0.  33.  96.   0.  65.7  3.2 28.6 69.5 54.4 24.2 32.2]
 [32.9  0.  75.2 18.6 54.4 21.9 18.4 48.8 22.8  0.   0.7]
 [68.8 51.1  0.  69.6 53.2 71.1 38.7  0.  31.7 57.  47.9]
 [ 0.  18.7 93.8  0.  63.6  1.2 27.5 71.2 45.3 15.  23. ]
 [33.3 22.1 45.  31.2  0.  33.1  0.  18.6 48.6 18.4 26.5]
 [ 2.  20.8 94.1  0.  64.3  0.  27.8 67.7 47.4 17.1 25.2]
 [13.8  3.7 48.1 12.7 17.6 14.2  0.  21.7 30.2  0.   8.1]
```

```
 [45.3 24.7  0.  47.  26.8 44.7 12.3  0.   5.3 30.5 21.5]
 [35.9  4.4 37.4 26.8 62.5 30.1 26.5 11.   0.   8.1  0. ]
 [24.1  0.  81.1 14.9 50.7 18.2 14.7 54.6 26.5  0.   4.4]
 [31.4  0.  71.3 22.2 58.1 25.6 22.1 44.9 17.7  3.7  0. ]]
[[ 0.  29.3 58.6  0.  48.1  2.  16.3 58.5 49.1 20.5 31.5]
 [30.9  0.  37.8  5.9 36.8 20.7  6.1 37.8 17.5  0.   0. ]
 [69.8 47.4  0.  56.9 35.6 69.9 26.4  0.  26.4 53.3 47.2]
 [ 0.  15.  56.4  0.  46.   0.  15.2 60.2 40.  11.3 22.3]
 [31.3 18.4  7.6 18.5  0.  31.9  0.   7.6 43.3 14.7 25.8]
 [ 0.  17.1 56.7  0.  46.7  0.  15.5 56.7 42.1 13.4 24.5]
 [11.8  0.  10.7  0.   0.  13.   0.  10.7 24.9  0.   7.4]
 [43.3 21.   0.  34.3  9.2 43.5  0.   0.   0.  26.8 20.8]
 [33.9  0.7  0.  14.1 44.9 28.9 14.2  0.   0.   4.4  0. ]
 [22.1  0.  43.7  2.2 33.1 17.   2.4 43.6 21.2  0.   3.7]
 [29.4  0.  33.9  9.5 40.5 24.4  9.8 33.9 12.4  0.   0. ]]
Path: [0, 3, 5, 9, 1, 10, 8, 7, 2, 6, 4, 0]
Total Distance: 293.2
```

[2]:

```
#step 9
import matplotlib.pyplot as plt

# Towns and their arbitrary positions
towns = ['KLC', 'Mbale', 'Bungoma', 'Kondele', 'Siaya',␣
 ↪'Nyalenda', 'Luanda', 'Mumias', 'Kakamega', 'Majengo',␣
 ↪'Chavakali']
positions = [(1, 2), (2, 3), (3, 5), (4, 6), (5, 5), (6, 4), (7,␣
 ↪3), (8, 2), (9, 1), (10, 3), (11, 5)]

# Creating a dictionary for town positions
town_positions = {town: pos for town, pos in zip(towns, positions)}

# Path (based on the indices of the towns in the list)
path_indices = [0, 3, 5, 9, 1, 10, 8, 7, 2, 6, 4, 0]
```

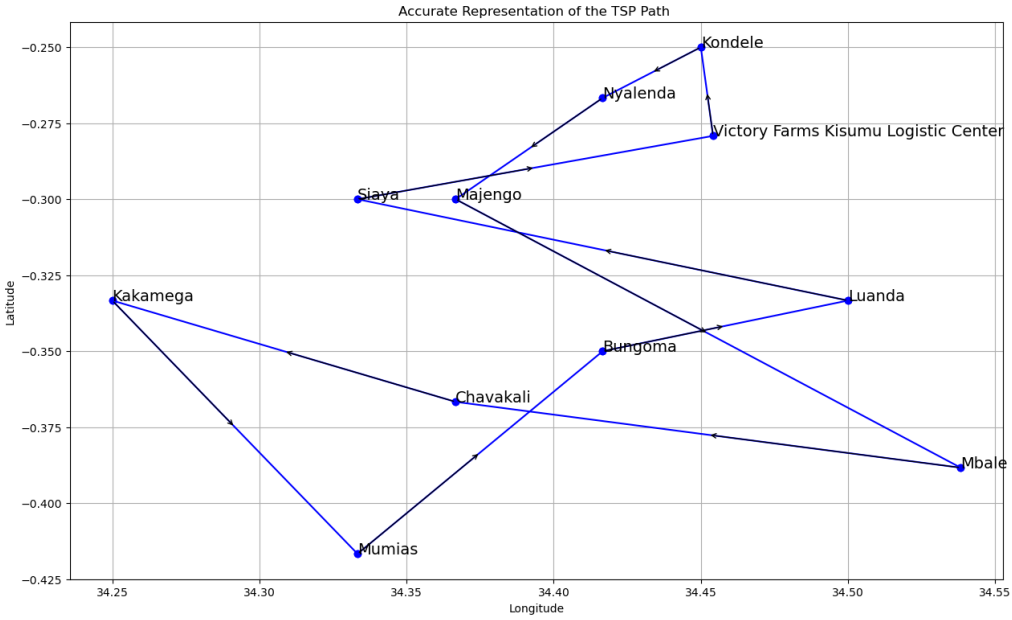
```
path = [towns[i] for i in path_indices]

# Plotting
plt.figure(figsize=(10, 6))
for i in range(len(path)-1):
    start = town_positions[path[i]]
    end = town_positions[path[i+1]]
    plt.plot([start[0], end[0]], [start[1], end[1]], 'bo-')

# Annotating towns
for town, pos in town_positions.items():
    plt.text(pos[0], pos[1], town)

plt.title('Schematic Representation of the TSP Path')
plt.xlabel('Arbitrary X Coordinate')
plt.ylabel('Arbitrary Y Coordinate')
plt.grid(True)
plt.show()
```



[1]:
```
#step 9 continuation
##visualization of the results with town coordinates
import matplotlib.pyplot as plt
```

```
# Function to convert DMS to decimal
def dms_to_decimal(dms_str):
    parts = dms_str.replace('°', ' ').replace('\'', ' ').
 ↪replace('"', ' ').split()
    d, m, s, direction = int(parts[0]), int(parts[1]),␣
 ↪float(parts[2]), parts[3]
    decimal = d + m / 60 + s / 3600
    if direction in ['S', 'W']:
        decimal = -decimal
    return decimal

# Town coordinates in DMS and their conversion to decimal
town_coordinates = {
    'Victory Farms Kisumu Logistic Center': '0°16\'45.0"S␣
 ↪34°27\'15.0"E',
    'Mbale': '0°23\'18.0"S 34°32\'18.0"E',
    'Bungoma': '0°21\'00.0"S 34°25\'00.0"E',
    'Kondele': '0°15\'00.0"S 34°27\'00.0"E',
    'Siaya': '0°18\'00.0"S 34°20\'00.0"E',
    'Nyalenda': '0°16\'00.0"S 34°25\'00.0"E',
    'Luanda': '0°20\'00.0"S 34°30\'00.0"E',
    'Mumias': '0°25\'00.0"S 34°20\'00.0"E',
    'Kakamega': '0°20\'00.0"S 34°15\'00.0"E',
    'Majengo': '0°18\'00.0"S 34°22\'00.0"E',
    'Chavakali': '0°22\'00.0"S 34°22\'00.0"E'
}

# Convert coordinates to decimal
decimal_coordinates = {town: (dms_to_decimal(coord.split()[0]),␣
 ↪dms_to_decimal(coord.split()[1]))
                       for town, coord in town_coordinates.items()}

# Path
#path = ['Victory Farms Kisumu Logistic Center', 'Kondele',␣
 ↪'Nyalenda', 'Majengo', 'Mbale','Chavakali', 'Luanda', 'Siaya',␣
 ↪'Bungoma', 'Mumias', 'Kakamega', 'Victory Farms Kisumu Logistic␣
 ↪Center']
path= ['Victory Farms Kisumu Logistic Center', 'Kondele',␣
 ↪'Nyalenda', 'Majengo', 'Mbale', 'Chavakali', 'Kakamega',␣
 ↪'Mumias', 'Bungoma', 'Luanda','Siaya', 'Victory Farms Kisumu␣
 ↪Logistic Center']
```

```
# Plotting
plt.figure(figsize=(15, 9))
for i in range(len(path)-1):
    start = decimal_coordinates[path[i]]
    end = decimal_coordinates[path[i+1]]

    # Plot line
    plt.plot([start[1], end[1]], [start[0], end[0]], 'bo-')

    # Calculate midpoint for arrow
    mid = [(start[0] + end[0]) / 2, (start[1] + end[1]) / 2]

    # Draw arrow
    plt.annotate('', xy=(mid[1], mid[0]), xytext=(start[1],␣
 ↪start[0]),
                 arrowprops=dict(facecolor='black',␣
 ↪arrowstyle="->"))

# Annotating towns
for town, (lat, lon) in decimal_coordinates.items():
    plt.text(lon, lat, town, fontsize=14)

plt.title('Accurate Representation of the TSP Path ')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid(True)
plt.show()
```



[ ]:

[1]:
```
#TIME OPTIMIZATION
import numpy as np
from scipy.optimize import linear_sum_assignment


# Given Time Matrix
B = np.array([
    [0, 43.9, 119.5, 4.3, 83.2, 8.2, 38.6, 87.7, 69.6, 33.4, 43],
    [43.9, 0, 94.7, 26.8, 69.7, 30.7, 26.5, 63, 31.8, 4.4, 5.3],
    [119.5, 94.7, 0, 116.9, 97.2, 118.7, 79.8, 33.4, 71.4, 101.8,␣
 ↪90.8],
    [4.3, 26.8, 116.9, 0, 80.6, 5.8, 37.3, 89.8, 58.7, 22.3, 31.9],
    [83.2, 69.7, 97.2, 80.6, 0, 82.9, 43.2, 65.5, 101.5, 65.3, 75],
    [8.2, 30.7, 118.7, 5.8, 82.9, 0, 39.1, 87, 62.6, 26.3, 36],
    [38.6, 26.5, 79.8, 37.3, 43.2, 39.1, 0, 48.1, 58.3, 22.1, 31.
 ↪8],
    [87.7, 63, 33.4, 89.8, 65.5, 87, 48.1, 0, 39.7, 70, 59.2],
    [69.6, 31.8, 71.4, 58.7, 101.5, 62.6, 58.3, 39.7, 0, 36.2, 26.
 ↪5],
    [33.4, 4.4, 101.8, 22.3, 65.3, 26.3, 22.1, 70, 36.2, 0, 9.7],
    [43, 5.3, 90.8, 31.9, 75, 36, 31.8, 59.2, 26.5, 9.7, 0]
])

# Set diagonal elements to infinity
B_temp = np.where(B == 0, np.inf, B)
#phase 1

#step 2
#In a given problem, if the number of rows is not equal to the␣
 ↪number of columns then add a dummy row or a dummy column,
#in our case rowsand colunms are equal


#step3
# Find the minimum non-zero element in each row
min_non_zero_per_row = np.min(B_temp, axis=1, keepdims=True)

#step 3
```

```
# Subtract the minimum non-zero value from each row, revert␣
 ↪infinity back to zero
B_row_reduced = np.where(B_temp == np.inf, 0, B -␣
 ↪min_non_zero_per_row)
print(B_row_reduced)

#phase 2

#step4
# Replace zeros with a very large number temporarily for column␣
 ↪reduction
B_temp = np.where(B_row_reduced == 0, np.inf, B_row_reduced)

# Find the minimum non-zero element in each column
min_non_zero_per_column = np.min(B_temp, axis=0, keepdims=True)

# Subtract the minimum non-zero value from each column, revert␣
 ↪infinity back to zero
B_col_reduced = np.where(B_temp == np.inf, 0, B_row_reduced -␣
 ↪min_non_zero_per_column)
print(B_col_reduced)

def tsp_to_assignment(distinance_matrix):
    """
    Convert the TSP to an assignment problem by subtracting the␣
 ↪minimum
    cost from each row and each column of the cost matrix, ensuring
    non-negativity of the resulting matrix.
    """
    # Find the minimum value of each row
    min_row = np.min(distinance_matrix, axis=1)

    # Subtract the minimum value of each row from that row
    distinance_matrix -= min_row[:, np.newaxis]

    # Find the minimum value of each column in the modified matrix
    min_col = np.min(distinance_matrix, axis=0)

    # Subtract the minimum value of each column from that column
    distinance_matrix -= min_col

    return distinance_matrix
```

```python
def assignment_to_tsp(assignment, original_distinance_matrix):
    """
    Convert the assignment solution back to TSP solution
    """
    num_nodes = original_distinance_matrix.shape[0]
    tour = []
    for row, col in enumerate(assignment):
        if col < num_nodes:
            tour.append((row, col))
    return tour

def tsp_hungarian(B, start=0):
    num_towns = len(B)
    path = [start]
    total_time = 0

    while len(path) < num_towns:
        last = path[-1]
        # Set time for already visited towns to infinity
        time = np.copy(B[last])
        time[path] = np.inf
        # Find the nearest town
        nearest = np.argmin(time)
        path.append(nearest)
        total_time += time[nearest]

    # Return to start
    total_time += B[path[-1], start]
    path.append(start)
    return path, total_time
# Solve TSP using Hungarian method
path, total_distance = tsp_hungarian(B)
print("Path:", path)
print("Total Distance:", total_distance)
```

```
[[  0.   39.6 115.2   0.   78.9   3.9  34.3  83.4  65.3  29.1  38.7]
 [ 39.5   0.   90.3  22.4  65.3  26.3  22.1  58.6  27.4   0.    0.9]
 [ 86.1  61.3   0.   83.5  63.8  85.3  46.4   0.   38.   68.4  57.4]
 [  0.   22.5 112.6   0.   76.3   1.5  33.   85.5  54.4  18.   27.6]
 [ 40.   26.5  54.   37.4   0.   39.7   0.   22.3  58.3  22.1  31.8]
 [  2.4  24.9 112.9   0.   77.1   0.   33.3  81.2  56.8  20.5  30.2]
```

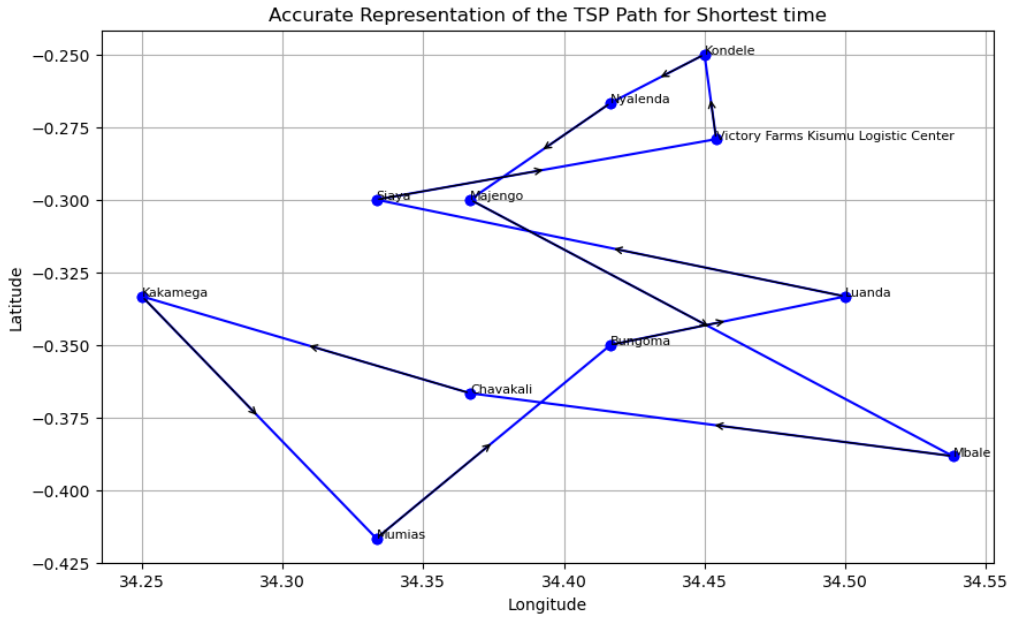
```
 [ 16.5   4.4  57.7  15.2  21.1  17.    0.   26.   36.2   0.    9.7]
 [ 54.3  29.6   0.   56.4  32.1  53.6  14.7   0.    6.3  36.6  25.8]
 [ 43.1   5.3  44.9  32.2  75.   36.1  31.8  13.2   0.    9.7   0. ]
 [ 29.    0.   97.4  17.9  60.9  21.9  17.7  65.6  31.8   0.    5.3]
 [ 37.7   0.   85.5  26.6  69.7  30.7  26.5  53.9  21.2   4.4   0. ]]
[[ 0.  35.2 70.3  0.  57.8  2.4 19.6 70.2 59.  24.7 37.8]
 [37.1  0.  45.4  7.2 44.2 24.8  7.4 45.4 21.1  0.   0. ]
 [83.7 56.9  0.  68.3 42.7 83.8 31.7  0.  31.7 64.  56.5]
 [ 0.  18.1 67.7  0.  55.2  0.  18.3 72.3 48.1 13.6 26.7]
 [37.6 22.1  9.1 22.2  0.  38.2  0.   9.1 52.  17.7 30.9]
 [ 0.  20.5 68.   0.  56.   0.  18.6 68.  50.5 16.1 29.3]
 [14.1  0.  12.8  0.   0.  15.5  0.  12.8 29.9  0.   8.8]
 [51.9 25.2  0.  41.2 11.  52.1  0.   0.   0.  32.2 24.9]
 [40.7  0.9  0.  17.  53.9 34.6 17.1  0.   0.   5.3  0. ]
 [26.6  0.  52.5  2.7 39.8 20.4  3.  52.4 25.5  0.   4.4]
 [35.3  0.  40.6 11.4 48.6 29.2 11.8 40.7 14.9  0.   0. ]]
Path: [0, 3, 5, 9, 1, 10, 8, 7, 2, 6, 4, 0]
Total Distance: 351.9
```

[ ]:

[2]:
```
#step 9 continuation
##visualization of the results with town coordinates
import matplotlib.pyplot as plt

# Function to convert DMS to decimal
def dms_to_decimal(dms_str):
    parts = dms_str.replace('°', ' ').replace('\'', ' ').
 ↪replace('"', ' ').split()
    d, m, s, direction = int(parts[0]), int(parts[1]),␣
 ↪float(parts[2]), parts[3]
    decimal = d + m / 60 + s / 3600
```

```
    if direction in ['S', 'W']:
        decimal = -decimal
    return decimal

# Town coordinates in DMS and their conversion to decimal
town_coordinates = {
    'Victory Farms Kisumu Logistic Center': '0°16\'45.0"S␣
 ↪34°27\'15.0"E',
    'Mbale': '0°23\'18.0"S 34°32\'18.0"E',
    'Bungoma': '0°21\'00.0"S 34°25\'00.0"E',
    'Kondele': '0°15\'00.0"S 34°27\'00.0"E',
    'Siaya': '0°18\'00.0"S 34°20\'00.0"E',
    'Nyalenda': '0°16\'00.0"S 34°25\'00.0"E',
    'Luanda': '0°20\'00.0"S 34°30\'00.0"E',
    'Mumias': '0°25\'00.0"S 34°20\'00.0"E',
    'Kakamega': '0°20\'00.0"S 34°15\'00.0"E',
    'Majengo': '0°18\'00.0"S 34°22\'00.0"E',
    'Chavakali': '0°22\'00.0"S 34°22\'00.0"E'
}

# Convert coordinates to decimal
decimal_coordinates = {town: (dms_to_decimal(coord.split()[0]),␣
 ↪dms_to_decimal(coord.split()[1]))
                       for town, coord in town_coordinates.items()}

# Path
#path = ['Victory Farms Kisumu Logistic Center', 'Kondele',␣
 ↪'Nyalenda', 'Majengo', 'Mbale','Chavakali', 'Luanda', 'Siaya',␣
 ↪'Bungoma', 'Mumias', 'Kakamega', 'Victory Farms Kisumu Logistic␣
 ↪Center']
path= ['Victory Farms Kisumu Logistic Center', 'Kondele',␣
 ↪'Nyalenda', 'Majengo', 'Mbale', 'Chavakali', 'Kakamega',␣
 ↪'Mumias', 'Bungoma', 'Luanda','Siaya', 'Victory Farms Kisumu␣
 ↪Logistic Center']


# Plotting
plt.figure(figsize=(10, 6))
for i in range(len(path)-1):
    start = decimal_coordinates[path[i]]
    end = decimal_coordinates[path[i+1]]

    # Plot line
    plt.plot([start[1], end[1]], [start[0], end[0]], 'bo-')
```

```
    # Calculate midpoint for arrow
    mid = [(start[0] + end[0]) / 2, (start[1] + end[1]) / 2]

    # Draw arrow
    plt.annotate('', xy=(mid[1], mid[0]), xytext=(start[1],␣
 ↪start[0]),
                 arrowprops=dict(facecolor='black',␣
 ↪arrowstyle="->"))

# Annotating towns
for town, (lat, lon) in decimal_coordinates.items():
    plt.text(lon, lat, town, fontsize=8)

plt.title('Accurate Representation of the TSP Path for Shortest␣
 ↪time')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid(True)
plt.show()
```



[ ]:

[1]:
```
#COST OPTIMIZATION
import numpy as np
from scipy.optimize import linear_sum_assignment

# cost Matrix
C = np.array([
    [0, 1830.0, 4980.0, 180.0, 3465.0, 340.0, 1610.0, 3655.0,␣
 ↪2900, 1390.0, 1790.0],
    [1830.0, 0, 3945.0, 1115.0, 2905.0, 1280.0, 1105.0, 2625.0,␣
 ↪1325.0, 185.0, 220.0],
    [4980.0, 3945.0, 0, 4870.0, 4050, 4945.0, 3325.0, 1390.0, 2975.
 ↪0, 4240.0, 3785.0],
    [180.0, 1115.0, 4870.0, 0, 3360.0, 240.0, 1555.0, 3740.0, 2445.
 ↪0, 930.0, 1330.0],
    [3465.0, 2905.0, 4050, 3360.0, 0, 3455.0, 1800, 2730.0, 4230.
 ↪0, 2720.0, 3125.0],
    [340.0, 1280.0, 4945.0, 240.0, 3455.0, 0, 1630.0, 3625.0, 2610.
 ↪0, 1095.0, 1500],
    [1610.0, 1105.0, 3325.0, 1555.0, 1800, 1630.0, 0, 2005.0, 2430.
 ↪0, 920.0, 1325.0],
    [3655.0, 2625.0, 1390.0, 3740.0, 2730.0, 3625.0, 2005.0, 0,␣
 ↪1655.0, 2915.0, 2465.0],
    [2900, 1325.0, 2975.0, 2445.0, 4230.0, 2610.0, 2430.0, 1655.0,␣
 ↪0, 1510.0, 1105.0],
    [1390.0, 185.0, 4240.0, 930.0, 2720.0, 1095.0, 920.0, 2915.0,␣
 ↪1510.0, 0, 405.0],
    [1790.0, 220.0, 3785.0, 1330.0, 3125.0, 1500, 1325.0, 2465.0,␣
 ↪1105.0, 405.0, 0]
])



# Set diagonal elements to infinity
np.fill_diagonal(C, np.inf)

# Replace zeros with a very large number temporarily
C_temp = np.where(C == 0, np.inf, C)
#phase 1

#step 2
#In a given problem, if the number of rows is not equal to the␣
 ↪number of columns then add a dummy row or a dummy column,
#in our case rows and colunms are equal
```

```
#step3
# Find the minimum non-zero element in each row
min_non_zero_per_row = np.min(C_temp, axis=1, keepdims=True)

#step 3
# Subtract the minimum non-zero value from each row, revert␣
 ↪infinity back to zero
C_row_reduced = np.where(C_temp == np.inf, 0, C -␣
 ↪min_non_zero_per_row)
print(C_row_reduced)

#step4
# Replace zeros with a very large number temporarily for column␣
 ↪reduction
C_temp = np.where(C_row_reduced == 0, np.inf, C_row_reduced)

# Find the minimum non-zero element in each column
min_non_zero_per_column = np.min(C_temp, axis=0, keepdims=True)

# Subtract the minimum non-zero value from each column, revert␣
 ↪infinity back to zero
C_col_reduced = np.where(C_temp == np.inf, 0, C_row_reduced -␣
 ↪min_non_zero_per_column)
print(C_col_reduced)
def tsp_to_assignment(cost_matrix):
    """
    Convert the TSP to an assignment problem by subtracting the␣
 ↪minimum
    cost from each row and each column of the cost matrix, ensuring
    non-negativity of the resulting matrix.
    """
    # Find the minimum value of each row
    min_row = np.min(cost_matrix, axis=1)

    # Subtract the minimum value of each row from that row
    cost_matrix -= min_row[:, np.newaxis]

    # Find the minimum value of each column in the modified matrix
    #min_col = np.min(cost_matrix, axis=0)
    min_col = np.min(distinance_matrix, axis=0)
    # Subtract the minimum value of each column from that column
    cost_matrix -= min_col
```

```
    return cost_matrix


def assignment_to_tsp(assignment, original_cost_matrix):
    """
    Convert the assignment solution back to TSP solution
    """
    num_nodes = cost_matrix_matrix.shape[0]
    tour = []
    for row, col in enumerate(assignment):
        if col < num_nodes:
            tour.append((row, col))
    return tour
def tsp_hungarian(C, start=0):
    num_towns = len(C)
    path = [start]
    total_time = 0

    while len(path) < num_towns:
        last = path[-1]
        # Set time for already visited towns to infinity
        time = np.copy(C[last])
        time[path] = np.inf
        # Find the nearest town
        nearest = np.argmin(time)
        path.append(nearest)
        total_time += time[nearest]

    # Return to start
    total_time += C[path[-1], start]
    path.append(start)
    return path, total_time
# Solve TSP using Hungarian method
path, total_distance = tsp_hungarian(C)
print("Path:", path)
print("Total Distance:", total_distance)
```

```
[[   0. 1650. 4800.    0. 3285.  160. 1430. 3475. 2720. 1210. 1610.]
 [1645.    0. 3760.  930. 2720. 1095.  920. 2440. 1140.    0.   35.]
 [3590. 2555.    0. 3480. 2660. 3555. 1935.    0. 1585. 2850. 2395.]
 [   0.  935. 4690.    0. 3180.   60. 1375. 3560. 2265.  750. 1150.]
```

```
 [1665. 1105. 2250. 1560.    0. 1655.    0.  930. 2430.  920. 1325.]
 [ 100. 1040. 4705.    0. 3215.    0. 1390. 3385. 2370.  855. 1260.]
 [ 690.  185. 2405.  635.  880.  710.    0. 1085. 1510.    0.  405.]
 [2265. 1235.    0. 2350. 1340. 2235.  615.    0.  265. 1525. 1075.]
 [1795.  220. 1870. 1340. 3125. 1505. 1325.  550.    0.  405.    0.]
 [1205.    0. 4055.  745. 2535.  910.  735. 2730. 1325.    0.  220.]
 [1570.    0. 3565. 1110. 2905. 1280. 1105. 2245.  885.  185.    0.]]
[[   0. 1465. 2930.    0. 2405.  100.  815. 2925. 2455. 1025. 1575.]
 [1545.    0. 1890.  295. 1840. 1035.  305. 1890.  875.    0.    0.]
 [3490. 2370.    0. 2845. 1780. 3495. 1320.    0. 1320. 2665. 2360.]
 [   0.  750. 2820.    0. 2300.    0.  760. 3010. 2000.  565. 1115.]
 [1565.  920.  380.  925.    0. 1595.    0.  380. 2165.  735. 1290.]
 [   0.  855. 2835.    0. 2335.    0.  775. 2835. 2105.  670. 1225.]
 [ 590.    0.  535.    0.    0.  650.    0.  535. 1245.    0.  370.]
 [2165. 1050.    0. 1715.  460. 2175.    0.    0.    0. 1340. 1040.]
 [1695.   35.    0.  705. 2245. 1445.  710.    0.    0.  220.    0.]
 [1105.    0. 2185.  110. 1655.  850.  120. 2180. 1060.    0.  185.]
 [1470.    0. 1695.  475. 2025. 1220.  490. 1695.  620.    0.    0.]]
Path: [0, 3, 5, 9, 1, 10, 8, 7, 2, 6, 4, 0]
Total Distance: 14660.0
```

[ ]:

[2]:
```python
#step 9 continuation
##visualization of the results with town coordinates
import matplotlib.pyplot as plt

# Function to convert DMS to decimal
def dms_to_decimal(dms_str):
    parts = dms_str.replace('°', ' ').replace('\'', ' ').
 ↪replace('"', ' ').split()
```

```
    d, m, s, direction = int(parts[0]), int(parts[1]),␣
 ↪float(parts[2]), parts[3]
    decimal = d + m / 60 + s / 3600
    if direction in ['S', 'W']:
        decimal = -decimal
    return decimal

# Town coordinates in DMS and their conversion to decimal
town_coordinates = {
    'Victory Farms Kisumu Logistic Center': '0°16\'45.0"S␣
 ↪34°27\'15.0"E',
    'Mbale': '0°23\'18.0"S 34°32\'18.0"E',
    'Bungoma': '0°21\'00.0"S 34°25\'00.0"E',
    'Kondele': '0°15\'00.0"S 34°27\'00.0"E',
    'Siaya': '0°18\'00.0"S 34°20\'00.0"E',
    'Nyalenda': '0°16\'00.0"S 34°25\'00.0"E',
    'Luanda': '0°20\'00.0"S 34°30\'00.0"E',
    'Mumias': '0°25\'00.0"S 34°20\'00.0"E',
    'Kakamega': '0°20\'00.0"S 34°15\'00.0"E',
    'Majengo': '0°18\'00.0"S 34°22\'00.0"E',
    'Chavakali': '0°22\'00.0"S 34°22\'00.0"E'
}

# Convert coordinates to decimal
decimal_coordinates = {town: (dms_to_decimal(coord.split()[0]),␣
 ↪dms_to_decimal(coord.split()[1]))
                       for town, coord in town_coordinates.items()}

# Path
#path = ['Victory Farms Kisumu Logistic Center', 'Kondele',␣
 ↪'Nyalenda', 'Majengo', 'Mbale','Chavakali', 'Luanda', 'Siaya',␣
 ↪'Bungoma', 'Mumias', 'Kakamega', 'Victory Farms Kisumu Logistic␣
 ↪Center']
path= ['Victory Farms Kisumu Logistic Center', 'Kondele',␣
 ↪'Nyalenda', 'Majengo', 'Mbale', 'Chavakali', 'Kakamega',␣
 ↪'Mumias', 'Bungoma', 'Luanda','Siaya', 'Victory Farms Kisumu␣
 ↪Logistic Center']


# Plotting
plt.figure(figsize=(7, 4))
for i in range(len(path)-1):
    start = decimal_coordinates[path[i]]
    end = decimal_coordinates[path[i+1]]
```

```
    # Plot line
    plt.plot([start[1], end[1]], [start[0], end[0]], 'bo-')

    # Calculate midpoint for arrow
    mid = [(start[0] + end[0]) / 2, (start[1] + end[1]) / 2]

    # Draw arrow
    plt.annotate('', xy=(mid[1], mid[0]), xytext=(start[1],␣
 ↪start[0]),
                 arrowprops=dict(facecolor='blue',␣
 ↪arrowstyle="->"))

# Annotating towns
for town, (lat, lon) in decimal_coordinates.items():
    plt.text(lon, lat, town, fontsize=10)

plt.title('A graph showing the optimal path for cost')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid(True)
plt.show()
```