universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Last-Mile Delivery via Truck and Trolley: a Traveling Salesman Problem with Parking and Trolley Packing Constraints

verfasst von | submitted by

Boryana Djarova BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 915
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Betriebswirtschaft
Betreut von | Supervisor:	emer. o. Univ.-Prof. Dipl.-Ing. Dr. Richard Hartl
Mitbetreut von | Co-Supervisor:	Dr. Yannick Scherr B.Sc. M.Sc.

Abstract

Der starke Anstieg des E-Commerce verstärkt die operativen Herausforderungen der Letzten-Meile-Zustellung, insbesondere in dicht besiedelten Gebieten. Begrenzte Haltemöglichkeiten, Verkehrsdichte und eine wachsende Paketmenge erschweren eine effiziente Tourengestaltung – Aspekte, die in klassischen Optimierungsmodellen wie dem Traveling-Salesman-Problem (TSP) kaum berücksichtigt werden.

Diese Arbeit stellt ein integriertes Modell des TSP mit Haltepunktauswahl und Trolley-Kapazitätsbeschränkungen (TSP-PTP) vor, das Fahrzeugrouten, Haltepunkte und fußläufige Zustelltouren gemeinsam optimiert. Das Modell kombiniert ein Bin-Packing-Teilproblem mit einer TSP-ähnlichen Routingstruktur und wird über eine heuristische Vorgehensweise gelöst, die einen Ausgleich zwischen Lösungsqualität und Rechenaufwand ermöglicht.

Die Rechenergebnisse zeigen, wie Haltezeiten, maximale Gehstrecken und Trolley-Kapazitäten das Verhältnis zwischen Fahr- und Fußwegen beeinflussen. Zudem wird deutlich, dass Tourenpläne ohne Halte- und Kapazitätsrestriktionen systematisch zu optimistisch ausfallen. Die integrierte Modellierung bildet die realen Bedingungen der Letzten-Meile-Zustellung deutlich besser ab und führt zu effizienteren Touren.

Abstract

The growth of e-commerce and the increasing volume of home deliveries are intensifying the challenges of last-mile logistics in dense urban areas. Delivery drivers face limited parking availability, congestion and the need to handle multiple parcels. These realities are rarely captured in classical optimization models such as the Traveling Salesman Problem (TSP). This thesis develops and tests an integrated model of the TSP with Parking and Trolley Packing (TSP-PTP), which jointly optimizes vehicle routing, parking location choices and walking tours under capacity-feasible trolley sets. The model combines a bin-packing subproblem with a TSP-style routing structure and is solved using a heuristic pipeline that balances solution quality and computational feasibility. Computational experiments on small and large instances demonstrate how parking time, walking distance limits and trolley capacity shape the trade-off between driving and walking in urban and rural delivery settings. The findings show that ignoring parking and packing constraints leads to overly optimistic delivery plans and that modeling them together provides more realistic and efficient solutions for last-mile operations.

Table of Contents

List of Tables and Figures

I.Part One

1.Introduction

E-commerce growth is driving parcel volumes to record levels. New York City already handles over 1.5 million daily deliveries (Haag & Hu, 2019), with forecasts of a 68% increase by 2045 (Danigelis, 2018). In dense urban areas, delivery drivers face severe challenges: congestion, scarce parking and growing competition for curb space. For example, in Seattle, parking search accounts for nearly 28% of trip time (Dalla Chiara & Goodchild, 2020). These conditions make last-mile delivery both costly and inefficient.

This thesis addresses a variation of the Traveling Salesman Problem (TSP) that captures two overlooked realities of last-mile logistics: (i) parking search time and (ii) trolley packing constraints. Delivery personnel often rely on hand trolleys with strict weight and volume limits, meaning not all packages can be grouped into a single walking loop. These constraints interact with parking: limited trolley capacity may require more frequent parking stops, while long parking times may encourage larger walking loops. To the best of my knowledge, no existing models jointly incorporate both parking-search time and trolley-capacity constraints.

The goal of this thesis is to design and test an optimization model - referred to here as the TSP-PTP (TSP with Parking and Trolley Packing) - that minimizes total delivery completion time by jointly optimizing vehicle routing, parking location choices, and walking tours with capacity-feasible trolley loads.

The thesis addresses two research questions:

1. Impact: How do trolley packing constraints and parking availability affect the efficiency of last-mile deliveries in urban areas?
2. Methods: How can heuristic and metaheuristic approaches be effectively applied to solve this TSP variant at realistic scale?

By investigating these questions, this work aims to contribute both a novel integrated model and computational evidence on the trade-offs between parking, walking and driving in dense cities.

This work is split into two parts. Part I gives the overall background needed to understand the problem. It starts by introducing the motivation and the key ideas behind last-mile delivery, then reviews the main concepts and related work on the TSP, parking constraints and trolley packing. After that, it lays out the different model components and

summarizes the solution approaches that are usually applied. Part I ends with a short discussion of research gaps and how this thesis fits into the existing literature.

Part II focuses on putting everything into practice. It presents the full model, the data requirements and the way the implementation was carried out. This part also includes the computational experiments, ranging from small test cases to larger instances using a scoring heuristic. The results are followed by an analysis of the heuristic's quality and several sensitivity tests. Part II finishes with a discussion of limitations and possible directions for future work.

2.Literature Review

This chapter reviews the key concepts and existing research that form the foundation for this thesis. The focus is on optimization approaches for last-mile delivery in dense urban areas, with particular attention to parking constraints and trolley packing, which are central to the problem studied here. The review begins by outlining the role and challenges of last-mile delivery and then introduces the Traveling Salesman Problem (TSP) as a fundamental optimization framework. It then considers extensions of the TSP, including parking and packing constraints, and examines related models such as the Capacitated Delivery Problem with Parking (CDPP) and the Two-Echelon Capacitated Vehicle Routing Problem (2E-CVRP). The chapter also surveys solution approaches, ranging from exact algorithms to heuristics and metaheuristics, and summarizes findings from computational studies. Finally, it discusses research trends, gaps, and directions that directly motivate the model developed in this thesis.

2.1.Last-Mile Delivery

Before turning to the Traveling Salesman Problem (TSP), it is useful to outline the role of last-mile delivery in urban logistics and the specific challenges of B2C parcel delivery to private households. The last-mile delivery, as defined by Boysen et al. (2021), refers to the final step in the retail supply chain, involving the actual delivery of shipments to private customer households with the focus on urban areas. This stage is crucial as it directly impacts customer satisfaction and plays a significant role in the overall logistics process due to the high logistics cost. Last-mile delivery is vital for retailers and logistics service providers, as it represents a key challenge due to trends such as the increasing urbanization, the expected growth in e-commerce and the need for sustainable delivery solutions. Specific challenges related to last-mile delivery, as discussed in Boysen et al. (2020), include:

Congestion: The increasing volume of deliveries leads to more delivery vans entering city centers, contributing to congestion and straining existing infrastructure.

Parking: As per the previously mentioned empirical study conducted in Seattle, commercial delivery drivers spend an average of 2.3 minutes searching for each parking spot, with parking search comprising 28% of the total trip time between parking locations (Dalla Chiara and Goodchild, 2020).

Costs: Traditional attended home deliveries by trucks are costly largely due to traffic jams, lack of parking, and failed deliveries and often account for a large fraction of logistics cost at around 50% (Klein, 2022). Consequently, alternative delivery concepts such as unattended delivery or customer self-services offer a promising solution to reduce these costs.

Time Windows: Customers often require specific time windows for deliveries, which can be challenging for logistics providers to manage efficiently. Balancing customer preferences for shorter time windows with the need for flexible routing poses a significant challenge.

Failed Deliveries: Instances where customers are not available to receive deliveries result in additional visits, increased traffic, and operational inefficiencies.

Sustainability: The environmental impact of last-mile delivery, such as emissions from delivery vehicles and the resulting pollution, is a growing concern. The European Commission has outlined in their White Paper the expected roadmap to essentially CO_2-free city logistics by 2030 (White Paper, 2011). This target is reinforced through the Sustainable Urban Logistics Plan (SULPs). Implementing sustainable delivery practices is a challenge faced by logistics providers.

Customer Expectations: Meeting customer expectations for fast, reliable deliveries within their preferred time windows while ensuring cost-effective operations poses a challenge in last-mile logistics.

Addressing these challenges requires innovative solutions, such as optimizing delivery routes, implementing sustainable delivery practices, leveraging technology like drones and autonomous vehicles and improving coordination between various stakeholders within the supply chain.

2.2.Traveling Salesman Problem (TSP)

As described by Laporte (2010), the Traveling Salesman Problem (TSP) is a widely studied problem in combinatorial optimization that involves finding the shortest possible tour that visits a set of cities exactly once and returns to the original city. The TSP is defined by determining a Hamiltonian tour, which is a tour that passes through each vertex of a graph precisely once, often interpreted as the optimal path for a salesman to navigate through cities and return to the origin.

The importance of the TSP in optimization problems lies in its wide range of applications across various disciplines ranging from manufacturing and logistics to distribution management and scheduling. The TSP serves as a fundamental problem that has led to the development of many exact and heuristic algorithms to find efficient solutions. The study of the TSP has resulted in the creation of well-known local improvement heuristics such as 2-opt and 3-opt, as well as exact algorithms such as branch-and-bound, branch-and-cut (e.g. Concorde) and branch-and-cut-and-price. These optimization techniques have also been applied to other complex optimization problems, making the TSP a cornerstone in the field of operational research and combinatorial optimization.

Mathematically, the symmetrical TSP can be formulated as an optimization problem where the goal is to minimize the total distance traveled. The TSP is defined on a complete undirected graph G =(V, E) if it is symmetric or on a directed graph G =(V, A) if it is asymmetric. In the classical metric TSP the cost matrix satisfies the triangle inequality, though in real street networks this assumption may not hold due to turn restrictions or traffic rules. Various constraints such as degree constraints, subtour elimination constraints and integrality constraints are applied to the formulation. The constraints ensure that each city is visited exactly once (Hamiltonian cycle constraint) and that the tour forms a closed loop returning to the starting city.

$$\text{(STSP) minimize} \sum_{i<j} c_{ij} x_{ij}$$

subject to

$$\sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 \qquad (k \in V)$$

$$\sum_{i,\, j \in S} x_{ij} \leqslant |S| - 1 \qquad (S \subset V, 3 \leqslant |S| \leqslant n-3)$$

$$x_{ij} = 0 \text{ or } 1 \qquad (i,\ j) \in E.$$

Figure 1.1: TSP Model formulation

There is an asymmetrical formulation of the problem in which the cost (distance) between two cities is different depending on the direction. Urban delivery networks are often asymmetric as there the routes are usually consisting of one-way streets, turn penalties and other directional constraints. This formulation provides the building block for modelling both the walking tours from parking spots and the truck's overall routing. The various exact and heuristic algorithms for solving the TSP will be examined in more detail later.

2.3.Parking Constraints in TSP

One of the main challenges in last-mile delivery route planning is the presence of parking constraints, i.e. the challenges in finding suitable and legal parking spots during the tour (Nguyen et al., 2018). These challenges can be observed as parking restrictions, limited availability of parking spaces, one-way road systems, traffic congestion and the need to balance walking distances with driving distances.

The impact of parking constraints on route planning is significant as it can affect the efficiency and effectiveness of last-mile delivery operations. In dense urban areas, where parking directly outside the delivery address may not always be possible or convenient, route planners must account for the additional walking distances that drivers may need to cover from the nearest parking spot to the delivery location. Failure to consider these constraints can lead to an overestimation of driving distances and suboptimal route plans.

To address parking constraints in route planning, a dual-mode routing model that incorporates both driving and walking components has been proposed in the literature. This model would require defining a walking network to accurately represent the walking distances between parking spots and delivery locations. By integrating parking constraints into the route optimization process, planners can create more realistic and efficient delivery routes that take into account the challenges posed by parking challenges in urban environments. In a later chapter the Capacitated Delivery Problem with Parking (CDPP) will be introduced as an example of a dual-model routing model for such routing problems (Reed et al., 2024).

To provide examples from urban delivery scenarios, we can look at the findings from Chiara et al. (2021). In this study, ride-alongs were conducted with various logistics carriers in Seattle, Washington to understand commercial vehicle (CV) driver behaviors in urban delivery settings.

Here are some examples based on the study's results:

Parking Behavior: The study found that urban CVs spent an average of 80% of their daily operating time being parked in a parking spot. This highlights the significant amount of time required to spend on parking activities during urban deliveries.

Parking Locations: Contrary to common belief, drivers tended to park in authorized parking locations. Less than 5% of stops occurred in the travel lane, emphasizing the importance of parking regulations.

Dwell Times: Dwell times associated with authorized parking locations were notably longer than those of other parking locations. Mail and heavy goods deliveries generally had longer dwell times due to the nature of the goods being transported.

Parking Criteria: CV drivers used three main criteria for choosing a parking location: avoiding unsafe maneuvers, minimizing conflicts with other road users and competition with other commercial drivers. These factors influence the decision-making process for selecting parking spots during deliveries.

These examples illustrate the diverse behaviors and challenges faced by commercial vehicle drivers in urban delivery operations, shedding light on the complexities of parking decisions and their impact on last-mile logistics.

For this thesis, parking constraints are not treated as a side issue but as a core element of the optimization model. Incorporating parking availability and associated walking distances into the decision process is essential to avoid overly optimistic travel times and to design feasible delivery tours for dense urban areas.

2.4.The Bin Packing Problem and Trolley Packing Constraints

The Bin Packing Problem (BPP) is a classical problem in combinatorial optimization and is known to be NP-hard. It involves assigning a set of items, each with a given weight (or size), to the minimum number of bins such that the total load of each bin does not exceed its capacity. The objective is to minimize the number of bins, subject to the constraint that the sum of the item weights in each bin does not exceed the bin capacity (Delorme et al., 2016).

A wide range of solution approaches has been developed:

- Exact methods such as Integer Linear Programming, Branch-and-Bound and Branch-and-Price guarantee optimal solutions but become computationally intractable for larger instances.

- Heuristics such as First Fit Decreasing (FFD), Best Fit Decreasing (BFD), Next Fit and greedy strategies provide fast, good-quality solutions but without optimality guarantee.
- Metaheuristics including Genetic Algorithms, Simulated Annealing and Ant Colony Optimisation offer scalable approaches for large instances and often yield near-optimal solutions within reasonable time.

While the BPP has been studied extensively, its classical formulation does not fully capture the trolley packing constraints relevant in last-mile delivery operations. In this context, packages are not only limited by weight but also by three-dimensional size, shape and stacking rules. Additional operational considerations include:

- Volume and dimensions: Items must fit within the finite space of the trolley, respecting length, width and height constraints.
- Stability and balance: Heavy packages must typically be placed at the bottom, while fragile items require careful positioning.
- Accessibility: Parcels may need to be arranged to facilitate delivery order, which introduces sequencing constraints absent from the classical BPP.
- Human factors: Trolley weight limits are tied to ergonomic and safety regulations, further constraining feasible packings.

In short, trolley packing generalises the classical BPP by adding further constraints. While exact algorithms are rarely used at scale, many of the heuristic and metaheuristic approaches developed for the BPP can be adapted to this setting.

For this thesis, trolley packing serves as a critical subproblem: the way parcels are packed determines how many customers can be served in a single walking tour, which in turn influences the number of parking stops required. Therefore, integrating packing constraints into routing decisions is essential for producing feasible and realistic delivery plans.

3. Problem Formulation and Models

3.1.TSP with Parking Constraints (CDPP)

Building on the discussion of parking constraints and the TSP/ATSP, Reed, Campbell and Thomas (2024) introduce the Capacitated Delivery Problem with Parking (CDPP) to capture a key operational reality of last-mile delivery: parking search time. The CDPP model incorporates parking time in the objective function and minimizes the completion time of the delivery tour by considering when and where to park the vehicle.The study provides insights into the trade-offs between walking and driving, the impact of parking time on routing decisions and the development of a heuristic to quickly find high-quality

solutions for the CDPP. The research highlights the importance of explicitly incorporating parking decisions in routing models to improve last-mile delivery practices.

Furthermore, the CDPP generalizes existing models in last-mile delivery literature, making the mixed-integer programming (MIP) formulation challenging to solve. The inclusion of parking time introduces additional constraints and variables, making the optimization problem more complex and computationally demanding. This complexity may lead to computational limitations, especially for large instances of the CDPP.

Moreover, the challenges introduced by the CDPP extend beyond computational complexity. Operational decisions such as where to park and how often to park become crucial factors in improving the productivity of the delivery person. These decisions require a deep understanding of customer locations, parking availability, and the trade-offs between driving, walking, and parking time. The paper primarily focuses on parking constraints in last-mile delivery optimization and does not specifically address packing constraints.

The mathematical formulation for the Parking Assignment and Routing (PA-R) problem in Reed et al. (2024) includes the objective function and constraints as follows:

$$\min \sum_{i \in \Pi \setminus \{0\}} p \cdot \hat{p}_i + \sum_{i \in \Pi \setminus \{0\}} \sum_{k \in C} W(i,k) \cdot \hat{a}_{ik}$$

$$\text{s.t.} \sum_{i \in \Pi \setminus \{0\}} \hat{a}_{ik} = 1 \qquad \forall k \in C$$

$$\hat{a}_{ik} \leq \hat{p}_i \qquad \forall i \in \Pi \setminus \{0\}, k \in C$$

$$\hat{p}_i \leq \sum_{k \in C} \hat{a}_{ik} \qquad \forall i \in \Pi \setminus \{0\}$$

$$\hat{p}_i \in \{0,1\} \qquad \forall i \in \Pi \setminus \{0\}$$

$$\hat{a}_{ik} \in \{0,1\} \qquad \forall i \in \Pi \setminus \{0\}, k \in C$$

Figure 1.2: PA-R (Reed et al.) model formulation

Key simplifying assumptions include:

- The model assumes a single delivery person and vehicle serving a set of customers.
- The cost of finding a parking spot is represented by the parking time p.
- The model aims to minimize vehicle travel + parking time at chosen stops + walking service cost.
- Customers are assigned to parking spots based on optimization decisions.
- The model does not consider time windows for customer service or fleet optimization.

These assumptions and simplifications help simplify the model and focus on the impact of parking time in last-mile delivery optimization and can help draw a comparison to a TSP with parking constraints.

In practice, many delivery services use a “park-and-loop” approach. The driver parks the vehicle and then completes a walking loop to serve a group of nearby customers before returning to the vehicle. The size of these loops usually depends on how difficult it is to find parking, how far the customers are spread out and how much the delivery person can carry at once. Models like the CDPP are basically trying to capture these kinds of decisions mathematically. By adding trolley-packing constraints, the model comes closer to what actually happens in dense urban areas, where loop size is limited not only by distance but also by how much fits on the trolley.

The CDPP is the closest match to my problem, so I use it as the starting point. However, it handles capacity in a fairly simple way. In this thesis I extend it by adding explicit trolley-packing constraints (volume and weight). This ensures that feasible walking subsets depend not only on proximity to the parking spot but also on what can realistically be packed and safely transported. In short, I combine the packing feasibility from the previous section with CDPP’s parking-focused routing to get a model that better reflects dense city conditions.

3.2.TSP with Trolley Packing Constraints

Having outlined parking and packing separately, I now turn to how existing models have tried to integrate such constraints. The traveling salesman problem with trolley packing constraints can be viewed as a two-echelon capacitated vehicle routing problem (VRP) because the delivery process is divided into two stages. In the first stage, the main vehicle is loaded at a depot with packages designated for a set of customers. At the second stage, at an intermediate point (a parking spot), selected packages are consolidated into a movable trolley and then delivered by foot to the customers. Both the capacity limitations of the main vehicle and the trolley must be considered. This ensures that the delivery time and transportation costs are minimized.

As a basis for this problem we can consider the article titled "The Two-Echelon Capacitated Vehicle Routing Problem" (Gonzales-Feliu et al., 2008) which introduces the Two-Echelon Capacitated Vehicle Routing Problem (2E-CVRP), which is an extension of the classical Vehicle Routing Problem (VRP). In this model, freight delivery is managed through a two-level system involving a depot, intermediate depots (satellites) and customers. The primary objective is to minimize the total transportation costs while adhering to the capacity constraints of both vehicles and satellites.

The objective function of the model aims to minimize the total transportation cost, which includes the traveling costs and handling operations at the satellites while satisfying capacity constraints of the vehicles in each level. It can be expressed as:

$$\min \sum_{i,j \in V_0 \cup V_s, i \neq j} c_{ij} x_{ij} + \sum_{k \in V_s} \sum_{i,j \in V_s \cup V_c, i \neq j} c_{ij} y_{ij}^k + \sum_{k \in V_s} S_k D_k$$

Where:
(c_{ij}) is the cost of traveling from node (i) to node (j)
(x_{ij}) is the number of 1st-level vehicles using the arc ((i, j))
(y_{kij}) is a binary variable indicating if a 2nd-level vehicle makes a route starting from satellite (k) to customer (j)
(S_k) is the cost for loading/unloading operations at satellite (k)
(D_k) is the total freight passing through satellite (k)

An assumption which is made is that for each level the vehicles at that level have the same capacity (are homogenous). Also, the fleet size at each level is fixed but the number of vehicles assigned to each satellite is not predetermined. There is one depot and a set number of capacitated satellites. All customer demands are fixed, known beforehand and must be fully met. Additionally, there are no time windows specified for deliveries or satellite operations. At the second level, each customer's demand is less than the capacity of a single vehicle and cannot be divided across multiple routes at the same level.

The constraints can be summarized shortly as:

1. Vehicle Route Constraints: Each 1st-level route begins and ends at the depot. The number of routes in each level must not exceed the number of available vehicles.

2. Capacity Constraints: Constraints limit the maximum number of 2nd-level routes starting from each satellite, which also limits the freight capacity of the satellites.

3. Flow Balance Constraints: The flow balance on each node must equal the demand of that node, except for the depot and satellites.

4. Subtour Elimination Constraints: Constraints are included to prevent subtours that do not contain the depot or a satellite.

5. Assignment Constraints:A customer can only be served by a satellite if it receives freight from that satellite.

6. Non-negativity Constraints: The flow variables must be non-negative.

The mathematical model is designed to ensure that all customer demands are met while optimizing the routing and handling of freight through the two-echelon system.

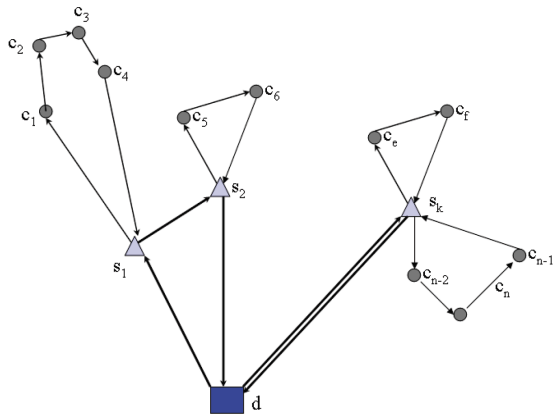

Figure 1.3: Example of 2E-CVRP transportation network

The key assumptions and simplifications in existing models of the Two-Echelon Capacitated Vehicle Routing Problem (2E-CVRP) include:

1. Fixed fleet: The number of vehicles at both levels is fixed.
2. Single freight type: All vehicles are identical and carry the same type of goods.
3. No time windows: Deliveries are not restricted by timing requirements.
4. No demand splitting: Each customer must be served in full by a single vehicle.
5. Satellite capacity: Each satellite has a maximum handling or routing capacity.
6. Static demand: Customer demands are known in advance and do not change.
7. Simplified higher-level routing: First-level routing is modelled in a simplified way, which may underestimate its cost.

The 2E-CVRP is a useful way to think about two-level distribution systems, but it does not fully match the realities of dense urban last-mile delivery with trolleys. In real operations, the “satellites” are not fixed depots but parking spots chosen by the driver, the second stage is done on foot rather than with vehicles, and each walking loop is limited not just by distance but also by trolley capacity in terms of volume, weight, and handling. On top of that, standard two-echelon models usually ignore parking search time, which plays a big role in urban efficiency. Because of these gaps, there is a need for a model that combines parking decisions, walking tours, and packing feasibility. This forms the basis for the model developed in the next chapter.

3.3.Integrated Model for Last-Mile Delivery

In this thesis, I aim to develop a model that combines the Travelling Salesman Problem (TSP) with both parking constraints and trolley bin-packing constraints. The goal is to optimize not only the vehicle's route but also the walking tours completed with a trolley. This approach reflects the realities of urban last-mile delivery, where efficiency depends on respecting walking distance limits, trolley capacity, and vehicle routing decisions.

The complexity of this model comes from combining these elements. The TSP is already NP-hard, and adding packing constraints increases the difficulty considerably. At the same time, truck routes and walking loops need to be optimized together, which makes the solution space very large—especially when many parking spots and possible customer groupings are involved. Because of this, exact integer programming can quickly become infeasible for larger problem instances, meaning that heuristics, metaheuristics, or matheuristic approaches (such as branch-and-cut) are more practical.

A further challenge is the link between parking choices and customer subsets. Parking can either occur directly at customer locations or at nearby points, which requires preprocessing and dynamic assignment. On top of that, parking time depends on location, adding extra non-linearities to the problem. In practice, urban conditions such as traffic or congestion may also force re-optimisation during operations.

Overall, a unified model that integrates parking and packing offers a much more realistic representation of urban delivery, but it also raises serious computational challenges. The key will be to strike a balance between solution quality and computational efficiency, especially for large-scale scenarios.

4. Solution Approaches

As described in earlier sections, the TSP is a well-known combinatorial optimization problem that seeks to determine the shortest possible route that visits a set of cities exactly once and returns to the original city. Various exact solution methods have been developed to tackle this problem, each with its own applicability and performance differences. In this section I will explore the available solution methods to the classical TSP problem described in the literature for exact, heuristic, metaheuristic and hybrid approaches.

4.1.Exact Algorithms

The following exact solution methods are described by (Laporte, 1992).

- (ILP) Integer Linear Programming

The TSP can be formulated as an ILP problem, where binary variables represent whether an edge between two nodes is included in the tour. The objective is to minimize the total travel cost while ensuring that each city is visited exactly once. The classic formulation by Dantzig, Fulkerson, and Johnson (1954) is a foundational approach in this area.

- Branch and Bound

This method systematically explores the solution space by dividing it into smaller subproblems (branches) and calculating bounds on the best possible solution within those subproblems. If a subproblem's bound exceeds the current best solution, it can be discarded (pruned). This approach is effective for smaller instances of the TSP and can be enhanced with valid inequalities to tighten the bounds.

- Cutting Plane Methods

These methods involve solving a relaxed version of the ILP and iteratively adding constraints (cuts) to eliminate infeasible solutions without excluding any feasible ones. The cutting planes can be derived from various formulations, such as the Miller-Tucker-Zemlin (MTZ) constraints, which help in eliminating subtours. MTZ is compact but typically weaker than subtour-elimination/comb cuts used in modern branch-and-cut.

- Dynamic Programming

The Held-Karp algorithm is a well-known dynamic programming approach that solves the TSP in O(n^2 * 2^n) time. It uses a recursive relation to build solutions for smaller subsets of nodes and combines them to find the optimal solution for the entire set. This method is particularly effective for smaller instances due to its exponential time complexity.

- Branch-and-Cut (Concorde)

This approach combines branch-and-bound with cutting planes. It is particularly effective for large instances of the TSP, as it can solve problems with thousands of vertices by efficiently managing the search space and incorporating cuts dynamically.

Exact methods are applicable to the TSP when an optimal solution is required, as these methods guarantee finding the best possible tour and the problem size is relatively small (typically up to 100 nodes for ILP and branch-and-bound methods).

The performance of exact solution methods can be evaluated based on:

- Computational Time: Exact methods often have exponential time complexity, making them impractical for large instances. For example, the Held-Karp algorithm has a time complexity of O(n^2 * 2^n), which limits its use to smaller problems. (Laporte, 1992)
- Optimality: Exact methods guarantee finding the optimal solution, which is a significant advantage over heuristic methods that may only provide approximate solutions.
- Scalability: While exact methods can solve small to medium-sized instances effectively, their performance worsens rapidly as the number of nodes increases. For larger instances, heuristic or approximation methods may be preferred due to their faster execution times, despite the lack of guaranteed optimality.

In conclusion, exact solution methods for the TSP provide a robust framework for finding optimal solutions, particularly for smaller instances. However, their applicability is limited by computational complexity, therefore the use of heuristic approaches is needed for larger problem sizes.

4.2.(Meta-)heuristic Methods

Before introducing the metaheuristic approaches commonly used for the TSP, it is important to note that these methods are not applied directly to my thesis. They are included here to provide context and to show which classes of algorithms are typically used when dealing with large routing problems. In this work, I rely on a simpler scoring-based heuristic for forming feasible walking subsets under trolley capacity and proximity constraints, which is the core step in my implementation. Such constructive heuristics are often used in packing and bin-packing settings because they are fast and easy to adapt. The discussion of metaheuristics below therefore serves mainly as background and helps position my scoring heuristic within the broader landscape of heuristic methods.

Heuristic approaches can handle larger instances of TSP that are impractical for exact algorithms, making them suitable for real-world applications. Many heuristics can be adapted to incorporate additional constraints or variations of the TSP, such as the combined TSP, where multiple objectives or constraints are considered. Heuristics typically provide solutions in a reasonable timeframe which is crucial for applications requiring quick decision-making.

However, heuristic methods do not guarantee optimal solutions, especially for larger problem sizes. Also, the performance of many heuristics is sensitive to the choice of parameters which can require extensive experimentation and tuning. Additionally, heuristics may get trapped in local optima, particularly in complex landscapes, leading to missed opportunities for better solutions.

Raman, 2017 summarizes the following heuristic approaches which are commonly applied to tackle the TSP:

- Genetic Algorithm (GA)

GA is inspired by the process of natural selection and employs mechanisms such as selection, crossover, and mutation to evolve solutions over generations. It is particularly effective for optimization problems, including TSP. GAs can adapt to various problem constraints and have shown good performance for small to mid-sized problems. On the negative side, GAs may suffer from premature convergence, where the population becomes too similar and fails to explore new areas of the solution space. They also require careful tuning of parameters (e.g., mutation rates, population size) to achieve optimal performance, which can be time-consuming.

- Ant Colony Optimization (ACO)

ACO mimics the foraging behavior of ants, using pheromone trails to guide the search for optimal paths. It is a meta-heuristic approach applicable to various optimization problems and has shown success in both small and large problem instances. ACO can be slow to converge, especially in larger problem instances, and may get trapped in local optima. The performance heavily relies on the proper tuning of parameters, such as pheromone evaporation rates and heuristic information.

- Simulated Annealing (SA)

SA is inspired by the annealing process in metallurgy and is effective in escaping local optima by allowing worse solutions at the beginning of the search. It can explore the solution space more broadly and has a probabilistic approach to finding solutions. On the negative side, the performance of SA is highly dependent on the cooling schedule and initial temperature settings. If not properly configured, it may converge too quickly or take too long to find a solution.

- Tabu Search (TS)

Tabu Search enhances local search methods by maintaining a list of previously visited solutions to avoid cycling back to them. It effectively explores new areas of the solution space and can lead to high-quality solutions. The effectiveness of Tabu Search relies on

the design of the tabu list and the neighborhood structure. It can be, however, computationally intensive and may require significant tuning to achieve optimal results.

- The Lin–Kernighan–Helsgaun (LKH)

LKH implements variable-depth k-opt with gain-guided backtracking and pruning, and restricts the search to candidate edges ranked by α-nearness computed from minimum 1-trees; the α-value of an edge is the increase in the 1-tree cost when the edge is forced (Helsgaun, 2000; 2009).

- Adaptive Large Neighborhood Search (ALNS) and Variable Neighborhood Search (VNS)

Beyond classical metaheuristics, Adaptive Large Neighborhood Search (ALNS) and Variable Neighborhood Search (VNS) are widely used for complex vehicle routing problems. ALNS adaptively selects destroy-and-repair operators based on their performance, making it effective for problems with side constraints (Ropke & Pisinger, 2006). VNS systematically changes neighborhood structures to escape local optima and improve diversification (Hansen & Mladenović, 2001). Both are regarded as state-of-the-art for large VRPs and offer flexibility that makes them promising for adapting to the TSP with parking and packing constraints considered in this thesis.

Computational time

SA: typically fastest among the three in the reported experiments. Hussain et al., reports SA had the shortest execution time (<1s on their tests) (Hussain et al., 2017). Antosiewicz reports SA processes the largest number of solutions per time unit (millions in 100s) and attains good solutions quickly (Antosiewicz, 2013).

GA: moderate runtime; population processing increases per‑iteration cost. Antosiewicz shows population methods tend to process far fewer solutions per unit time than single‑solution methods (Antosiewicz, 2013). Hussain et al., reports GA faster than ACO but slower than SA on their benchmarks (Hussain et al., 2017).

ACO: generally slowest in the Hussain experiments - gave best distances but with the longest execution times (Hussain et al., 2017). Population‑style and pheromone computations increase per‑iteration cost.

Convergence behavior and stability

SA: converges quickly in limited time (fast traversal of many candidate solutions) and often reaches very good solutions, but can show higher run-to-run variability (Antosiewicz, 2013). Hussain notes SA's fast convergence but sometimes not the best final distance (Hussain et al., 2017).

GA: tends to give more stable results (lower variance) across runs when well designed; convergence speed depends on population size and operators. Antosiewicz reports GA and tabu search produced stable runs; GA had relatively low standard deviation (Antosiewicz, 2013).

ACO: often finds high-quality solutions (good accuracy) and can converge to excellent tours, but convergence is slower (requires more time/iterations) and runtime/quality sensitive to parameters (pheromone, evaporation, number of ants). Hussain emphasizes ACO produced best distances but at much higher execution time (Hussain et al., 2017).

Overall takeaways

SA: fastest, very effective in short runs, often best mean quality in limited time but can show variability.
GA: stable and tunable, solution quality depends strongly on operator choice, moderate runtime.
ACO: can deliver best final tour lengths but needs more computational time and careful parameter tuning.

Although metaheuristics provide strong performance for many TSP variants, the scoring heuristic used later in this thesis was chosen for its simplicity and its ability to directly handle packing feasibility before solving the main routing problem.

5.Computational Studies

Computational studies play a central role in evaluating the performance of algorithms for the Traveling Salesman Problem (TSP) and its variants, including those with parking or packing constraints. Three representative studies are particularly relevant: Pasquier et al. (2007), Antosiewicz et al. (2013), and Reed and Campbell (2024). Together, they provide insights into benchmark datasets, comparative algorithmic performance, and the role of modeling parking time in last-mile delivery.

Benchmark instances

Classical benchmarks such as TSPLIB remain the standard for testing TSP algorithms (Antosiewicz et al., 2013). Pasquier et al. (2007) implemented metaheuristics on TSP instances of 30 cities, focusing on solution quality and runtime. Reed and Campbell (2024) extend benchmarking to urban grid-based customer networks for the Capacitated Delivery Problem with Parking (CDPP), capturing realistic delivery geographies and explicitly modeling parking times. These studies illustrate the range of benchmarks from small-scale synthetic datasets to structured urban delivery settings.

Performance comparison

The three studies consistently highlight the trade-off between solution quality and computational time. Pasquier et al. (2007) compared Genetic Algorithms (GA), Ant Colony Optimization (ACO), and Simulated Annealing (SA), finding that carefully tuned GA and ACO produced near-optimal solutions, whereas SA was less consistent. Antosiewicz et al. (2013) broadened the analysis to six metaheuristics (GA, SA, Tabu Search, Quantum Annealing, Particle Swarm Optimization, and Harmony Search), concluding that SA and Tabu Search outperform newer methods when computational time is limited, with SA yielding the best quality solutions and Tabu Search converging more quickly. Finally, Reed and Campbell (2024) demonstrate that explicitly modeling parking time changes the optimal routing structure: when parking times exceed 1.6 minutes, the CDPP outperforms a standard TSP approach, showing that ignoring parking constraints underestimates total delivery time.

Tools and software

Computational experiments relied on a mix of general-purpose solvers and custom implementations. Antosiewicz et al. (2013) validated metaheuristic performance against Concorde, a leading exact TSP solver, which ensured that the heuristic results could be benchmarked against optimal solutions. Pasquier et al. (2007) emphasized implementation efficiency, comparing algorithm runtime and code complexity to assess ease of deployment. Reed and Campbell (2024) developed both Mixed-Integer Programming (MIP) formulations and a tailored heuristic for the CDPP, demonstrating scalability limits of exact solvers and the necessity of heuristics in realistic urban delivery contexts.

Study	Problem Focus	Benchmarks	Methods	Key Findings	Tools/Solvers
Pasquier et al. (2007)	Classical TSP	30-city test maps	GA, ACO, SA	GA & ACO near-optimal; SA less stable	Custom implementations
Antosiewicz et al. (2013)	TSP under time limits	TSPLIB, medium-size instances	GA, SA, Tabu, PSO, Harmony, QA	SA best quality; Tabu fastest convergence	Compared to Concorde
Reed & Campbell (2024)	CDPP (TSP + parking time)	Urban grid networks	MIP + heuristic	Parking >1.6 min alters optimal strategy; heuristics scale better	MIP solver + custom heuristic

Figure 1.4: Metaheuristics performances

In summary, the computational evidence highlights that while metaheuristics remain the most practical approach for large-scale TSPs, their performance depends on problem size, time restrictions, and additional constraints. Importantly, studies such as Reed and Campbell (2024) demonstrate that incorporating realistic features like parking time fundamentally alters computational outcomes, reinforcing the need for problem-specific benchmarking in last-mile delivery optimization.

6.Discussion

6.1.Current trends

From the studies reviewed, one clear trend is that metaheuristic algorithms are still the most common and practical way of solving the TSP and its variants. Pasquier et al. (2007) tested Genetic Algorithms, Ant Colony Optimization, and Simulated Annealing on small TSP instances and found that GA and ACO were able to get very close to optimal solutions, while SA gave less consistent results. A few years later, Antosiewicz et al. (2013) ran a much broader comparison of six different metaheuristics and came to a similar conclusion: even though new methods such as particle swarm optimization and harmony search have become popular, “classic” algorithms like Simulated Annealing and Tabu Search actually performed better when there was limited time to compute a solution. This indicates that despite the appearance of new techniques, older metaheuristics are still very competitive, especially when decisions need to be made quickly.

At the same time, more recent work has started to move away from purely abstract TSP formulations and look at more realistic delivery settings. Reed and Campbell (2024) introduced the Capacitated Delivery Problem with Parking (CDPP), which for the first time includes the time it takes drivers to find and secure parking spots. Their computational experiments showed that parking time can significantly change the best delivery strategy: if parking takes more than about 1.6 minutes, the traditional approach of parking at each customer becomes less efficient. This finding highlights that adding

real-world elements such as parking into routing models can completely change the results, something that standard TSP models tend to ignore.

6.2.Research gaps

While these studies provide valuable insights, there are still some important gaps. First, most computational studies of the TSP are based on overly optimistic conditions. For example, both Pasquier et al. (2007) and Antosiewicz et al. (2013) test algorithms on standard datasets like TSPLIB, where the assumption is that every customer can be directly accessed by vehicle. This ignores the very real challenges faced in dense cities, such as traffic, limited parking, and the need for delivery workers to walk part of their route.

Second, while Reed and Campbell (2024) take an important step by showing that parking time matters, their model does not consider package handling constraints. In practice, delivery workers often rely on hand trolleys to deliver multiple packages at once, and these trolleys have weight and volume limits. Decisions about where to park are therefore closely linked to how much can actually be carried on a trolley in a single walking loop, yet this connection has not been explored in the literature.

The most significant gap, then, is that parking and packing constraints have not been studied together. The algorithmic studies (Pasquier et al., 2007; Antosiewicz et al., 2013) focus only on computational performance without considering urban delivery realities, and the parking study (Reed & Campbell, 2024) looks only at parking in isolation. In practice, though, the two are deeply linked: if a trolley can only carry a few packages, more parking stops will be needed; if parking is hard to find, drivers may try to load the trolley more heavily to avoid moving the truck. This interdependence is exactly what is missing in the current research.

6.3.Relevance to this thesis

This thesis aims to fill that gap by developing a model that brings both parking availability and trolley packing constraints into a single optimization framework. In doing so, it builds directly on Reed and Campbell's (2024) insight that parking has a major effect on delivery efficiency, but extends their work to account for how much can actually be carried on a trolley at one time. At the same time, the thesis will draw on lessons from metaheuristic studies such as Pasquier et al. (2007) and Antosiewicz et al. (2013), since these methods are best suited to handling large, complex problems where exact solutions are unrealistic. By integrating these two strands of research, the thesis hopes to bridge the

gap between abstract computational experiments and the reality of last-mile delivery in crowded urban areas.

7.Future Directions

This thesis provides a baseline model for last-mile delivery with both parking and trolley packing constraints, but several natural extensions can make it more realistic and open up new research opportunities. At the same time, recent work on metaheuristics in transportation shows that optimization research is moving toward more complex, data-driven, and sustainable models (Bueno-Ferrer et al., 2025).

One straightforward extension concerns the packing problem itself. In the current model, packages are simplified to two dimensions (length × height) plus weight. In practice, however, trolleys must handle full 3D arrangements where stability matters - e.g., heavy items need to be placed at the bottom, fragile packages should go on top, and only certain orientations are feasible. Extending the model to a proper 3D bin packing problem would therefore be more realistic. As Bueno-Ferrer et al. (2025) note, “metaheuristics have repeatedly proven effective in addressing complex combinatorial optimization problems”, so heuristic or matheuristic strategies such as first-fit-decreasing or hybrid greedy + MIP methods could be a practical way to start.

Additionally, there is a methodological opportunity. As the Buenno-Ferrer (2025) study emphasizes, “the ongoing evolution of metaheuristics, combined with advances in AI and machine learning, is shaping the future of transport optimization”. For this thesis, metaheuristics such as Simulated Annealing, Tabu Search, or Genetic Algorithms already look promising. But future work could explore hybrid AI–metaheuristic approaches, for example, combining reinforcement learning for parking-time prediction with a metaheuristic for routing and packing. This would connect operational decision-making with predictive analytics, something the literature identifies as a major trend.

8.Conclusion

In summary, the literature on last-mile delivery and the TSP provides valuable insights but also reveals important gaps. While classical models such as the TSP and 2E-CVRP offer useful structures, they often simplify or ignore practical issues like parking availability, walking distances, and trolley capacity. Studies such as Reed and Campbell

(2024) highlight the importance of parking constraints, but do not consider packing feasibility, while bin-packing research addresses capacity without linking it to routing. Similarly, computational work shows that metaheuristics are the most practical tools for large-scale problems, yet they are rarely applied to integrated parking-and-packing scenarios. This thesis addresses these gaps by developing a model that combines routing, parking, and trolley packing into a single framework and by exploring suitable heuristic and metaheuristic solution approaches.

II.Part Two

1.Model Formulation

1.1.Model Description

I will begin by presenting a comprehensive overview of the problem, outlining its primary goals and challenges. Following this general description, the problem is deconstructed into a two-stage approach: a separate model formulation for the bin packing subproblem together with a TSP-like routing, which focuses on identifying feasible customer service sets, their designated parking location and walking routing and, secondly, the traveling salesman problem, which optimizes the driving routing of these service sets.

TSP - PTP (Traveling Salesman Problem with Parking and Trolley Packing Constraints)

This is a traveling salesman problem with parking and packing constraints. At the first stage, the vehicle is loaded at a depot with packages designated for a set of customers. At the second stage, at an intermediate point (a parking spot), selected packages are consolidated into a movable trolley and then delivered by foot to the customers. The time to search a parking spot and park is fixed. The loaded packages need to comply with the length, height, width and weight constraints of the trolley and a walking distance proximity constraint. The postal worker serves customers using a hand trolley on a route which takes the shortest amount of time starting at the parked vehicle, visiting each customer once and returning to the parked vehicle to load packages for the next route until all packages are delivered. The tour starts and ends at the depot.

Loading and Driving Stage:

- A vehicle is loaded at a depot with packages designated for a set of customers.
- A parking spot is determined for each feasible subset in advance.

- At an intermediate point (a parking spot), the vehicle must park and selected packages are consolidated into a movable trolley for on-foot delivery.

On-foot Delivery Stage:

- The tour taken by the trolley starts and ends at the parked vehicle's location, and all customers on the trolley route must be visited exactly once. The route is predetermined in advance to minimize walking time.

Objective:

- The objective function minimizes the total service time, which is a combination of the truck's driving time and a service cost at each candidate stop (this service cost consists of the fixed parking time plus the walking time).

Decisions:

- Determine the sequence in which the customers are visited in the walking tour (solved via a separate TSP routing problem).
- Determine which packages are loaded onto the trolley for the tour, depending on the size of the packages and the capacity of the trolley with regard to size and weight (solved via a separate bin packing problem).
- Select which pre-defined parking stops (from the feasible subsets) are visited and in what order (solved via a separate TSP routing problem).
- Determine the sequence in which the customers are visited in the truck tour.

Constraints:

- Customer coverage: Each customer must be assigned to exactly one feasible service set (subset).
- Visit each customer exactly once (walking): Every customer in a selected subset must be included in the walking tour exactly once. Formally, for n customers, each customer i must be visited exactly once, where i ranges from 1 to n.
- Form a tour: The walking tour must be a single cycle that starts and ends at the parked vehicle. This ensures that there are no disjoint cycles or sub-tours within the solution.
- Trolley capacity: Packages loaded onto the trolley must fit within the trolley's constraints, including length, width, height, and weight.

1.2.Model Formulation

Bin Packing Problem (Separate Subproblem) with TSP-like routing:

The bin packing problem needs to be solved separately to determine the feasible customer service sets Sk that can be delivered on foot in a single trip, keeping both the trolley and proximity constraints. Then for each feasible subset a designated parking location is selected which then optimizes the walking route for each subset and selects the shortest loop for each subset. The resulting customer sets will be implemented later into the main traveling salesman problem.

Parameters:

C: Set of all customers

wi: Weight of the package for customer $i \in C$

vi: Volume of the package for customer $i \in C$

Wmax: Maximum weight capacity of the trolley

Vmax: Maximum volume capacity of the trolley

(xi, yi): Coordinates of customer $i \in C$

Dmax: Maximum allowed walking distance for the entire service set

Nmax: Maximum number of customers allowed in a single feasible set

Definitions:

k: Index of a feasible subset

S□: Set of customers contained in feasible subset *k*, where $S\square \subseteq C$

p□: Parking location assigned to subset *k*

π□: Permutation of customers in $S\square$ that defines the walking sequence

Constraints:

1. Weight capacity constraint of the trolley:

$$\sum_{i \in S_k} w_i \leq W_{\max}$$

2. Volume capacity constraint of the trolley:

$$\sum_{i \in S_k} v_i \leq V_{\max}$$

3. TSP Tour Length (Proximity) Constraint using the Euclidean distance formula:

For subset Sk with parking location pk and customer order πk, the walking tour length is defined as:

$$\text{TourLength}(S_k) = d(p_k, \pi_k(1)) + \sum_{j=1}^{|S_k|-1} d(\pi_k(j), \pi_k(j+1)) + d(\pi_k(|S_k|), p_k)$$

where

$$d(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

The total walking distance must satisfy:

$$\mathrm{TourLength}(S_k) \le D_{\max}$$

4. Customer count constraint

$$|S_k| \le N_{\max}$$

TSP routing for each feasible subset

For a given feasible subset Sk and a designated parking location p ∈ Sk:

1.Define the set of customers to be served in the route

$$S'_k = S_k \setminus \{p\}.$$

2.Route representation (The full route (loop) starts at p, visits all customers in the order determined by π, and returns to p)

$$\mathrm{Route}(\pi) = (p,\, \pi(1),\, \pi(2),\, \ldots,\, \pi(|S'_k|),\, p)\,.$$

4. **Objective**: For each Sk, select the designated parking location p ∈ Sk and determine the permutation π that minimizes the walking tour:

$$\min_{p \in S_k,\ \pi \in \text{Permutations}(S_k \setminus \{p\})} TWD(S_k, p, \pi),$$

The output will be a set of feasible customer service sets S = (S1, S2, …,Sk) which meet the constraints as well as their assigned designated parking location in a way that minimizes the walking tour for each subset. These will be used in the main problem to select which sets are to be served.

Main model formulation (TSP with parking constraints)

Sets:

- P: Set of parking locations, the depot is designated as $0 \in P$
- C: Set of customers, where $C \subseteq P \setminus \{0\}$
- **S:** Set of pre-computed feasible service subsets (obtained via the bin packing model)

  For each subset $S_k \in S$:

  $S_k \subseteq C$ is the set of customers in the subset k

  $p_k \in P \setminus \{0\}$ is the designated (randomly assigned) parking location for S_k

  $D_k \in R+$ is the pre-computed minimal walking tour cost for S_k (the TSP loop that starts and ends at p_k)

Parameters:

- t_{ij}: Travel time between parking locations i and j (either distance or time), for i,j $\in$ P
- parking_time_i: Time required to park at location i $\in$ P
- n = $|P|$: Total number of parking locations

Decision Variables:

- x_{ij}^P: Binary variable, 1 if the truck travels directly from parking location i to parking location j, 0 otherwise
- u_i^P: Continuous variable, representing the order in which the truck visits parking location i
- z_{S_k}: Binary variable, 1 if subset S_k $\in$ S is selected, 0 otherwise

Objective Function:

Minimize the total time, which includes travel time, parking time and walking time for the selected subsets:

$$\min \quad \sum_{i,j \in P} t_{ij}\, x_{ij}^{P} \; + \; \sum_{i \in P} \text{parking\_time}_i \left(\sum_{j \in P} x_{ij}^{P} \right) \; + \; \sum_{S_k \in S} D_k \, z_{S_k}$$

Constraints:

1. **Depot Start:**

$$\sum_{j \in P \setminus \{0\}} x_{0j}^{P} = 1.$$

2. **Depot End:**

$$\sum_{i \in P \setminus \{0\}} x_{i0}^{P} = 1.$$

3. **Truck route subtour elimination MZT**

$$u_i^P - u_j^P + n \cdot x_{ij}^P \leq n - 1 \quad \forall i, j \in P,\ i \neq j$$

4. **Parking subset linking:** ensure that the truck leaves the designated parking location pk if subset Sk is selected

$$\forall p_k \in P, \quad \sum_{j \in P} x_{p_k,j}^P = z_{S_k}$$

5. **Parking subset linking:** ensure that the truck arrives at the designated parking location pk if subset Sk is selected

$$\forall p_k \in P, \quad \sum_{i \in P} x_{i,p_k}^P = z_{S_k}$$

6. **Subset selection constraint to serve all customers (**ensure that each customer is served exactly once by one of the selected subsets:**):**

$$\sum_{S_k \in S : i \in S_k} z_{S_k} = 1, \quad \forall i \in C.$$

2.Solution Approach and Implementation

In this chapter, I first describe how the model and heuristic are implemented and combined into a practical solution approach. I then present the computational

experiments and discuss the results. Subsections 2.1 focuses on data and implementation details, while Subsections 2.2–2.5 report the experimental results. I built everything in Python 3 in Google Colab, using pandas/NumPy for data handling and OR-Tools for optimization. The implementation follows the same two-stage structure as the mathematical model formulation:

1. The Bin Packing Sub-Problem: The goal is to determine the feasible customer subsets that can be delivered on foot, complying to both packing and proximity constraints.
2. Main TSP Problem: Then subsets are selected that cover every customer exactly once and sequence stops into a single truck route that minimizes driving, parking, and walking time.

I start small to make sure everything works (10 customers), then scale up. For 50 customers, enumerating all subsets becomes heavy so I added a simple scoring heuristic that ranks feasible subsets by (normalized) volume, distance and a tiny random term for diversity. Only the top-ranked subsets feed the main model. Finally, for 100 customers, I keep the same pipeline and run sensitivity analyses on key parameters (parking time, allowed walking distance, trolley volume) to see how they shift the balance between parking, walking and driving.

The rest of the chapter walks through the code at a practical level (what files are loaded, how variables are built and how the solver is configured), then reports results for the 10-, 50-, and 100-customer experiments, including the heuristic's quality loss and the sensitivity findings.

2.1.Data and implementation setup

This section summarises all data inputs and implementation settings used for both stages of the solution approach. It describes the customer, parking, travel-time and trolley data required by the code, as well as the format in which these data are loaded into Python before running the bin-packing subproblem and the main TSP model. Since the implementation relies on pre-computed feasible subsets, this section also lists the structure of the subset files generated by the bin-packing script. Together, these inputs form the basis for the experiments shown in the following subsections.

1.Customer data

- Locations (xi,yi): $j \in C$

  Latitude and longitude (e.g. 41.40338, 2.17403)

  Source: Adams_100_1_customers (Reed)

- Package information (vj, wj) $j \in C$

  Volume and weight of the package (units: L or m^3, weight in kg)

  Note: Reed reports package counts. Volume and weight of the parcels are created synthetically by sampling parcel dimensions within sensible ranges (e.g., L: 10–100 cm, W: 10–50 cm, H: 5–50 cm; Weight: 0.1–20 kg) and computed Volume=L×W×H, https://leventozturk.com/engineering/random/

  File used: Adams_100_1_customers_packages (synthetic)

2.Parking location data

- Parking coordinates (xi,yi) :i ∈ P

  Latitude and longitude (e.g. 41.40338, 2.17403)

  Parking location = customer locations

  Source: Adams_100_1_parking (Reed)

- Parking search time P (fixed in minutes)

  Reed takes a fixed parking time depending on the area, e.g. 5 min, 9 min, vary by urban/rural context

3.Travel data

- Between nodes dij : i ∈ P, j ∈ C and tij : i, j ∈ C

  Note: Reed provides base data. In this implementation, both truck and walking times are derived from coordinates plus speed assumptions for consistency

4.Trolley data

- Trolley capacity V,W

  Maximum volume and weight the trolley can carry per walking tour.

  Example used: PT400, V=230 L (≈0.23 m3), W=30 kg

  Source: product brochure (e.g., Viadukt) ->

  http://viadukt.eu/brochures/brochure_en.pdf e.g. below

Figure 1.5: Postal trolley example

5.Customer service sets S

Each subset is a set of customers that fits the trolley constraints (V,W) and respects the maximum walking distance Dmax. Each feasible subset is associated with a designated parking location and a computed walking time/distance.

Generated by: bin-packing subproblem script

6.Depot data

- Depot coordinates (x0,y0, latitude and longitude)

  Source: available from Reed

2.2.Experiment 1: 10-customer instance

Before describing the computational steps, the following experiment settings apply to all test instances: in Stage 1 I use a planar Euclidean approximation for walking tours. In Stage 2 I use Haversine distances for truck travel. I tested this mismatch and the effect on total time was under 1% at neighborhood scale, therefore the decision was made to keep

it for simplicity and speed. Unless otherwise stated, I set walking speed to 5 km/h and use a 5-minute per-stop parking time. Solver runs are time-limited and for the larger instances I accept the best feasible solution found within the limit.

Step 1: Bin Packing Sub-Problem (feasible subsets generation)

I implement Stage 1 in Python (Google Colab, Jupyter, Python 3) and test on the first 10 customers from Adams_100_1_customers. The Jupyter Notebook uses Python 3 programming language. The feasible sets are restricted both in volume and in proximity. A walking tour is considered feasible if the total distance a delivery person must walk to complete the route is less than 1.5 km or 1500m. The weight and volume of the trolley are based on the selected trolley model above 'PT400': 230l volume and 30kg weight. The instances are uploaded from a CSV file named Adams_customers10.csv. The Euclidean formula is used to calculate the distance between two latitude/longitude points. In order to shorten the execution time, the maximum subset size is limited to 5 deliveries per parking location. As a way to handle the parking location allocation, the code looks through each feasible subset and randomly chooses one customer location to be the designated parking location for this subset. The output of the code provides us with all feasible customer subsets (and their designated parking location), that are within the set maximum total distance, comply with the trolley constraints and are limited to 5 deliveries per subset.

Assumptions for the Python code:

- D_max = 1500 m (max distance allowed walking distance)

- trolley_capacity_volume = 230 l (230 liters from the selected trolley model above 'PT400')
- trolley_capacity_weight = 30 (kg from the selected trolley model above 'PT400')
- customers = latitude and longitude for 10 test instances from Adams_100_1_customers, Adams_customers10.csv files
- designated_parking = randomly select one customer from the subset as the designated parking stop

Firstly, the code imports all necessary libraries (pandas, math, etc) and then a CSV file Adams_customers10.csv is uploaded which contains all customer data (location, volume and weight). Then the customer data is converted into a dictionary for easy access by mapping each customer ID to their coordinates, volume and weight. I made sure to specify the customer ID as an integer, which will be later used to list the customer subsets. Then a function is defined to compute the Euclidean distance between two points based on their Cartesian coordinates.

The trolley constraints define the limit of the trolley's volume and weight (230 l and 30kg) and sets a maximum walking distance of 1500 m. Then a feasibility check is done to check if a subset satisfies the trolley packing constraints as well as the proximity rule (Euclidean distance). As I wanted to display the total walking distance for each subset, another function was added. The function is called tsp_min_distance and treats each subset as a TSP. It chooses the designated parking spot from the subset, generates all possible permutations in which to visit the remaining customers in the subset, calculates the route distances for each in a way that the route starts at the parking spot, visits all customers and returns back to the parking spot. The route for each subset is selected in a

way that minimizes the walking time. This function calculates the total walking distance for a single subset. The code generates all possible customer subsets (up to the specified maximum size) that satisfy both the capacity constraints and the proximity constraint - where the proximity constraint is based on the TSP tour length being less than or equal to Dmax. For each subset, the code randomly selects one customer as the parking spot. The feasible subsets, their designated parking location and total walking distance in meters are stored in a list Adams_feasible_subsets_10.csv.

Output CSV file Adams_feasible_subsets_10.csv (totals 26 feasible subset, containing both single customer and multiple customer subsets):

feasible_subsets_output10_141.csv

```
SubsetNo,ParkingLocation,CustomerIDs,TotalWalkingDistance
1,1,1,0.00
2,2,2,0.00
3,3,3,0.00
4,4,4,0.00
5,5,5,0.00
6,6,6,0.00
7,7,7,0.00
8,8,8,0.00
9,9,9,0.00
10,10,10,0.00
11,3,"1, 3",1483.80
12,2,"2, 3",626.74
13,2,"2, 8",310.92
14,2,"2, 9",573.67
15,3,"3, 8",315.82
16,3,"3, 9",53.07
17,7,"4, 7",1363.31
18,10,"4, 10",920.08
19,10,"7, 10",443.23
20,9,"8, 9",262.75
21,2,"2, 3, 8",626.74
22,3,"2, 3, 9",626.74
23,2,"2, 8, 9",573.67
24,3,"3, 8, 9",315.82
25,10,"4, 7, 10",1363.31
26,8,"2, 3, 8, 9",626.74
```

Figure 1.6: 10 customers output

Step 2: Main TSP Problem (optimal route generation)

Brief Explanation of the Code:

The code begins by installing all necessary packages (NumPy, pandas, and OR-Tools) and importing them. I then mounted Google Drive to access a CSV file containing candidate customer subsets Adams_feasible_subsets_10.csv. Each row in this CSV represents a candidate service set with a predetermined parking location, a list of customers it serves, and the total walking distance in meters associated with that stop. A preview of this data is printed for verification.

Fixed geographic coordinates for several parking locations are defined, as well as the coordinates for the customers (identified by IDs). The parking and customer locations are identical, e.g. P1 is identical to C1. The code also establishes several parameters such as the walking speed (in km/h), a fixed parking time (5 minutes), and the truck's driving speed (in km/h). In addition, a Haversine function is used to compute the driving distances (in kilometers) between any two geographic points.

It is worth disclosing, that in Stage 1 (The Bin Packing Sub-Problem), walking distances were computed using a planar Euclidean approximation, whereas here, truck travel times are based on Haversine distances. Although this introduces a slight inconsistency, when running a comparison check, the results differ by less than 1% in total time (see section Limitations) for 50 customers as we focus on single neighborhoods.

The code then processes each row from the CSV file to create a list of candidate stops. For each candidate stop it converts the parking location from a numeric format into a key (like "P1"), parses the string of customer IDs into a list of integers, records the total walking distance (from the CSV) and computes the corresponding walking time in minutes (based on the walking speed), as well as, it stores the coordinates of the fixed parking location.
It also builds a mapping for each customer that identifies which candidate stops serving that customer.

A truck routing model is set up where the truck starts and ends at a depot (with a fixed coordinate, here taken as 40.157444, -91.100838. All candidate stops become nodes in this routing model. A travel time matrix is created that specifies the truck driving time (in minutes) between every pair of nodes (the depot and each candidate stop) using the Haversine function and the truck's speed.

The optimization model is built using OR-Tools. It defines binary decision variables to determine whether each candidate stop is selected and additional binary variables to indicate whether the truck travels directly between any two nodes. To ensure a valid route without disconnected cycles, the model uses MTZ (Miller-Tucker-Zemlin) constraints for subtour elimination.

The constraints include:

- Ensuring that every customer is covered exactly once by one of the candidate stops,
- Linking the candidate stop selection with the truck routing (i.e., if a stop is selected, the route must enter and leave it exactly once),
- Guaranteeing that the route starts and ends at the depot.

The objective function minimizes the total service time, which is a combination of the truck's driving time (calculated from the travel time matrix) and a service cost at each candidate stop (this service cost consists of the fixed parking time plus the walking time).

After formulating the model, the code solves it using OR-Tools' SCIP solver. If an optimal/feasible solution is found, it prints out the total time (including driving, walking, and parking times), the details of the selected candidate stops (showing which customers are served, the parking location, and the associated walking time and distance), the truck's route in order, from the depot through the selected stops and back to the depot, the driving times for each leg of the route, along with a final summary of the total truck travel time.

Output:

```
Truck Route (in order):
Depot -> Subset 26 @ P3 -> Subset 5 @ P5 -> Subset 4 @ P4 -> Subset 10 @ P10 -> Subset 7 @ P7 -> Subset 6 @ P6 -> Subset 1 @ P1 -> Depot

Truck Travel Times Between Stops:
  Leg from Depot to Subset 26 @ P3: 1.19 minutes
  Leg from Subset 26 @ P3 to Subset 5 @ P5: 1.88 minutes
  Leg from Subset 5 @ P5 to Subset 4 @ P4: 1.57 minutes
  Leg from Subset 4 @ P4 to Subset 10 @ P10: 0.86 minutes
  Leg from Subset 10 @ P10 to Subset 7 @ P7: 0.41 minutes
  Leg from Subset 7 @ P7 to Subset 6 @ P6: 1.18 minutes
  Leg from Subset 6 @ P6 to Subset 1 @ P1: 2.45 minutes
  Leg from Subset 1 @ P1 to Depot: 0.00 minutes
Total truck travel time (driving only): 9.54 minutes
Total Parking Time: 35.00 minutes
Final Total Time (Driving + Walking + Parking): 52.06 minutes
Total Walking Time = 7.52 minutes
```

Figure 1.7: 10 customers result

2.3.Experiment 2: 50-customer instance and scoring heuristic

Step 1: Bin Packing Sub-Problem

In order to test a larger number of customers, a heuristic was applied to limit the number of feasible subsets. When running the trolley-packing subproblem to generate feasible subsets, the script resulted in over 1,900 feasible subsets. Listing all feasible subsets is computationally expensive; therefore I implemented a heuristic that scores each subset based on normalized volume, normalized distance (shorter is better), and a small random term to introduce diversity. The weighted scoring to shortlist feasible subsets is as follows:

score = (1/3)*vol_score + (1/3)*dist_score + (1/3)*rand_score

score_S = (vol_score + dist_score + rand_score) / 3

Subsets are then sorted by their score in descending order and I can decide how many of the subsets to insert into the main model in subsequent steps. The obtained solution is not guaranteed to be optimal, however, based on a later quality-loss check (presented in a later chapter), it performs well in most cases.

Below is an updated version of the Bin packing subproblem's script that replaces the lengthy generation of all feasible subsets with the proposed scoring heuristic. The instances are uploaded from a CSV file named customers50.csv which lists the first 50 customers from Adams_100_1_customers.csv. The proposed modification allows to limit the subset size to a specific number of quality subsets and overall reduces the number of subsets into a manageable set to insert into the main model. The feasible subsets, their designated parking location, total walking distance in meters and score are stored in a list Adams_feasible_subsets_scored_50.csv. The walking distance for each subset is computed by a nearest-neighbor tour initiated at a randomly selected customer, which also serves as the subset's parking label.

Output CSV file Adams_feasible_subsets_scored_50.csv (totals 1599 feasible subset, containing both single customer and multiple customer subsets):

Adams_feasible_subsets_scored_50.csv

```
SubsetNo,ParkingLocation,CustomerIDs,TotalWalkingDistance,TotalVolume,Score
1,50,"36, 38, 50",344.48,193.40,0.8371
2,38,"2, 35, 38",592.25,218.10,0.8274
3,45,45,0.00,147.00,0.8254
4,7,"7, 10, 15, 18",570.40,213.30,0.8216
5,15,"7, 15, 45",737.64,224.90,0.8189
6,8,"3, 8, 9, 36",315.82,153.40,0.8176
7,9,"9, 36, 38, 39",608.23,198.30,0.8115
8,16,"16, 21, 41",228.93,146.50,0.8114
9,28,"8, 28, 36",116.31,143.00,0.8067
10,22,22,0.00,156.30,0.8011
11,36,36,0.00,97.00,0.8009
12,15,"7, 15, 29",174.64,133.70,0.7967
13,38,"3, 9, 36, 38",543.99,181.70,0.7943
14,36,"2, 36, 50",427.23,186.10,0.7923
15,36,"17, 28, 36, 38",344.48,179.70,0.7915
16,3,"3, 28, 36, 37",396.28,152.20,0.7886
17,36,"2, 8, 36, 38",427.23,216.80,0.7793
18,35,"2, 35",509.50,166.30,0.7778
19,36,"36, 37",196.77,105.10,0.7758
20,22,"22, 31",41.82,223.30,0.7735
21,38,"8, 36, 38",344.48,172.30,0.7706
22,8,"2, 8, 36",427.23,165.00,0.7683
23,37,"8, 36, 37, 38",344.48,180.40,0.7679
24,17,"2, 17, 36",427.23,149.90,0.7666
25,29,"10, 18, 29",395.77,191.20,0.7654
26,36,"12, 36, 38",344.48,162.20,0.7621
27,35,"35, 47",252.12,154.30,0.7576
28,36,"8, 9, 36",262.75,128.80,0.7554
29,36,"17, 36, 37",306.48,113.50,0.7550
```

Figure 1.8: 50 customers subset output

To limit the 1599 scored and sorted subsets to a manageable number for the main TSP script while keeping runtime in check, I ran an intermediate greedy heuristic to keep the highest-quality subsets while guaranteeing coverage. The script (greedy_limit_subsets.py) sorts all candidate subsets by score (best first), walks down the list and picks a subset only if it covers at least one customer not yet covered. It stops when all customers are covered or when 200 subsets have been picked. If needed, the script adds synthetic single-customer subsets for any customers that still are not covered. This is less critical with 50 customers but becomes important for larger instances. The output file Adams_best_200_50c.csv selects the highest-quality subsets:

Adams_best_200_50c.csv

```
SubsetNo,ParkingLocation,CustomerIDs,TotalWalkingDistance,TotalVolume,Score,CustomerList
1,50,"36, 38, 50",344.48,193.4,0.8371,"[36, 38, 50]"
2,38,"2, 35, 38",592.25,218.1,0.8274,"[2, 35, 38]"
3,45,45,0.0,147.0,0.8254,[45]
4,7,"7, 10, 15, 18",570.4,213.3,0.8216,"[7, 10, 15, 18]"
6,8,"3, 8, 9, 36",315.82,153.4,0.8176,"[3, 8, 9, 36]"
7,9,"9, 36, 38, 39",608.23,198.3,0.8115,"[9, 36, 38, 39]"
8,16,"16, 21, 41",228.93,146.5,0.8114,"[16, 21, 41]"
9,28,"8, 28, 36",116.31,143.0,0.8067,"[8, 28, 36]"
10,22,22,0.0,156.3,0.8011,[22]
12,15,"7, 15, 29",174.64,133.7,0.7967,"[7, 15, 29]"
15,36,"17, 28, 36, 38",344.48,179.7,0.7915,"[17, 28, 36, 38]"
16,3,"3, 28, 36, 37",396.28,152.2,0.7886,"[3, 28, 36, 37]"
20,22,"22, 31",41.82,223.3,0.7735,"[22, 31]"
26,36,"12, 36, 38",344.48,162.2,0.7621,"[12, 36, 38]"
27,35,"35, 47",252.12,154.3,0.7576,"[35, 47]"
38,29,"15, 18, 29, 42",797.09,170.3,0.7339,"[15, 18, 29, 42]"
41,10,"7, 10, 15, 42, 46",1014.92,203.5,0.7281,"[7, 10, 15, 42, 46]"
42,47,"24, 47",170.52,153.9,0.728,"[24, 47]"
48,31,"26, 31",16.15,107.4,0.7193,"[26, 31]"
50,34,34,0.0,44.3,0.7177,[34]
53,23,"23, 24",270.87,142.0,0.7149,"[23, 24]"
67,40,"22, 26, 40",934.89,202.3,0.7041,"[22, 26, 40]"
75,36,"3, 19, 36",760.77,138.6,0.6992,"[3, 19, 36]"
79,49,"25, 49",81.1,41.1,0.6967,"[25, 49]"
82,7,"7, 10, 18, 27",878.97,202.7,0.6911,"[7, 10, 18, 27]"
98,41,"14, 16, 41",595.71,160.7,0.6817,"[14, 16, 41]"
99,44,"10, 15, 18, 44",742.17,191.3,0.6813,"[10, 15, 18, 44]"
104,20,"10, 20",952.04,156.6,0.6766,"[10, 20]"
116,6,"6, 48",222.17,41.2,0.6708,"[6, 48]"
```

Figure 1.9: 50 customers subset output - best

Step 2: Main TSP Problem

Importing the file above into the main model formulation results in finding the best feasible route. It is worth mentioning that I have adjusted the main model to not necessarily find the best proven solution but instead to stop at a feasible one with a time limit of 5 min.

```python
# Solve
solver.SetTimeLimit(300000)  # 5 minutes in milliseconds

status = solver.Solve()

if status in [pywraplp.Solver.OPTIMAL, pywraplp.Solver.FEASIBLE]:
    print('============================================================')
    print('                SOLUTION FOUND (Feasible or Optimal)                ')
    print('============================================================')
    total_time = solver.Objective().Value()
```

Figure 1.10: Time limit code snippet

```
Truck Route (in order):

Depot -> Subset 247 @ P1 -> Subset 40 @ P40 -> Subset 10 @ P22 ->
Subset 48 @ P31 -> Subset 19 @ P19 -> Subset 8 @ P16 -> Subset 14 @
P14 -> Subset 30 @ P30 -> Subset 116 @ P6 -> Subset 34 @ P34 ->
Subset 45 @ P45 -> Subset 27 @ P27 -> Subset 42 @ P42 -> Subset 44 @
P44 -> Subset 12 @ P15 -> Subset 18 @ P18 -> Subset 4 @ P4 -> Subset
154 @ P43 -> Subset 79 @ P49 -> Subset 11 @ P11 -> Subset 13 @ P13
-> Subset 33 @ P33 -> Subset 716 @ P5 -> Subset 42 @ P47 -> Subset
23 @ P23 -> Subset 39 @ P39 -> Subset 2 @ P38 -> Subset 50 @ P50 ->
Subset 17 @ P17 -> Subset 12 @ P12 -> Subset 37 @ P37 -> Subset 6 @
P8 -> Subset 28 @ P28 -> Depot

Total truck travel time (driving only): 13.39 minutes
Total Parking Time: 165.00 minutes
Final Total Time (Driving + Walking + Parking): 213.55 minutes
Total Walking Time = 35.17 minutes
```

Figure 1.11: 50 customers result

In order to serve all 50 customers, the postal worker would need 213.55 minutes or about 3 hours and 56 minutes. The parking time is a big portion of the total time with a total of 165 minutes.

2.4.Experiment 3: Quality-loss analysis of the scoring heuristic

The primary goal of this analysis was to evaluate the loss in solution quality resulting from this reduction. To do so, I compared results from three model configurations:

1. **TSP Upper Bound** – a pure truck-routing model that visits every customer directly by vehicle. It uses the same truck speed and a fixed parking time p = 5min per stop as in the integrated model but ignores walking and trolley-packing

decisions. Because the integrated model can consolidate customers into fewer stops and trade off driving and walking, this TSP benchmark provides an **upper bound** on the integrated model's performance (except for additional constraints present only in the full model)

2. **Model with Scoring Preprocessing** – incorporating a scoring heuristic to limit subset generation.
3. **Full Model** – the complete formulation without heuristic-based subset reduction.

Tests were conducted on small instances of 10, 15, and 20 customers, using the Adams_100_1_customers dataset from Reed's study. I selected the Adams dataset over others such as Cook's (strictly urban) and Cumberland's (strictly rural) to balance computational manageability with realistic routing complexity, ensuring some instances involved multiple customer stops per route. For each customer size (10, 15, 20), five distinct instances were generated by randomly selecting subsets from the 100-customer pool. The results are reported in Table 1.12. as average values across these samples. The runtime for each scenario is also provided in Table 1.13. to compare performance efficiency across models.

Key Observations

The Full Model consistently yields the lowest total service time, optimizing parking, routing, and walking decisions jointly. It serves as the performance benchmark for quality. The Scoring-based approach offers a heuristic simplification that significantly reduces runtime for larger instances, which are of interest here, while maintaining comparable solution quality:

- In 10-customer instances, total time is often within 1% of the Full Model.
- In 15-customer instances, solution quality remains close, with differences averaging under **1%**, proving that the heuristic is effective in mid-sized cases.
- In 20-customer instances, deviations are more noticeable, averaging around **3–4%**, depending on spatial layout and parking constraints, but still a comparable result.

Because the TSP-only model must visit every customer by truck and account for the same parking time at each stop, it consistently overestimates the true operational cost and therefore acts as an upper bound for the integrated model.

Computational Efficiency

TSP runtimes are generally low across all scenarios (under 20 seconds). The Full Model runtimes increase substantially with problem size, reaching up to 600 seconds in 20-customer cases. The Scoring Model achieves substantial savings - runtimes stay below 200 seconds even for 20-customer problems, cutting solve time by nearly a third compared to the Full Model.

Trade-off Analysis: Scoring Model vs. Full Model

The Scoring Model sacrifices very little solution quality (within 0–4% on average) while gaining significant computational efficiency. In operational settings where rapid planning is needed such as real-time delivery scheduling or same-day route adjustments, the scoring approach offers a practical and effective alternative. Overall, the heuristic strikes a strong balance: near-optimal quality with a fraction of the runtime cost.

Instance	Number of Customers	TSP (Traveling Salesman Problem) Time (UB)				Full Model (Best Found Solution)				Heuristic Model with Scoring (Novelty Model)			
		(Total Time)	(Parking Time)	(Travel time)	(Walking time)	(Total Time)	(Parking Time)	(Travel time)	(Walking time)	(Total Time)	(Parking Time)	(Travel time)	(Walking time)
A	**10 Average**	**57.81**	50	7.736	0	**46.51**	33	8.08	5.43	**46.80**	32	7.748	7.0
A1	10	58.61	50	8.61	0	46.29	35	8.6	2.69	46.29	35	8.6	2.69
A2	10	57.66	50	7.66	0	47.13	35	7.55	4.58	47.13	35	7.55	4.58
A3	10	56.84	50	6.48	0	41.78	25	6.69	10.09	41.87	25	6.78	10.09
A4	10	57.06	50	7.06	0	48.67	35	8.78	4.89	50.03	30	7.04	12.99
A5	10	58.87	50	8.87	0	48.66	35	8.77	4.89	48.66	35	8.77	4.89
B	**15 Average**	**83.93**	75	8.928	0	**64.28**	48	8.72	7.56	**64.5**	48.0	8.9	7.6
B1	15	84.73	75	9.73	0	65.32	50	9.71	5.61	65.33	50	9.71	5.62
B2	15	83.43	75	8.43	0	60.69	45	8.31	7.38	60.69	45	8.31	7.38
B3	15	84.76	75	9.76	0	73.9	60	9.6	4.3	74.06	60	9.76	4.3
B4	15	83.93	75	8.93	0	55.5	40	8.29	7.21	56.11	40	8.9	7.21
B5	15	82.79	75	7.79	0	65.99	45	7.69	13.3	66.09	45	7.79	13.3
C	**20 Average**	**109.55**	100	9.546	0	**81.66**	57	8.67	15.98	**84.5**	**57.0**	**8.7**	**18.7**
C1	20	108	100	8	0	81.18	55	7.8	18.38	81.33	55	7.95	18.38
C2	20	110.74	100	10.74	0	84.94	55	9.01	20.93	85.2	55	9.26	20.93
C3	20	109.38	100	9.38	0	81.06	60	9.3	11.76	82.01	60	9.25	12.76
C4	20	109.12	100	9.12	0	80.7	55	8.36	17.34	93.13	55	8.03	30.1
C5	20	110.49	100	10.49	0	80.4	60	8.9	11.51	80.58	60	9.07	11.51

Table 1.12: Analysis of the quality loss: result

Instance	Number of Customers	Gap Analysis		Runtime Comparison (in seconds)		
		% Change (Heuristic vs. TSP)	% Change (Heuristic vs. Full Model)	TSP Model	Full Model	Heuristic Model
A	**10 Average**	**-19.0%**	**0.6%**	**7.6**	**5.4**	**8.6**
A1	10	-21.0%	0.0%	7	5	8
A2	10	-18.3%	0.0%	12	6	9
A3	10	-26.3%	0.2%	9	5	9
A4	10	-12.3%	2.8%	6	5	9
A5	10	-17.3%	0.0%	4	6	8
B	**15 Average**	**-23.2%**	**0.3%**	**14.2**	**22.4**	**25.8**
B1	15	-22.9%	0.0%	44	13	17
B2	15	-27.3%	0.0%	7	28	31
B3	15	-12.6%	0.2%	9	11	18
B4	15	-33.1%	1.1%	5	18	22
B5	15	-20.2%	0.2%	6	42	41
C	**20 Average**	**-22.9%**	**3.4%**	**153.2**	**265**	**184.6**
C1	20	-24.7%	0.2%	180	300	300
C2	20	-23.1%	0.3%	86	162	263
C3	20	-25.0%	1.2%	180	23	25
C4	20	-14.7%	15.4%	20	240	135
C5	20	-27.1%	0.2%	300	600	200

Table 1.13: Analysis of the quality loss: percentage change and runtime

2.5.Experiment 4: Sensitivity analysis with 100 customers

Impact of parking time

This section analyzes the impact of parking time on the obtained solution using data from Reed's paper. Given variations in delivery areas (urban, suburban, rural) and time of day, parking time is expected to influence delivery efficiency. Urban routes during peak hours likely experience higher parking times compared to suburban areas with readily available parking.

To conduct a sensitivity test on parking time, I will use data from Reed's paper and analyze 100 customers in different location types. First, I tested for Cook County, an urban area, with parking times of 0, 3, 6, and 9 minutes. Then, I will apply the same test to Cumberland County, a rural area with more dispersed customers, using parking times of 0, 1.5 and 3 minutes. The packages which each of the customers is set to receive have been kept identical for both Cook and Cumberland County.

After confirming that the proposed Scoring Model produces results comparable to the Full Model, I ran an analysis of larger instances with 100 customers. To do this, I ran a sensitivity analysis to examine how small changes in key parameters (parking time, allowed walking distance, and trolley volume) influence the outcomes. The bin packing subproblem was solved across five separate runs and the results presented here represent the averages. For each run, only the top 200 high-quality feasible subsets were retained and subsequently used as input for the main problem, following the same logic as for the prior 50 customer testing.

Cook County

"Cook County is the most populous county in the U.S. state of Illinois and the second-most-populous county in the United States, after Los Angeles County, California. More than 40 percent of all residents of Illinois live within Cook County. As of 2020, the population was 5,275,541. The county seat is Chicago, the most populous city in Illinois and the third most populous city in the United States. The county is at the center of the Chicago metropolitan area." (Wikipedia, n.d.)

Figure 1.14: Cook

Cumberland County

"Cumberland County is a county in the Commonwealth of Pennsylvania. As of the 2020 census, the population was 259,469 with a population density of 470/sq mi (180/km2). According to the U.S. Census Bureau, the county has a total area of 550 square miles (1,400 km2), of which 545 square miles (1,410 km2) is land and 4.8 square miles (12 km2) (0.9%) is water." (Wikipedia, n.d.)

Figure 1.15: Cumberland county

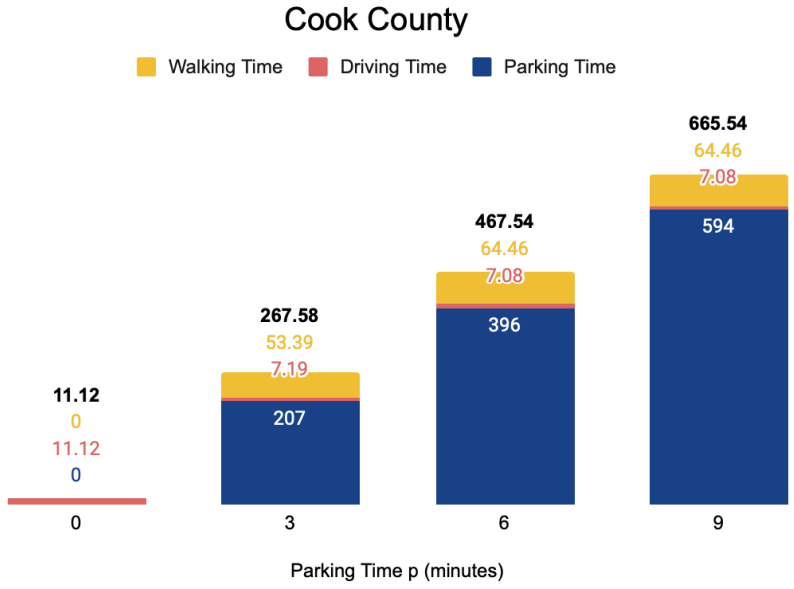

Figure 1.16: Parking time results: Cook county

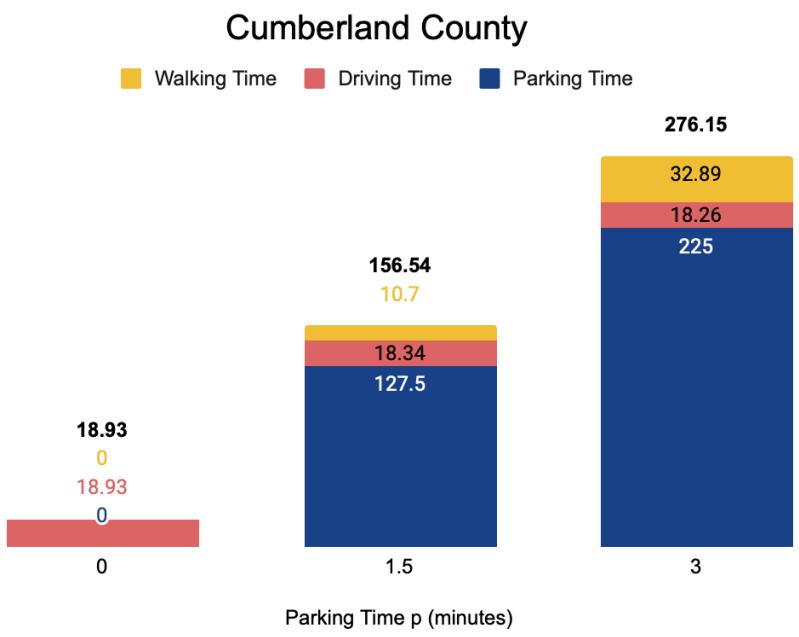

Figure 1.17: Parking time results: Cumberland county

In Cook County, where density is high and parking is more competitive, increases in parking time had a dramatic impact on total route time. As the fixed parking time increased from 0 to 9 minutes, the total time increased from 11.12 minutes to 665.54 minutes. The longer parking time encouraged the algorithm to favor walking over driving to reduce the number of parking stops. This trade-off shifted the balance: walking time increased from 0 to over 64.46 minutes, while driving time remained the same. Therefore, in urban areas, higher parking time can lead to fewer vehicle stops and greater reliance on on-foot delivery.

In **Cumberland County**, where parking is more readily available, the effect of increased parking time was less extreme but still significant. A parking time of 3 minutes raised the total time from **18.93 minutes** to **276.15 minutes**. The walking time remained relatively low, **32.89 minutes** at the highest and driving time stayed roughly constant. This indicates that in rural areas the system remained more vehicle-centric even with modest parking time increases.

These findings highlight that parking time is a critical parameter in route planning. In dense urban areas even small increases in parking time can substantially change the route structure, while in rural settings the effect is much more contained.

Impact of walking time

This section analyzes the impact of walking time on the obtained solution using data from Reed's paper. Given variations in delivery areas (urban, suburban, rural) and time of day, maximum allowed walking distance is expected to influence delivery efficiency. Urban customers are more densely dispersed and rural customers are further apart which will affect how the subtours are built.

To conduct a sensitivity test on walking time, I used data from Reed's paper and analyze 100 customers in different location types. First, I will test for Cook County, an urban area, with a maximum allowed walking distance of 125, 250, 500, 1500 and 2000 meters. Then, I applied the same test to Cumberland County, a rural area with more dispersed customers. The packages which each of the customers is set to receive have been kept identical for both Cook and Cumberland County. As we are testing for 100 customers, the version of the trolley packing sub-problem which uses the Scoring heuristic.

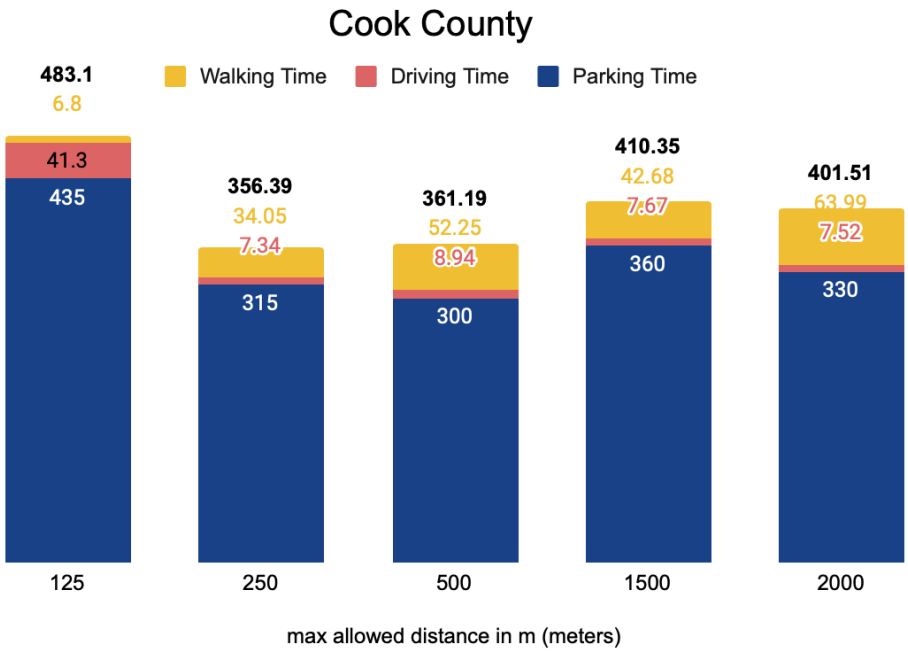

Figure 1.18: Walking time results: Cook county

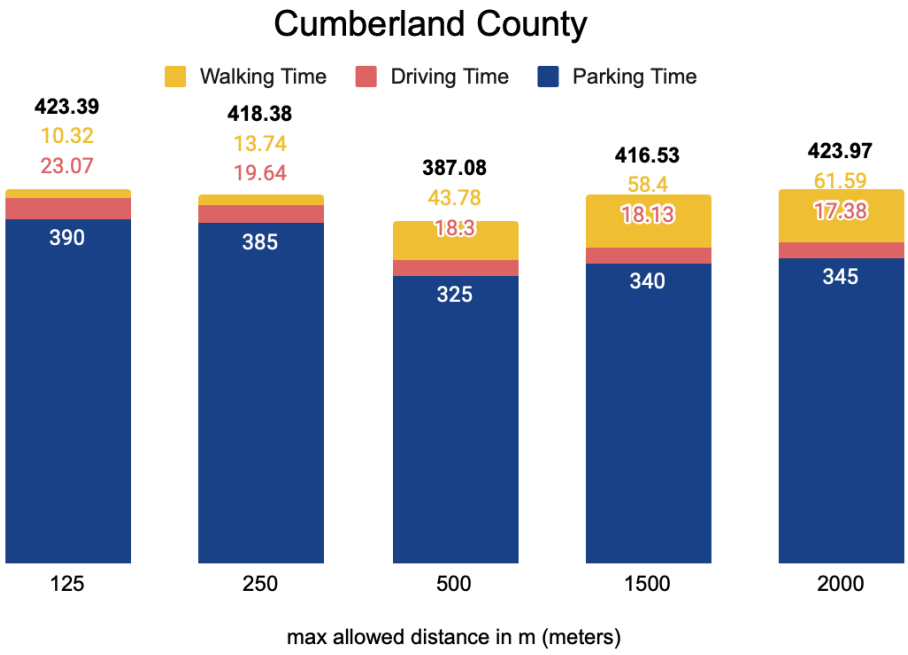

Figure 1.19: Walking time results: Cumberland county

Findings:

In Cook County, where density is high and stops are closely spaced, adjusting the maximum allowed walking distance had a strong influence on the balance between parking stops and on-foot delivery. When the threshold was very restrictive (125 meters), the system was forced to park frequently, resulting in excessive parking time (435 minutes) and the highest total time (483.1 minutes). As the threshold increased to 250–500 meters, the algorithm was able to consolidate more customers per stop. This reduced parking time and driving substantially while shifting some of the burden to walking, resulting in the lowest total times (356–361 minutes). However, when the threshold became too loose (1500–2000 meters), walking time grew steeply (up to 64 minutes), and total time worsened again (401–410 minutes).

Overall, these results show that a moderate walking distance allowance (250–500 meters) yields the best trade-off and minimizes total service time. Restrictive walking thresholds overemphasize parking, while overly loose ones overload the walking component.

In Cumberland County, where customers are more spread out and parking is less restrictive, the effect of the maximum walking distance was present but less pronounced than in Cook County. At very short limits (125 meters), parking time was high (390 minutes), and driving time was also relatively large (23.1 minutes), leading to a total of 423.4 minutes. Increasing the threshold to 250 meters produced only marginal improvements, as parking remained nearly unchanged and walking stayed low. The best performance occurred at 500 meters, where total time dropped to 387.1 minutes due to a reduction in parking (325 minutes) and a controlled increase in walking (43.8 minutes).

When walking thresholds became too loose (1500–2000 meters), the burden shifted heavily toward walking (58–62 minutes), and total time worsened again (416–424 minutes).

Overall, Cumberland shows the same general pattern as Cook: moderate walking threshold (around 500 meters) provides the best trade-off. However, the effect is less extreme in rural settings, since parking is less of a bottleneck and driving remains the dominant component.

These findings underline the importance of tuning walking distance parameters, with urban areas benefiting more from flexible walking limits than rural ones.

Impact of trolley volume

This section analyzes the impact of adjusting the trolley's capacity on the obtained solution using data from Reed's paper. Changing the trolley size, the capacity in terms of volume and weight, directly affects how many customers can be served in a single tour. If we increase the trolley size, we can pack more customer stops into one walking tour. This generally means fewer routes overall, less frequent parking and potentially shorter overall travel time since the truck doesn't have to return as often to reload. On the other hand, if we reduce the trolley size, we have to create more tours because fewer customers can be served at once. This can lead to more parking events and increased fixed costs per tour, but the walking distance per stop might be lower. Therefore, there is a trade-off: a larger trolley may improve overall efficiency by reducing the number of stops, while a smaller trolley might lead to shorter walks but more frequent stops and increased total travel time.

To conduct a sensitivity test on the trolley's capacity, I used data from Reed's paper and analyzed 100 customers in different location types. First, I will evaluate the model in Cook County, an urban area, by using two different trolley capacity setups: one with 230 L and 30 kg, and another with 300 L and 30 kg. The decision to test these configurations comes from an online brochure that states the trolley can boost its package volume from 230 L to 300 L by using additional bags. In this analysis, only the volume and weight of the packages are taken into account; the actual dimensions of the packages are not considered. Then, I applied the same test to Cumberland County, a rural area with more dispersed customers. The packages which each of the customers is set to receive have been kept identical for both Cook and Cumberland County. As we are testing for 100 customers, the version of the trolley packing sub-problem which uses the Scoring heuristic.

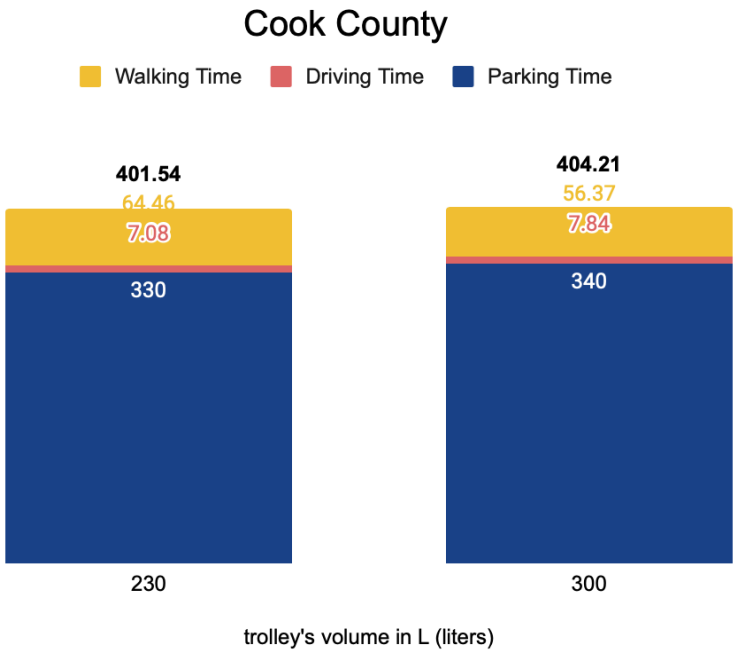

Figure 1.20: Trolley volume results: Cook county

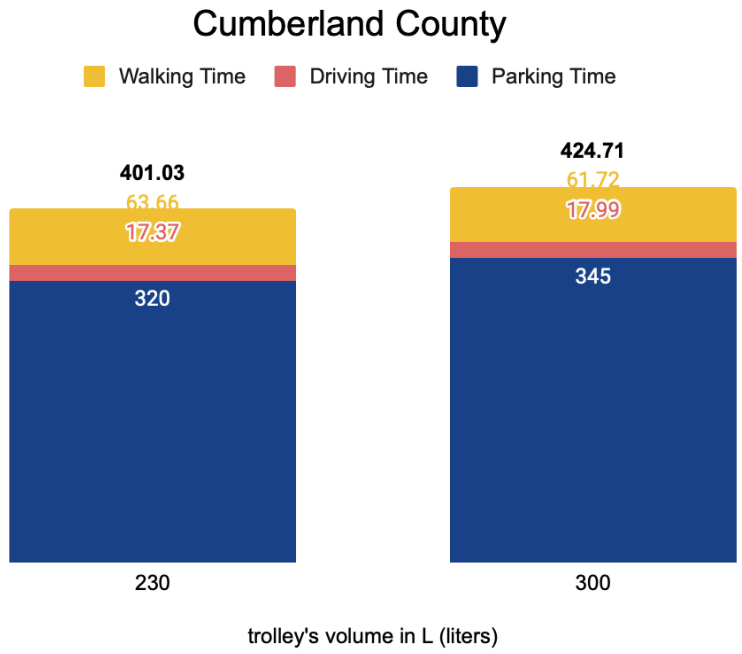

Figure 1.21: Trolley volume results: Cumberland county

Findings:

In Cook County, increasing the trolley's capacity from 230 to 300 units had only a minor effect on performance. Parking time stayed roughly the same (330–340 minutes), and driving time was nearly unchanged. Walking time dropped slightly (from 64.5 to 56.4 minutes), but this was offset by the added parking, so the total service time barely moved (401.5 → 404.2 minutes). This suggests that in dense urban areas, where stops are already tightly packed and the bottleneck is parking rather than capacity, enlarging the trolley provides little benefit.

In Cumberland County, the effect was also minimal but slightly different. Parking time actually increased (320 → 345 minutes), implying more stops, while walking time

decreased only slightly (63.7 → 61.7 minutes). The total service time worsened (401.0 → 424.7 minutes).

Overall, the results indicate that increasing trolley size does not systematically improve efficiency in either urban or rural contexts. In dense areas like Cook, capacity is not the limiting factor. In rural areas like Cumberland, the structure of travel patterns means larger trolleys can even reduce efficiency. This suggests that optimizing routing and parking strategies is more impactful than simply increasing trolley capacity.

3.Limitations

Optimality Gaps in the TSP UB Model

In the pure TSP benchmark model, the CBC solver was used with default settings. CBC does not report MIP optimality gaps unless explicitly configured and in this project the solver output did not provide a gap measure. Therefore, I cannot state a numerical optimality gap for the TSP runs. However, all TSP instances solved extremely quickly (under 20 seconds) and the solver terminated with status OPTIMAL in every run. Based on this, I assume that the obtained TSP solutions are optimal for the tested instances. If a gap had been present, CBC would normally return status FEASIBLE.

Distance Metrics

A minor inconsistency exists between the distance metrics used in the scripts (Euclidean in Stage 1 vs. Haversine in Stage 2). This difference results from how the two stages of the pipeline operate. In Stage 1, the code must generate and evaluate a very large number

of potential walking subsets. Because these walking distances are short and lie within a compact neighbourhood, a simple Euclidean approximation keeps the computation fast and is fully sufficient for feasibility checks. In Stage 2, the model computes truck travel times between parking locations, which occurs far less frequently, so using the more accurate Haversine formula is computationally inexpensive.

Since the two stages work on different scales, the use of two metrics ends up being a practical rather than a conceptual limitation. A validation check showed that the mismatch affected total times by less than 1%, so the impact is negligible at the neighbourhood scale. For this reason, I kept Euclidean distances in Stage 1 and Haversine distances in Stage 2.

Mix metrics result:

```
Total truck travel time (driving only): 13.39 minutes
Total Parking Time: 165.00 minutes
Final Total Time (Driving + Walking + Parking): 213.55 minutes
Total Walking Time = 35.17 minutes
```

Figure 1.22: Mix metrics result

Euclidean only result for the same 50 customers:

```
Total truck travel time (driving only): 11.85 minutes
Total Parking Time: 165.00 minutes
Final Total Time (Driving + Walking + Parking): 211.56 minutes
Total Walking Time = 34.71 minutes
```

Figure 1.23: Euclidean only result

4.Future Work and Extensions

This thesis builds a workable baseline for last-mile delivery with parking and trolley packing. Below I outline a few realistic extensions that would make the model closer to practice and open up new research questions. There are three straightforward ways to make the model more realistic.

First, the packing step is effectively 2D (length × height) plus weight, the width is ignored. A natural extension is full 3D bin packing with limited item orientations and simple stability checks (e.g., heavy items at the bottom). Because 3D packing is harder, a practical start would be a greedy or first-fit-decreasing heuristic adapted to 3D or a matheuristic that mixes greedy placement with a MIP.

Secondly, the model does not include loading/unloading time or stacking/handling rules. Adding a fixed loading time per walking tour, a per parcel handling tim, and a basic dependence of walking speed on trolley weight would already help. Simple stacking rules (fragile on top, heavy below, items for the same address grouped) and optional ergonomics limits (max. push weight, max. continuous walking minutes) would make the timings more credible.

Third, parking time is currently fixed. In practice it depends on context (urban vs. suburban) and time of day and it is uncertain. Two extensions are (i) context-aware parking times using area/time multipliers, and (ii) a stochastic version where parking time is drawn from a distribution and the objective optimizes expected time or adds a risk term (e.g., penalizing very long searches). This would likely push the model toward fewer truck stops in dense areas.

If implemented, I would evaluate these additions by re-running the 10/50/100-customer experiments reporting the total time and the split between driving/parking/walking, plus runtime.

5.Conclusion

This thesis set out to answer two questions: (i) what is the impact of trolley-packing constraints and parking availability on last-mile deliveries in dense urban settings and (ii) how (meta)heuristics can be used to solve the resulting routing problem at realistic scale. I approached this by building an integrated model that combines a TSP-style truck tour with explicit parking decisions and a trolley bin-packing subproblem. In short, the model links where to park, which customers to bundle into a single walking loop and how to pack the trolley and then sequences those loops into a single truck route.

The main novelty lies in three parts. First, I unify parking choice and packing feasibility in one framework: feasible walking subsets are generated by a bin-packing-with-proximity step (capacity + distance) and those subsets are then selected and sequenced by a truck-level TSP. This directly addresses a gap in the literature, where parking (CDPP) and capacity/packing (BPP/2E-CVRP) are typically treated separately. Second, I propose a scoring heuristic that ranks feasible subsets and guarantees customer coverage while keeping the search space manageable. This makes the integrated model practical for 50–100 customers without giving up much solution quality. Third, I provide a sensitivity analysis that quantifies how parking time, maximum walking distance and trolley volume shift the optimal balance between parking, walking and driving.

The results are clear. Parking time is the dominant driver in dense areas: even small increases push the solution toward fewer truck stops and longer (but still efficient) walking loops. Maximum walking distance has a non-linear effect: very tight limits cause excessive parking; very loose limits overload walking; moderate thresholds perform best (consolidation without long detours). By contrast, simply increasing the size of the trolley had little systematic benefit in either urban or rural layouts - capacity was not the main bottleneck once parking and walking were modeled explicitly. These findings confirm that ignoring parking and packing leads to overly optimistic plans and that the interaction between the two must be modeled to get credible times.

On the algorithmic side, the proposed scoring heuristic delivered near-optimal solutions (≈0–4% gap in tests) with substantial runtime savings versus the full model. This supports the use of metaheuristics/matheuristics for larger instances: methods like LKH for local tour improvement and ALNS/VNS for subset selection and routing are natural next steps.

There are, however, limitations. I used a fixed parking time per context, a simplified (2D + weight) packing model and a single-vehicle, no-time-windows setting, walking distances were approximated in Stage 1. Still, testing showed metric differences had <1% effect at neighborhood scale. Future work should extend to 3D packing with stability rules, stochastic/context-aware parking times, time windows,and multi-vehicle fleets.

Overall, the thesis contributes an operationally grounded, scalable framework for last-mile routing that jointly optimizes parking, walking and packing. It demonstrates that modeling these elements together changes routing structure and time estimates in

meaningful ways and it offers a practical heuristic pipeline that can be adapted and strengthened with state-of-the-art metaheuristics.

Appendix: Code and Results

All Python scripts, pseudocode, data sets used and result files (including CSV outputs and charts) used in this thesis are provided in a supplementary ZIP archive. The archive is organized as follows:

10 customers script – Python scripts and supporting files for the 10 customer tests.

50 customers script – Python scripts and supporting files for the 50 customer tests.

Others – Parcels data set and results from the parking, walking time and trolley volume comparisons

Pseudocodes – Pseudocodes of the mode

References

A Mukhairez, H. H., & A Maghari, A. Y. (2015). *Performance Comparison of Simulated Annealing, GA and ACO Applied to TSP*.

Amaya, J., Encarnación, T., & Delgado-Lindeman, M. (2023). Understanding Delivery Drivers' Parking Preferences in Urban Freight Operations. *Transportation Research Part A: Policy and Practice*, *176*. https://doi.org/10.1016/j.tra.2023.103823

Boysen, N., Fedtke, S., & Schwerdfeger, S. (2021). Last-mile delivery concepts: a survey from an operational research perspective. *OR Spectrum*, *43*(1). https://doi.org/10.1007/s00291-020-00607-8

Bueno-Ferrer, Á., de Pablo Valenciano, J., & de Burgos Jiménez, J. (2025). Unveiling the Potential of Metaheuristics in Transportation: A Path Towards Efficiency, Optimization, and Intelligent Management. *Infrastructures*, *10*(1). https://doi.org/10.3390/infrastructures10010004

Chauhan, V. K., Bass, M., Parlikad, A. K., & Brintrup, A. (2022.). *Trolley optimisation: An extension of bin packing to load PCB components*.

Chiara, G. D., Krutein, K. F., Ranjbari, A., & Goodchild, A. (2021). Understanding urban commercial vehicle driver behaviors and decision making. In *Transportation Research Record* (Vol. 2675, Issue 9, pp. 608–619). SAGE Publications Ltd. https://doi.org/10.1177/03611981211003575

Christensen, H. I., Khan, A., Pokutta, S., & Tetali, P. (2017). Approximation and online algorithms for multidimensional bin packing: A survey. In *Computer Science Review* (Vol. 24, pp. 63–79). Elsevier Ireland Ltd. https://doi.org/10.1016/j.cosrev.2016.12.001

Delorme, M., Iori, M., & Martello, S. (2016). Bin packing and cutting stock problems: Mathematical models and exact algorithms. In *European Journal of Operational Research* (Vol. 255, Issue 1, pp. 1–20). Elsevier. https://doi.org/10.1016/j.ejor.2016.04.030

Gambella, C., Lodi, A., & Vigoa, D. (2018). Exact solutions for the carrier-vehicle traveling salesman problem. *Transportation Science*, *52*(2), 320–330. https://doi.org/10.1287/trsc.2017.0771

Garone, E., Determe, J.-F., & Naldi, R. (2012). *A Travelling Salesman Problem for a class of Heterogeneous Multi-vehicle Systems*. https://doi.org/10.0/Linux-x86_64

Gonzalez-Feliu, J., Perboli, G., Tadei, R., & Vigo, D. (2008). *The two-echelon capacitated vehicle routing problem.2008.* https://shs.hal.science/halshs-00879447

Hansen, P., & Mladenovi, N. (2001). *Variable neighborhood search: Principles and applications*. www.elsevier.com/locate/dsw

Helsgaun, K. (2009). General k-opt submoves for the Lin-Kernighan TSP heuristic. *Mathematical Programming Computation*, *1*(2–3), 119–163. https://doi.org/10.1007/s12532-009-0004-6

Hussain, A., Muhammad, Y. S., Nauman Sajid, M., Hussain, I., Mohamd Shoukry, A., & Gani, S. (2017). Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator. *Computational Intelligence and Neuroscience*, *2017*. https://doi.org/10.1155/2017/7430125

Kamiński, B., Antosiewicz, M., & Koloch, G. (2013). Choice of best possible metaheuristic algorithm for the travelling salesman problem with limited computational time: quality, uncertainty and speed. *Journal of Theoretical and Applied Computer Science* (Vol. 7, Issue 1). http://www.jtacs.org

Kim, S., & Moon, I. (2019). Traveling salesman problem with a drone station. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(1), 42–52. https://doi.org/10.1109/TSMC.2018.2867496

Klein, P., & Popp, B. (2022). Last-Mile Delivery Methods in E-Commerce: Does Perceived Sustainability Matter for Consumer Acceptance and Usage? *Sustainability (Switzerland)*, *14*(24). https://doi.org/10.3390/su142416437

Laporte, G. (1992). The Traveling Salesman Problem: An overview of exact and approximate algorithms. *European Journal of Operational Research* (Vol. 59).

Laporte, G. (2010). A concise guide to the Traveling Salesman Problem. *Journal of the Operational Research Society*, *61*(1), 35–40. https://doi.org/10.1057/jors.2009.76

Lee, K., Chae, J., & Kim, J. (2019). A courier service with electric bicycles in an Urban Area: The case in Seoul. *Sustainability (Switzerland)*, *11*(5). https://doi.org/10.3390/su11051255

Leong, K. H., Abdul-Rahman, H., Wang, C., Onn, C. C., & Loo, S. C. (2016). Bee inspired novel optimization algorithm and mathematical model for effective and efficient route planning in railway system. *PLoS ONE*, *11*(12). https://doi.org/10.1371/journal.pone.0166064

Matai, R., Singh, S. P., & Mittal, M. L. (2010). *Traveling Salesman Problem: An Overview of Applications, Formulations, and Solution Approaches*. www.intechopen.com

Mondal, M., & Srivastava, D. (2022). Solving a Multi-Conveyance Travelling Salesman Problem using an Ant Colony Optimization Method. *Indian Journal Of Science And Technology*, *15*(45), 2468–2475. https://doi.org/10.17485/IJST/v15i45.1506

Moshref-Javadi, M., & Winkenbach, M. (2021). Applications and Research avenues for drone-based models in logistics: A classification and review. *Expert Systems with Applications*, *177*. https://doi.org/10.1016/j.eswa.2021.114854

Neumann, F., Polyakovskiy, S., Skutella, M., Stougie, L., & Wu, J. (2019). A Fully Polynomial Time Approximation Scheme for Packing While Traveling. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11409 LNCS*, 59–72. https://doi.org/10.1007/978-3-030-19759-9_5

Nguyên, T. B. T., Bektaş, T., Cherrett, T. J., McLeod, F. N., Allen, J., Bates, O., Piotrowska, M., Piecyk, M., Friday, A., & Wise, S. (2019). Optimising parcel deliveries in London using dual-mode routing. *Journal of the Operational Research Society*, *70*(6), 998–1010. https://doi.org/10.1080/01605682.2018.1480906

Olivier, P., Lodi, A., & Pesant, G. (2018). A comparison of optimization methods for multi-objective constrained bin packing problems. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10848 LNCS*, 462–476. https://doi.org/10.1007/978-3-319-93031-2_33

Ostermeier, M., Heimfarth, A., & Hübner, A. (2022). Cost-optimal truck-and-robot routing for last-mile delivery. *Networks*, *79*(3), 364–389. https://doi.org/10.1002/net.22030

Pasquier, J. L., Balich, I. K., Carr, D. W., & López-Martín, C. (2007). A comparative study of three metaheuristics applied to the traveling salesman problem. *Proceedings - 2007 6th Mexican International Conference on Artificial Intelligence, Special Session, MICAI 2007*, 243–254. https://doi.org/10.1109/MICAI.2007.14

Laporte G. (2010). A concise guide to the Traveling Salesman Problem. *Journal of the Operational Research Society,* pp. 35-40. https://www.jstor.org/stable/40540226

Raman, V., & Gill, N. S. (2017). Review of different heuristic algorithms for solving Travelling Salesman Problem. *International Journal of Advanced Research in Computer Science*, *8*(5). www.ijarcs.info

Ramirez-Rios, D. G., Kalahasthi, L. K., & Holguín-Veras, J. (2023). On-street parking for freight, services, and e-commerce traffic in US cities: A simulation model incorporating demand and duration. *Transportation Research Part A: Policy and Practice*, *169*. https://doi.org/10.1016/j.tra.2023.103590

Reda, M., Onsy, A., Elhosseini, M. A., Haikal, A. Y., & Badawy, M. (2022). A discrete variant of cuckoo search algorithm to solve the Travelling Salesman Problem and path planning for autonomous trolley inside warehouse. *Knowledge-Based Systems*, *252*. https://doi.org/10.1016/j.knosys.2022.109290

Reed, S., Campbell, A. M., & Thomas, B. W. (2024). Does parking matter? The impact of parking time on last-mile delivery optimization. *Transportation Research Part E: Logistics and Transportation Review*, *181*. https://doi.org/10.1016/j.tre.2023.103391

Rego, C., Gamboa, D., Glover, F., & Osterman, C. (2011). Traveling salesman problem heuristics: Leading methods, implementations and latest advances. *European Journal of Operational Research*, *211*(3), 427–441. https://doi.org/10.1016/j.ejor.2010.09.010

Rhiat, A., Aggoun, A., & Lachere, R. (2020). Combining Mobile Robotics and Packing for Optimal deliveries. *Procedia Manufacturing*, *44*, 536–542. https://doi.org/10.1016/j.promfg.2020.02.258

Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, *40*(4), 455–472. https://doi.org/10.1287/trsc.1050.0135

Sadykov, R., & Vanderbeck, F. (2013). Bin packing with conflicts: A generic branch-and-price algorithm. *INFORMS Journal on Computing*, *25*(2), 244–255. https://doi.org/10.1287/ijoc.1120.0499

Sathya, N., & Muthukumaravel, A. (2015). A review of the optimization algorithms on traveling salesman problem. *Indian Journal of Science and Technology*, *8*(29). https://doi.org/10.17485/ijst/2015/v8i1/84652

Sluijk, N., Florio, A. M., Kinable, J., Dellaert, N., & van Woensel, T. (2023). Two-echelon vehicle routing problems: A literature review. In *European Journal of Operational Research* (Vol. 304, Issue 3, pp. 865–886). Elsevier B.V. https://doi.org/10.1016/j.ejor.2022.02.022