CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

This section uses insights from several studies and the relevance of efficiency, from the cost’s perspective in transportation with sustainable delivery optimization (Thipparthy et al., 2024).

In this chapter, present this comprehensive methodology for optimizing delivery routes in the e-commerce industry using linear programming (LP) alongside the advanced tools, frameworks and techniques. Incorporated into this project are insights from scholarly articles to further expand and broaden the scope of this project. It proposes to achieve the cost efficiency, environmental sustainability and enhanced delivery performance of the methodology through mathematical modelling and technological innovation.

3.2 Research Design

3.2.1 Definition:

The research design follows as it combines the principles from (Xue et al., 2021); focusing on optimizing rider scheduling and improving delivery performance through a structured methodology.

Generally, this is referred to as research design, which is the structured plan and methodology used to achieve project objectives. The research design adopted in this study is quantitative with a dependence on mathematical modelling.

a) **Objective Definition**:

On one hand, clearly define the goals related to minimizing operational costs, delivery time and environmental impacts.

Frame these goals within the logistical problems of e-commerce delivery systems.

b) **Data Integration**:

Use data from many sources such as Geographic Information Systems (GIS) historical delivery logs, traffic patterns.

Combine both real time and historical data as a way to improve the decision-making potential.

c) **Model Development**:

Linear Programming (LP) is the bedrock, where constraints and restricted resources are formulated to optimize the transportation challenges.

Mix Hull sore Programming (MIP) address Round more specific constraints Round time windows and resources limitations.

d) **Validation and Sensitivity Analysis**:

The proposed model must be properly tested, and should be proved to work well under different environments.

Adapt the model to changes in traffic, demand patterns, and environmental variables.

3.3 Framework Overview

3.3.1 Definition:

The approach to the framework is multi phase and on integrating supplier relation management and technological tools for good logistics outcomes (Grant, 2024).

The structured phases for the development, optimization and validation of the delivery optimization model are described in the framework.

a) **Problem Formulation**:

Set goals (e.g., get cheapest, or fastest).

Constraint such as vehicle capacity, time window, environmental consideration is defined.

b) **Data Collection and Pre-processing**:

Get raw data from delivery logs or external traffic system.

The data will be process to remove errors and make the data compatible with optimization algorithms.

c) **Model Development**:

Solve cost and route optimization with Linear Programming.

Solve complex routing challenges with heuristic and meta heuristic algorithms (Genetic Algorithm etc).

d) Optimization:

Apply advanced solvers such as Gurobi or Python libraries to find optimal routes.

Be defined for static (fixed schedule) as well as dynamic (real time updates).

e) Validation and Implementation:

Compare results to known, proven data from the historical past and the field.

They implement findings in simulation environments and evaluate the performance in real world applications.

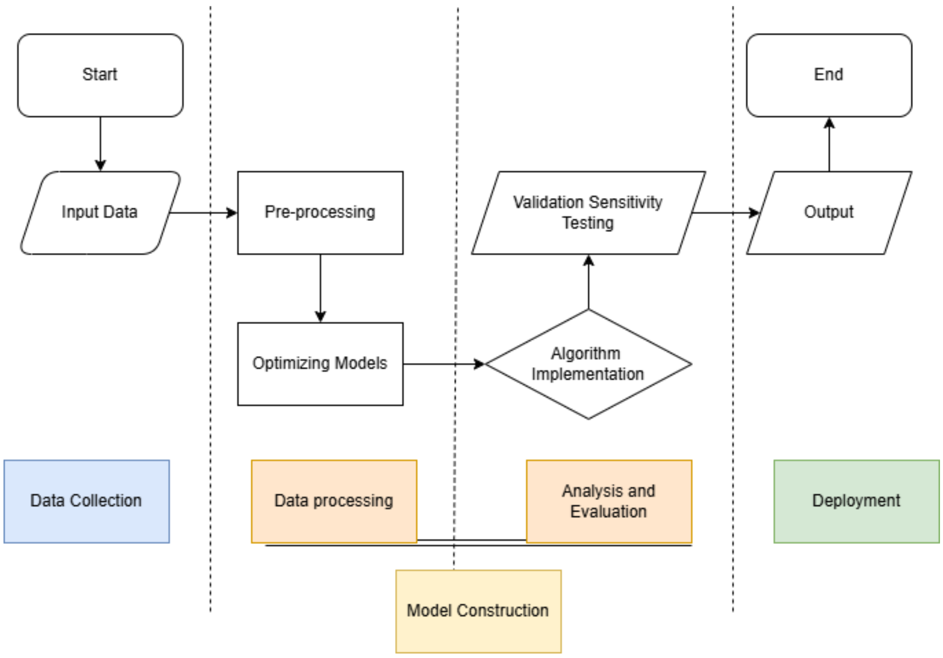

Figure 3.1 Research Frame work of Optimization

3.4 Problem Formulation

3.4.1 Definition:

Linear programming (LP) is a mathematical method to opt for the best outcome of a model containing linear links between variables subject to restrictions (Tang, 2023). The delivery route optimization problem is defined as a linear programming model.

Set up an objective function to minimize costs, such as:

a) **Objective Function**:

The optimization model aims to achieve multiple goals, such as minimizing costs, time, and environmental impact:

$$Z = \alpha \sum_{i,j} c_{ij} x_{ij} + \beta \sum_{i,j} t_{ij} x_{ij} + \gamma \sum_{i,j} e_{ij} x_{ij} \tag{3.1}$$

b) **Travel Cost Component (c_{ij})**:

c_{ij}: Represents the monetary cost associated with traveling from node i to node j. For example, it could account for fuel, tolls, or wear and tear on the vehicle.

$\sum_{i,j} c_{ij} x_{ij}$: Aggregates the total travel costs across all selected routes.

Weighted by α: This factor determines how much importance is given to minimizing the travel cost in the overall optimization.

c) **Travel Time Component (t_{ij})**:

t_{ij}: Represents the time required to travel from node i to node j.

$\sum_{i,j} t_{ij}x_{ij}$: Aggregates the total travel time across all selected routes.

Weighted by β: This factor controls the emphasis on minimizing travel time in the overall objective.

d) Environmental Cost Component (e_{ij}):

e_{ij}: Represents the environmental cost associated with traveling from node i to node j. It could be measured in terms of greenhouse gas emissions, energy consumption, or other sustainability metrics.

$\sum_{i.j} e_{ij}x_{ij}$: Aggregates the total environmental cost across all selected routes.

Weighted by γ: This factor adjusts how much importance is placed on minimizing the environmental cost.

e) Decision Variable (x_{ij})

x_{ij}: A binary decision variable that determines whether the route from node i to node j is selected:

$x_{ij} = 1$: Route $i \rightarrow j$ is selected.

$x_{ij} = 0$: Route $i \rightarrow j$ is not selected.

The objective function is therefore the total weighted sum of the costs (travel cost, travel time, environmental cost) for all the selected routes.

f) Weight Factors (α, β, γ)

These parameters allow for prioritization among the three objectives:

A higher α emphasizes minimizing monetary travel costs.

A higher β emphasizes minimizing travel time.

A higher γ emphasizes reducing environmental impact.

The choice of α, β, γ depends on the priorities of the delivery company or stakeholders. For instance, a company focused on eco-friendly operations might assign a higher γ compared to α or β.

g) **Constraints**: The optimization problem would also typically include constraints to ensure feasibility, such as:

Each delivery must be served only once.

Vehicle capacity limits.

$$\sum_i d_i x_{ij} \leq C \ \forall j$$

Time windows for deliveries.

$$e_i \leq t_i \leq l_i \ \forall i$$

This formulation can be applied in:

E-commerce logistics: Optimizing delivery routes with minimal cost, time, and environmental impact.

Fleet management: Or assigning routes to vehicles such that the vehicle routes satisfy cost, time and sustainability goals.

Urban planning: Reduction of congestion and greenhouse emissions from on street delivery vehicles in the city.

3.5 Optimization Techniques

3.5.1 Gurobi: Gurobi is considered a powerful commercial optimization solver, it is particularly quick at solving large scale linear programming (LP), mixed integer programming (MIP), and quadratic programming (QP) problems. Here's how Gurobi is applied:

a) Implementation in Gurobi:

```
from gurobipy import Model, GRB

# Define the model
model = Model("Minimize_Delivery")
```

Figure 3.2 Define the Model

```
# Decision variables
x = model.addVars(n, m, vtype=GRB.CONTINUOUS, name="x")
```

Figure 3.3 Decision Variables

```
# Objective function
model.setObjective(sum(c[i][j] * x[i, j]
                       for i in  range(n)
                       for j in range(m)),
                   GRB.MINIMIZE)
```

Figure 3.4 Objective Function

```
# Constraints
for i in range(n):
    model.addConstr(sum(x[i, j]
                        for j in range(m)) == demand[i], "Demand")

# Solve
model.optimize()
```

Figure 3.5 Constraint and Optimal Solution

b) Advantages of Gurobi:

It handles large datasets with complex constraints in a fast manner.

Provides advanced features such as sensitivity analysis, parameter tuning and multi objective optimization.

3.5.2 PuLP

PuLP is an open-source linear programming library in Python. Unlike Gurobi, it's easier but effective in solving smaller scale problems or for academic considerations.

a) **Define the Model:** Use $L_P Problem$ to create a linear programming problem:

```
from pulp import LpProblem, LpMinimize, LpVariable

# Define the problem
problem = LpProblem("Delivery_Route_Optimization", LpMinimize)
```

Figure 3.6 LP Problem

b) **Set Decision Variables:** Define binary variables x_{ij} for route selection:

```
x = LpVariable.dicts("Route", (nodes, nodes), cat="Binary")
```

Figure 3.7 Binary Variable

c) **Set Objective Function:** Minimize the total cost:

```
problem += lpSum(c[i][j] * x[i][j] for i in nodes for j in nodes)
```

Figure 3.8 Minimization of Cost

d) **Define Constraints:** Add constraints for vehicle capacity and demand fulfillment:

```
for i in nodes:
    problem += lpSum(x[i][j] for j in nodes) == 1
```

Figure 3.9 Constraint

e) Solve the Problem: Use $problem.solve()$ to fine the optimal solution:

```
from pulp import PULP_CBC_CMD

# Solve the problem
problem.solve(PULP_CBC_CMD())
```

Figure 3.10 Optimal Solution

f) Output Results: Retrieve the optimal routes and costs:

```
for i in nodes:
    for j in nodes:
        if x[i][j].value() == 1:
            print(f"Route selected: {i} -> {j}")
```

Figure 3.11 Result

Feature	Gurobi	PuLP
Scalability	Suitable for large-scale problems.	Best for small-to-medium problems.
Speed	Faster due to proprietary algorithms.	Slower but sufficient for basic cases.
Cost	Commercial; requires a license.	Open-source and free.
Advanced Features	Sensitivity analysis, multi-objective optimization, callbacks.	Basic LP/MIP problem-solving
Ease of Use	More complex API but powerful.	Simple and intuitive.

Figure 3.12 Comparison of Gurobi and PuLP

g) Practical Usage:

(a) **-** Small-Scale Optimization (Using PuLP): Perfect for academic demo or to optimize delivery routes for a small fleet or dataset.

- Large-Scale, Real-Time Scenarios (Using Gurobi): This approach is efficient for large e-commerce delivery data with many delivery points, complex constraints (time windows and traffic), and dynamic updates.

3.5 Data Collection and Pre-processing

3.5.1 Definition:

This step married findings, considered the part played by robotics in improving data driven logistics operations, and the significance of accurate supplier data for delivery performance. To the optimization models, data collection and pre-processing describe data collection, and the preparation of it to use in the optimization models (Kumar Davala et al., 2024).

a) Data Sources:

Historical Delivery Logs: Provides baseline metrics and patterns.

GIS Data: Offers spatial coordinates and distances.

Traffic Patterns: Supplies real-time road conditions and delays.

b) Pre-processing:

Data Cleaning: Remove anomalies, duplicates, and incomplete records.

Normalization: Standardize data formats and scales to ensure compatibility.

Distance Matrix Creation: Use GIS to calculate distances and travel times between nodes.

3.6 Advanced Tools and Technologies

3.6.1 Definition:

Using the advanced tools (Thipparthy et al., 2024) guarantees scalability and efficiency in logistics solutions, and also the robotics and IoT plays an important role in modern e-commerce logistics (Kumar Davala et al., 2024).

These tools and technology improve model implementation, model optimization and model visualization.

a) **Programming Tools**: **Python**: Libraries like **PuLP**, **Gurobi**: Advanced solvers for large-scale optimization.

b) **Visualization Tools**: Tableau and presenting results.

c) **Simulation Tools**: Any Logic for testing models under simulated environments.

3.7 Validation and Sensitivity Testing

3.7.1 Definition:

The validation methods we use match up with (Xue et al., 2021) and we’re sanity checking that models are working on real conditions and sensitivity testing includes variables that (Tang, 2023) has found.

Firstly, validation makes sure the models are accurate, and secondly, sensitivity testing tests the model on changing conditions.

a) **Validation**: Compare optimized routes with historical data to measure improvements and Validate cost savings and efficiency gains through real-world pilot programs.

b) **Sensitivity Testing**: Assess the model's performance under, Increased demand, Variable traffic conditions and Changes in vehicle capacity or time constraints.

REFERENCES

Grant, O. (2024). *Exploring the Impact of Supplier Relationship Management on E-Commerce Delivery Performance*. https://doi.org/10.20944/preprints202407.0603.v1

Kumar Davala, K., Ganesh, B., Tarun Raj, C., Chandra Prasad, S., & Balasem, Z. A. (2024). E-Commerce Door Delivery Robot. *E3S Web of Conferences*, *552*. https://doi.org/10.1051/e3sconf/202455201143

Tang, P. (2023). Minimization of transportation costs using linear programming. *Theoretical and Natural Science*, *25*(1), 226–231. https://doi.org/10.54254/2753-8818/25/20240975

Thipparthy, K. R., Khalaf, M. I., Yogi, K. S., Alghayadh, F. Y., Yusupov, A., Maguluri, L. P., & Ofori-Amanfo, P. (2024). Optimizing delivery routes for sustainable food delivery for multiple food items per order. *Discover Sustainability*, *5*(1). https://doi.org/10.1007/s43621-024-00326-y

Xue, G., Wang, Z., & Wang, G. (2021). Optimization of Rider Scheduling for a Food Delivery Service in O2O Business. *Journal of Advanced Transportation*, *2021*. https://doi.org/10.1155/2021/5515909