VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY
THE INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

PARALLEL ANT COLONY OPTIMIZATION FOR VEHICLE ROUTING WITH PARCEL LOCKERS

By
Chau An Phu
ITDSIU22158
Advisor: Assoc. Prof. Nguyen Thi Thuy Loan

A thesis submitted to the School of Computer Science and Engineering in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Data Science

Ho Chi Minh City, Vietnam
2025

PARALLEL ANT COLONY OPTIMIZATION FOR VEHICLE ROUTING WITH PARCEL LOCKERS

APPROVED BY:

_____,
Assoc. Prof. Dr. Nguyen Thi Thuy Loan

_____
Committee name here

_____
Committee name here

_____
Committee name here

_____
Committee name here

THESIS COMMITTEE

Acknowledgments

I would to extend my deepest appreciation to Assoc. Prof. Dr. Nguyen Thi Thuy Loan for her instrumental guidance and feeback. Her enthusiasm and empathy were crucial to the completion of this thesis. It has been distinct priviledge to work under her supervision. Furthermore, Seminar Bay-Loan Group has always played a vital role in providing encouraging environemnt as the seniors have granted me with great inspiration and experiences for conducting research in this field of study.

I also acknowledge for the central Interdisciplinary Laboratory in Electronics and Information Technology (AI and Cooperation Robot) of the School of Computer Science and Engineering at International University, VNU-HCM, along with its staffs and faculty, for equipping me with foundation in this field. The resources provided and the knowledge acquired during my tenure here have significantly influenced my professional outlook and personal growth.

Lastly, I am indebted to my parents and friends for their unwavering support, both physical and emotional, particularly during the most challenges phases of this journey.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	**Definition**
3D-PACO	3D Enhanced Parallel Ant Colony Optimization
ACO	Ant Colony Optimization
LMD	Last-mile Delivery
OFAT	One-Factor-at-A-Time
OOHD	Out-of-Home Delivery
SA	Simlated Annealing
TSP	Travelling Salesman Problem
VECOM	Vietnam E-Commerce Association
VRP	Vehicle Routing Problem
VRPPL	Vehicle Routing Problem with Parcel Lockers

Abstract

The recent surge in Vietnam's e-commerce has significantly strained urban infrastructure, leading to increased last-mile delivery times, higher transportation costs, and diminished customer satisfaction. Traditional delivery methods are struggling to meet modern demands, facing challenges such as rising expectations for same-day delivery, high operational costs from failed deliveries due to customer absence, and the inefficiency of door-to-door service in high-density areas (e.g., apartments, offices). In response, delivery consolidation strategies, such as parcel locker networks, are being adopted as a sustainable solution. However, integrating parcel lockers introduces a new, complex variant to the classic routing problem: the Vehicle Routing Problem with Parcel Lockers (VRPPL). This problem requires managing multiple, interdependent delivery options (home or locker), which significantly complicates route optimization. Each decision impacts subsequent travel costs and potentially the solution's feasibility under time-window and capacity constraints. Moreover, VRPPL introduces a critical customer preference constraint, requiring the solution to respect the customer's selected receiving location (home, locker, or flexible). To address these challenges, this research proposes a novel 3D Enhanced Parallel Ant Colony Optimization (3D-PACO) model. The core contribution is a novel multidimensional pheromone matrix that extends traditional Ant Colony Optimization (ACO) to effectively manage the VRPPL's multiple delivery options. This study further enhances this with an adaptive mechanism for dynamic ant allocation per thread, optimizing computational resource utilization. Finally, this thesis adopts a master-slave parallel schema to deploy a large-scale ant population, enabling a more comprehensive search of the solution space. Experimental results on VRPPL benchmark datasets show statistically significant improvements in solution quality, achieving computational runtime savings of 46.4% to 83.5% compared to state-of-the-art methods. The proposed 3D-PACO model not only addresses the unique challenges of integrating parcel lockers into delivery routes but also sets a new benchmark for future research in this domain.

Chapter 1

Introduction

1.1 Background and Motivation

Vietnam's e-commerce sector has demonstrated transformative growth, evolving from a nascent market into a significant pillar of the national economy. This expansion is largely propelled by a young, tech-native demographic and the explosive rise of social commerce. The sector's rapid ascent is quantified in recent reports: according to Vietnam E-Commerce Association (VECOM) [1], Vietnam's e-commerce grew by over 25% in 2023 to reach $25 billion. Within this, online retail sales contributed $17.3 billion, representing approximately 10% of the nation's total retail sales. This trend is predicted to continue; a subsequent 2025 VECOM report [3] estimates the market will climb to $32 billion in 2025, reaching a 30% increase over the previous year. This sustained boom in online transactions has consequently placed immense pressure on logistics, necessitating a parallel development in the scale, quality, and efficiency of fulfillment and last-mile delivery networks [1].

While these issues are highlighted in the Vietnamese market, they reflect a global challenge in Last-mile Delivery (LMD) optimization. LMD refers to the final step of order fulfillment, which involves transporting goods from a distribution center to the customer's final address. Despite its critical role in customer satisfaction, LMD is widely considered the most expensive and inefficient segment of the logistics chain, accounting for up to 53% of total shipping costs [4]. This financial burden is evident in Vietnam; as shown in Fig 1.1, while most companies maintain LMD costs below 10%, a significant one-third still reported losing 10% to 20% of their revenue to last-mile shipping.

The most common method, traditional door-to-door delivery, is a primary source of this inefficiency [5]. Its first drawback is the strict requirement for the

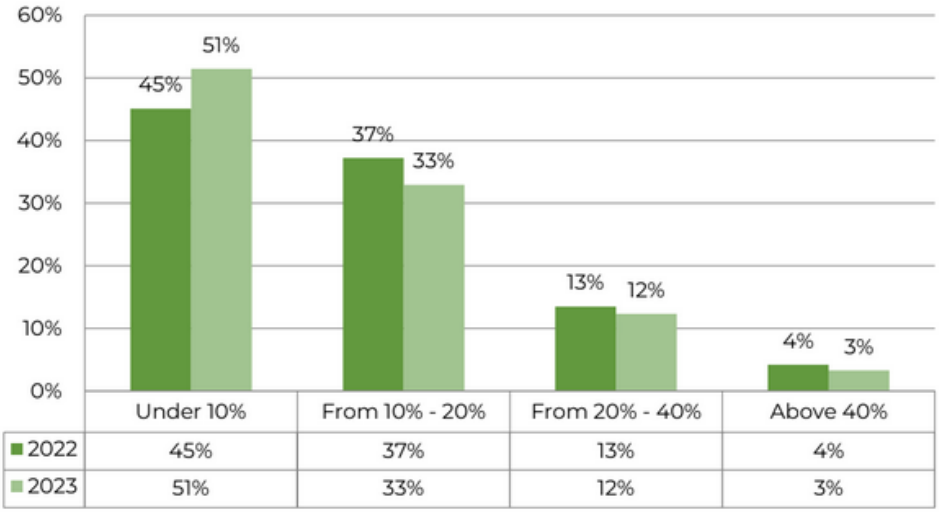


Figure 1.1: Order fulfillment and last-mile delivery costs as a percentage of enterprise e-commerce revenue [1]

customer's physical presence at a specific, pre-arranged time. This rigidity directly leads to the second major limitation: a high rate of failed delivery attempts, which occur when customers are not present or are unavailable at the designated time [6]. Thirdly, while deliveries to workplaces may reduce failure rates, they introduce other issues, such as internal building congestion and negative impacts on employees' productivity. Consequently, many companies are now placing restrictions on personal deliveries [5]. Finally, security and privacy are significant concerns. Drivers often leave packages unsupervised at the doorstep, exposing them to the risk of theft which is a phenomenon known as porch piracy [7]. As a result, these shortcomings of traditional LMD necessitate research into alternative models, particularly focusing on the optimization field of Vehicle Routing Problem (VRP) and the implementation of smart locker systems.

VRP is a study regarding finding the optimal routes for set of vehicles to distribute goods from a transporation center to scattered geography of customers. VRP is highly studied topic in operational research, since the distribution of the goods is influenced by multiple factors originating from transportation companies, customers and the external environment. The distribution costs in VRP contribute significantly in a product's final selling price, and can be cateogorized into:

- Fixed costs: Consisting the driver's wage or vehicle usage costs, which burden the company simply by using the vehicle, regardless of the specific route.
- Variable costs: Mostly resulting from fuel costs or the traveling time of each

route, which are affected by the route's length and duration.

VRP is considered to be the extension of Travelling Salesman Problem (TSP), therefore, it is inherently a NP-hard problem, meaning finding the optimal solution is exponentially computationally expensive, and potentially infeasible if incorporated with constraints.

1.2 Problem Statement

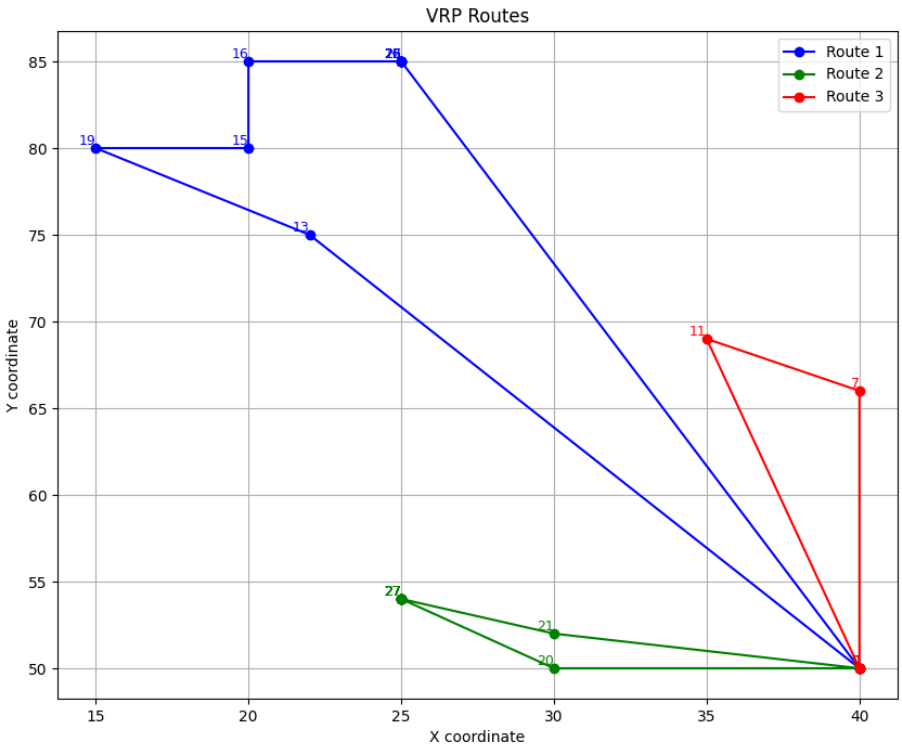


Figure 1.2: An example of a VRPPL solution. Node 26 and 27 are lockers and the remaining nodes are customer homes

The integration of smart lockers into routing problem has introduced a new variant of VRP, known as VRPPL [8]. Unlike traditional VRP in which each customer is associated with a unique delivery point, there can be multiple alternative locations for a customer's request. Furthermore, these delivery options might be preoccupied or capacity-limited. Such challenges demand the routing algorithm not only be able to find near-optimal permutations of customers to be served by also considering the route options.

The state-of-the-art solutions is Simlated Annealing (SA) of Yu et.al [8]. However, this SA algorithm posesses a critical structual weakness. The multi-decision challenge of VRPPL is solving two problems simultaneously, which are the assignment problem (which type-III customers should be served at their preferred lockers?) and the routing problem (which is the optimal route for the assignment?).

The Yu et al. SA evades the problem by not optimizing it. Instead, it uses a static, random probability parameter p to assign Type-III customers during initial solution construction. The parameter is only tuned but not non-optimizable, random assignment can trap the single-solution-based SA in a local optimum, where it may find the best route for a fundamentally suboptimal assignment. Furthermore, the original work's assumption is that type-III customers only chooses a single closest locker as their preferred pickup point. This prevent the mathemtical model from scaling to abitrary number of preferred locations.

The thesis proposes that a swarm-based metaheuristics, specificcally ACO, can overcome this by learning the optimal assignment. However, a traditional 2D ACO, which only optimizes pheromone trails between nodes (ρ_{ij}), is insufficient as it contains no mechanism to model the "delivery mode" choice. Therefore, the central problem of this research is how to design an optimization framework that simultaneously learns and solves both the HLC customer assignment and the vehicle routing components of the VRPPL.

This requires a novel, multi-dimensional ACO pheromone model that can reinforce both good routes and good delivery mode choices. This new model introduces a secondary challenge: a combinatorial explosion of the search space. Consequently, this thesis must also address the problem of computational feasibility, necessitating the use of a parallel computing framework to efficiently navigate this high-complexity, multi-decision landscape.

1.3 Scope and Limitations

1.3.1 Scope of Research

This thesis is scoped to the design and implementation and rigiorous evaluation of the novel 3D-PACO algorithm to solve the VRPPL.This research includes the

developemnt of a new pheromone structure that explicitly manages the type-III assignment decision, moving beyond the traditional 2D pheromone matrix. Next, this research also address the implementation of this ACO algorithm within a computing framework to manage the expanded search complexity introduced by the multi-dimensional decision problem. The performance of the proposed 3D-PACO will be benchmarked directly against the SA from Yu et al.. To provide a more comprehensive evaluation, the proposed solver will also be compaired against several other swarm-based metaheuristics solvers using random assignment presented by Yu et al. The evaluation will exclusively use the identical small, medium, and large VRPPL benchmark instances generated and used by Yu et al. to ensure a valid, direct comparison. The comparison will be based on the same metrics: solution quality (minimum total distance) and the computational time required to find that solution. Statistical significance will be evaluated using the same Wilcoxon signed-rank test.

1.3.2 Limitations of Research

This thesis primarily focuses on algorithm model design and implementation, thus it will not contribute to the expansion of VRPPL mathematical model and dataset. All assumptions from the original paper are adopted. This includes, but is not limited to: a homogeneous fleet of vehicles, known customer time windows, and the assumption that Type-II and Type-III customers have already pre-selected their preferred parcel locker. The model will not optimize which locker a customer should use or where lockers should be placed. This work focuses exclusively on the VRPPL as defined. It does not extend to other VRP variants, such as considering reverse logistics (product returns via lockers), which was noted as a potential future direction by Yu et al.. The parallel implementation will be focused on a multi-core CPU architecture. A broader investigation into other parallel paradigms (e.g., GPGPU, distributed-memory clusters) is outside the scope of this work. Furthermore, despite the mathematical model of Yu et al. mentions and defines constraints for locker capacity, the published dataset does not contain such information therefore this work ommit the locker capacity constraints during implementation.

1.4 Research Contributions

This research offers three key contributions. First, this research proposes a novel 3D pheromone matrix that significantly extends the ACO model's applicability to VRPs featuring multiple delivery options, explicitly incorporating customer service preferences as constraints. This ACO model is further enhanced with a local search mechanism to effectively escape local optima. Second, this research implements a master-slave parallel architecture designed to efficiently deploy a large population of ants. An elitist update strategy not only reduces pheromone synchronization runtime at the master process but also enhances exploitation, leading to faster convergence and improved solution quality for large-scale problems. Finally, to overcome critical scaling and hyperparameter tuning challenges, this research introduce an adaptive mechanism for dynamically adjusting the number of ants per thread. This significantly enhances the model's robustness and ensures economical computational resource utilization.

1.5 Thesis Structure

The article is further organized as follows. Sections 2 reviews the problem variants, different approaches to VRPs, ACO extensions and parallel schema for ACO. Section 3 focuses on presenting 3D-pheromone matrix and proposed algorithm. In section 4, this research perform analysis on the model's robustness, scalability and performance comparison to the published results of Yu et al. [8]. Lastly, section 5 summarizes the article, states the limitations of the proposed method and suggests future works.

Chapter 2

Literature Review

2.1 Parcel Lockers and OOHD

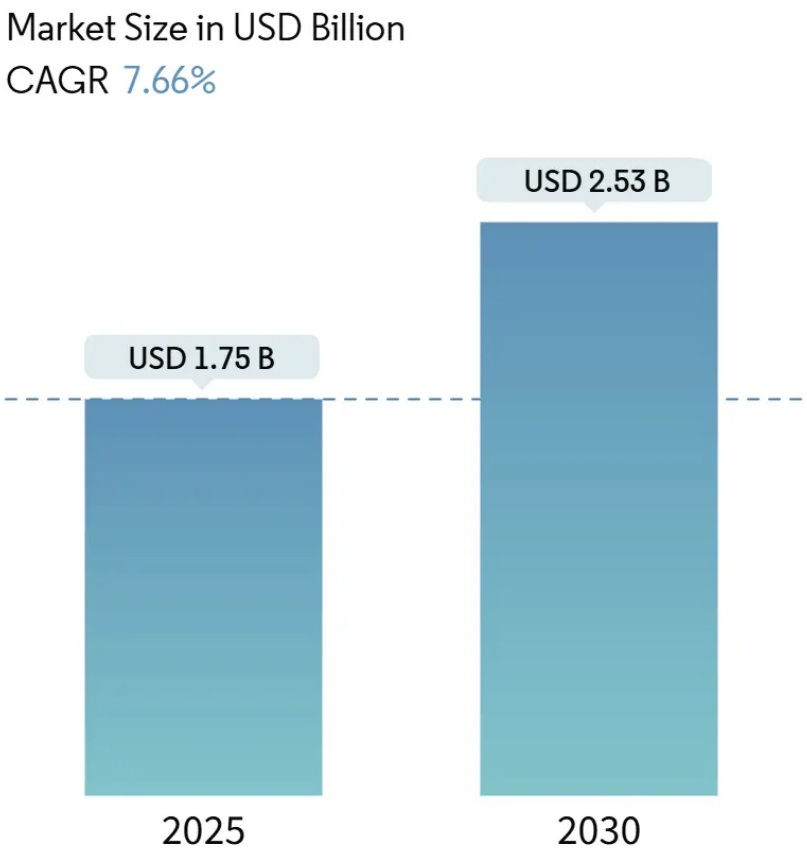


Figure 2.1: Vietnam Courier, Express and Parcel (CEP) Market growth from 2025 to 2030

The Vietnamese market presents a compelling, high-growth environment for the deployment of smart locker and automated parcel collection systems. Fuelled by robust e-commerce expansion, the Courier, Express, and Parcel (CEP) market in Vietnam is projected to grow significantly, representing a high-potential opportunity for logistics innovation. [9] The market is expected to expand from USD 1.75 billion in 2025 to USD 2.53 billion by 2030 as shown by Fig 2.1, re-

flecting a Compound Annual Growth Rate (CAGR) of 7.66% during this forecast period. [9] This sustained expansion creates substantial tailwinds for efficiency-enhancing technologies designed to address the challenges inherent in last-mile delivery within rapidly urbanizing areas.

Despite the macroeconomic drivers, the current penetration of automated parcel lockers remains nascent. Lockboxes, while common in many developed economies, are currently "not wildly available in Vietnam". [10] This low existing penetration defines the market as a significant opportunity for disruptive entry. The current state of smart locker deployment in Vietnam is characterized by its relative infancy. In contrast to countries like Germany, where DHL operates a network exceeding 340,000 lockers, Vietnam has not yet reached a point of widespread availability. Deployment efforts are currently concentrated in major commercial hubs such as Hanoi and Ho Chi Minh City [11, 12], aligning with the high density and e-commerce volume generated by these commercial centers.

Figure 2.2: A smart locker unit of Viettel (SmartBox)

The domestic deployment of smart locker infrastructure is currently being spearheaded by ViettelPost. The company has proactively established a network comprising 2,000 smart lockers [9]. This significant investment provides a practical operational template for scaled deployment within the country. ViettelPost's deployment strategy strategically embeds parcel retrieval points within "urban micro-malls". This approach yields a critical advantage: it addresses the regulatory complexity of public space allocation. Deploying lockers in public right-of-way locations often requires addressing regulatory constraints, stakeholder coordination, and complex permitting. By leveraging commercial or partnered real estate

(micro-malls), ViettelPost mitigates the bureaucratic delays associated with public land use, focusing their assets in high-traffic, existing consumer destinations. For new market entrants, this suggests that an effective strategy must prioritize a "real-estate-first" approach through partnerships with property developers (residential and commercial) to secure viable, non-public installation sites.

OOHD refers to the delivery of parcels to facilitates, including parcel shops and parcel lockers instead of recipient's home [7]. Smart locker systems, also known as automated boxes or parcel lockers, are a specific type of OOHD facility. The capacity of a smart locker is strictly limited by the number and size of installed compartments; parcels from multiple recipients cannot be placed in a single compartments. Compartments may be heterogeneous in size. OOHD offers several advantages to achieve efficient routing strategy. Firstly, packages can be consolidated to reduce the number of delivery points, as a result, could save up 55-66% compared to traditional shipping [13]. Secondly, OOHD decouples the delivery process from the receiving process, which greatly reduces the incidence of costly failed deliveries that occur when recipients are absent [14], necessitating further delivery attempts. Flexibility is the third advantages of OOHD since the customers can pickup the packages at their preferred time, as well as flexible in route planning since customers may be willing to pickup from multiple smart lockers. Finally, the use of smart lockers is proved to be a green solution, helping to reduce emissions and traffic volume in congested city centers. The solution fosters sustainable development by reducing air pollution, noise pollution, and traffic congestion, as well as saving energy.

2.2 Combinatorial Optimization on Graphs

Combinatorial Optimization (CO) problems present in diverse domains ranging from transportation and telecommunications to resource management. Formally a CO problem can be defined by a set of feasible solutions F and a cost function c, denoted as $C = \{f, c\}$, where $c : F \to \mathbb{R}$ serves as a cost function. The primary object is to identify a feasible solution $f \in F$ that minimizes (or maximizes) the cost function (optimization version), calculate the cost of the optimal solution (evaluation version) or to determine if a solution exists within a specific cost threshold $c(f) \leq L$, where L is some integer (recognition version) [15].

The primary objective of combinatorial optimization is the design of efficient algorithms, where efficiency is defined by the number of elementary steps growing polynomially with the size of the input. Based on this criterion, problems are categorized into specific complexity classes. Computational complexity classes establish a framework for understanding the inherent difficulty of problem-solving. The class $\mathcal{P}$ encompasses decision problems (those with a yes/no answer) that are solvable by a deterministic Turing machine in polynomial time, meaning the solution time is bounded by a polynomial function of the input size. Problems classified within $\mathcal{P}$ are considered efficiently solvable in practice. However, the majority of combinatorial optimization problems encountered in practical scenarios, such as routing and scheduling, are considered computationally intractable, as no exact polynomial-time algorithm has yet been devised for them.

To categorize these intractable problems, the class $\mathcal{NP}$ (Nondeterministic Polynomial time) is defined. A problem belongs to $\mathcal{NP}$ if and only if its corresponding decision problem which is formulated as a question of whether a feasible solution exists within a certain cost limit is solvable in polynomial time by a nondeterministic algorithm. Within this framework, a problem is classified as $\mathcal{NP}$-hard if every problem in $\mathcal{NP}$ can be reduced to it in polynomial time. Finally, a problem is designated as $\mathcal{NP}$-complete ($\mathcal{NPC}$) if it satisfies two conditions: it belongs to the class $\mathcal{NP}$, and it is also $\mathcal{NP}$-hard.

The classification of VRP as *NP-hard* can be understood through its structural relationship to the TSP. The Travelling Salesman Problem (TSP) is a fundamental combinatorial optimization problem. The objective is to identify the shortest possible route for a salesman who must depart from a home city, visit every city on a specific list exactly once, and subsequently return to the starting location. Mathematically, the problem is defined given an integer $n \geq 3$ and an $n \times n$ matrix $C = (c_{ij})$, where each element c_{ij} represents a non-negative integer cost between cities. The goal is to find a cyclic permutation π of the integers from 1 to n that minimizes the sum $\sum_{i=1}^{n} c_{i\pi(i)}$ [16]. The computational difficulty of the TSP arises from the magnitude of the search space, which comprises $\frac{(n-1)!}{2}$ possible tours for a problem with n cities. The TSP is a classic example of an $\mathcal{NP}$-complete problem, meaning that no polynomial-time algorithm is currently known to solve it.

The VRP is mathematically defined as a "common extension to the TSP" [17]. By definition, a problem is $\mathcal{NP}$-hard if every problem in $\mathcal{NP}$ can be reduced to it in polynomial time. Since the VRP generalizes the TSP structure by effectively

encompassing the TSP as a special case where constraints such as vehicle capacity are relaxed, hence the computational intractability inherent to the TSP is inherited by the VRP, classifying it as $\mathcal{NP}$-hard.

These distinctions are critical for justifying the use of meta-heuristics, such as Ant Colony Optimization, as exact approaches for $\mathcal{NP}$-hard problems often result in prohibitive execution times for large-scale instances [18].

2.3 From VRP to VRPPL

The Vehicle Routing Problem (VRP) constitutes a cornerstone of operations research due to its critical role in optimizing logistics and transportation networks. The classic VRP, first introduced in the seminal work of Dantzig and Ramser (1959) as "The Truck Dispatching Problem" [19], modeled how a fleet of homogeneous trucks could efficiently fulfill oil demand at multiple stations from a central hub, focusing on minimizing total travel distance. Clarke and Wright (1964) [20] extended this foundation to address a broader class of logistics problems, optimizing route costs for geographically dispersed customers served by trucks with varying capacities. While the fundamental objective remains consistent, determining efficient routes starting and ending at a central depot to serve all customers, real-world applications have introduced complex constraints involving both objective and subjective supply chain factors. Consequently, the VRP has evolved into numerous variants over the decades to address these specific needs [21].

Among the numerous variants, Capacitated Vehcile Routing Problem (CVRP) stands out as one of the most extensively studied foundations [21]. CVRP enforces strict capacity constraints, mandating that each customer is serviced by exactly one vehicle and that the aggregated demand of assigned customers does not exceed the vehicle's capacity. Often, additional constraints regarding fleet size or maximum travel distance are imposed. Formally, CVRP is defined as follows [22]: A set of n customers requires service from a central depot. Each customer i is associated with a non-negative demand q_i and a service time s_i.

A fleet of m homogeneous vehicles, each with capacity Q and a maximum service duration D, operates from the depot. Given known locations, the travel distance d_{ij} and travel time t_{ij} between any two points are deterministic. The

CVRP consists of designing a set of at most m delivery routes such that:

1. Each route starts and ends at the depot.
2. Each customer is visited exactly once by exactly one vehicle.
3. The total demand of each route does not exceed Q.
4. The total duration of each route (including travel and service times) does not exceed a preset limit D.
5. The total routing cost is minimized.

A critical evolution of the standard model is the Vehcile Routing Problem with Time Windows (VRPTW), which incorporates temporal constraints into the routing logic. In this variant, each customer specifies a service time interval during which the delivery must occur. These constraints typically fall into two categories: hard and soft time windows. Hard time windows strictly enforce the schedule; if a vehicle arrives before the time window opens, it must wait, and arriving after the window closes is prohibited [23]. Conversely, soft time windows allow for late arrivals but incur a penalty cost, effectively modeling a trade-off between strict punctuality and operational flexibility. The VRPTW is paramount in modern logistics, as the ability to adhere to specific delivery slots is inextricably linked to delivery accuracy and overall customer satisfaction.

The increasing rate of first-time delivery failures in e-commerce, driven by the absence of customers at their designated location, necessitated a paradigm shift from purely optimizing the delivery route to optimizing the delivery transaction itself. This challenge led to the introduction of the Vehicle Routing Problem with Delivery Options (VRPDO), as formalized by Tilk et al. (2021) [24]. The VRPDO explicitly models the modern reality where a customer request n, can be fulfilled by selecting exactly one option from a set of available delivery options O_n. Each option $o = (n_o, l_o, p_o)$ is defined by the request, a specific delivery location l_o (which could be the customer's home or a Shared Delivery Location (SDL) like a locker), and a customer-assigned priority p_o. This formulation generalizes previous models by integrating two critical, interlinked factors:

1. Customer Preference and Service Level: The optimization objective expands from purely minimizing cost (distance/time) to also considering a service level

constraint that requires a minimum percentage of deliveries to be fulfilled using options with a high priority level $p \leq p_{max}$.

2. Shared Location Capacity: The model must respect capacity constraints, particularly when multiple delivery options are assigned to a common resource, such as the limited size of a parcel locker.

The VRPDO, therefore, represents a crucial conceptual step in the literature, as it formally models the trade-off between carrier efficiency (cost minimization) and customer convenience (preference/service level) by integrating decision-making (option selection) into the core routing problem.

While VRPDO introduces the concept of selecting delivery locations, it often simplifies the physical capacity constraints of those locations. Addressing this limitation, Grabenschweiger et al. (2021) [25] proposed the Vehicle Routing Problem with Heterogeneous Locker Boxes (VRPHLB). This variant acknowledges that real-world logistics involve parcels of varying dimensions and locker stations containing slots of heterogeneous sizes (e.g., small, medium, large). Consequently, the problem expands beyond pure routing into a dual optimization problem involving routing and packing. The authors model the packing component as a capacitated bin packing problem with conflicts, where parcels must be fitted into slots without mixing orders from different customers. Formally, let u_{gh} be a binary variable equal to 1 if parcel g with size a_g is assigned to slot h with capacity h_b. The model enforces size compatibility via the constraint:

$$\sum_{g} a_g u_{gh} \leq b_h w_h \quad \forall h \in H$$

where w_h indicates if slot h is utilized. Crucially, to ensure security, the model enforces a conflict constraint preventing parcels from different customers ($i_g \neq i_f$) from sharing the same locker slot:

$$u_{gh} + u_{fh} \leq w_h$$

The objective of the VRPHLB is to minimize the sum of operational routing costs and the compensation costs paid to customers for accepting locker delivery, creating a balance between logistic efficiency and customer incentives.

Building upon the integration of lockers into the routing network, Yu et al.

(2022) proposed the Vehicle Routing Problem with Parcel Lockers (VRPPL). Unlike previous models that often treat customers as static points, the VRPPL distinguishes between three specific customer behaviors to optimize the last-mile network: Type-1 customers (N_{HC}) requiring strictly home delivery, Type-2 customers (N_LC) requiring locker delivery, and Type-3 customers (N_{HLC}) who are flexible and can accept either method. This flexibility is mathematically formalized by introducing binary decision variables h_i (equal to 1 if customer i is served at home) and l_i (equal to 1 if served at a locker). The model enforces that exactly one delivery mode is selected for every customer $i \in N_C$:

$$h_i + l_i = 1 \quad \forall i \in N_C$$

Furthermore, the model explicitly links customer demand to locker capacity. If a customer is assigned to a locker ($l_i = 1$), they must be mapped to a specific parcel locker station j via the assignment variable y_{ij}. The aggregated demand at any locker station j must not exceed its capacity ξ_j:

$$\sum_{i \in N_C} y_{ij} \leq \xi_j \quad \forall j \in N_{PL}$$

The objective of the VRPPL differs slightly from the VRPHLB; rather than balancing compensation costs, it focuses on minimizing the total travel distance of the fleet while satisfying these hybrid service constraints.

2.4 Common solvers for VRPs

2.4.1 Exact methods and Heuristics

The optimization landscape for the Vehicle Routing Problem (VRP) is generally categorized into exact methods, heuristics, metaheuristics, and hybrid approaches. Foundational research primarily relied on exact methods, including Branch-and-Bound (B&B), Integer Linear Programming (ILP), and Dynamic Programming. These approaches are critical as they provide a guarantee of optimality. Historically, Mixed-Integer Programming (MIP) formulations, solved via column generation [26] or branch-and-cut algorithms [27], have established the benchmarks for solution quality. For example, ILP has been successfully applied to complex

variants such as the VRP with Simultaneous Delivery and Pickup, handling heterogeneous fleets and time window constraints [28]. However, despite their theoretical soundness, exact methods suffer from the problem's $\mathcal{NP}$-hard nature. As problem instance sizes grow, the computational time required increases exponentially, making these methods prohibitive for large-scale, real-world logistics.

To mitigate the computational intractability of exact methods on large-scale instances, heuristic approaches prioritize computational efficiency over guaranteed optimality. These methods generally fall into the category of constructive heuristics, which generate feasible solutions through iterative, rule-based procedures. Prominent examples include the Nearest Neighbor and the Clarke-Wright Savings algorithms. The Nearest Neighbor algorithm offers rapid implementation by greedily selecting the closest unvisited node [29]; however, its myopic nature often leads to suboptimal routing decisions. Similarly, while the Clarke-Wright Savings algorithm remains a staple in industrial applications due to its speed [20], these deterministic heuristics share a fundamental limitation: they lack the mechanisms to retract decisions or explore the solution space broadly. Consequently, they are prone to becoming trapped in local optima, necessitating the use of more advanced search strategies.

2.4.2 Metaheuristics

To overcome the local optima entrapment inherent in constructive heuristics, metaheuristics provide a robust framework for navigating the solution space through advanced search strategies. As classified by Stamadianos et al. [30], these methods are generally distinguished into two categories based on their search trajectory: Population-based and Local Search methods. Population-based approaches, such as Evolutionary Algorithms and Swarm Intelligence, maintain a diverse pool of candidate solutions to explore the global search space effectively. Prominent algorithms in this domain include Genetic Algorithms (GA), Particle Swarm Optimization (PSO) [31], Artificial Bee Colony (ABC) [32], Cuckoo Search [33], and Ant Colony Optimization (ACO). Conversely, Local Search methods—such as 2-opt, Tabu Search, and Simulated Annealing—focus on exploitation by iteratively refining a single solution through neighborhood moves. While population methods excel at identifying promising regions within the search space, local search techniques are crucial for intensifying the search to uncover high-quality local optima.

As a prominent Local Search technique, Tabu Search (TS, introduced by Glover [34], employs a memory structure known as a "tabu list" to record recent moves and prevent cycling back to previously visited solutions. This mechanism allows the algorithm to escape local optima by facilitating the exploration of less promising areas of the search space in the short term. In the VRP domain, TS has been successfully adapted to handle complex constraints, such as those found in the Heterogeneous VRP (HVRP) [35, 36]. Building upon these advancements, Ahmed et al. [37] developed an improved tabu search algorithm to address the Heterogeneous Fixed Fleet Open Vehicle Routing Problem with Time Windows (HFOVRPTW). Their results demonstrated that this enhanced TS approach outperformed existing methods in terms of both solution quality and computational efficiency.

Another fundamental Local Search strategy is Simulated Annealing (SA), proposed by Kirkpatrick et al. [38]. Drawing inspiration from the thermodynamic process of annealing in metallurgy, SA distinguishes itself from strict descent methods by allowing the acceptance of inferior solutions based on a probabilistic criterion. This probability is governed by a "temperature" parameter that gradually decreases over time, a mechanism that enables the algorithm to escape local optima, particularly in the early stages of execution. Its utility in complex logistics networks is well-documented; for instance, Kancharla and Ramadurai [39] utilized SA to address the multi-depot two-echelon capacitated VRP. Their work highlights SA's robustness in handling the hierarchical constraints often present in modern distribution systems which is a structure that shares similarities with parcel locker networks.

While methods like Tabu Search and Simulated Annealing typically operate through small, local neighborhood moves, Adaptive Large Neighborhood Search (ALNS), proposed by Ropke and Pisinger (2006) [40], adopts a broader search trajectory by iteratively destroying and repairing significant portions of the solution. This metaheuristic distinguishes itself by maintaining a portfolio of removal and insertion operators, using an adaptive layer to select the most effective operators based on their varying performance during the search process. ALNS has proven particularly effective for complex VRP variants that share structural similarities with parcel locker networks. For instance, Hemmelmayr et al. (2012) [41] successfully applied ALNS to the Two-Echelon Vehicle Routing Problem (2E-VRP), dealing with intermediate satellites that function analogously to lockers. Similarly,

Masson et al. (2013) [42] utilized ALNS for the Pickup and Delivery Problem with Transfers, demonstrating the algorithm's capability to manage the tight synchronization constraints required when parcels must be handed off at specific locations.

Particle Swarm Optimization (PSO), originally proposed by Kennedy and Eberhart (1995) [43], has emerged as one of the most popular and widely cited swarm intelligence algorithms due to its simplicity and rapid convergence. Inspired by the social behavior of bird flocking and fish schooling, PSO operates by initializing a population of particles that move through the search space, updating their velocities based on their own best-known position (cognitive component) and the swarm's global best position (social component). While originally designed for continuous optimization, the method has been extensively adapted for combinatorial problems like the VRP. As detailed in the survey by Marinakis et al. (2025) [44], PSO is particularly favored for its ease of implementation and ability to quickly locate promising regions of the solution space. However, its high convergence speed can lead to premature convergence to local optima.

Harris Hawks Optimization (HHO), a recent addition to swarm intelligence proposed by Heidari et al. (2019) [45], mimics the cooperative hunting behavior of Harris' hawks, specifically their "surprise pounce" strategy. Unlike many predators that hunt alone, Harris' hawks work together to trace, encircle, and attack prey, dynamically adjusting their chasing patterns based on the prey's escaping energy. Mathematically, HHO models this process through two primary phases: exploration, where hawks randomly perch to detect prey, and exploitation, which involves soft and hard besiege tactics governed by the prey's depleting energy. In the context of vehicle routing, Alweshah et al. (2022) [46] adapted HHO to solve the classical VRP, utilizing its distinct exploration-exploitation balance to navigate complex constraints. Their comparative study demonstrated that HHO achieves superior convergence speed and solution quality compared to both single-solution methods like SA and other population-based methods like Artificial Bee Colony (ABC). This suggests that HHO's cooperative mechanism is particularly effective at avoiding local optima in combinatorial problems, making it a promising candidate for complex variants such as the VRPPL.

Despite numerous advancements in meta-heuristic and machine learning, the VRPPL remains a challenging problem due to its complex characteristics. A key bottleneck is the lack of efficient mapping mechanisms to translate customer permutations into delivery options, which is crucial for improving solution quality.

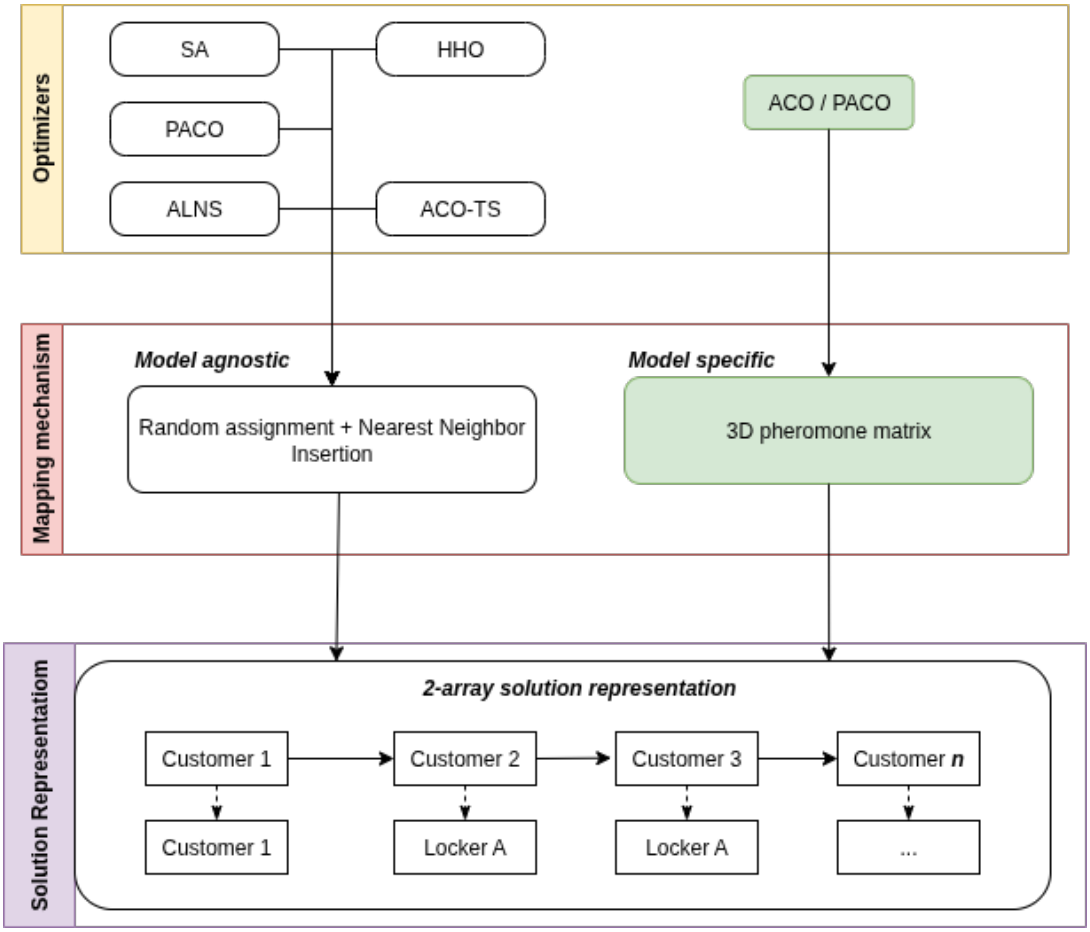


Figure 2.3: Meta-heuristic VRPPL Approaches: From Optimizer to Solution Representation

Yu et.al [8] introduces probabilistic assignment based on parameter p, and nearest neighbor heuristic insertion for initial solution. While the advantage of this method relies in its simplicity and model-agnostic, it suffers from a significant limitation: the mapping mechanism is not adaptive. This static nature requires extensive tuning for p and can lead to suboptimal solutions. Moreover, the stochastic nature of the method can result in significant variation between runs. Nguyen et.al [47] propose a two-level allocation algorithm, which employs the assignment-first, routing-second mechanism. However, similar to the approach by Yu et al. [8], this two-level mechanism also struggles to effectively address the intricate relationship between delivery options and the corresponding optimal routes.

To tackle these challenges, this research presented a multidimensional pheromone matrix as a model-specific mapping mechanism for ACO, or any constructive graph-based metaheuristics that can optimize the solution quality by effectively capturing the relationships between customer permutations and their preferred delivery options.

2.5 ACO and its extensions

Based on indirect pheromones trails, the artificial ants in ACO probabilistically construct solutions, and the pheromone trails are subsequently updated through both evaporation and deposition, guided by the optimality of the generated solutions. In ACO, a colony of artificial ants constructs solutions to a combinatorial problem by traversing a graph $G = (N, E)$. Each ant incrementally builds a path by moving from node i to node j with probability shown in belows:

$$P_{i\to j}^{(k)} = \begin{cases} \dfrac{\left[\tau_{ij}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{u\in\mathcal{N}_i^{(k)}}\left[\tau_{iu}\right]^{\alpha}\left[\eta_{iu}\right]^{\beta}} & \text{if } j \in \mathcal{N}_i^{(k)}, \\ 0 & \text{otherwise,} \end{cases} \tag{2.1}$$

where:

- τ_{ij} is the pheromone trail intensity on edge (i, j),
- $\eta = \frac{1}{d_{ij}}$ is the heuristic desirability (e.g., inverse distance),
- $\alpha, \beta > 0$ control the relative influence of pheromone vs. heuristic,
- $\mathcal{N}_i^{(k)}$ is the feasible neighborhood for ant k at node i.

After all ants complete their tours, pheromone trails are updated via

$$\tau_{ij} \leftarrow (1-\rho)\,\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{(k)}, \tag{2.2}$$

$$\Delta\tau_{ij}^{(k)} = \begin{cases} \dfrac{Q}{L^{(k)}} & \text{if ant } k \text{ used } (i,j), \\ 0 & \text{otherwise,} \end{cases} \tag{2.3}$$

where
$rho \in (0, 1]$ is the evaporation rate, Q is a pheromone deposit constant, and $L(k)$ is the tour length of ant k. Eq (2.1), (2.2) form the backbone of ACO positive feedback and distributed search mechanism

A key advantage of ACO for the VRPPL, when compared to other swarm-based metaheuristics, is its constructive nature and explicit incorporation of heuristic

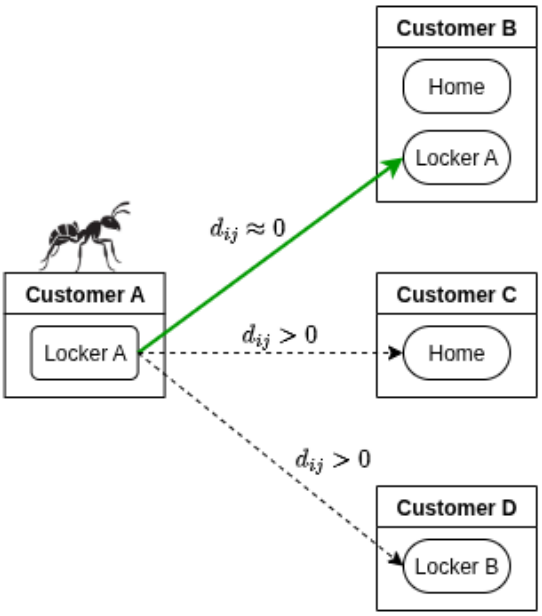


Figure 2.4: Demonstration of ACO decision making process

desirability. In this context, desirability is defined as the inverse of the distance between delivery locations. This feature allows ACO to effectively consolidate customers served by the same locker, even during the initial search stages, without heavy reliance on learned pheromone trails. As demonstrated in Figure 2.4, since the distance between consecutive demands delivered to the same locker is approximately zero (a small value is added to the denominator to prevent division by zero), the desirability value η_{ij} becomes exceptionally high. This inherent mechanism efficiently guides ants to create compact, consolidated routes from the outset.

An extensive body of research [48] has explored numerous extensions and variants of ACO for addressing both classical routing problems and their increasingly complex modern extensions. The first ACO algorithm is Ant System (AS) [49], followed by Ant Colony System [50] which introduces the indirect pheromone communication. Despite initial encouragement, it was not proved to be competitive against other state-of-the-art methods at that time. Elitist strategy or Elitist Ant System (EAS), mentioned in the same paper, extends AS by only allow best solutions to update pheromone trails, such solutions can be in within current iteration or global-best. $\mathcal{MAX} - \mathcal{MIN}$ Ant System ($\mathcal{MMAS}$) can be considered as a further improvement of EAS, introducing limitation interval for pheromone value $[\tau_{min}, \tau_{max}] \quad \forall \tau_{ij}$ to balance between exploration and exploitation.

Recent advancements in ACO frequently involve hybridization with other meta-

heuristics and the incorporation of adaptive strategies. For instance, Ren et al. [51] proposed a hybrid approach where ACO constructs initial solutions, which are then enhanced with adaptive tabu operators, and global pheromone is updated using Simulated Annealing (SA) as a replacement for elitist ants. Similarly, Wan et al. [52] leverage SA to refine solutions generated by elite ACO ants, aiming to escape local optima. Stodola et al. [53] integrate an adaptive pheromone evaporation rate, dynamically adjusted based on information entropy to maintain population diversity. Similarly, Xue [54] employs an Adaptive ACO (ADACO) for the Multi-Depot Vehicle Routing Problem with Time Windows (MDVRPTW), where the evaporation rate dynamically decreases with iteration count. Finally, Zhou et al. [55] model the Traveling Salesman Problem (TSP) as a Reinforcement Learning (RL) framework and implements Stochastic Gradient Descent (SGD) as an adaptive learning strategy for ACO.

2.6 Parallel Computing in ACO

While ACO has proven effective for $\mathcal{NP}$-hard combinatorial problems, its inherent iterative nature presents a significant computational bottleneck when applied to large-scale instances. The sequential process of solution construction and pheromone updates is computationally intensive, often restricting the algorithm's scalability in complex domains. Parallel computing addresses this limitation by distributing the workload across multiple processing elements, thereby facilitating the discovery of high-quality solutions within feasible execution times [56]. Crucially, parallelization in ACO serves a dual purpose: it not only accelerates the search process but also introduces distinct exploration patterns—often improving solution quality by mitigating the stagnation risks associated with sequential execution.

To navigate the diverse landscape of parallel ACO implementations, this research adopts the comprehensive taxonomy proposed by Pedemonte et al [2]. Unlike earlier classifications based solely on granularity, this modern framework categorizes strategies according to two primary dimensions: the number of colonies (single vs. multiple) and the presence of cooperation mechanisms. This distinction yields four fundamental architectural models: the Master-Slave model (centralized management), the Cellular model (structured diffusion), Parallel Independent Runs (isolated execution), and the Multicolony model (cooperative dis-

tributed search). This classification provides a robust structure for analyzing the trade-offs between computational efficiency and search diversity inherent in each approach.

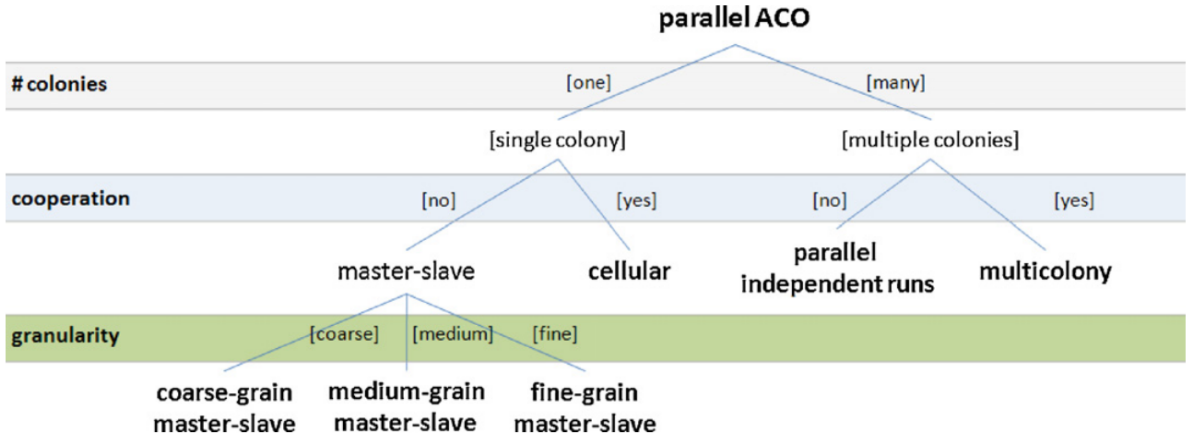


Figure 2.5: A hierarchical view of the new taxonomy for parallel ACO [2]

2.6.1 Master-Slave model

The Master-Slave model operates on a hierarchical structure where a single 'master' process manages global information—such as the pheromone matrix and the global best solution—while 'slave' processes execute the search. This model is typically categorized by task granularity.

In coarse-grained model, the master manages the central pheromone matrix while slaves work on complete solutions. An early example is the ANTabu method by Talbi et al. [57], which applied a coarse-grain ACO combined with Tabu Search to the Quadratic Assignment Problem (QAP) and demonstrated strong solution quality. Delisle et al. [58] used OpenMP to implement a multithreaded coarse-grain model for an industrial scheduling problem, achieving significant but sublinear speedups. To improve efficiency, a common strategy is to group several ants on a single processor. This approach was used by Doerner et al. [59] for the VRP and by Li et al. [60] for vector quantization codebook design, with the latter reporting high efficiency values (> 0.8). This strategy effectively amortizes the communication overhead between the master and slaves, as a single communication event can manage the results of multiple ants, thereby improving the efficiency-to-workload ratio. More recently, coarse-grain models have been adapted for Graphics Processing Units (GPUs). Zhu and Curry's [61] GPU implementation for continuous optimization reported remarkable speedups ranging from 128 to 403, showcasing

the potential of modern hardware architectures.

The medium-grained approach is based on domain decomposition, where the main problem is broken down into smaller subproblems. Mocholí et al. [62] implemented this model for the Orienteering Problem (OP) in a grid environment, where slaves found solutions for different clusters of the problem. A notable example is the D-ant algorithm developed by Doerner et al. [63, 64] for the VRP. In this model, slaves compute partial solutions that are then merged by the master. The improved implementation of D-ant outperformed other parallel models, including coarse-grain and multicolony approaches, by achieving superior efficiency with only a very small degradation in solution quality.

The fine-grained model is characterized by high-frequency communication, as slaves process individual solution components. This presents a significant challenge to scalability. Randall and Lewis [65] reported poor scalability for a TSP implementation due to the communication overhead of frequent local pheromone updates. To mitigate this, Delisle et al. [66] used shared memory and spaced out information updates, which improved performance without degrading solution quality. Recently, Fu et al. [67] developed a GPU-based implementation that achieved speedups of up to 30 but identified the communication between the CPU and GPU as a persistent bottleneck. The recent rise of GPU implementations [61, 67–69] represents a paradigm shift for the master-slave model. While early parallel ACO focused on distributed-memory clusters, these newer implementations leverage massively parallel shared-memory architectures. This fundamentally alters the communication cost model, making fine-grain approaches newly viable despite their historical scalability issues on traditional platforms.

2.6.2 The Cellular model

The cellular model is a recent addition to the parallel ACO landscape, inspired by similar models in evolutionary algorithms. It features a single colony structured in a grid, with pheromone updates localized to neighborhoods. The only implementation found in the literature is by Pedemonte and Cancela [70] for a network design problem. Their work reported very high computational efficiency (> 0.9) but noted a slight deterioration in solution quality compared to a standard sequential ACO. This suggests that while promising for performance, the model may require further tuning to match the solution quality of other approaches.

2.6.3 The Parallel Independent Runs model

This is the most straightforward parallelization strategy, consisting of a multi-start approach where several sequential ACO algorithms are executed concurrently with no communication between them. This model trivially achieves near-ideal efficiency and almost linear speedup because it has zero communication overhead. Stützle [71] demonstrated that this approach could find better solutions for the TSP than a single, longer sequential run. Similarly, studies by Alba et al. [72, 73] confirmed that this model had the highest efficiency but was sometimes outperformed in terms of solution quality by cooperative models that involve communication.

2.6.4 The Multicolony model

The multicolony model is an extensively used cooperative strategy that often yields superior results. In this model, multiple colonies with their own pheromone matrices explore the search space and periodically exchange information. The "standard" configuration, introduced by Michel and Middendorf [74, 75], involves a synchronous algorithm with a structured neighborhood and a fixed frequency for exchanging the best solutions. Numerous studies have demonstrated the effectiveness of this model. Middendorf et al. [76] applied it to the TSP and QAP, Chu et al. [77] developed the PACS algorithm for the TSP, and Xiong et al. [78] used a multicolony $\mathcal{MMAS}$ for large TSP instances. All of these works reported better solution quality compared to sequential versions. Researchers have also explored variants of the standard model. Chen and Zhang [79] used an adaptive frequency for information exchange. In-depth analyses by Manfrin et al. [80] and Twomey et al. [81] concluded that reducing the communication frequency can enhance exploration and ultimately lead to better results. This counter-intuitive finding suggests that excessive information exchange can lead to premature convergence, where all colonies begin to exploit the same region of the search space. Lowering the communication rate promotes greater diversity, allowing each colony to conduct a more independent and thorough exploration of different search space regions before sharing its findings, ultimately leading to higher-quality solutions.

2.6.5 The Hybrid model

This category comprises models that combine characteristics from two or more of the other parallel models. For example, Delisle et al. [66] proposed a theoretical two-level hybrid with a multicolony model at the upper level and a fine-grain master-slave model at the lower level. This theoretical proposal was recently adopted in the work by Liu et al. [82], where it was used to solve a routing problem in mobile networks. Another example is the ASqueen implementation by Iimura et al. [83], which combined master-slave and multicolony models to improve solution quality for the TSP. This detailed survey of individual implementations reveals a clear landscape of competing strategies, whose relative merits regarding computational efficiency and solution quality demand a direct comparative analysis.

2.6.6 Comparative Evaluation of Parallel ACO Strategies

Table 2.1: Characteristics of Parallel ACO Models

Model	**Population Organization**	**# Col.**	**# Matrices**	**Freq.**
Coarse-grain master-slave	Hierarchical, non-cooperative	1	1	Med.
Medium-grain master-slave	Hierarchical, non-cooperative	1	1	Med-High
Fine-grain master-slave	Hierarchical, non-cooperative	1	1	High
Cellular	Structured, cooperative	1	Many	Med.
Parallel independent runs	Distributed, non-cooperative	Several	Several	Zero
Multicolony	Distributed, cooperative	Several	Several	Low
Hybrids	Hierarchical D/P	D/P	D/P	D/P

For the hybrids category, D/P stands for "depends on the proposal,".

The selection of an appropriate parallel ACO architecture for complex optimization problems, such as the VRP with Parcel Lockers, necessitates a critical trade-off between two primary objectives: maximizing computational efficiency and achieving superior solution quality. This section provides a comparative evaluation of the reviewed models against these criteria, synthesizing strategic conclusions from the literature.

The computational efficiency of parallel ACO models is predominantly gov-

erned by the volume and frequency of communication required between processes. For practitioners aiming to balance high efficiency with scalability, particularly on large computing clusters, Coarse-grained Master-Slave and Multi-colony models represent the most viable choices. The Multi-colony model, in particular, demonstrates robust scalability, rendering it highly effective for computationally intensive problems.

In contrast, the efficiency of the Fine-grained Master-Slave model is inherently constrained by significant communication overhead. However, this bottleneck can be effectively mitigated through the utilization of shared memory architectures or modern GPU platforms, which facilitate low-latency communication. A more novel approach, the Medium-grained Master-Slave model, offers a compromise for large, decomposable problem instances, yielding acceptable to significant speedups depending on the underlying hardware.

At the extreme end of the efficiency spectrum, the Parallel Independent Runs model achieves near-linear speedup. By eliminating communication entirely, this "embarrassingly parallel" approach often outperforms Multi-colony architectures in terms of raw computational throughput, though it functions as a brute-force mechanism. Finally, the Cellular model exhibits high potential for speedup on modern parallel platforms, though its performance is strictly coupled with the defined neighborhood size, which dictates both communication costs and search granularity.

The impact of parallelization on solution quality is determined by how the chosen model alters the algorithmic search behavior relative to a sequential implementation.

Models such as the Master-Slave and Parallel Independent Runs generally yield solution qualities comparable to sequential ACO. In these architectures, the fundamental algorithmic behavior remains largely unchanged; however, the multi-start nature of independent runs provides a probabilistic advantage in escaping local optima and avoiding stagnation.

Conversely, the Multi-colony model is widely regarded as superior for achieving high-quality solutions across diverse problem domains. This advantage is attributed to its cooperative search mechanism, where distinct colonies exchange information to promote diversity and enhance exploration. This "cooperative

learning" approach contrasts sharply with the isolated search of independent runs. Furthermore, findings by Twomey et al. [81] suggest that regulating communication rates to prevent excessive homogenization is crucial for maintaining the exploration capabilities required to locate global optima.

Building upon this, Hybrid models, often constructed atop a multi-colony framework—typically inherit the capacity for superior accuracy. By integrating domain-specific heuristics or local search operators, hybrid approaches frequently outperform both standard parallel models and sequential algorithms. Ultimately, no single model is universally optimal; the architectural choice depends on whether the application prioritizes raw computational speed, solution accuracy, or a specific equilibrium between the two.

Chapter 3

Methodology

3.1 Mathematical Model

Table 3.1: Notation for parameters and decision variables

Parameters	
N	Set of all nodes, including depot, customers, and parcel lockers
A	Set of undirected edges connecting nodes
N_{HC}	Set of home delivery customers (type 1)
N_{LC}	Set of locker delivery customers (type 2)
N_{HLC}	Set of flexible customers (type 3)
N_{PL}	Set of parcel lockers
N_C	Set of all customers, $N_{HC} \cup N_{LC} \cup N_{HLC}$
N_{CPL}	Set of customers and parcel lockers, $N_C \cup N_{PL}$
d_i	Demand of customer $i \in N_C$
c_{ij}	Travel cost between nodes i and j
t_{ij}	Travel time between nodes i and j
$[a_i, b_i]$	Time window for service at node $i \in N_{PL}$
s_i	Service duration at node $i \in N_{PL}$
ℓ_{ij}	Binary parameter: 1 if the customer i chooses a locker j, 0 otherwise
ξ_j	Capacity of parcel locker $j \in N_{PL}$ (omitted in our model)
K	Set of vehicles
Q	Maximum capacity of each vehicle
M	A sufficiently large constant for time-dependent constraints
Decision Variables	
x_{ijk}	1 if the vehicle k travels from node i to node j, 0 otherwise
h_i	1 if the customer i receives home delivery, 0 otherwise
l_i	1 if the customer i receives locker delivery, 0 otherwise
y_{ij}	1 if the customer i is assigned to a locker j, 0 otherwise
μ_{ik}	Time vehicle k starts service at node i
∂_{k0}	Departure time of vehicle k from the depot
v_k	Time vehicle k completes its final delivery
ψ_{jk}	Total demand loads vehicle k delivers to locker j

The Vehicle Routing Problem with Parcel Lockers (VRPPL) is formulated

following the original model of Yu et al., [8] without further extensions. As shown in Table 3.1, in this problem, a homogeneous fleet of vehicles operates from a central depot to serve customer demands either at their homes or at parcel locker stations. The delivery network is represented as an undirected graph $G = (N, A)$, where N is the set of nodes and A is the set of edges between nodes.

Customers are divided into three types:

1. **Home delivery customers** (type 1) who must receive deliveries at home,
2. **Locker delivery customers** (type 2) who must receive deliveries at a parcel locker,
3. **Flexible customers** (type 3) who can receive deliveries either at home or at a parcel locker.

Parcel lockers are subject to time windows for service and predefined service durations. Each vehicle has a maximum load capacity and must start and end its route at the depot. The objective is to minimize the total travel distance while satisfying vehicle capacity and time-window constraints.

In the study of VRPPL and VRP generally, each delivery node can be annotated by i for departure node or j for the destination node. The depot within the delivery graph G is usually assigned the index 0, $N_0 = \{0\}$. For instance, x_{i0k} will be interpreted as the vehicle k returning to the depot after servicing the customer i. While there can be multiple indices for the depots in the field of Multi-Depot VRP (MD-VRP), this study focuses on the single-depot case. As a result, the customer nodes' index will be denoted as $i, j \in \{1, ..., n_C\}$ with n_C is the number of customers in N_C.

The main objective is to minimize the total travel distance of all vehicles:

$$D = \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \tag{3.1}$$

As this study follows the original model without further extensions, it presents only the key constraints. Constraints 3.2 - 3.7 guarantee that each customer is served exactly once, either at home or at a parcel locker, in accordance with their

specified delivery preference:

$$\sum_{k\in K}\sum_{j\in N} x_{ijk} = 1, \qquad \forall i \in N_{HC} \tag{3.2}$$

$$\sum_{k\in K}\sum_{j\in N} x_{ijk} \leq h_i, \qquad \forall i \in N_{HLC} \tag{3.3}$$

$$\sum_{k\in K}\sum_{j\in N} x_{ijk} = 0, \qquad \forall i \in N_{LC} \tag{3.4}$$

$$h_i + l_i = 1, \qquad \forall i \in N_C \tag{3.5}$$

$$h_i = 1, \qquad \forall i \in N_{HC} \tag{3.6}$$

$$l_i = 1, \qquad \forall i \in N_{LC} \tag{3.7}$$

Constraints 3.8 and 3.9 ensure that the total load to customer homes and consolidated demands to the lockers (ψ_{jk}) carried by each vehicle k do not exceed its capacity Q:

$$\sum_{k\in K} \psi_{jk} = \sum_{i\in N_{LC}\cup N_{HLC}} y_{ij} \times d_i, \qquad \forall j \in N_{PL} \tag{3.8}$$

$$\sum_{i\in N}\sum_{\substack{j\in N_C \\ i\neq j}} d_j \times x_{ijk} + \sum_{p\in N}\sum_{\substack{m\in N_{PL} \\ p\neq m}} \psi_{mk} x_{pmk} \leq Q, \qquad \forall k \in K \tag{3.9}$$

Constraints 3.10 - 3.12 track the arrival time at each visited node, while Constraint 3.13 ensures that the delivery time windows are respected for home-delivery customers:

$$\partial_{k0} + t_{0i} - \mu_{ik} \leq M(1 - x_{0ik}), \qquad \forall k \in K,\ i \in N_{CPL} \tag{3.10}$$

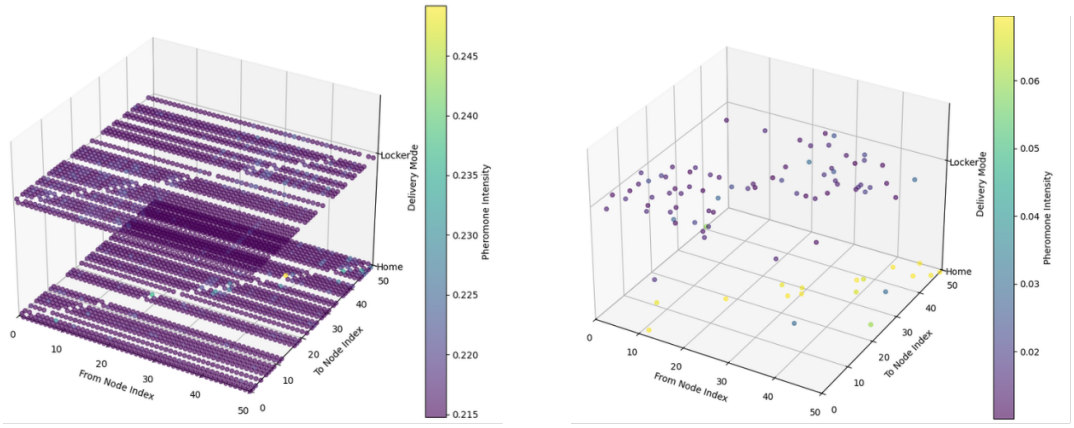
$$\mu_{ik} + s_i + t_{ij} - \mu_{jk} \leq M(1 - x_{ijk}), \qquad \forall k \in K,\ i \in N_{CPL},\ j \in N_{CPL} \tag{3.11}$$

$$\mu_{jk} - v_k \leq M(1 - x_{j0k}), \qquad \forall j \in N_{CPL},\ k \in K \tag{3.12}$$

$$a_i \times h_i \leq \sum_{k\in K} \mu_{ik} \leq b_i \times h_i, \qquad \forall i \in N_{HC},\ k \in K \tag{3.13}$$

Despite the mathematical model of the original paper defines ξ_j as the limited capacity of each locker, the published dataset does not include this information. Therefore, this thesis also omits the locker capacity constraint and primarily focus on operational cost such as vehicle capacity and time-windows.

3.2 3D Pheromone Matrix



(a) (b)

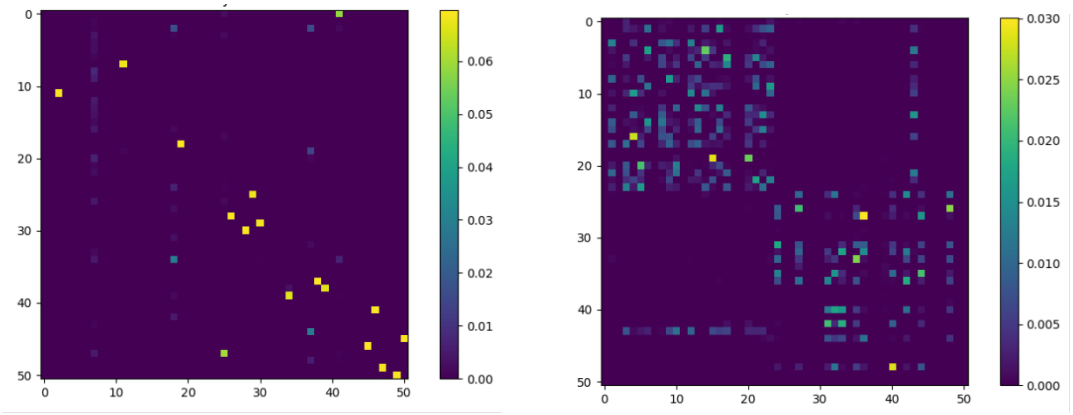
Figure 3.1: A visualization of a 3D pheromone matrix: (a) the initial pheromone density and (b) the pheromone density after being converged. The matrix has 3 dimensions: i-axis represents the departure delivery node index ($i \in \{0, ...n_C\}$), j-axis represents the destination delivery node index ($i \in \{0, ...n_C\}$) and o-axis represents the delivery options at each delivery node ($o \in \{0, 1\}$)

Despite widely applied, the traditional 2D pheromone matrix τ_{ij} cannot distinguish among multiple delivery options for each customer: it encodes only the desirability of traversing arc (i, j), not which delivery options (e.g., home, locker type) should be used. [8] addressed this by assigning delivery modes at random, however, this approach has three major drawbacks. Firstly, it requires careful manual tuning of the random-assignment probabilities regarding the problem scale. Secondly, it breaks down when there are more than two delivery options, forcing the model to default to "nearest locker" even if that is suboptimal. Thirdly, it is inherently unstable: the optimal customer permutation may be discovered but still fail to deliver efficiently because the randomly chosen mode may not match the best service option.

Addressing these limitations, this study proposes a 3D pheromone matrix τ_{ijo}, as shown in Figure 3.1, that explicitly tracks and learns the efficacy of each delivery option o for the customer j, enabling ants to make jointly optimized routing and mode decisions without manual tuning or restrictive assumptions.

To represent delivery-option effectiveness, the 2D pheromone matrix is extended to a three-dimensional tensor: $\tau_{ijo}, i, j \in N_C, o \in O$. The 3D matrix introduces O as a set the alternative delivery options (e.g., home delivery, locker

1, locker 2, ...). Each element τ_{ijo} measures the desirability for the vehicle k to travel from the customer i and serve the customer j via option o. Since VRPPL only considers two delivery options (home delivery and nearest locker delivery), we can simplify O to be 0 if home delivery is chosen and 1 if locker delivery is selected. A key advantage of the ACO family in comparison with other swarm-based meta-heuristics is its decision interpretability, using the pheromone density, as shown in Figure 3.2. The brighter color represents the higher density, thus indicating a higher probability that the ant agent will choose such an arc.



(a) (b)

Figure 3.2: An illustration of a converged pheromone density by layers: (a) the pheromone density at the home layer $o = 0$ and (b) the converged pheromone density at the locker layer $o = 1$

A feasibility mask $M_{ijo} \in \{0, 1\}$ is applied to enforce both home-delivery and locker-delivery constraints. $M_{ij1} = 0, \forall i \in N_C$ If the customer j is type I, and $M_{ij0} = 0, \forall i \in N_C$ if the customer j is type II. Before computing transition probabilities, we zero out all infeasible entries by masking $\tilde{\tau}_{ijo} = \tau_{ijo} \times M_{ijo}$. An advantage of the feasibility mask is that it only needs to be applied once during initialization, since a zeroed-out node will have no probability of being visited.

Both transitional probabilities and the pheromone update function are generalized for a 3D matrix. The generalized transition probability is given by:

$$P_{i \to j,o}^{(k)} = \begin{cases} \dfrac{\left[\tilde{\tau}_{ijo}^{(k)}\right]^{\alpha} \left[\eta_{ijo}\right]^{\beta}}{\sum\limits_{u \in \mathcal{N}_i^{(k)}} \sum\limits_{o' \in O} \left[\tilde{\tau}_{iuo'}^{(k)}\right]^{\alpha} \left[\eta_{iuo'}\right]^{\beta}}, & \text{if } j \in \mathcal{N}_i^{(k)} \\ 0, & \text{otherwise} \end{cases} \tag{3.14}$$

Equation (3.14) ensures that only feasible arc-option triplets (i, j, o) are considered

and that the probabilities across all valid (j, o) sums equal one.

3.3 Routes Construction

Algorithm 1 Sequential Construction with 3D Pheromones

Require: Graph G, 3D Pheromone Matrix τ_{ijo}, Heuristic η_{ijo}
Ensure: A complete solution S (set of routes)

$S \leftarrow \emptyset$, $U \leftarrow N_C$ (Unvisited customers)
$k \leftarrow 1$ (Current vehicle), $R_k \leftarrow [0]$ (Current route)
while $U \neq \emptyset$ **do**
  $i \leftarrow$ last node in R_k
  Calculate probabilities for all feasible $(j, o) \in U$ using Eq. (3.13)
  Select next demand (j, o) via roulette wheel selection
  if Adding (j, o) violates Capacity OR Time Windows **then**
    Close route R_k (add depot 0)
    $S \leftarrow S \cup \{R_k\}$
    $k \leftarrow k + 1$, $R_k \leftarrow [0]$
    $i \leftarrow 0$ (Reset to depot)
  end if
  if o is Locker AND $o ==$ location of node i **then**
    Aggregate demand j into current stop (Locker bundling)
  else
    Append (j, o) to R_k
    Update current load and time
  end if
  $U \leftarrow U \setminus \{j\}$
end while
Close final route R_k
return S

This study denotes each demand by the pair (i, o), where $i \in N_C$ identifies the customer and $o \in O$ specifies the chosen option. Routes are constructed incrementally using a Sequential Insertion Heuristic (SIH). Beginning at the depot $(i = 0)$, the next demand is selected (j, o) according to the transition probability defined in Equation 3.14. Upon inserting each new demand into the route, the vehicle's load and the current time are updated. This insertion process continues until adding any remaining demand would violate either the vehicle's capacity or the prescribed time windows. At that point, the route is deemed saturated, and a new route is initiated by resetting both load and time, modeling parallel vehicle deployments.

To streamline bundled deliveries to the same locker, any consecutive demands for that locker are aggregated into a single, larger demand (Figure 1.2). However, revisits to the same locker after servicing an intervening location are treated as violations and hence new vehicle will be deployed. Finally, if unsatisfied demands remain once all vehicles have been deployed, the overall solution is declared infeasible.

3.4 Coarse-grain parallel schema

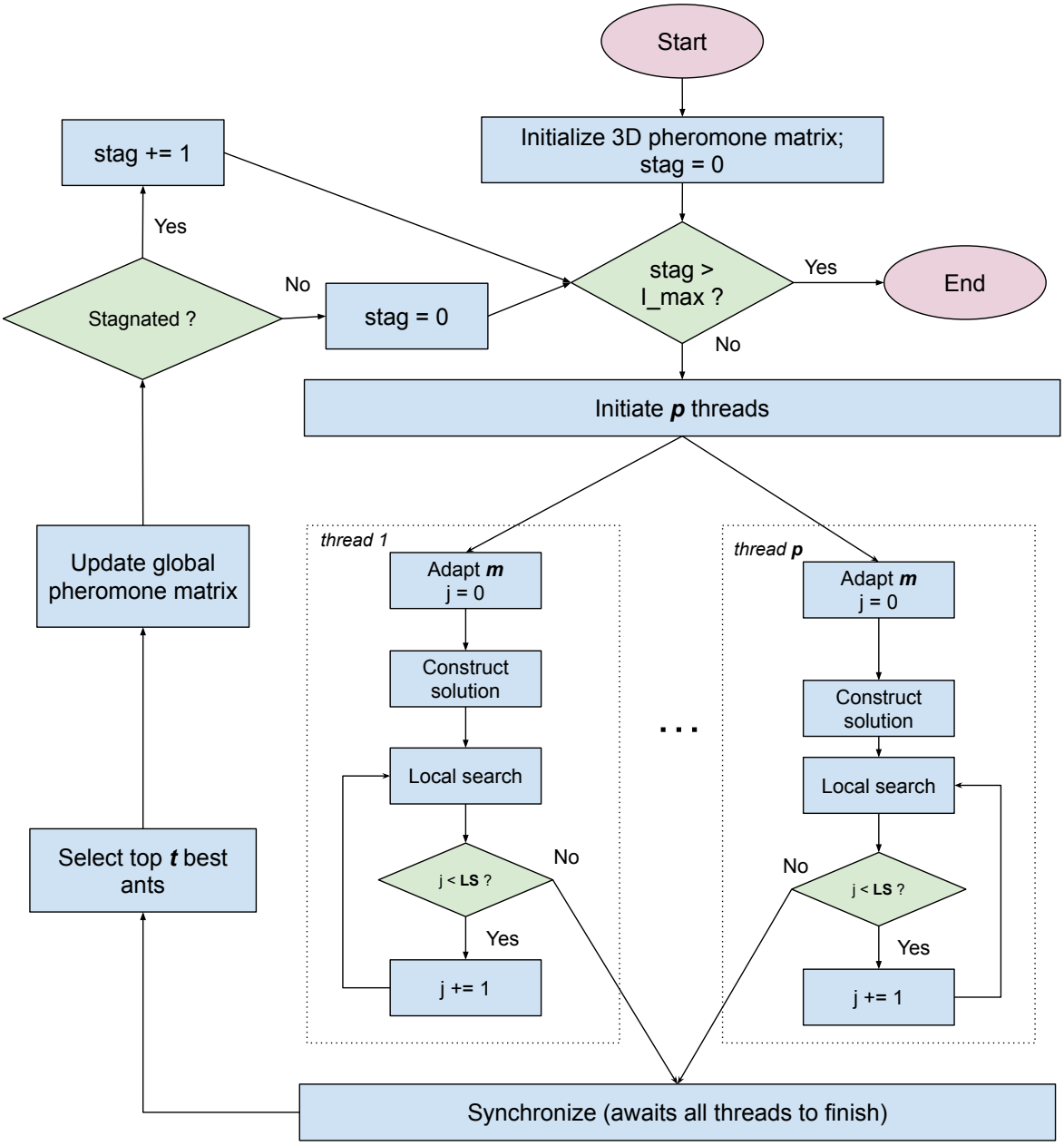


Figure 3.3: Coarse-grain parallel model

Under this 3D representation, the memory complexity is $O(|N|^2 \times |K| \times |O|)$ with $|N| = N_C + N_0$. In practice, since each ant k also carries its own tour data, the overall memory footprint grows linearly with the number of ants m, which can become prohibitive for large-scale instances or fine-grained parallelism. To mitigate these challenges, this thesis adopts a coarse-grain master–slave model, as shown in Fig 3.3.

In coarse-grain schema, Master process maintains the global resources, such as best solution so far, objective value of such solution, and shared memory pheromone matrix. Meanwhile, each Slave, corresponding to a processing unit, independently constructs complete solutions with objective values, using enhanced ACO model. The detailed implementation of the model will be explained in the next section.

Within each iteration, the solutions produced by all slave processes are asynchronously generated and then synchronously aggregated upon their completion. A key mechanism for managing convergence is the stagnation iteration counter, I_{stag}, which is incremented whenever the current iteration fails to discover a new global best solution. Following this aggregation, an elitist strategy is employed: only the top-t ants, identified by their superior objective values, are permitted to update the global pheromone matrix. By restricting pheromone deposition to this elite subset, the proposed model not only promotes effective ex of the search space but also significantly reduces the computational cost associated with pheromone synchronization.

Finally, by utilizing a shared-memory architecture for the global pheromone matrix, the algorithm eliminates the need to distribute copies of the matrix to slave processes. This significantly reduces both memory complexity from $O(m \times |N|^2 \times |O|)$ to $O(|N|^2 \times |O|)$, and the overhead associated with inter-process communication.

3.5 Complete 3D-PACO algorithm for VRPPL

The complete implementation of the proposed 3D-PACO algorithm for the VRPPL is detailed in Algorithm 2. The algorithm begins by initializing the 3D pheromone matrix τ_{ijo}, which encodes the transition desirability from customer i

Algorithm 2 3D-PACO Algorithm

Input: Graph $G = (N, E)$
Parameters: $\alpha, \beta, \rho, Q, m_{min}$ (ants per thread), p (processors), t (top elite ants), I_{max} (max stagnation), I_{LS} (local search iters)
Output: Best solution S^* and objective value D^*

Initialize pheromone matrix τ_{ijo}
Construct infeasibility mask M_{ijo}
$S^* \leftarrow \emptyset$, $D^* \leftarrow \infty$, $I_{stag} \leftarrow 0$
while $I_{stag} \leq I_{max}$ **do**
  $S_{\text{iter}} \leftarrow \emptyset$
  $m_p \leftarrow m_{min} \cdot \min(I_{stag} + 1, 8)$
  for each processor $j = 1$ to p **in parallel do**
    $S_{\text{local}}^{(j)} \leftarrow \emptyset$
    for each ant $a = 1$ to m_p **do**
      Construct solution S_a using 3D-ACO transition rule (Algorithm 1)
      for I_{LS} local search iterations **do**
        Apply local search (swap, insert, 2-opt) to S_a
      end for
      Evaluate D_{S_a}
      Add S_a to $S_{\text{local}}^{(j)}$
    end for
  end for
  Wait for all p processors to finish
  $S_{\text{iter}} \leftarrow \bigcup_{j=1}^{p} S_{\text{local}}^{(j)}$
  Sort S_{iter} by D
  Select top t elite solutions from S_{iter}
  Update τ_{ijo} using selected solutions
  Let S_{best} be the best in S_{iter}, D_{best} its cost
  if $D_{\text{best}} < D^*$ **then**
    $S^* \leftarrow S_{\text{best}}$, $D^* \leftarrow D_{\text{best}}$
    $I_{stag} \leftarrow 0$
  else
    $I_{stag} \leftarrow I_{stag} + 1$
  end if
end while
return S^*, D^*

to j under delivery option o. A feasibility mask M_{ijo} is constructed to eliminate invalid transitions. The algorithm maintains a global best solution S^* and its corresponding objective value D^*, initialized to an empty or infeasible solution.

While the stagnation counter does not reach its predefined limit, the algorithm initiates p threads to construct solutions independently. First, the number of ants per thread is adaptively scaled up based on stagnation level, with a maximum increase of 8-fold. This is formalized by $m = max[m_p * (I_{stag} + 1), 8]$, where m_p represents initial number of ants per thread. Each ant constructs a complete delivery route using a probabilistic transition rule (Eq. 3.14) that combines pheromone values with heuristic desirability. Crucially, infeasible solutions are accepted during this phase to enable a broader exploration of the solution space, recognizing that optimal or near-optimal solutions may emerge from transiently infeasible paths.

The output of ACO is a permutation of served customers and their mapping to delivery choices. However, a high pheromone concentration can lead to ants favoring specific trails, thereby restricting the exploration to particular regions of the solution space and resulting in the generation of repeated or less diverse solutions. To mitigate this limitation, each ACO customer permutation is subsequently explored using local search operations to create more diverse outcomes, with number of neighbors is LS. These local search operations are the same as those employed by Yu et al. [8], comprising insertion, swapping, and inversion.

Once all processors complete their assigned construction tasks, their local solutions are merged into a global solution pool S_{iter}. The master process then selects the top t elite solutions from S_{iter} to perform the global pheromone update using Eq. 3.16:

$$\Delta\tau_{ijo}^{(k)} = \begin{cases} \dfrac{Q}{L^{(k)}}, & \text{if } (i,j,o) \in \text{solution path of } k \\ 0, & \text{otherwise} \end{cases} \tag{3.15}$$

$$\tau_{ijo} \leftarrow (1-\rho) \cdot \tau_{ijo} + \sum_{k \in t} \Delta\tau_{ijo}^{(k)} \tag{3.16}$$

If the best solution in the current iteration improves upon D^*, it is saved as the new global best and reset the stagnation counter. Otherwise, it will increase the counter. If terminated, algorithm ultimately returns the best solution found and its objective value.

3.6 Software and Supporting Tools

To ensure high computational efficiency and precise memory management, the proposed 3D-PACO algorithm was implemented using the C++ programming language (specifically the C++17 standard). C++ was selected for its low-level memory manipulation capabilities and zero-overhead abstraction, which are critical when handling the iterative and stochastic nature of metaheuristics applied to $\mathcal{NP}$-hard problems like the VRPPL.

To address the computational intensity of the VRPPL and to exploit modern multi-core processor architectures, the implementation integrated the OpenMP (Open Multi-Processing) application programming interface. OpenMP was utilized to parallelize the ant construction phase, facilitating a shared-memory multiprocessing environment. This allowed multiple ant agents to construct solutions concurrently while accessing a shared pheromone matrix, significantly reducing the overall wall-clock time required for convergence.

The 3D-PACO model, as well as all comparative benchmark solvers used in this study, were implemented entirely from scratch, available in GitHub repository.[1]. No external black-box optimization frameworks or pre-existing metaheuristic libraries were employed. This approach ensured complete control over the algorithmic logic, data structures, and the specific constraint handling mechanisms required for the Parcel Locker components of the routing problem.

Python (3.12) was utilized as the primary environment for data analysis and visualization, serving as a flexible interface to interpret the extensive results generated by the metaheuristics solvers.

To visualize the VRPPL instances and the resulting routing topologies, the Matplotlib plotting library is employed. This tool allowed for the generation of high-resolution graphical representations of the solution space, mapping the specific locations of the central depot, parcel lockers, and customers. By plotting the optimized routes as directed edges between these nodes, Matplotlib assists in visually inspecting the feasibility of the solutions and the geometric characteristics of the tours produced by the 3D-PACO compared to the benchmark solvers. Furthermore, Matplotlib was instrumental in conducting the statistical performance

[1] https://github.com/chauanphu/routing-PL

analysis. It was used to generate line charts for sensitivty and scalability analysis, bar charts and box plots for benchmark performance comparison. These plots summarize the distribution of solutions across different solvers, highlighting the median performance, variance, and any outliers among the tested algorithms in term of solution quality and speed.

To further facilitate the dynamic inspection of these problem instances, a lightweight web-based interface was developed using the Streamlit framework. This application functions as an interactive visualization layer, allowing for the rapid verification of dataset topology and the qualitative assessment of route feasibility. By rendering the solutions generated by the C++ solver, this interface provides an immediate visual validation of the geometric characteristics of the tours, complementing the statistical performance analysis conducted via Matplotlib.

Chapter 4

Experiments and Discussion

This section is presented as follows: Subsection 4.1 presents dataset availability and benchmark for comparison; Subsection 4.2 presents computational and the source code repository for reproduction; Subsection 4.3 explains the hyperparameter tuning method for different optimizers; Subsection 4.4 and 4.5 discusses the robustness and scalability of 3D-PACO respectively; finally, Subsection 4.6 compares the proposed model's performance with original SA of Yu et al. as well as other methods.

Given that the original work does not benchmark its proposed Simulated Annealing against other meta-heuristic methods, this research adapted several others with the same random locker mapping as Yu et al. [8] to ensure consistent comparison. There methods include the Ant Colony Optimization with Tabu Search (ACO-TS) [84], Harris Hawk Optimization [46], Adaptive Large Neighborhood Search (ALNS) [85] and the Parallel Ant Colony Optimization (PACO) but without the multidimensional pheromone matrix. Next, the thesis performed a dedicated parameter tuning for this adapted algorithm using Bayesian Optimization (BO), consistent with the approach used for 3D-PACO.

4.1 Data Availability

The datasets used in this research are original benchmark datasets from Yu et al., available at the author's website.[1] These datasets are derived from Solomon CVRP instances, preserving their geographical distribution and customer demands. The author created three sizes: small (25 customers, 2 lockers), medium (50 customers, 3 lockers), and large (100 customers, 5 lockers). Each size includes 56 testing instances across three distribution types: clustered (C), random (R), and random-

[1] https://www.vincentyu.info/data

clustered (RC). The locker time window is identical to the depot's, and the service time at a locker is half of a customer's. A key assumption made in this study is that lockers have infinite capacity, as the provided datasets do not specify locker constraints.

The detailed computational benchmark results of Subsection 4.6 is published on Mendeley Data [86]. The dataset includes the best-known solutions, average distances, and runtimes for each method across all instances. This benchmark serves as a reference for future research in VRPPL and related optimization problems.

To facilitate the reproducibility of experiments and ensure consistent benchmarking against the state-of-the-art Simulated Annealing approach proposed by Yu et al. [8], this study adopts a standardized plaintext input format. The datasets employed are derived from the Solomon CVRP instances, adapted to include parcel locker characteristics while preserving the original geographical distributions and customer demands.

```
<num_customers> <num_lockers>
<num_vehicles> <vehicle_capacity>
<demand_1>
<demand_2>
...
<demand_n>
<x> <y> <earliest> <latest> <service_time> 0  % Depot (Type 0)
<x> <y> <earliest> <latest> <service_time> <type>  % Customer 1 / 2 / 3
...
<x> <y> <earliest> <latest> <service_time> 4  % Locker 1 (Type 4)
...
<locker_assignment_row_1>
<locker_assignment_row_2>
...
```

Figure 4.1: Structure of the VRPPL benchmark instance files used in this study.

As illustrated in Figure 4.1, the file is organized into four distinct logical blocks: global configuration, demand vectors, node attributes, and locker assignment constraints.

The first two lines establish the problem scale and fleet constraints. The parameters `num_customers` and `num_lockers` define the cardinality of the sets N_C and N_{PL}, respectively. The vehicle fleet K is constrained by `vehicle_capacity`

(Q), which is strictly enforced as per the mathematical model defined in Equation 3.8. Following the header, the demand vector lists the specific load requirements d_i for each customer $i \in N_C$.

The core topology of the graph $G = (N, A)$ is defined in the subsequent block. Each row corresponds to a node i and contains the following attributes:

- **Coordinates (x, y):** Euclidean coordinates used to compute the travel cost matrix c_{ij} and travel time matrix t_{ij}.
- **Time Windows (`earliest`, `latest`):** These values define the service interval $[a_i, b_i]$. Violations of these hard time windows render a solution infeasible, consistent with the constraints applied to home delivery customers and locker stations.
- **Service Time (s_i):** The duration required to service node i. Note that for locker nodes, the service time is typically half that of a customer node.
- **Node Type:** An integer identifier that categorizes the node into specific subsets for the VRPPL model:
  - **Type 0:** The central depot.
  - **Type 1-3:** Customer nodes, corresponding to N_{HC} (Home-only), N_{LC} (Locker-only), and N_{HLC} (Flexible) respectively.
  - **Type 4:** Parcel locker nodes (N_{PL}).

4.2 Empirical Configuration

Experiments were conducted on a computer with an Intel(R) Xeon(R) CPU E5-2696 v3 @ 2.30GHz, 62 GB of RAM, and 36 threads, achieving a performance of 196.17 GFLOPS. No GPU acceleration was used. The operating system was Ubuntu 24.04.

4.3 Hyperparameter tuning

To ensure fair and optimized performance comparisons, the hyperparameters of ACO-TS, ALNS, HHO and the 3D-PACO model were systematically tuned. Bayesian

Optimization [87] was chosen for this process, as it offers superior computational efficiency compared to exhaustive methods like Grid Search. PACO (without structured 3D matrix) utitlized the same parameters as 3D-PACO.

This study employs a Gaussian Process (GP) as a surrogate model to approximate the complex mapping between the 3D-PACO hyperparameters and the resulting solution quality. To navigate this search space, the Expected Improvement (EI) acquisition function is selected to balance exploration (investigating high-uncertainty regions) and exploitation (refining regions with known high performance).

The evaluation resolves the trade-off between solution quality (distance) and computational cost (runtime). For feasible solutions, the score is calculated by treating the runtime as a normalized penalty scaled by the objective value. Let D_{avg} be the average distance and T_{avg} be the average runtime over N samples. The valid score S_{valid} is defined as:

$$S_{valid} = D_{avg} \cdot (1 + (\mathcal{W}_{time} \cdot \frac{T_{avg}}{T_{\max}})$$

where $\mathcal{W}_{time}$ represents the weight of runtime penalty and T_{max} is the timeout budget.

Table 4.1: Parameter Search Ranges for 3D-PACO Bayesian Optimization

Parameter	**Small Dataset**	**Medium Dataset**	**Large Dataset**
m	[3, 20]	[5, 55]	[10, 70]
I_{max}	[3, 20]	[5, 30]	[15, 70]
α	[0.75, 1.75]	[0.75, 1.75]	[0.75, 1.75]
β	[0.75, 1.75]	[0.75, 1.75]	[0.75, 1.75]
ρ	[0.2, 0.8]	[0.2, 0.8]	[0.2, 0.8]
Q	[1.0, 3.0]	[1.0, 3.0]	[1.0, 3.0]
LS	[10, 100]	[10, 100]	[10, 100]
t	[20, 60]	[30, 80]	[30, 100]
p	64 (Fixed)	64 (Fixed)	64 (Fixed)

However, standard Bayesian Optimization does not inherently address the evaluation of infeasibility (caused by timeouts or invalid solutions). Since the global optima frequently reside within the neighborhood of infeasible regions, assigning an arbitrarily high static cost (e.g., a large constant) may create sharp discontinu-

ities in the surrogate model, discouraging the Bayesian Optimizer from exploring these promising neighborhoods. To mitigate this, a Dynamic Penalty Strategy is applied. Infeasible configurations are assigned a penalty proportional to the worst valid score observed so far:

$$S_{invalid} = \begin{cases} 1.5 \times \max(S_{valid}) & \text{if } S_{valid} \neq \emptyset \\ 100,000 & \text{otherwise} \end{cases} \tag{4.1}$$

Given that native C++ implementations for BO are not readily available, this thesis leveraged the scikit-optimize library within a Python environment to manage the optimization routines. Recognizing the substantial computational expense of independently optimizing parameters for all 56 instances within each dataset, a pragmatic approach is adopted: five instances were randomly sampled from each dataset. For every sampled instance, the average objective distance over five independent runs served as the fitness value for the BO algorithm. The number of threads used will always be set to the maximum available $p = 64$ (Table 4.1). The tuned set of parameters is shown in Table 4.2.

Table 4.2: Tuned parameters of 3D-PACO

	m	I_{max}	LS	p	α	β	ρ	Q	t
small	10	10	20	64	1.25	1.0	0.5	2.0	40
medium	30	10	50	64	1.25	1.0	0.5	2.0	40
large	50	10	100	64	1.25	1.0	0.5	2.0	40

4.4 Sensitivity Analysis

Next, a One-Factor-At-a-Time (OFAT) approach is employed to analyze the impact of parameter variance across different datasets. Using the BO-tuned parameter set as the baseline, this study systematically varied each parameter within a predefined range while keeping all other parameters fixed. The percentage deviation PD (%) was recorded for each variation, reflecting the parameter's influence on the model's performance. For computational efficiency, 5 instances were randomly selected from each dataset, and the experiment was conducted five times on each instance. The performance were aggregated by averaging the results across these runs.

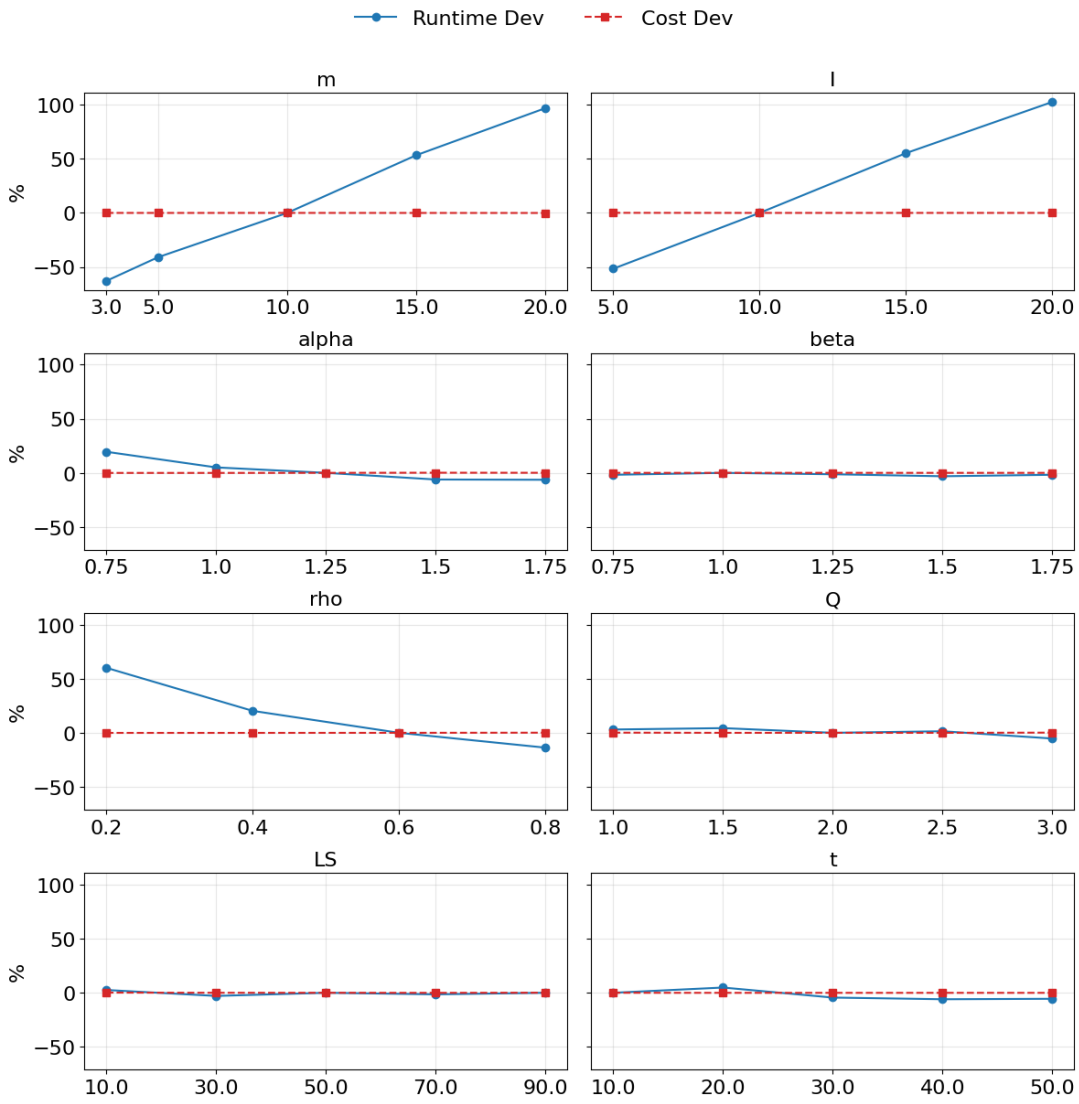


Figure 4.2: Sensitivity on small dataset

4.4.1 m (Ants per thread) and I (Stagnation Iterations)

Firstly, this study inspects the two parameters that most directly control the search budget and sampling effort: the number of ants per thread (m) and the maximum allowed stagnation iterations (I). Across all instance sizes these parameters show the clearest trade-off between runtime and solution quality: increasing either m or I raises runtime in an approximately linear manner while providing diminishing returns in solution cost beyond a moderate threshold. In practice this means that modest populations and iteration budgets are sufficient to capture most of the attainable improvement, whereas overly small values can be harmful - on the medium and large tests. It can be observed that there was a dramatic quality degradation when I was set too low (costs doubled when $I < 5$ on medium and when $I < 10$ on

large instances) and some cost increases occurred when m dropped below about 30 on large instances. Thus, when tuning for larger problems it is safer to increase m and I moderately rather than to scale them down to reduce runtime; the observed thresholds are useful starting points for a problem-dependent calibration.

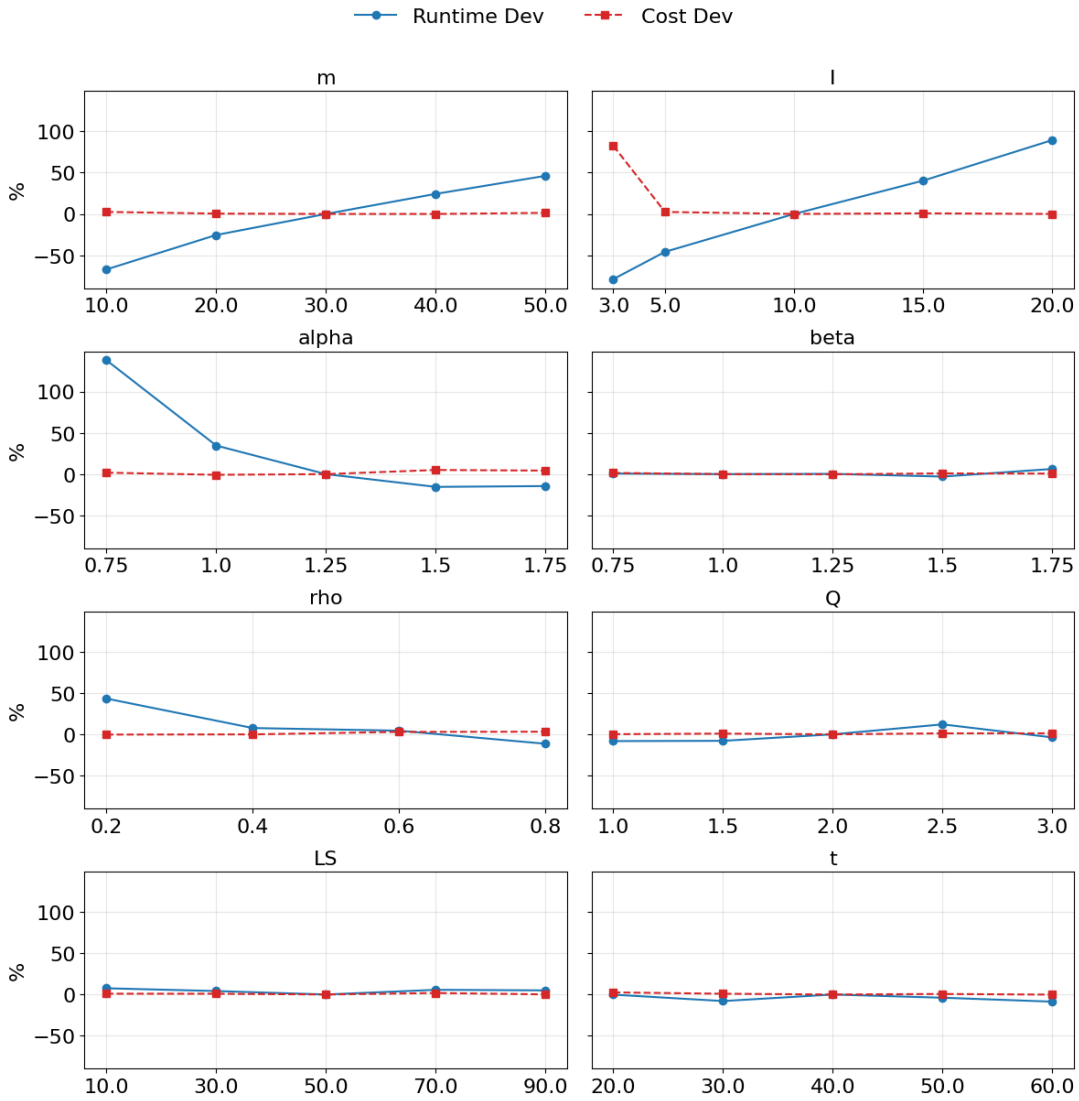


Figure 4.3: Sensitivity on medium dataset

4.4.2 α (Pheromone Importance) and β (Heuristics Importance)

The pheromone–heuristic balance, encapsulated by α (pheromone importance) and β (heuristic importance), exhibits markedly different behaviors: α has strong, dataset-dependent effects while β is comparatively inert within the tested range. Specifically, very low pheromone influence ($\alpha \leq 0.75$) caused severe runtime penal-

ties and poorer quality on larger instances. Specifically, runtime increases of roughly 50% to 150% and cost rises of the order of 50% were observed on medium/large sets, and on the large set the runtime curve is non-monotonic with a local minimum near $\alpha = 1.25$.

This indicates that pheromone strength interacts with problem scale and search stability: too little pheromone leaves the search overly heuristic-driven and unstable (longer, oscillatory searches), while a moderate pheromone influence helps stabilize convergence. By contrast, β produced only negligible fluctuations in both runtime and cost (typically under 3%), implying that the heuristic used is already informative and that modest changes to its weight do not qualitatively alter behavior. The practical implication is to prioritize careful tuning of α (especially for large problems), using a narrow grid around 1.0–1.5, while keeping β at a reasonable default (e.g., 1–2) and revisiting it only if the heuristic definition changes.

4.4.3 ρ (Evaporation Rate)

Increasing ρ (faster evaporation) tends to shorten runtime and reintroduce exploration by preventing long-lived reinforcement of suboptimal trails, while very low ρ values permit pheromone to accumulate on poor solutions and can cause premature convergence. The results show a steady reduction in runtime with higher ρ values, and premature fixation at low ρ particularly on the large dataset. Practically, this recommends a moderate-to-high evaporation setting for complex instances (so that the algorithm can escape plateaus), while too-aggressive evaporation should be avoided because it can wash out valuable learned information too quickly.

4.4.4 Q (Deposit Rate) and LS (Number of Local Search)

Two parameters that had comparatively small effects across the tested ranges are the pheromone deposit constant Q and the local-search (LS) frequency / intensity. Variations in Q produced only small runtime and cost fluctuations, suggesting that the absolute scale of pheromone updates matters less than the relative distribution of pheromone across edges; therefore Q may be set proportional to the objective magnitude and need not be exhaustively tuned. Local search showed only small sensitivity as well, but this hides an important conditional observation: the cost of invoking LS matters. When LS is cheap relative to the iteration cost, frequent use

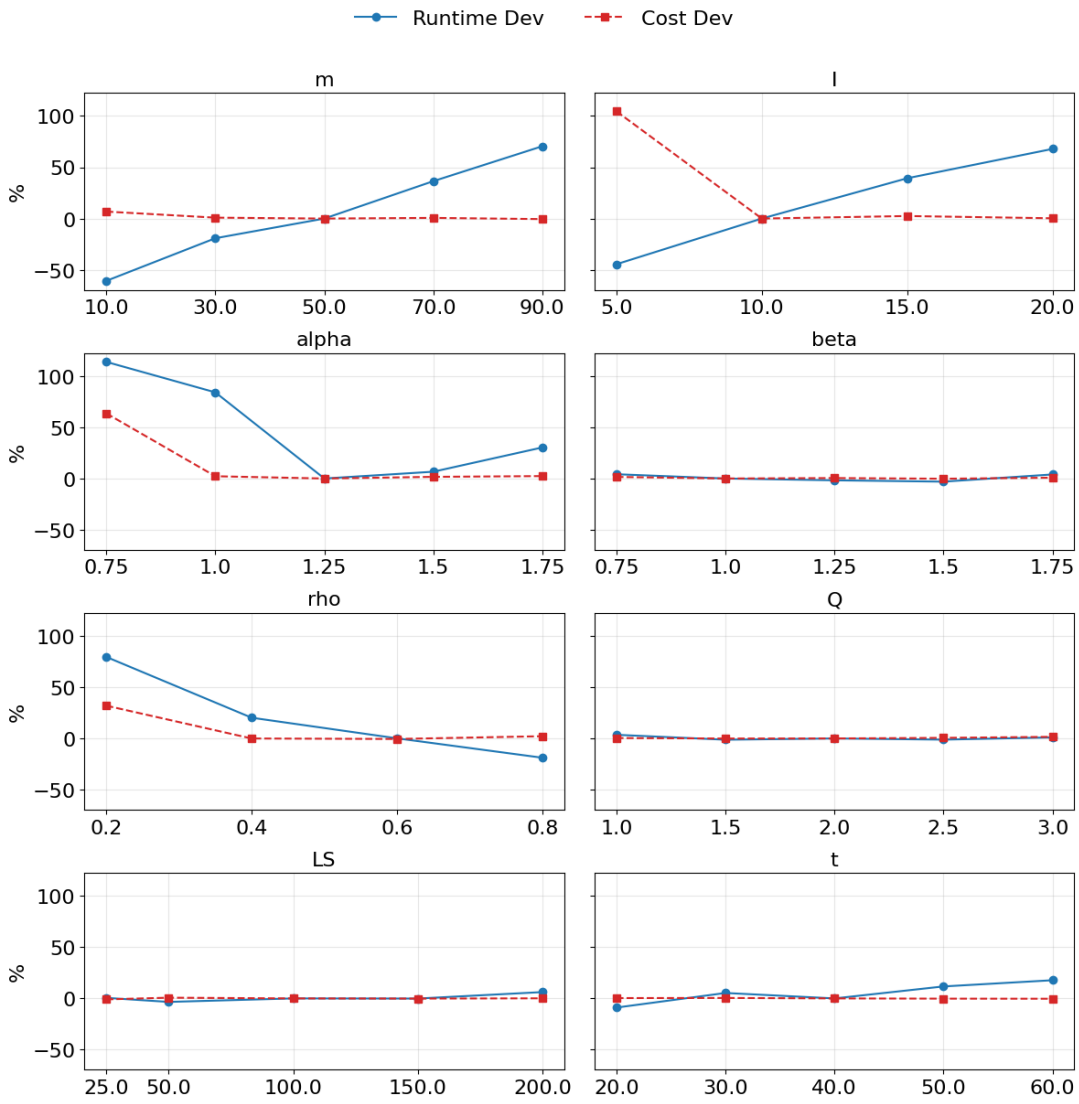
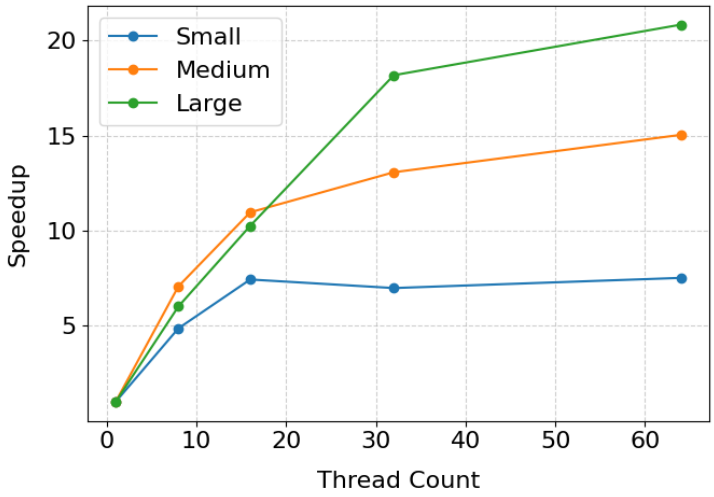


Figure 4.4: Sensitivity on large dataset

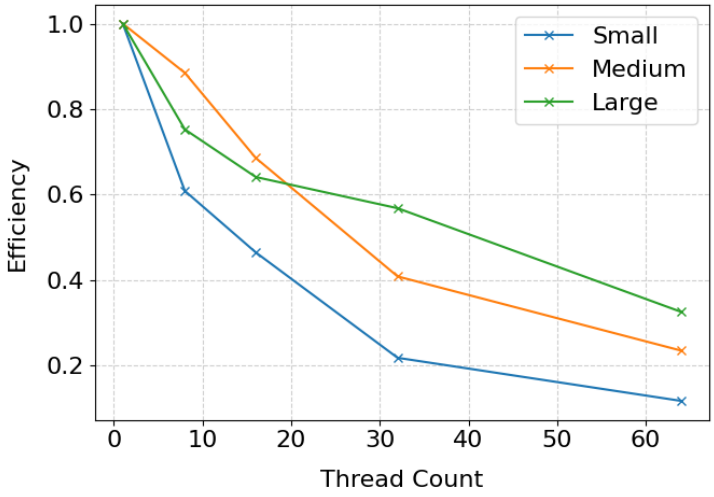
yields net gains; when expensive, it increases runtime with modest benefit. Thus the sensible rule is to use inexpensive or lightweight LS variants more aggressively and reserve heavy LS for occasional refinement of elite solutions.

4.5 Scalability Analysis

Model scalability will be assessed based on speedup (relative to its sequential version ($p = 1$), and resource utilization efficiency across varying thread counts (8, 16, 32 and 64). The number of ants per thread m is adjusted proportionally (m = total ants/p) to maintain a consistent global ant population. For instance, with a large-scale dataset and 32 threads, $m = 3200/32 = 100$.



(a)



(b)

Figure 4.5: Scalability Analysis: (a) Speedup and (b) Efficiency across data sizes.

$$S(p) = T(1)/T(p) \tag{4.2}$$

$$E(p) = S(p)/p = T(1)/(p \times T(p)) \tag{4.3}$$

where $T(p)$ is the runtime (measured in seconds) with p threads.

Fig. 4.5 offers insights into the scalability of the adaptive parallel ACO model, revealing distinct behaviors based on problem size. Fig. 4.5a demonstrates that for small problem instances, gains quickly saturate, peaking around 7-8x speedup with modest thread counts (approx. 16-20). This rapid plateau is attributed to the increasing dominance of parallelization overheads, such as communication and pheromone updates, over the limited workload. For medium instances, speedup is more sustained, reaching about 15x at 64 threads, indicating better resource

utilization. However, the model's true strength in scalability is evident with large instances, demonstrating robust scalability, especially for large datasets, achieving over 20x speedup with 64 threads despite sublinear scaling.

Fig. 4.5b further clarifies these trends. The overall decrease in efficiency as the thread count scales is a drawback of the master-slave schema, as mentioned by Pedemonte et al. [2]. Despite the coarse-grain model having the least communication frequency compared to the medium-grain and fine-grain models, the synchronization still has to wait for the slowest worker to finish before collecting the results. For small problems, efficiency sharply declines, confirming the poor utilization of additional threads, since overhead communication, which includes collecting the workers' results and updating the pheromone matrix, accounts for a larger proportion compared to the runtime per thread. Medium-scale problems show a more gradual efficiency drop. Critically, large-scale problems maintain significantly higher efficiency across the entire range of thread counts, reflecting a much more effective and economical use of parallel computing resources, since the runtime of each worker outweighs the synchronization cost.

These findings collectively establish the proposed adaptive parallel ACO as a highly promising solution for large-scale, computationally intensive VRPPL challenges. This indicates that despite the model's speedup being sublinear, 3D-PACO still displays remarkable runtime improvement.

4.6 Computational results

In this experiment, we measure the Wall-Clock Time as the primary standard for the solver's runtimes. Despite CPU Time being the benchmark to compare the algorithmic intensity across different algorithms, we argue that logistics stakeholders require near real-time response from the solver for making immediate actions; thus, providing the CPU Time may cause misleading impressions.

4.6.1 Interactive Visualization and Solution Output

To facilitate the qualitative assessment of the solutions produced by the 3D-PACO solver, a custom interactive dashboard was developed using the Streamlit frame-

work. This tool parses the solver's output to present key performance metrics, specifically total travel distance, fleet size, and runtime, alongside a detailed textual log of the routing logic and a graphical representation of the network topology.

Textual Route Representation

The solver outputs the optimal routing plan in a structured text format, as illustrated in Table 4.3. Each line corresponds to a single vehicle's route, starting and terminating at the depot (Node 0). A critical feature of this output is its ability to explicitly verify the customer-to-locker assignment decisions made by the 3D pheromone matrix.

Table 4.3: Detailed routing plan generated by the 3D-PACO solver. The notation $i(j)$ indicates that customer i is serviced at parcel locker j.

Route ID	**Sequence of Nodes**
Route 1	$0 \to 29 \to 9 \to 34 \to 24(52) \to 3(52) \to 23(52) \to 39(52) \to 22(52) \to 4(52) \to 35(52) \to 25 \to 0$
Route 2	$0 \to 5 \to 18 \to 45(53) \to 46(53) \to 16(53) \to 27(53) \to 37(53) \to 6(53) \to 8(53) \to 30(53) \to 36(53) \to 31(53) \to 47(53) \to 0$
Route 3	$0 \to 21(51) \to 33(51) \to 26(51) \to 12(51) \to 15(51) \to 28(51) \to 1(51) \to 41(51) \to 43(51) \to 2(51) \to 50(51) \to 40 \to 13 \to 0$
Route 4	$0 \to 11 \to 19 \to 10 \to 48(53) \to 38(53) \to 7(53) \to 32(53) \to 20(53) \to 49(53) \to 0$
Route 5	$0 \to 42 \to 14 \to 44 \to 17 \to 0$

As shown in Route 1 of Table 4.3, the notation `24(52)` indicates that Customer 24 is not visited at their home location but is instead served at Locker Node 52. Conversely, a standard entry such as `29` implies a traditional home delivery. This distinguishing format allows for immediate verification that Type-II (Locker-only) and Type-III (Flexible) customers are being correctly mapped to valid locker locations, while Type-I customers remain strictly bound to their home nodes.

Graphical Network Visualization

Complementing the textual logs, the dashboard renders the solution on a 2D Euclidean plane, as depicted in Figure 4.6. This visualization color-codes nodes based on their service type constraints, allowing for rapid inspection of route feasibility

and clustering behavior:

- **Depot (Type 0):** Marked by a red star (Node 0).
- **Home-only Customers (Type I):** Represented as blue circles (e.g., Node 11).
- **Locker-only Customers (Type II):** Represented as orange circles (e.g., Node 20).
- **Flexible Customers (Type III):** Represented as green circles (e.g., Node 32).
- **Parcel Lockers (Type IV):** Marked as purple squares (e.g., Nodes 51, 53).

The directed edges illustrate the sequence of visits. For instance, the visualization highlights how multiple customers are often consolidated into a single stop at a locker node (e.g., the convergence of routes at Node 53), visually validating the efficiency of the locker aggregation strategy.

4.6.2 Statistical test against Yu et al.

Since the distance and runtime may vary significantly within each dataset, the improvement of the proposed 3D-PACO compared to original SA method will be calculated as percent deviation (PD) of each instance:

$$\mathrm{PD}_{(\mathrm{BKS})}\,(\%) = \frac{D^{\mathrm{best}}_{\text{3D-PACO}} - D^{\mathrm{best}}_{\mathrm{SA}}}{D^{\mathrm{best}}_{\mathrm{SA}}} \times 100\% \tag{4.4}$$

$$\mathrm{PD}_{(\mathrm{D})}\,(\%) = \frac{D^{\mathrm{avg}}_{\text{3D-PACO}} - D^{\mathrm{avg}}_{\mathrm{SA}}}{D^{\mathrm{avg}}_{\mathrm{SA}}} \times 100\% \tag{4.5}$$

$$\mathrm{PD}_{(\mathrm{T})}\,(\%) = \frac{T_{\text{3D-PACO}} - T_{\mathrm{SA}}}{T_{\mathrm{SA}}} \times 100\% \tag{4.6}$$

For the statistical comparison of model performances, this study first conducts a Shapiro-Wilk test for normality on the differences between the paired observations to assess their distributional symmetry. Subsequently, a Wilcoxon signed-rank test or a paired t-test will be selected based on the outcome of the normality

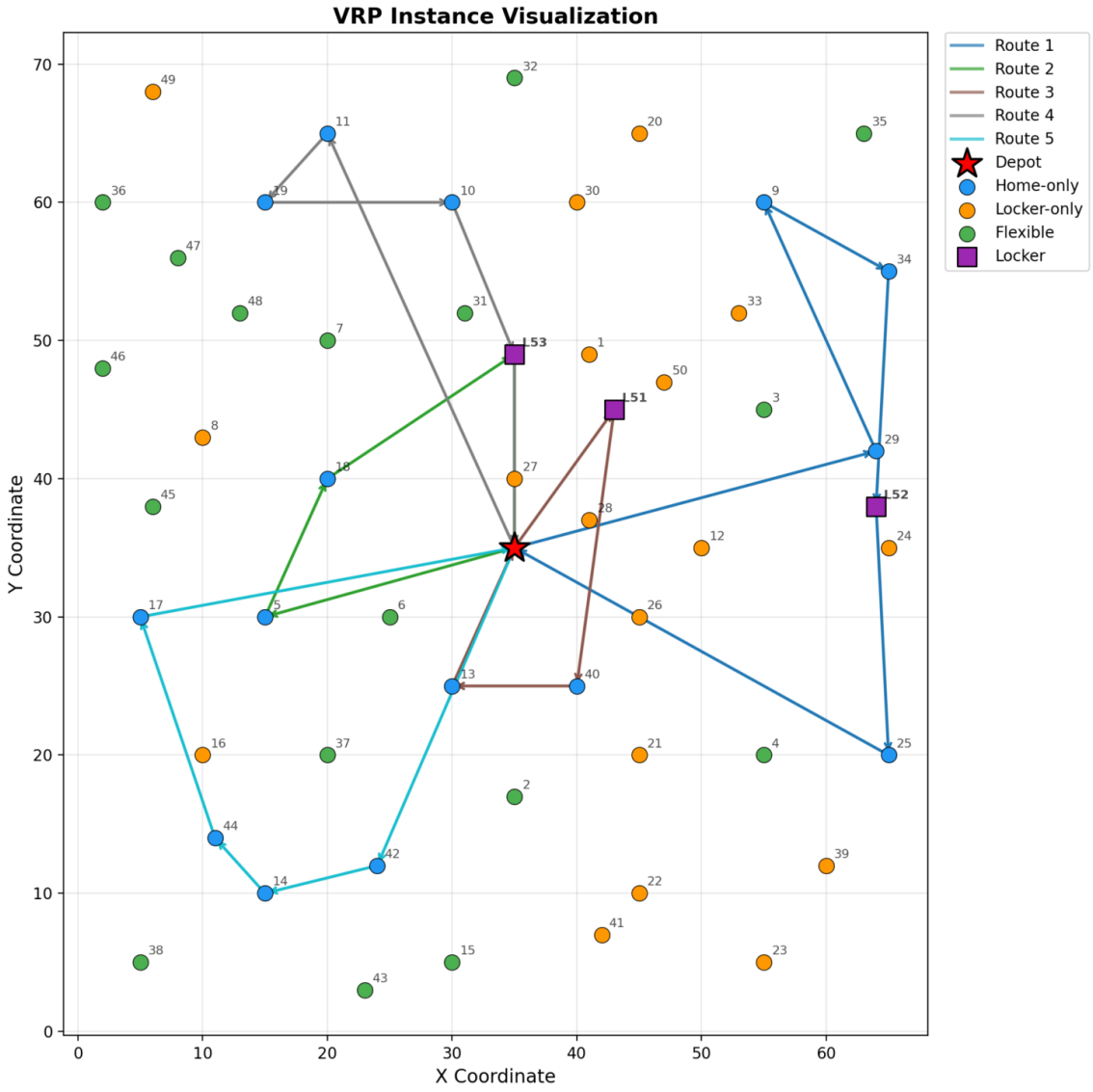


Figure 4.6: Graphical visualization of the R101_co_50.txt VRPPL solution.

test, with a significance level of $\alpha = 0.05$. The primary objective is to test whether the proposed 3D-PACO model significantly outperforms SA:

- $H_0 : \hat{PD}_{metric} = 0$
- $H_a : \hat{PD}_{metric} < 0$

where $\hat{PD}_{metric} = \frac{\sum \hat{PD}_{(metric)}}{N}(\%)$ represents the average performance difference between 3D-PACO and SA. A negative value indicates that 3D-PACO achieved a better performance for the given metric.

As shown in Table 4.4, for small instances, 3D-PACO, while achieving lower average distance and BKS, did not exhibit statistical significant advantage over

Table 4.4: Summary of Statistical Test Results

Dataset	Metric	Avg. PD	Normality Test (p-value)	Statistical Test	p-value	Conclusion
Small	BKS	-0.68	0.0000	Wilcoxon	0.7721	Not Significant
	Avg D	-0.51	0.0000	Wilcoxon	0.8178	Not Significant
	Avg T	-46.39	0.3680	T-test	0.0000	**Significant**
Medium	BKS	-1.30	0.0000	Wilcoxon	0.0005	**Significant**
	Avg D	-0.53	0.0000	Wilcoxon	0.9870	Not Significant
	Avg T	-83.52	0.0150	Wilcoxon	0.0000	**Significant**
Large	BKS	-2.65	0.5120	T-test	0.0000	**Significant**
	Avg D	-0.94	0.5080	T-test	0.0490	**Significant**
	Avg T	-49.96	0.0000	Wilcoxon	0.0000	**Significant**
Overall	BKS	-1.54	0.0000	Wilcoxon	0.0000	**Significant**
	Avg D	-0.66	0.0000	Wilcoxon	0.0770	Not Significant
	Avg T	-59.96	0.0000	Wilcoxon	0.0000	**Significant**

Denotes the significant difference exists.

SA on these metrics, respectively by both Pair t-test and Wilcoxon test. However, PACO achieves statistically significant runtime reduction by 46.4% and $p \approx 0$.

The performance differentials became more pronounced with increasing dataset complexity. On medium datasets, 3D-PACO significantly surpassed SA in terms of finding best routes (-1.3%), and remarkably reduces runtime by 83.52%. The average distance for medium dataset remained non-significant.

The most compelling performance advantages for 3D-PACO were observed on large datasets. Here, the proposed model consistently delivered statistically significant improvements across all evaluated metrics: a lower average distance ($-0.94\%, p = 0.049$), superior BKS (-2.65%) with half of runtime (-49.96%). T-tests were used for BKS and Avg D, while Wilcoxon was used for Avg T on large datasets.

Considering the overall aggregated performance, 3D-PACO maintained its significant advantage in average runtime ($\hat{PD} = -59.96\%$, p-value 0.001). However, the overall best known solution and average distance differences, while numerically favoring 3D-PACO, did not achieve statistical significance (p-values 0.772 for BKS and 0.077 for average distance, respectively), suggesting that while 3D-PACO generally performs better in quality, this improvement is not consistently significant across all dataset sizes when aggregated.

4.6.3 Benchmark comparison

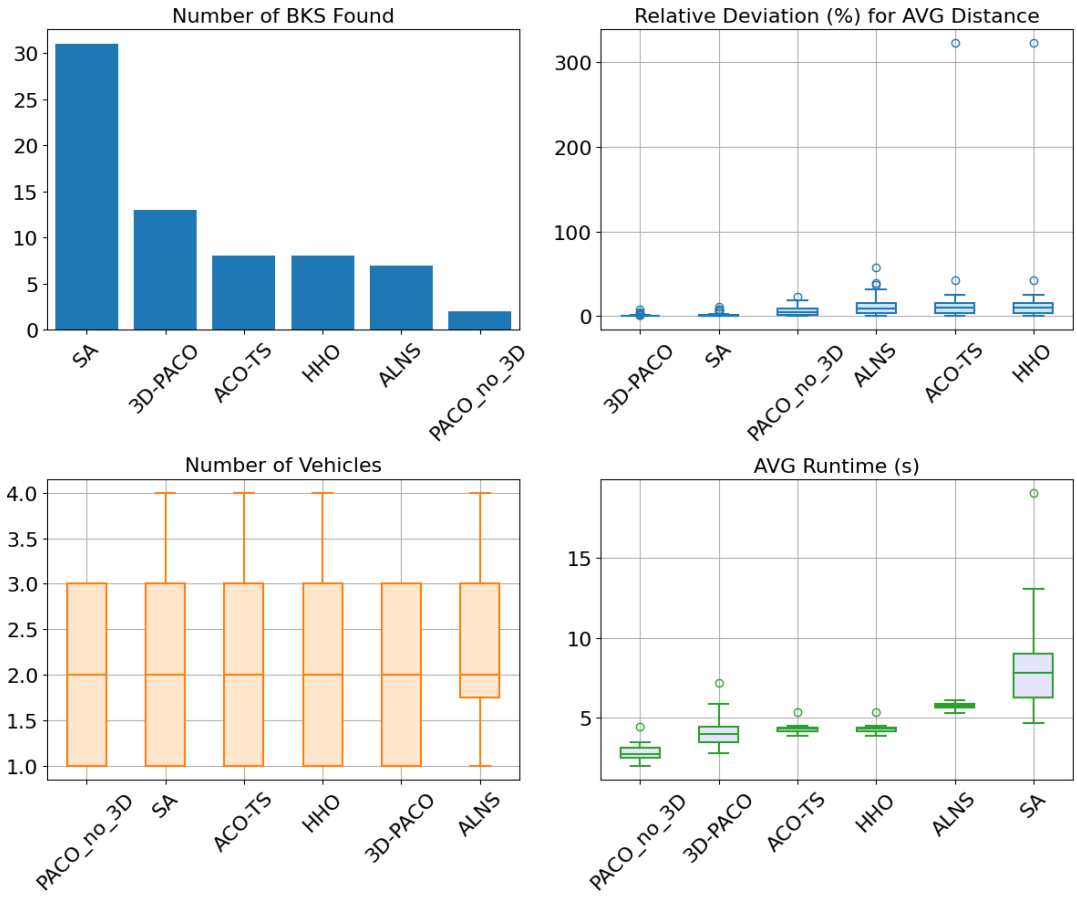


Figure 4.7: Benchmark comparison on the small dataset.

To evaluate the effectiveness of the proposed 3D-PACO, the thesis compared it against its non-3D counterpart (PACO), as well as four baselines: SA, ACO-TS, ALNS, and HHO. Performance was assessed using three metrics: number of best-known solutions (BKS) recovered, relative deviation from best average distance, and computational efficiency in terms of runtime. Figures 4.7–4.9 illustrate the results across the small, medium, and large benchmark datasets, while the aggregated statistics are summarized in Table 4.5.

On the small dataset (Fig. 4.7), SA maintains its dominance in terms of BKS discovery, recovering 31 out of 56 best-known solutions, followed by 3D-PACO with 13. The other metaheuristics trail behind, with ACO-TS, ALNS, and HHO each solving roughly eight instances, and PACO without the 3D pheromone matrix solving only two. Despite SA's clear strength in finding best solutions, the PACO family demonstrates remarkable consistency in average performance: the mean distance produced by 3D-PACO is slightly better than SA, and PACO

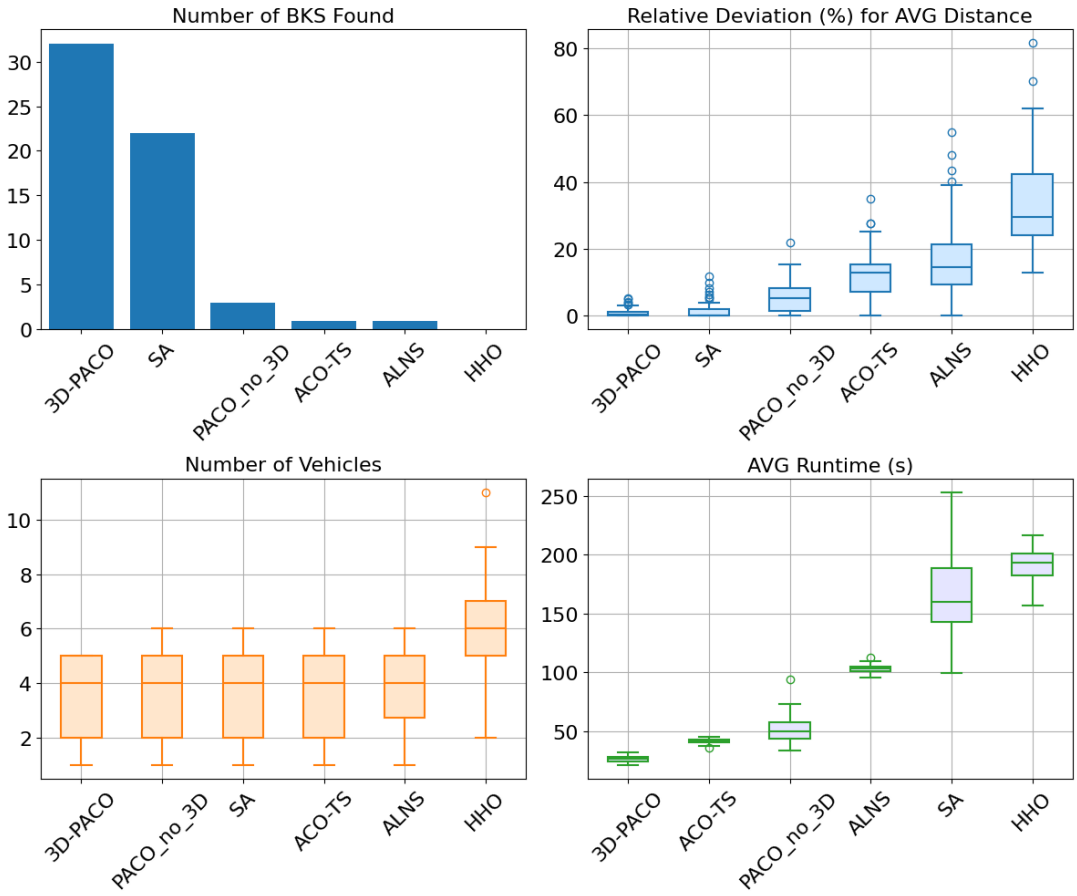


Figure 4.8: Benchmark comparison on the medium dataset.

(no 3D) remains competitive against ALNS, ACO-TS, and HHO. Where PACO clearly outperforms is in efficiency: 3D-PACO completes in just over four seconds on average, while PACO without 3D achieves the fastest runtime at 2.8 seconds. In contrast, SA is significantly slower and less predictable, ranging from 4 to 18 seconds depending on the instance. Vehicle usage was generally stable across methods, averaging two vehicles, but PACO solutions showed tighter distributions.

When moving to the medium dataset (Fig. 4.8), 3D-PACO emerges as the clear leader across all metrics. It discovers 32 BKS, the highest among all solvers, and delivers the lowest average distance (382.6). More importantly, 3D-PACO demonstrates strong runtime efficiency, solving instances in an average of 26 seconds, substantially faster than SA (166 seconds), HHO (191 seconds), and ALNS (103 seconds). PACO without 3D remains competitive in terms of runtime (52 seconds) but fails to achieve strong solution quality, recovering only three BKS and yielding worse average distances. The performance gap between PACO-family methods and the rest widens as problem scale increases, highlighting the importance of structured pheromone representation for handling larger neighborhoods efficiently.

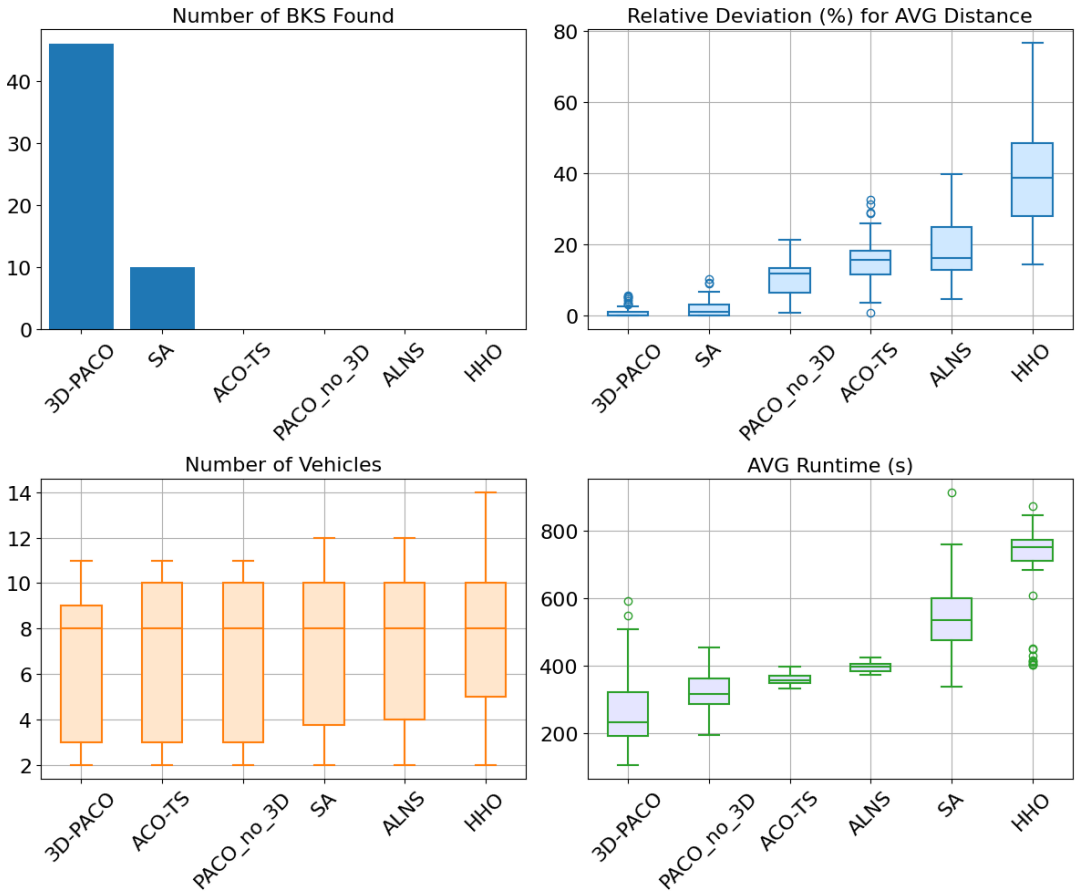


Figure 4.9: Benchmark comparison on the large dataset.

Vehicle usage results further confirm this advantage: 3D-PACO consistently identifies routes with fewer vehicles compared to ALNS and HHO, demonstrating better packing and assignment quality.

On the large dataset (Fig. 4.9), 3D-PACO solidifies its superiority, recovering 46 BKS out of 56 and maintaining the best average distance (693.2). SA performs relatively well, solving 10 instances, but suffers from prohibitively high runtimes (over 540 seconds on average) and much higher deviation from the best distances. Other methods, including ACO-TS, ALNS, and HHO, fail to scale: none discover more than a handful of BKS, and their average distances are substantially worse (all above 790). Runtime comparisons reinforce this trend: 3D-PACO remains efficient at scale, averaging 267 seconds, which is an order of magnitude faster than SA and significantly better than HHO (703 seconds). PACO without 3D shows slightly faster runtime than SA but still lags well behind in solution quality, reinforcing that the 3D matrix structure is not merely an incremental addition but a decisive factor in scaling solution quality and stability.

Table 4.5: Aggregated performances across datasets

Methods	**Metrics**	**Large**	**Medium**	**Small**
3D-PACO	Runtime (s)	**266.97**	**26.08**	4.08
	CPU Time (s)	17,086.08	1,669.12	261.12
	Average Distance	**693.17**	**382.55**	**214.44**
	# Best Distance Found	**46**	**32**	13
SA	Runtime (s)	546.40	165.73	8.09
	CPU Time (s)	546.40	165.73	8.09
	Average Distance	702.80	384.82	215.89
	# Best Distance Found	10	22	**31**
PACO (no 3D)	Runtime (s)	322.59	51.82	**2.81**
	CPU Time (s)	20,645.76	3,316.48	179.84
	Average Distance	756.52	400.40	225.93
	# Best Distance Found	3	3	2
ACO-TS	Runtime (s)	359.21	41.34	4.31
	CPU Time (s)	359.21	41.34	4.31
	Average Distance	792.31	427.63	241.19
	# Best Distance Found	0	1	8
ALNS	Runtime (s)	395.90	103.46	5.75
	CPU Time (s)	395.90	103.46	5.75
	Average Distance	808.37	440.76	236.52
	# Best Distance Found	0	1	7
HHO	Runtime (s)	702.69	191.23	4.31
	CPU Time (s)	702.69	191.23	4.31
	Average Distance	947.69	512.61	241.19
	# Best Distance Found	0	0	8

Bold indicates the best performance among methods for each metric.

Taken together as shown in Table 4.5, the results indicate that 3D-PACO offers a balanced and scalable solution strategy. In small instances, it is slightly behind SA in terms of best-solution discovery but already competitive in average solution quality and markedly superior in runtime. As the problem size grows, 3D-PACO surpasses SA and all other solvers decisively, combining robustness in discovering BKS with consistently lower average distances and highly efficient runtimes. The comparison between 3D-PACO and PACO without the 3D matrix makes clear that the novel multidimensional pheromone representation is responsible for escaping suboptimal solutions while providing timely responses, confirming the design motivation of our proposed method.

However, despite 3D-PACO and PACO outperforming other solvers in terms of wall-clock time, their architecture relies on mobilizing numerous threads ($p = 64$).

This leads to the average CPU Time, which is estimated by $AVG(T_{CPU}) \approx AVG(T) \times p$, displaying tremendous computational effort on different datasets. As a result, the benchmark in Table 4.5 suggests that the implementation of master-slave coarse-grain architecture achieves an advantage in reducing runtime by exchanging for significant costs for algorithmic intensity.

Chapter 5

Conclusion

5.1 Summary of Findings

This study introduced a 3D Parallel Ant Colony Optimization (3D PACO) model incorporating a share-memory multidimensional pheromone matrix to effectively address routing problems with complex delivery decisions, such as the VRPPL. Additionally, a coarse-grained master-slave architecture is implemented, leveraging an elitist strategy to enhance solution quality, and scaled adaptively number of ants per thread and evaporation rate based on stagnation.

As the problem complexity increases, 3D PACO exhibit gradual superiority in term of BKS and tremendous runtime compared to SA [8] and ACO-TS [84], due to optimizable locker assignment without relying on probabilistic assignment and nearest neighbor grouping.

5.2 Contributions of Research

5.2.1 Algorithmic and Scientific Impact

The core algorithmic contribution of this study lies in solving the interedependnece challenge inherent in the VRPPL. Traditional solvers frequently decouple the problem into sequential stages by assigning the deliveries to lockers, and sequently optimizing the route [8]. This sequential decomposition is algorithamtically brittle since it ignores the fact that optimal assignment is contigent upon the routing, and vice versa. This study advances the field by introducing the 3D pheromone matrix that allows ACO to navigate the assignment and routing decision space simultaneously. By extending the pheromone topology into a third dimension,

this study transforms two disjoint sub-problems into a single, holistic optimization task. This ensures that the algorithm escapes the local optima common in hierarchical heuristics.

Furthermore, the implementation of coarse-grain Parallel Architecture demonstrates that this expanded search space can be traversed without prohibitive computational costs, establishing a new precedent for solving highly coupled combinatorial problems with both precision and scalability.

5.2.2 Economic Impact

From an economic perspective, the proposed framework translates algorithmic precision directly into tangible Operational Expenditure (OPEX) reductions. In the low-margin logistics industry, even marginal gains in routing efficiency compound into significant fiscal savings. The proposed model achieves a reduced average travel distance of 693.17, compared to the baseline of 702.80. While statistically distinct, this reduction materially lowers variable costs associated with fuel consumption, vehicle depreciation, and carbon taxation. Furthermore, the computational efficiency (runtime) of the Parallel Architecture implies a reduction in overhead for route planning systems. By delivering high-quality solutions in reduced timeframes, logistics operators can achieve greater dynamic responsiveness—re-optimizing routes in real-time without incurring prohibitive server costs or delaying fleet deployment. Consequently, this solution offers a dual economic advantage: lowering the direct cost per mile while maximizing the operational throughput of the delivery fleet.

5.2.3 Social and Environmental Impact

Socially, this research directly supports the sustainable transition toward Out-of-Home Delivery (OOHD) models by maximizing the utility of parcel locker infrastructure. By rigorously optimizing the assignment and routing logic, the proposed algorithm mitigates the friction often associated with locker adoption, making OOHD a more viable alternative to traditional logistics. This consolidation of drop-off points is particularly critical for high-density urban environments, such as Ho Chi Minh City, where the prevalence of door-to-door delivery necessitates frequent stops and obstructive double-parking. By significantly reducing the stop

frequency of courier vehicles, this study's approach alleviates strain on municipal infrastructure and mitigates the traffic bottlenecks caused by last-mile logistics.

Environmentally, this shift reduces the idle time and stop-and-go driving patterns inherent in dense delivery routes, thereby lowering local carbon emissions and aligning commercial logistics with broader smart-city sustainability goals.

5.3 Limitations

Despite these advantages, the multidimensional pheromone matrix remains tied to the ACO framework and incurs additional computational overhead when synchronizing. Furthermore, due to expanding the search space to optimize assignment and routing simultaneously through the 3D pheromone matrix, 3D-PACO requires tremendous computational effort despite parallelism. Future research should explore approaches to optimize the energy efficiency without sacrificing the runtime and solution optimality.

Lastly, a key limitation of VRPPL's dataset is the lack of locker capacity information. Future work should address this limitation to ensure the practical application and investigate its application in multi-objective scenarios [88]. In terms of algorithm improvement, potential directions include exploring asynchronous parallelization (e.g., multi-colony models) to reduce synchronization costs and extending the model-agnostic mapping mechanism to other optimization frameworks.

5.4 Suggestions for Future Work

To advance the findings of this study, several future research directions can be explored to enhance both the parallel efficiency and the practical applicability of the algorithm. First, to mitigate the synchronization delays inherent in the current architecture, future investigations should prioritize asynchronous parallel schemes, such as multicolony models or independent parallel runs. Concurrently, the algorithmic exploration strategy requires further refinement to prevent early termination and premature convergence; promising enhancements include integrating the MMAS framework, adopting hybrid metaheuristics, or implementing objective-based adaptive mechanisms. Finally, to better align the model with

complex real-world logistics, empirical studies should assess the potential of the 3D-PACO model for multi-objective optimization [88], as well as its performance under limited locker capacity constraints, similar to the constraints modeled by Grabenschweiger et al. [25].

Appendix A

Awards & Publications

During my years studying Bachelor degree, I have participated in Parcel Locker Project as the project manager and researcher, lead by Assoc. Prof. Dr. Nguyen Thi Thuy Loan. The project has been run for 2 years and have gained several achievements, some of which establish the foundations for this thesis.

A.1 VNICT XVIII 2024 Conference

The project had its first publication in **VNICT XVIII 2024 Conference in Nha Trang** [89]. This research involves solving the multi-decision Vehicle Routing Problem using Graph Neural Network (GNN) and Reinforment Learning (RL). The dataset used in this proplem is generated, therefore, requiring for extensive searches of existing benchmark instances in further researches, and leading to eventual discovery of Yu et al.'s VRPPL instances.

Hội thảo quốc gia lần thứ XXVII: Một số vấn đề chọn lọc của Công nghệ thông tin và truyền thông – Nha Trang, 11-12/10/2024

Enhancing E-commerce Logistics with AIoT Smart Lockers: An AI-Technology Implementation for Urban Vietnam

An-Phu Chau, Tien-Son Nguyen, Phu-Quoc Pham, Tran-Khanh Huynh, Thanh-Danh Le, Hong-Son Nguyen, Kien.T Le, Thuy-Tien Trinh, Phuc-Quan Nguyen Minh, Chi Thanh Vi, Loan T.T. Nguyen

1School of Computer Science and Engineering, International University, Ho Chi Minh City, Viet Nam

2Vietnam National University, Ho Chi Minh City, Vietnam

chauanphu@gmail.com, tienson2003@gmail.com, ppquoc23@gmail.com, qcdz9r6wpcbh59@gmail.com, ltdanh.gdsciu@gmail.com, nguyenhongson5067@gmail.com, lt.kien@hutech.edu.vn, trinhthuytien1208@gmail.com, miqua2004@gmail.com, vcthanh@hcmiu.edu.vn, nttloan@hcmiu.edu.vn

Figure A.1: A published paper on VNICT Conference

A.2 Student Scientific Research Competition Eureka XXVI 2024

Figure A.2: Certificate of Participation in Eureka 2024

Based on the previous work of the conference paper, the Parcel Locker Project shifted the research for VRPPL solver from GNN-RL to metaheuristics. We implemented the Particle Swarm Optimization (PSO) solver with random assignment for VRPPPL problem. Furthermore, the research group also developed a zero-touch authorization protocol for smart locker system, allowing verification using QR scanning with minimal hardware installation. The functional smart locker with routing algorithm were submitted to the Student Scientific Research Competition Eureka XXVI 2024, which lead us to the **semi-final round**.

A.3 HUTECH IT GOT TALENT 2024

The project furthers combines PSO with Genertic Algorithm, creating the PSO-GA solver. The project products including a new algorithm, along with the refined parcel locker system was submitted to HUTECH IT GOT TALENT 2024 Competition. The work has won for the team the **Second Prize, City-level Award**, issued by Ho Chi Minh City Computer Association (HCA).

Figure A.3: Certificate for the 2nd Prize in IT Got Talent 2024

A.4 Swarm and Evolutionary Computation Q1 Journal

Figure A.4: A paper under review in Swarm and Evolutionary Computation

Recently, the work of the thesis has been submitted to **Swarm and Evolutionary Computation Q1 Journal**. The paper is already under the review of the second round with myself as the first author and Mrs. Nguyen Thi Thuy Loan as the corresponding author.

References

[1] V. E. commerce Association (VECOM), "Vietnam e-commerce business index report (ebi 2024)," Vietnam E-commerce Association, Tech. Rep., 2024. [Online]. Available: http://en.vecom.vn/vietnam-e-commerce-business-index-report-ebi-2024

[2] M. Pedemonte, S. Nesmachnow, and H. Cancela, "A survey on parallel ant colony optimization," *Applied Soft Computing*, vol. 11, no. 8, pp. 5181–5197, Dec. 2011.

[3] V. E. commerce Association (VECOM), "Vietnam e-commerce business index report (ebi 2025)," Vietnam E-commerce Association, Tech. Rep., 2025. [Online]. Available: https://vecom.vn/bao-cao-chi-so-thuong-mai-dien-tu-viet-nam-2025

[4] Wise Systems, "Last mile statistics 2023: Insights, innovations & trends," Wise Systems, Tech. Rep., 2023.

[5] T. T. Huong and B. N. Thiet, "Smart locker-a sustainable urban last-mile delivery solution: Benefits and challenges in implementing in vietnam," in *12th NEU-KKU International Conference Socio-Economic and Environmental Issues in Development*, 2020.

[6] "Parcel lockers have great potential in vietnam's delivery industry: Jll," *Hanoi Times*, 2025. [Online]. Available: https://hanoitimes.vn/parcel-lockers-have-great-potential-in-vietnams-delivery-industry-jll-45611.html

[7] L. Janinhoff, R. Klein, D. Sailer, and J. M. Schoppa, "Out-of-home delivery in last-mile logistics: A review," *Computers & Operations Research*, vol. 168, p. 106686, 2024.

[8] F. Y. Vincent, H. Susanto, P. Jodiawan, T.-W. Ho, S.-W. Lin, and Y.-T. Huang, "A simulated annealing algorithm for the vehicle routing problem with parcel lockers," *IEEE Access*, vol. 10, pp. 20 764–20 782, 2022.

[9] Mordor Intelligence, "Vietnam Courier, Express, And Parcel (CEP) Market Size & Share Analysis - Growth Trends And Forecast (2025 - 2030)," https://www.mordorintelligence.com/industry-reports/vietnam-courier-express-and-parcel-cep-market, 2025, accessed: November 24, 2025.

[10] M. An, "Parcel lockers have great potential in Vietnam's delivery industry: JLL," *Hanoi Times*, Sep. 2019, accessed: November 24, 2025.

[11] P. Krishn and Ken Research, "Vietnam Express Delivery Market Outlook to 2030," https://www.kenresearch.com/industry-reports/vietnam-express-delivery-market, Jul. 2024, accessed: November 24, 2025.

[12] B&Company, "Vietnam's Smart Lock Market: Unlocking Potential for Foreign Brands," https://b-company.jp/vietnam-smart-lock-market-unlocking-potential-for-foreign-brands/, Dec. 2024, accessed: November 24, 2025.

[13] Y. Deutsch and B. Golany, "A parcel locker network as a solution to the logistics last mile problem," *International Journal of Production Research*, vol. 56, no. 1-2, pp. 251–261, 2018.

[14] S. F. W. Lim, Q. Wang, and S. Webster, "Do it right the first time: Vehicle routing with home delivery attempt predictors," *Production and Operations Management*, vol. 32, no. 4, pp. 1262–1284, 2023.

[15] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[16] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: A review of representations and operators," *Artificial intelligence review*, vol. 13, no. 2, pp. 129–170, 1999.

[17] N. Vesselinova, R. Steinert, D. F. Perez-Ramirez, and M. Boman, "Learning combinatorial optimization on graphs: A survey with applications to networking," *IEEE Access*, vol. 8, pp. 120 388–120 416, 2020.

[18] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Computing*, vol. 8, no. 2, pp. 239–287, 2009.

[19] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[20] G. Clarke and J. W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations Research*, vol. 12, no. 4, pp. 568–581, 1964.

[21] G. D. Konstantakopoulos, S. P. Gayialis, and E. P. Kechagias, "Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification," *Operational research*, vol. 22, no. 3, pp. 2033–2062, 2022.

[22] T. J. Ai and V. Kachitvichyanukul, "Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem," *Computers & Industrial Engineering*, vol. 56, no. 1, pp. 380–387, 2009.

[23] H. C. Brandão de Oliveira and G. C. Vasconcelos, "A hybrid search method for the vehicle routing problem with time windows," *Annals of Operations Research*, vol. 180, no. 1, pp. 125–144, 2010.

[24] C. Tilk, K. Olkis, and S. Irnich, "The last-mile vehicle routing problem with delivery options," *OR Spectrum*, vol. 43, no. 4, pp. 877–904, 2021.

[25] J. Grabenschweiger, K. F. Doerner, R. F. Hartl, and M. W. Savelsbergh, "The vehicle routing problem with heterogeneous locker boxes," *Central European Journal of Operations Research*, vol. 29, pp. 113–142, 2021.

[26] T. I. Faiz, C. Vogiatzis, and M. Noor-E-Alam, "A column generation algorithm for vehicle scheduling and routing problems," *Computers & Industrial Engineering*, vol. 130, pp. 222–236, 2019.

[27] X. Zhang, L. Chen, M. Gendreau, and A. Langevin, "A branch-and-cut algorithm for the vehicle routing problem with two-dimensional loading constraints," *European Journal of Operational Research*, vol. 302, no. 1, pp. 259–269, 2022.

[28] S. Ropke and J.-F. Cordeau, "Branch and cut and price for the pickup and delivery problem with time windows," *Transportation science*, vol. 43, no. 3, pp. 267–286, 2009.

[29] G. Laporte, S. Ropke, and T. Vidal, "Chapter 4: Heuristics for the vehicle routing problem," in *Vehicle Routing: Problems, Methods, and Applications, Second Edition*. SIAM, 2014, pp. 87–116.

[30] T. Stamadianos, A. Taxidou, M. Marinaki, and Y. Marinakis, "Swarm intelligence and nature inspired algorithms for solving vehicle routing problems: a survey," *Operational Research*, vol. 24, no. 3, p. 47, 2024.

[31] Y. Marinakis, M. Marinaki, and A. Migdalas, "A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows," *Information Sciences*, vol. 481, pp. 311–329, 2019.

[32] D. Trachanatzi, M. Rigakis, M. Marinaki, Y. Marinakis, and N. Matsatsinis, "Distance related: a procedure for applying directly artificial bee colony algorithm in routing problems," *Soft Computing*, vol. 24, pp. 9071–9089, 2020.

[33] G.-H. Wu, C.-Y. Cheng, P. Pourhejazy, and B.-L. Fang, "Variable neighborhood-based cuckoo search for production routing with time window and setup times," *Applied Soft Computing*, vol. 125, p. 109191, 2022.

[34] F. Glover and M. Laguna, "Tabu search. handbook of combinatorial optimization," *Handbook of Combinatorial Optimization*, pp. 2093–2229, 1998.

[35] J. Brandão, "A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem," *Computers & Operations Research*, vol. 38, no. 1, pp. 140–151, 2011.

[36] D. S. Lai, O. C. Demirag, and J. M. Leung, "A tabu search heuristic for the heterogeneous vehicle routing problem on a multigraph," *Transportation Research Part E: Logistics and Transportation Review*, vol. 86, pp. 32–52, 2016.

[37] Z. H. Ahmed and M. Yousefikhoshbakht, "An improved tabu search algorithm for solving heterogeneous fixed fleet open vehicle routing problem with time windows," *Alexandria Engineering Journal*, vol. 64, pp. 349–363, 2023.

[38] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[39] S. R. Kancharla and G. Ramadurai, "Multi-depot two-echelon fuel minimizing routing problem with heterogeneous fleets: Model and heuristic," *Networks and Spatial Economics*, vol. 19, no. 3, pp. 969–1005, 2019.

[40] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transportation science*, vol. 40, no. 4, pp. 455–472, 2006.

[41] V. C. Hemmelmayr, J.-F. Cordeau, and T. G. Crainic, "An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics," *Computers & operations research*, vol. 39, no. 12, pp. 3215–3228, 2012.

[42] R. Masson, F. Lehuédé, and O. Péton, "An adaptive large neighborhood search for the pickup and delivery problem with transfers," *Transportation Science*, vol. 47, no. 3, pp. 344–355, 2013.

[43] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. ieee, 1995, pp. 1942–1948.

[44] Y. Marinakis, M. Marinaki, and A. Migdalas, "Particle swarm optimization for the vehicle routing problem: a survey and a comparative analysis," in *Handbook of heuristics*. Springer, 2025, pp. 1563–1596.

[45] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future generation computer systems*, vol. 97, pp. 849–872, 2019.

[46] M. Alweshah, M. Almiani, N. Almansour, S. Al Khalaileh, H. Aldabbas, W. Alomoush, and A. Alshareef, "Vehicle routing problems based on harris hawks optimization," *Journal of Big Data*, vol. 9, no. 1, p. 42, 2022.

[47] N. N. T. Nguyen and N. Van Hop, "Hierarchical allocation-routing heuristic algorithm for crowd-shipping problem with time windows, transshipment nodes, and delivery options," *Expert Systems with Applications*, vol. 268, p. 126325, 2025.

[48] M. Dorigo and T. Stützle, "Ant colony optimization: overview and recent advances," *Handbook of Metaheuristics*, vol. 272, pp. 311–351, 2018.

[49] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.

[50] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[51] T. Ren, T. Luo, B. Jia, B. Yang, L. Wang, and L. Xing, "Improved ant colony optimization for the vehicle routing problem with split pickup and split delivery," *Swarm and Evolutionary Computation*, vol. 77, p. 101228, 2023.

[52] F. Wan, H. Guo, W. Pan, J. Hou, and S. Chen, "A mathematical method for solving multi-depot vehicle routing problem," *Soft Computing*, vol. 27, no. 21, pp. 15 699–15 717, 2023.

[53] P. Stodola and J. Nohel, "Adaptive ant colony optimization with node clustering for the multidepot vehicle routing problem," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 6, pp. 1866–1880, 2022.

[54] S. Xue, "An adaptive ant colony algorithm for crowdsourcing multi-depot vehicle routing problem with time windows," *Sustainable Operations and Computers*, vol. 4, pp. 62–75, 2023.

[55] Y. Zhou, W. Li, X. Wang, Y. Qiu, and W. Shen, "Adaptive gradient descent enabled ant colony optimization for routing problems," *Swarm and Evolutionary Computation*, vol. 70, p. 101046, 2022.

[56] E. Alba, *Parallel metaheuristics: a new class of algorithms*. John Wiley & Sons, 2005.

[57] E.-G. Talbi, O. Roux, C. Fonlupt, and D. Robillard, "Parallel ant colonies for the quadratic assignment problem," *Future Generation Computer Systems*, vol. 17, no. 4, pp. 441–449, 2001.

[58] P. Delisle, M. Krajecki, M. Gravel, and C. Gagné, "Parallel implementation of an ant colony optimization metaheuristic with openmp," in *Proceedings of the 3rd European Workshop on OpenMP (EWOMP'01), Barcelona, Spain*, 2001, pp. 1–7.

[59] K. F. Doerner, R. F. Hartl, G. Kiechle, M. Lucka, and M. Reimann, "Parallel ant systems for the capacitated vehicle routing problem," in *Evolutionary Computation in Combinatorial Optimization: 4th European Conference, EvoCOP 2004, Coimbra, Portugal, April 5-7, 2004. Proceedings 4*. Springer, 2004, pp. 72–83.

[60] X. Li, X. Yu, and X. Luo, "Parallel implementation of ant colony optimization for vector quantization codebook design," in *Third International Conference on Natural Computation (ICNC 2007)*, vol. 4. IEEE, 2007, pp. 787–791.

[61] W. Zhu and J. Curry, "Parallel ant colony for nonlinear function optimization with graphics hardware acceleration," in *2009 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2009, pp. 1803–1808.

[62] J. A. Mocholí, J. Jaen, and J. H. Canos, "A grid ant colony algorithm for the orienteering problem," in *2005 IEEE congress on evolutionary computation*, vol. 1. IEEE, 2005, pp. 942–949.

[63] K. F. Doerner, R. F. Hartl, and M. Lucka, "A parallel version of the d-ant algorithm for the vehicle routing problem," *Parallel Numerics*, vol. 5, pp. 109–118, 2005.

[64] K. F. Doerner, R. F. Hartl, S. Benkner, and M. Lucka, "Parallel cooperative savings based ant colony optimization—multiple search and decomposition approaches," *Parallel processing letters*, vol. 16, no. 03, pp. 351–369, 2006.

[65] M. Randall and A. Lewis, "A parallel implementation of ant colony optimization," *Journal of Parallel and Distributed Computing*, vol. 62, no. 9, pp. 1421–1432, 2002.

[66] P. Delisle, M. Gravel, M. Krajecki, C. Gagné, and W. L. Price, "Comparing parallelization of an aco: message passing vs. shared memory," in *Hybrid Metaheuristics: Second International Workshop, HM 2005, Barcelona, Spain, August 29-30, 2005. Proceedings 2*. Springer, 2005, pp. 1–11.

[67] J. Fu, L. Lei, and G. Zhou, "A parallel ant colony optimization algorithm with gpu-acceleration based on all-in-roulette selection," in *Third International Workshop on Advanced Computational Intelligence*. IEEE, 2010, pp. 260–264.

[68] H. Bai, D. OuYang, X. Li, L. He, and H. Yu, "Max-min ant system on gpu with cuda," in *2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC)*. IEEE, 2009, pp. 801–804.

[69] A. Catala, J. Jaen, and J. A. Mocholi, "Strategies for accelerating ant colony optimization algorithms on graphical processing units," in *2007 IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 492–500.

[70] M. Pedemonte and H. Cancela, "A cellular ant colony optimisation for the generalised steiner problem," *International Journal of Innovative Computing and Applications*, vol. 2, no. 3, pp. 188–201, 2010.

[71] T. Stützle, "Parallelization strategies for ant colony optimization," in *International Conference on Parallel Problem Solving from Nature*. Springer, 1998, pp. 722–731.

[72] E. Alba, G. Leguizamón, and G. Ordonez, "Analyzing the behavior of parallel ant colony systems for large instances of the task scheduling problem," in *19th IEEE International Parallel and Distributed Processing Symposium*. IEEE, 2005, pp. 14–pp.

[73] E. Alba, G. Leguizamon, and G. Ordonez, "Two models of parallel aco algorithms for the minimum tardy task problem," *International Journal of High Performance Systems Architecture*, vol. 1, no. 1, pp. 50–59, 2007.

[74] R. Michel and M. Middendorf, "An island model based ant system with lookahead for the shortest supersequence problem," in *International conference on parallel problem solving from nature*. Springer, 1998, pp. 692–701.

[75] ——, "An aco algorithm for the shortest common supersequence problem," in *New ideas in optimization*, 1999, pp. 51–62.

[76] M. Middendorf, F. Reischle, and H. Schmeck, "Multi colony ant algorithms," *Journal of Heuristics*, vol. 8, no. 3, pp. 305–320, 2002.

[77] S.-C. Chu, J. F. Roddick, and J.-S. Pan, "Ant colony system with communication strategies," *Information sciences*, vol. 167, no. 1-4, pp. 63–76, 2004.

[78] X. Jie, L. CaiYun, and C. Zhong, "A new parallel ant colony optimization algorithm based on message passing interface," in *2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application*, vol. 2. IEEE, 2008, pp. 178–182.

[79] L. Chen and C. Zhang, "Adaptive parallel ant colony algorithm," in *International Conference on Natural Computation*. Springer, 2005, pp. 1239–1249.

[80] M. Manfrin, M. Birattari, T. Stützle, and M. Dorigo, "Parallel ant colony optimization for the traveling salesman problem," in *International workshop on ant colony optimization and swarm intelligence*. Springer, 2006, pp. 224–234.

[81] C. Twomey, T. Stützle, M. Dorigo, M. Manfrin, and M. Birattari, "An analysis of communication policies for homogeneous multi-colony aco algorithms," *Information Sciences*, vol. 180, no. 12, pp. 2390–2404, 2010.

[82] C. Liu, L. Li, and Y. Xiang, "Research of multi-path routing protocol based on parallel ant colony algorithm optimization in mobile ad hoc networks," in *Fifth International Conference on Information Technology: New Generations (itng 2008)*. IEEE, 2008, pp. 1006–1010.

[83] I. Iimura, K. Hamaguchi, T. Ito, and S. Nakayama, "A study of distributed parallel processing for queen ant strategy in ant colony optimization," in *Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05)*. IEEE, 2005, pp. 553–557.

[84] B. Yu, Z. Yang, and B. Yao, "A hybrid algorithm for vehicle routing problem with time windows," *Expert Systems with Applications*, vol. 38, no. 1, pp. 435–441, 2011.

[85] S. Voigt, "A review and ranking of operators in adaptive large neighborhood search for vehicle routing problems," *European Journal of Operational Research*, vol. 322, no. 2, pp. 357–375, 2025.

[86] A. P. Chau, L. Nguyen, and B. Vo, "Benchmark results for vrp with parcel locker," 2025.

[87] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, pp. 455–492, 1998.

[88] W. Chen, Y. Liu, and J. Liu, "The multi-objective vehicle routing problems with parcel lockers for simultaneous pick-up and delivery," in *2024 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2024, pp. 1–8.

[89] A.-P. Chau, T.-S. Nguyen, P.-Q. Pham, T.-K. Huynh, T.-D. Le, H.-S. Nguyen, K. T. Le, T.-T. Trinh, P.-Q. Nguyen Minh, C. T. Vi, and L. T. T. Nguyen, "Enhancing e-commerce logistics with aiot smart lockers: An ai-technology implementation for urban vietnam," in *The 27th National Conference on Selected Issues in Information and Communication Technology (VNICT 2024)*, Nha Trang, Vietnam, October 11-12 2024, pp. 162–167.