# LAST MILE ROUTING FOR LTL SHIPMENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

In this paper we consider a problem of last mile routing of LTL shipments based on our real life experience in route planning system development. The developed system processes over 100,000 transportation tasks across 300 cities daily. The goal of this paper is to combine the existing methods of route planning, adapt them for daily application, and add new constraints necessary for practical employment. The route planning system contains a risk model which fosters route stability over optimality. Experimental results, based on Homberger tests, demonstrate that the constructed algorithm performs remarkably well, with minimal deviations from best known solutions.

## 1 INTRODUTION

The Vehicle Routing Problem (VRP) was first formulated in Dantzig & Ramser (1959) as a generalization of the Traveling Salesman Problem. Due to the growth in e-commerce and logistics industry, the VRP has gained significant relevance and research in this field became of high importance.

In this context, the problem itself evolved and was divided into separate subtasks: Capacitated Vehicle Routing Problem (CVRP), Vehicle Routing with Time Windows (VRPTW), Capacitated Pickup and Delivery Problem with Time Windows (CPDPTW), Team Orienteering Problem With Time Windows (TOPTW), Vehicle Routing Problems with Backhauls (VRPB), Multi-depot vehicle routing problem with time windows (MDVRPTW), Asymmetric Distance Matrix (AVRP), and others. The algorithms evolved to be increasingly flexible with each new subtask, allowing for the solution of broader sets of real-world problems. The complexity of the problem grew with introduction of new constraints and the number of customers going from dozens initially to thousands now. This trend reflects the ongoing developments in the real world, where numerous businesses ship tens or even hundreds of thousands of items daily, all aiming to maximize their shipping efficiency.

The Vehicle Routing Problem (VRP) is an NP-hard problem making it practically impossible to find an exact solution to such a problem on large real data within a reasonable calculation time. Current approaches predominantly rely on simplified methods, such as heuristics, metaheuristics and reinforcement learning to achieve solutions that are close to accurate and realistic, as shown by articles Nagata & Bräysy (2009), Vidal et al. (2013), Duan et al. (2020), Beling et al. (2022). It appears to be a successful technique. We selected a metaheuristic based approach since we wanted to be able to quickly and flexibly adjust the planning logic and to ”justify” the reasons for forming routes, which is a crucial step in the system implementation. Therefore this paper focuses specifically on the application of metaheuristics to planning task.

The goal of this paper is to combine the existing methods of route planning, adapt them for daily application, and add necessary limitations for practical employment. To achieve this, a description of the specific business constraints and application-related features of the modeling results, which are necessary for real-world applicability, will be given. The paper primary focuses on minimizing risks to fail the route rather than pursuing the optimal solution, prioritizing the stability of routes. It's important to note that this paper only addresses the process of the last mile of LTL shipments, specifically the delivery and pickup of cargo. This narrow scope allows us to highlight the features of the problem and identify the difficulties in implementing optimization algorithms. We will provide the mathematical formulation of the problem for these restrictions and discuss high-level algorithms tailored to address these restrictions. It's crucial to take into account the hardware requirements and maximum calculation time for practical applications. Therefore, the algorithms should approach near-optimal solutions quickly while agreeing with performance constraints.

The work draws on practical experience in the development and implementation of a planning system, that handles over 125,000 transportation tasks daily for 300 cities. For over seven years, this system has been in continuous operation while undergoing regular modifications to meet evolving business requirements.

The paper is structured as follows: Section 2 outlines the problem formulation, Section 3 presents a mixed-integer programming (MIP) formulation of the problem, and Section 4 highlights main features of the implemented system. In Appendix we provide Algorithm description and Data Preparation.

## 2 THE MODEL AND NOTATION

In the classical interpretation, this is a task of creating a set of routes from a depot to a group of customers with the lowest price (number of vehicles, distance, time or cost). The routes must satisfy the following set of constraints: Each customer is visited exactly once; All routes start and end at the depot.

The last mile for LTL shipments involves delivering cargo from a depot to clients and picking up cargo from clients for further transportation, all within specified client time windows, using a heterogeneous vehicle fleet. One of the interpretations of this problem, known as the Pickup and Delivery Problem (PDP), includes tasks for both delivering cargo from the depot and picking up cargo to drop it off at the depot. It is also referred in the literature as the Vehicle Routing Problem with (Mixed) Backhauls (Parragh et al. (2008a), Parragh et al. (2008b)). When considering this problem we provide the following additional constraints:

- The total load handled by each vehicle on a route must not exceed the capacity of the vehicle.
- Each client must be served within their designated time window.

We use the following notation:

- Graph $G(V, E)$, where $V = \{v_0, v_1, ..., v_n\}$ is a set of vertices ($v_0$ — depot, $v_{1,..,N,N+1,...,n}$ — customers), $E$ — is a set of edges $\{(v_i, v_j)|i \neq j\}$.
- Delivery tasks are indexed by $i = 1, .., N$ and pickup tasks by $i = N + 1.., n$.
- $w^i_{cargo}$, $v^i_{cargo}$, $l^i_{cargo}$, $wi^i_{cargo}$, $h^i_{cargo}$ are the weight, volume, length, width, and height of customer's $i$ cargo. We assume that the length is always greater than or equal to the width of the cargo.
- For delivery tasks, the weight $w^i_{cargo} < 0$ and volume $v^i_{cargo} < 0$, $i = 1, .., N$.
- $T$ is the matrix of non-negative distances (path costs) $t_{ij}$ between customers $i$ and $j$, where $t_{ij} \neq t_{ji}$.
- $m$ is the number of vehicles.
- Vehicle $k$ has capacity $w_k$, volume $v_k$, length $l_k$, width $wi_k$ and height $h_k$ of the trailer.
- $W^k_i$ and $V^k_i$ are the weight and volume load of vehicle $k$ after visiting point $i$ respectively.
- $R_i$ is the route of vehicle $i$ ($i = 1, .., m$).
- $C(R_i)$ is the cost of route $R_i$.
- $[a_i, b_i]$ is the minimum and maximum possible arrival time at the customer's location.
- $\tau^k_i$ is the start time of serving customer $i$ by vehicle $k$.
- $serv^k_i$ is the duration of serving customer $i$ by vehicle $k$.

We define $x^k_{ij}$ as

$$x^k_{i,j} = \begin{cases} 1, & \text{if vehicle } k \text{ travels from } v_i \text{ to } v_j, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

### 2.1 TASKS PROPERTIES

It is important to consider the unique business requirements related to tasks and customers while planning the routes. These requirements can be categorized into two types: flexible (might to be violated, but that lowers solution quality) and strict (must be followed). First, let's outline the strict limitations:

- Customer's Time Window: The customer's time window may include a lunch break, dividing the delivery interval into two parts. Let $l^i_{start}$, $l^i_{end}$ denote the start and end times of the lunch break for customer $i$ respectively. If the client $i$ does not have a lunch break, then $l^i_{start} = b_i$, $l^i_{end} = a_i$.
- Vehicle Requirements: Some customers may choose a particular method of loading or unloading cargo or require an access pass at their address (Nag et al., 1988).
- Entry Restrictions: Not all vehicles may be able to access certain customer addresses due to, for example, an archway in the driveway. Let $arch^i_h$ and $arch^i_w$ be the height and width of the archway at client $i$'s address.
- Prohibition on Transit Cargo: Certain clients may restrict access to their address unless the vehicle only contains their orders. This condition can be represented as $\{transit_i\}$, where $i \in \{1, .., n\}$ shows whether the task belongs to the transit or not.
- Delivery Time to Depot: Pickup orders must be delivered to the depot by a set time. This deadline is denoted as $e_i$ for client $i$. If not specified, $e_i = b_0$ for client $i$.
- Second Address: Some orders have a second address $\{second_i\}, i \in \{1, .., n\}$, it indicates that there is a second address that the vehicle must visit immediately after unloading at the main address (for clarity, if $second_i = 1$, then the main address is indexed $i-1$). This may be related, for example, to the requirement to deliver the shipment to a depot and register documents in the office.

These strict requirements might only be applicable to a limited number of clients, however if ignored the route planning process may necessitate human intervention. This means that some of these operations might need human route design for the whole city, hence prevent the use of the planning system. For example, with 1000 tasks and 30 vehicles, we can plan an average of just 6 cars out of 30 using 5 limits, each with a 1% probability of occurrence. This completely negates the benefit of automatic scheduling.

Flexible limitations:

- Client Categories: Clients can be grouped according to certain criteria, such as those with high penalties for delays, standard clients, and those with extended service times.
- Execution Scenarios: Orders can be completed either within their time windows or past their end times, depending on client category.
- Risk Time: Risk time is a period of time that ensures the vehicle does not arrive late for the order. Risk time is subtracted from the order window's end time to accommodate potential delays. Let $[a_i, b_i]$ be the work interval with risk time.
- Order Priority: Tasks with higher priority earn greater penalties for missing their time windows with risk time. This prioritization considers task order in a route and potential traffic delays.

The major purposes of flexible restrictions are risk reduction and route replanning. The presence of such restrictions allows partial violations to occur during execution, but minimizes routes disruptions. It's important to note that when reorganizing orders within the executing route, the sequence in which orders were loaded into the vehicle must be carefully considered. Some rearrangements may not be feasible because they could potentially obstruct the unloading of other tasks. Therefore, calculation configuration should be designed to allow for easy adjustment of these restrictions for different cities.

### 2.2 TRAFFIC JAMS

When planning routes, it is necessary to take into account the forecast of traffic jams for the next day. For example, deliveries scheduled for a busy Friday evening will experience substantially longer travel times than those scheduled for a quieter Saturday midday. Travel time is an important variable that affects the optimization criteria as well as other route-related aspects. While travel times without jams greatly simplifies the problem and facilitates analysis of the solution, it introduces significant errors due to variations of up to $100\%$ or even $200\%$ in travel times depending on the hour and day of the week.

At the beginning of the planning process, the time interval for each order is not yet assigned. Constantly querying a separate service or database for real-time traffic forecasts would slow down the optimization execution. As a result, a method was selected where the time function $\tilde{t}_{i,j}(x)$ is defined as a piecewise smooth function, with $i, j$ denoting the indices of two tasks, and $x$ representing the start time of the route. Traffic jams forecasts $p_{i,j}$ were obtained for 24 points, one per hour, to construct this function. Between these points, interpolation is used:

$$\tilde{t}_{i,j}(x) = p_{i,j}(\lfloor x \rfloor) + (p_{i,j}(\lceil x \rceil) - p_{i,j}(\lfloor x \rfloor))(x - \lfloor x \rfloor) \tag{2}$$

We can quickly and directly calculate the travel time at any given time and for any pair of locations, following the creation of matrices $[p_{i,j}]$ for each hour. Real routes with both estimated and calculated travel times were tested to confirm the accuracy of this approximation. The Mean Absolute Error (MAE) between actual and calculated travel times using this approximation was less than 3%. For cities prone to heavy traffic jams and/or rapidly changing traffic conditions, it may be necessary to increase the number of data points or adjust the granularity during peak hours (e.g., creating matrices every 2 hours at night, and every 30 minutes in the morning). Adapting well-known heuristics becomes essential when dealing with non-constant travel times, as numerous algorithms rely on this feature to accelerate their execution.

### 2.3 VEHICLE RESTRICTIONS

When creating routes for a variety of vehicles, it's vital to consider more than just their volumes or load capacities. Nowadays, many different vehicle characteristics must be considered during planning. For instance, clients might request side loading, a hydraulic lifts or manipulators. Additionally, constraints at specific addresses, like narrow archways limiting height and width of the vehicle, further complicate route planning. Beyond technical limitations, there are non-technical ones to consider. For example, access to certain locations may require a pass, which must be processed within a minimum notice of 24 hours. In addition to constraints, vehicles are classified based on ownership—whether they are owned or rented. The system has to prioritize assigning orders to owned vehicles over rented ones.

### 2.4 ARRANGEMENT OF TASKS IN A VEHICLE

When considering a problem where all orders are delivered from a depot to various addresses, arranging tasks within the vehicle is straightforward. According to the Last In, First Out (LIFO) principle, the sequence of visiting tasks dictates the order in which they are loaded at the depot onto the truck. However, when pickups from addresses to the depot are included in the model, constraints must be defined to ensure that retrieving a pickup order does not obstruct unloading other orders from the vehicle further in the route. This article does not address arranging cargo within a trailer, which involves considerations like axle pressure, it focuses specifically on the ability to retrieve the order from the vehicle, considering the cargo compartment's characteristics and the orders already loaded. Only a few routes may tackle the three-dimensional packing issue, for example Maarouf et al. (2008), Bortfeldt & Homberger (2013), Ceschia et al. (2013), Bortfeldt & Yi (2020). Often, the volume model and the characteristics of the pickup orders are unknown, therefore calculations rely solely only on the outer dimensions of orders, leading to significant errors and providing results of similar quality to those obtained by considering the total dimensions of all orders in the vehicle.

## 3 FORMULATION

The solution to the problem is the minimum of the optimization function $F$, where

$$F = \sum_{i=1}^{m} C(R_i) \tag{3}$$

with the following constraints:

$$\sum_{k=1}^{m}\sum_{j=0}^{n} x_{ij}^{k} = 1, \forall i \in \{1,..,n\} \tag{4}$$

$$\sum_{j=0}^{n} x_{0j}^{k} = 1, \forall k \in \{1,..,m\} \tag{5}$$

$$\sum_{i=0}^{n} x_{i0}^{k} = 1, \forall k \in \{1,..,m\} \tag{6}$$

$$\sum_{i=0}^{n} x_{il}^{k} - \sum_{j=0}^{n} x_{lj}^{k} = 0, \forall k \in \{1,..,m\}, \forall l \in \{1,..,n\} \tag{7}$$

Equation (4) shows that each client is visited exactly once. Equations (5), (6), and (7) impose constraints on the route, ensuring that each vehicle departs from the depot, returns to the depot, and leaves the customer's location, respectively.

$$a_i \leq \tau_i^k \leq l_{start}^i, \ \ \forall i = \{0,..,n\}, \forall k \in \{1,..,m\} \tag{8}$$

$$l_{end}^i \leq \tau_i^k \leq b_i, \ \ \forall i = \{0,..,n\}, \forall k \in \{1,..,m\} \tag{9}$$

$$\tau_i^k + serv_i^k + \tilde{t}_{i,j} - \tau_j^k \leq M(1 - x_{ij}^k), \ \ \forall i = \{0,..,n\}, \forall j = \{1,..,n\}, \forall k \in \{1,..,m\} \tag{10}$$

$$\tau_i^k + serv_i^k + \tilde{t}_{i,0} - e_i \leq M(1 - x_{i0}^k), \ \ \forall i = \{0,..,n\}, \forall k \in \{1,..,m\} \tag{11}$$

Constrains (8) and (9) enforce that client $i$ is visited within its time window including lunch time, constrains (10) and (11) determine the order of customers and depot in a route. $M$ is a scalar value for linearity.

$$W_j^k \geq W_i^k + w_{cargo}^j - W(1 - x_{ij}^k) \ \ \forall i,j = \{0,..,n\}, \forall k \in \{1,..,m\} \tag{12}$$

$$V_j^k \geq V_i^k + v_{cargo}^j - V(1 - x_{ij}^k) \ \ \forall i,j = \{0,..,n\}, \forall k \in \{1,..,m\} \tag{13}$$

$$0 \leq W_i^k, W_i^k \leq w_k, W_i^k \leq w_k + w_{cargo}^i, \\ \forall i \in \{1,..,n\}, \forall k \in \{1,..,m\} \tag{14}$$

$$0 \leq V_i^k, V_i^k \leq v_k, V_i^k \leq v_k + v_{cargo}^i, \\ \forall i \in \{1,..,n\}, \forall k \in \{1,..,m\} \tag{15}$$

$$l_{cargo}^i \sum_{j=0}^{n} x_{ij}^k \leq l_k, \forall k \in \{1,..,m\}, \forall i \in \{1,..,n\} \tag{16}$$

$$wi_{cargo}^i \sum_{j=0}^{n} x_{ij}^k \leq wi_k, \forall k \in \{1,..,m\}, \forall i \in \{1,..,n\} \tag{17}$$

$$h^i_{cargo} \sum_{j=0}^{n} x^k_{ij} \leq h_k, \forall k \in \{1, .., m\}, \forall i \in \{1, .., n\} \tag{18}$$

$$Kw^j_{cargo} + W^k_i - w_k \leq W(1 - x^k_{ij}) \ \ \forall i = \{0, .., n\}, \\ \forall j = \{N+1, .., n\}, \forall k \in \{1, .., m\} \tag{19}$$

$$Kv^j_{cargo} + V^k_i - v_k \leq V(1 - x^k_{ij}) \ \ \forall i = \{0, .., n\}, \\ \forall j = \{N+1, .., n\}, \forall k \in \{1, .., m\} \tag{20}$$

Load and volume limitations are ensured by equations (12),(13), where $W$ and $V$ are constants (Cordeau, 2006). Equations (14), (15), (16), (17), (18) prevent exceeding vehicle capacity and check load against all parameters. We use inequalities (19) and (20) to determine if order $j$ can be picked up after visiting customer $i$. The constant $K$ in expressions (19) and (20) is a real number and is chosen based on historical statistics on pickups. Furthermore, this constant allows a flexible approach for handling heavy or large orders inside an empty trailer, thus enabling order relocation in trailer during route execution. The positioning of the orders inside the vehicle does not need to remain static the entire duration of the route; for instance, a cargo initially placed across the trailer can be rearranged to accommodate subsequent pickups without hindering unloading. Based on experience, the acceptable value of this constant falls within the range of 3 to 5, and can be tailored to specific city planning details by analyzing historical route data.

$$arch^i_w \geq wi_k \sum_{j=0}^{n} x^k_{ij}, \forall k \in \{1, .., m\}, \forall i \in \{1, .., n\} \tag{21}$$

$$arch^i_h \geq h_k \sum_{j=0}^{n} x^k_{ij}, \forall k \in \{1, .., m\}, \forall i \in \{1, .., n\} \tag{22}$$

We use inequalities (21) and (22) to show, that vehicle $k$ can access client $i$'s address if there is an archway present.

$$W^k_i \leq w^i_{cargo} + T(1 - transit_i), V^k_i \leq v^i_{cargo} + T(1 - transit_i), \forall k \in 1, .., m, \ \ \forall i \in N+1, .., n \tag{23}$$

$$W^k_i \leq T(1 - transit_i), V^k_i \leq T(1 - transit_i), \forall k \in 1, .., m, \ \ \forall i \in 1, .., N \tag{24}$$

If task $i$ is a pickup and transit order, then the vehicle must arrive empty (24), and if task $i$ is a delivery and transit order, then the vehicle must only carry orders belonging to client $i$ (23). Here, $T$ is introduced as a scalar value for linearity.

$$x^k_{ij} - x^k_{i-1i} - (1 - second_i) \leq 0, \forall i \in 1, .., n, \forall k \in 1, .., m, \forall j \in 1, .., n \tag{25}$$

The equation (25) ensures that if client $i$ has a second address, the route must include a visit to that address.

### 3.1 Strict assignment restrictions

In general, let's assume that each task has a set of flags $\{f_{u,i}\}$, where $f_{u,i} = 0$ or $f_{u,i} = 1$, indicating which flags the assigned vehicle must satisfy, and each vehicle has two sets of flags: a required set $\{r_{u,k}\}$, indicating which flags the assigned task must satisfy, and an optional set $\{o_{u,k}\}$, indicating which flags the assigned task can satisfy, which are also binary. We will assume that all vehicle and task constraints are combined and indexed as $u \in \{1, .., s\}$ ($f_u$ corresponds to $r_{u,k}$ if $u \leq |\{r_{u,k}\}|$ or $o_{u,k}$ if $u > |\{r_{u,k}\}|$). These flags represent vehicle specifications required by the client or possessed by the vehicle. To assign an order to a vehicle, the following conditions must be satisfied:

$$f_{u,i} \sum_{j=0}^{n} x^k_{i,j} \leq r_{u,k} + o_{u,k}, \forall u \in \{1, .., s\}, \forall i \in \{1, .., n\}, \forall k \in \{1, .., m\} \tag{26}$$

$$r_{u,k} \sum_{j=0}^{n} x_{i,j}^{k} \leq f_{u,i}, \forall u \in \{1,..,s\}, \forall i \in \{1,..,n\}, \forall k \in \{1,..,m\} \tag{27}$$

During the data preparation stage of the calculation, all of the previously mentioned vehicle requirements or restrictions can be reduced to these constraints. The second constraint is necessary for implementing business cases such as, for example, delivering tasks using a manipulator only to customers who have requested it. The search space for solutions can be significantly reduced by using these constraints in a unified form. However, depending on the city, they only apply to 5 $\%$ to 15 $\%$ of the tasks.

### 3.2 Objective function and initial routes construction

In traditional models, the primary metrics of performance that needs to be minimized is the number of routes and their total duration. Additionally, it is important to mitigate the risk of route disruptions and acknowledge errors accumulated during route planning. There are two main reasons for this:

- The travel times between orders in a route are not independent. Any incident, such as an accident, affects not just one route edge but also adjacent route edges, potentially accumulating errors.
- Order execution times can exhibit significant outliers towards increased work time. Compensating for such outliers within the same route can be challenging. For example, in a sequence of 50 orders, 49 might each last two minutes, while one order lasts 102 minutes. This results in an average duration of 4 minutes per event. However, if there is a downtime of 100 minutes, this time cannot be compensated within the same sequence. This requires planning with increased flexibility towards the end of the route.

Due to these factors, constructing clustered routes with short travel times between orders is an effective strategy (see B, B.1). This approach ensures that routes can continue smoothly even if disruptions occur on specific route edges. Consequently, traditional algorithms like the Clarke-Wright approach (Clarke & Wright, 1964) may not be practical. It becomes exceedingly challenging to reassign orders to different vehicles in cases of accidents or delays along a route. Each order must be evaluated for inclusion in a route not only based on proximity but also on the corresponding time slot. The nearest neighbor algorithm (Rosenkrantz et al., 1977) is not suitable for this scenario, as it inserts orders closest to the last added order. In the developed system in general the order closest to all existing orders in the route is inserted. The algorithm constructs the route starting with the farthest order, this policy however can be adjusted based on the specific requirements of small or large vehicles involved. Orders can be sorted based on weight or volume, so that small vehicles only carry light orders and large vehicles only carry heavy orders. The sorting of orders within a cluster for inclusion in the route can be chosen during the initial phase of route creation.

Depending on the type of heuristic used (whether optimizing a single route or multiple routes), different aspects of the optimization function are minimized, specifically:

- Travel time, idle time and task execution time on a route (WTS).
- Time spent carrying heavy cargo (HT).
- Return to an address in the route (R).
- Average driving time between route points (AvgT).
- Work time of the route (WT).
- Number of clusters in the route (NC).

For heuristics optimizing a single route, the objective function is:

$$F_{single} = \gamma_1 WTS + \gamma_2 HT + \gamma_3 R. \tag{28}$$

The objective function for the heuristics, working with multiple routes, is the following:

$$F_{mult} = \alpha_1 WT + \alpha_2 HT + \alpha_3 R + \alpha_4 AvgT + \alpha_5 NC. \tag{29}$$

The coefficients $\gamma_i$ and $\alpha_j$ are specific to each city and are calculated statistically.

### 3.3 HEURISTICS

Due to the strict constraint on route calculation duration, parameters for early termination and a blacklist for potential insertions are introduced for each heuristic. The following heuristics are utilized in the developed system:

- 2-opt: This heuristic reverses a part of the route, replacing the arcs $\{i, i+1\}, \{j, j+1\}$ with the arcs $\{i, j\}, \{i+1, j+1\}$ (Croes, 1958).
- Or-opt: This heuristic moves a part of the route (one or multiple arcs) to another location, for example, the sequence of nodes $\{i, j\}$ between nodes $i-1$ and $j+1$ (Or, 1976).
- I2: This heuristic redistributes some or all tasks from one route to other routes (Solomon, 1987).
- CrossExchange: It exchanges subsets of tasks between two routes (Taillard et al., 1997).
- Squeeze: Originally proposed in Nagata & Bräysy (2009), this method is integrated into all heuristics used in the system. It employs recovery heuristics within a route to restore its validity if order insertion or permutation compromises the route's integrity.
- InsertEject: The basic idea is to destroy a route by distributing all its tasks to other routes (Nagata & Bräysy, 2009).

## 4 CONCLUSION

The developed model has been successfully operational for over seven years. Due to its numerous parameters, the system can be readily customized to accommodate a variety of business cases. By assigning drivers to specific clusters, the system mitigates the likelihood of drivers deviating from algorithmically generated routes. In case of an unforeseeable event, clustering aids logistic managers in visually distributing order groups among neighboring clusters, thereby reducing their workload and minimizing the number of order failures or delays across all routes. The route generation process of the system relies on reproducible algorithms, which guarantees transparency and justification for each route created. Grouping tasks at the same address reduces calculation time, enhances route resilience against visit order manipulation, and improves overall customer service (see Appendix B, B.1).

The table 3 in Appendix presents the algorithm's performance on the Gehring & Homberger tests, indicating deviations from the best-known solutions. The calculation time was limited to two minutes (on one CPU Intel i7-6900k). We unite all datasets to C1, C2, R1, R2 and RC1, RC2 datasets and the average deviations are presented in Table 3. Despite clustering, the algorithm achieved an average deviation of $0.61\%$ in the number of routes and $3.98\%$ in their duration within such a short time.

## REFERENCES

Mikhail Batsyn, Ekaterina Batsyna, I. Bychkov, and Panos Pardalos. Vehicle assignment in site-dependent vehicle routing problems with split deliveries. *Operational Research*, 21, 03 2021. doi: 10.1007/s12351-019-00471-7.

Piotr Beling, Piotr Cybula, Andrzej Jaszkiewicz, Przemysław Pełka, Marek Rogalski, and Piotr Sielski. Deep infeasibility exploration method for vehicle routing problems. In Leslie Pérez Cáceres and Sébastien Verel (eds.), *Evolutionary Computation in Combinatorial Optimization*, pp. 62–78, Cham, 2022. Springer International Publishing. ISBN 978-3-031-04148-8.

Andreas Bortfeldt and Jörg Homberger. Packing first, routing second—a heuristic for the vehicle routing and loading problem. *Computers & Operations Research*, 40(3):873–885, 2013. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2012.09.005. URL `https://www.sciencedirect.com/science/article/pii/S0305054812002018`. Transport Scheduling.

Andreas Bortfeldt and Junmin Yi. The split delivery vehicle routing problem with three-dimensional loading constraints. *European Journal of Operational Research*, 282(2):545–558,

2020. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2019.09.024. URL https://www.sciencedirect.com/science/article/pii/S0377221719307647.

Sara Ceschia, Andrea Schaerf, and Thomas Stützle. Local search techniques for a routing-packing problem. *Computers & Industrial Engineering*, 66(4):1138–1149, 2013. ISSN 0360-8352. doi: https://doi.org/10.1016/j.cie.2013.07.025. URL https://www.sciencedirect.com/science/article/pii/S0360835213002404.

G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4):568–581, 1964. ISSN 0030364X, 15265463. URL http://www.jstor.org/stable/167703.

Jean-François Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586, 2006. ISSN 0030364X, 15265463. URL http://www.jstor.org/stable/25146992.

G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6): 791–812, 1958. ISSN 0030364X, 15265463. URL http://www.jstor.org/stable/167074.

G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6(1):80–91, 1959.

Moshe Dror, Gilbert Laporte, and Pierre Trudeau. Vehicle routing with split deliveries. *Discrete Applied Mathematics*, 50(3):239–254, 1994. ISSN 0166-218X. doi: https://doi.org/10.1016/0166-218X(92)00172-I. URL https://www.sciencedirect.com/science/article/pii/0166218X9200172I.

Lu Duan, Yang Zhan, Haoyuan Hu, Yu Gong, Jiangwen Wei, Xiaodong Zhang, and Yinghui Xu. Efficiently solving the practical vehicle routing problem: A novel joint learning approach. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3054–3063, 2020.

Wissam F. Maarouf, Aziz M. Barbar, and Michel J. Owayjan. A new heuristic algorithm for the 3d bin packing problem. In Khaled Elleithy (ed.), *Innovations and Advanced Techniques in Systems, Computing Sciences and Software Engineering*, pp. 342–345, Dordrecht, 2008. Springer Netherlands. ISBN 978-1-4020-8735-6.

Barin N. Nag, Bruce L. Golden, and Arjang A. Assad. Vehicle routing with site dependencies. vehicle routing: Methods and studies. studies in management science and systems - volume 16. 1988. URL https://api.semanticscholar.org/CorpusID:107384641.

Yuichi Nagata and Olli Bräysy. A powerful route minimization heuristic for the vehicle routing problem with time windows. *Oper. Res. Lett.*, 37:333–338, 2009.

Ilhan Or. Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking, 1976. [3 Mikrofiches].

Sophie Parragh, Karl Doerner, and Richard Hartl. A survey on pickup and delivery problems: Part i: Transportation between customers and depot. *Journal für Betriebswirtschaft*, 58:21–51, 04 2008a. doi: 10.1007/s11301-008-0033-7.

Sophie Parragh, Karl Doerner, and Richard Hartl. A survey on pickup and delivery problems: Part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58:81–117, 06 2008b. doi: 10.1007/s11301-008-0036-4.

Daniel J. Rosenkrantz, Richard Edwin Stearns, and Philip M. Lewis II. An analysis of several heuristics for the traveling salesman problem. *SIAM J. Comput.*, 6(3):563–581, 1977. doi: 10.1137/0206041. URL https://doi.org/10.1137/0206041.

Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987. ISSN 0030364X, 15265463. URL http://www.jstor.org/stable/170697.

Eric Taillard, Philippe Badeau, Michel Gendreau, François Guertin, and Jean-Yves Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31:170–186, 05 1997. doi: 10.1287/trsc.31.2.170.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research*, 40(1):475–489, 2013. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2012.07.018. URL `https://www.sciencedirect.com/science/article/pii/S0305054812001645`.

Thibaut Vidal, Maria Battarra, Anand Subramanian, and Günes Erdogan. Hybrid metaheuristics for the clustered vehicle routing problem. *Computers & Operations Research*, 58, 04 2014. doi: 10.1016/j.cor.2014.10.019.

## A ALGORITHM

The algorithm's flow is depicted in Figure 1. Upon initialization, the following information is available: all tasks with their properties, such as coordinates, categories, service times, vehicle requirement, cluster number; all vehicles with their properties: weight and dimensions characteristics, types of loading, access passes and cluster numbers; traffic jam duration matrices and calculation settings.

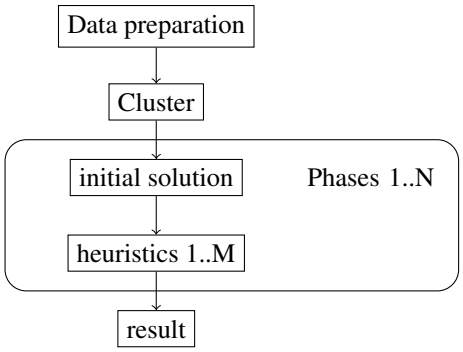


Figure 1: Algorithm

## B DATA PREPARATION AND CLUSTERS

A key component of a real-world route planning problem is data processing. It allows to increase route stability, include business restrictions efficiently, and minimize manual manipulations in route planning. Since the route planning system was developed for practical application, time constraints are imposed on route calculations, namely 5 minutes for 1000 orders and 30 vehicles after calculation starts. One obvious way to reduce problem dimension is to introduce clusters. Clustering facilitates more effective distribution of unallocated orders within a route in the event of delays, client absences, or accidents. The article Vidal et al. (2014) deals with clusters, that must be visited once, and the assignment problem is discussed in Batsyn et al. (2021). In our model clusters are built in two phases: initially, clusters are predefined to segment the city geographically before route planning begins, based on drivers' experience across the city, regular clients, and traffic situation. Subsequently, clusters are formed after the start of planning, upon receiving complete information on orders and vehicles to allocate unique orders to specific vehicles.

Due to the fact that multiple customers may be located at a single address and a single customer may place multiple distinct orders, it is not possible to solely rely on coordinates. Additionally, concentrating multiple vehicle arrivals at the same customer address, despite potentially reducing overall time, can lower customer service quality by subjecting them to multiple vehicle interactions (Dror et al. (1994)). Therefore, the approach considers consolidating clients at one address while accounting for vehicle type restrictions for each client. Orders may be consolidated according to

one of the two rules: either until all vehicles can carry them, or until one or more vehicles can carry them.

Orders at the same address are consolidated using the following rules:

1. Using permutations, we choose an ordered subset of tasks that can be merged with the longest possible time window. This time window has been created to allow for the processing of chosen orders in a sequential manner, each within its own time window and without any idle time in between.
2. Delivery and pickup orders can be merged into a single order of the combined type. Instead of using the total sum of the weight and volume for the address, distinct values for linehauls and backhauls are used to evaluate vehicle capacity limits for such orders.
3. Define other features of united order as:
   - Total weight, volume for delivery and pickup orders separately.
   - Binary flag sums.
   - Maximum dimensions (length, width, and height).
   - Combined client category.
   - Total work time.
4. Verify vehicle availability for executing the consolidated order. If successful, merge the tasks; otherwise, divide the tasks into smaller groups at the address.

For orders at one address belonging to the same client some rules are different:

1. We could combine the client work window with the earliest work window, the latest work window, or the intersection of work windows as determined by calculation settings.
2. The combined work time is typically less than the sum of individual order work times, according to business requirements.
3. If there is no vehicle that can carry all tasks for a client at one address, these orders are excluded from calculations, producing an error message.

Client aggregation precedes common aggregation as a separate step. This approach ensures efficient order consolidation and allocates them to available vehicles within separate geographic clusters.

### B.1 Clusterization Example

Let's consider a scenario with a depot (0) and customers (1), (2), and (3) located on the same road, where customers (1) and (3) share the same location (Fig.2). Assume travel times from point (0) to (1,3), from (1,3) to (2), from (2) to (1,3), and from (1,3) to (0) are 1 hour each, and the travel time from 1 to 3 is 0 minutes (Table 1). The work time at each address is 1 hour with the same work window for all $[0, 7]$. In this case, two routes, (0→1→3→2→0) and (0→1→2→3→0), have the same cost. However, the first route is preferred for several reasons. Firstly, if customer 1 is absent, the driver can complete the task for customer 3 first and then proceed to customer 1 without changing the sequence of the rest part of the route. This manipulation of orders without changing the overall sequence of routes minimizes the risk of trip failure. Secondly, the driver parks once and delivers two orders. Considering additional parking time is not sufficient when the travel time in the model depends on traffic jams. Table 2 illustrates the travel time in hour 3. The costs of the two routes can thus be calculated, the route (0→1→2→3→0) is preferable with a cost of 7 hours.

Table 1: The travel time in hours without traffic jams

| | **0** | **1** | **2** | **3** |
|---|---|---|---|---|
| **0** | 0 | 1 | 2 | 1 |
| **1** | 1 | 0 | 1 | 0 |
| **2** | 2 | 1 | 0 | 1 |
| **3** | 1 | 0 | 1 | 1 |

Table 2: The travel time in hour 3

| | **0** | **1** | **2** | **3** |
|---|---|---|---|---|
| **0** | 0 | 2 | 4 | 2 |
| **1** | 2 | 0 | 2 | 0 |
| **2** | 4 | 2 | 0 | 2 |
| **3** | 2 | 0 | 2 | 2 |

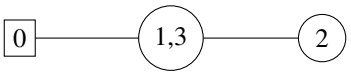


Figure 2: Route

## C PHASES

The system includes sequential computation phases that allow flexible adjustments tailored to the specific characteristics of the city under consideration. In these phases, routes are created, optimized, divided, destroyed and merged. This approach ensures that routes are calculated consecutively and with respect with the load capacities of different vehicle types.

For instance, if creating routes for small vehicles before considering others is a priority, the computational phases are structured as follows:

```
{
    "phaseType": "heuristic",
    "name": "small_vehicels",
    "sourceTags": "bad",
    "resultTag": "good",
    "chooseVehiclesPolicy": {
        "vehicleKind": "own",
        "minWeight": 0,
        "maxWeight": 1500
    },
    "canMergeOldAndNewRoutesBySameTag": true,
    "useUnservedClaims": true,
    "heuristics": [
        "Initial_for_small_vehicle",
        "I2",
        "2Opt"
    ]
},
{
    "phaseType": "heuristic",
    "name": "other",
    "sourceTags": "bad",
    "resultTag": "good",
    "chooseVehiclesPolicy": {
        "vehicleKind": "own",
        "minWeight": 1500
    },
    "canMergeOldAndNewRoutesBySameTag": true,
    "useUnservedClaims": true,
    "heuristics": [
        "Initial",
        "I2",
        "2Opt"
    ]
}
```

The ”small_vehicles” phase generates routes for owned vehicles with capacities ranging from 0 to 1500 kg. Following this, the ”other” phase creates routes for the rest of the owned vehicles, excluding those tagged as ”good”. The final set of routes is merged based on identical result tags.

Additionally, the system supports various phases that merge, destroy, or divide routes based on selected criteria. For example, routes can be categorized by vehicle ownership status—whether owned or hired—before being merged or divided as needed.

# D RESULTS

Table 3: Gehring & Homberger comparison

| Dataset | vehicles (average 3 runs all instance) | duration (average 3 runs all instance) | vehicles diff from WR | duration diff from WR |
|---|---|---|---|---|
| **C1_10_*** | 100 | 42502,59657 | 1,0% | 2,0% |
| **C2_10_*** | 91 | 17035,03417 | 1,7% | 2,4% |
| **R1_10_*** | 91,9 | 49307,02281 | 0,0% | 4,8% |
| **R2_10_*** | 19 | 30131,50813 | 0,0% | 4,1% |
| **RC1_10_*** | 90 | 46184,82728 | 0,0% | 4,9% |
| **RC2_10_*** | 18,4 | 25329,29559 | 1,0% | 5,7% |