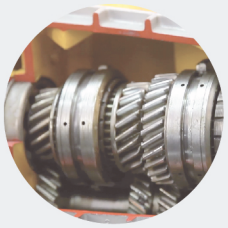
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA E GESTÃO INDUSTRIAL






Using districting and a data driven TSP to improve last mile delivery

BEATRIZ BARBOSA DOS SANTOS
Setembro de 2023

POLITÉCNICO
DO PORTO

Instituto Superior de
Engenharia do Porto

Using districting and a data driven TSP to improve last mile delivery

Instituto Superior de Engenharia do Porto
Master in Industrial Engineering and Management

Beatriz Santos
ID number 118718

Advisor
António Galrão Ramos

Academic Year 2022/2023

Thesis not yet defended

Using districting and a data driven TSP to improve last mile delivery
Instituto Superior de Engenharia do Porto

This thesis has been typeset by LaTeX and the isepthesis class.

Author's email: 1180718@isep.ipp.pt

Acknowledgments

The development of this dissertation (and my academic journey as a whole) was only possible due to the contribution of some people.

First and foremost, I would like to thank my advisor, António Ramos, for being available and for all the guidance.

I thank INESC TEC, for the opportunity of developing this dissertation with a research grant and for the all resources provided.

Next, I must express deep gratitude to my parents and my sister, for making all this possible by being my backbone, for every dinner at home after a stressful day.

To my boyfriend, Rafael, for always reminding me that everything will be fine and that I will succeed. Thank you for always making me laugh.

And, to my friends, that made the last five years a lot easier and more fun.

Abstract

This dissertation considers how a parcel delivery company can improve last mile delivery services' performance using historical data. To tackle this challenge, we start by proposing a framework for data cleaning, in order to produce reliable data for vehicle routing problems.

Data on the historical geographical location of clients is used to model a hierarchical districting problem, mid-level districts (named micro districts) are limited to an eight hour shift, representing a daily route. Using the districting solution as a procedure for package to driver/vehicle assignment, it is possible to achieve a 14% decrease in the number of vehicles needed, while keeping daily routes more balanced in terms of working times.

Using a transition probabilities based TSP to sequence nano zones (the lower-level districts), the preferences of drivers are used as a cost function. The transition probabilities based TSP produces solutions with a total distance 12% higher, comparing with a distance based TSP. Moreover, sequencing the nano zones using the maximum likelihood routing enables the incorporation of the driver's tacit knowledge.

Contents

List of Abbreviations

ConVRP	Consistent Vehicle Routing Problem
CVRP	Capacitated Vehicle Routing Problem
DVRP	Dynamic Vehicle Routing Problem
EVRP	Electric Vehicle Routing Problem
HFVRP	Heterogeneous Fleet Vehicle Routing Problem
HVRP	Hybrid Vehicle Routing Problem
LMD	Last Mile Delivery
MDVRP	Multi Depot Vehicle Routing Problem
MTVRP	Multi Trip Vehicle Routing Problem
OVRP	Open Vehicle Routing Problem
PVRP	Periodic Vehicle Routing Problem
SDVRP	Split Delivery Vehicle Routing Problem
SVRP	Stochastic Vehicle Routing Problem
TSP	Travelling Salesman Problem
TTVRP	Truck and Trailer Vehicle Routing Problem
VRP	General Purpose Operating System
VRPPB	Vehicle Routing Problem with backhauls
VRPTW	Vehicle Routing Problem with Time Windows

List of Tables

List of Figures

Chapter 1

Introduction

This chapter presents the research problem that will be the focus of this dissertation, by contextualizing it and exploring its relevance. Then, the research question is established, while specifying the overall objective and the specific objectives of the project. The methodological option used is also presented. And, finally, the structure of the dissertation is detailed.

1.1 Research problem, framework, and relevance

E-commerce is the activity of buying and selling products through the Internet. This type of business is expected to account for 24% of total retail sales by 2026 (Cramer-Flood, 2022).

E-commerce activities rely on parcel delivery companies for product delivery. As e-commerce activity rises, the service level required by customers regarding product delivery also increases. Thus, parcel delivery companies are more and more seeking to improve processes. Last Mile Delivery (LMD) is the the last phase in parcel delivery, during which a driver transports a package (or mail) from the depot to the end customer. Research in the field of delivery route optimization is considerable, and it is mainly focused on minimizing the distance or time of the proposed route.

However, empirical knowledge shows that the routes actually traveled by drivers deviate from the routes suggested by the optimization algorithms. This discrepancy can be explained by the multiple preferences a driver may have: whether it is finding a suitable parking space, avoiding traffic lights, or roads with many curves. These kinds of peculiarities are difficult to consider in traditional optimization strategies, and therefore the tacit knowledge of an experienced driver is discarded. On the other hand, for the driver to be experienced and learn the service zone, the LMD has to be consistent, since the efficiency of a route, especially in an urban environment, is dependent on how familiar the driver is with the service area (Holland et al., 2017).

In order to achieve consistency, the service territory should be divided into zones that must be repeatedly assigned to the same driver (Vidal et al., 2020).

Thus, this paper seeks to fill a gap in the existing literature by combining the concepts of consistency and inference in an approach to the last mile problem. In general, it is intended to infer the natural sequence of drivers between zones, using a set of districts (e.g., zones) and a set of performed routes as an input.

1.2 Research question and objectives

Considering the problem description detailed in section 1.1, the present dissertation aims to answer the following question: how can districting and zone-sequencing be used to incorporate the tacit knowledge of a driver.

In that sense, the general objective of the project is the conception and validation of an algorithm of zone sequence. To accomplish the proposed general objective, the following specific objectives must be fulfilled:

1. Analyse the provided data set.
2. Pre-process noisy data.
3. Design districts with three levels of aggregation.
4. Evaluate the districting solution.
5. Design a zone sequence algorithm.

1.3 Methodological options

The methodological options will be described using the framework presented by Caixeta and Fabricio (2018). Regarding, the methodology, the research has a quantitative perspective, since we understand the research question has an objective solution, whose development will be anchored in mensurable data. The research method will be the case study, because the research relates to a specific case and context, the case of a national company of parcel delivery.

1.4 Structure

The structure of this dissertation relates to its purpose. It starts with chapter 1, Introduction, in which the research problem and its relevance is presented, the research question and the objectives are defined, and the methodology and the structure of the dissertation are detailed. In chapter 2, Literature Review, the state

of art in the areas regarding the project is reviewed. Chapter 3 gives an overview of the dataset representing the case study. In chapter 4 the data cleaning process for the dataset is explained. Chapter 5 treats the districting part of the districting. The mathematical model, the recursive algorithm, the parameterization and the result analysis are presented. Chapter 6 focus on the sequencing of nano districts. The mathematical model, the representation of the solution and the result analysis are depicted. Finally, in chapter 7 conclusions considered relevant to the dissertation are drawn.

Chapter 2

Literature Review

Throughout this chapter, a literature review of the elements considered relevant to the development of the research is exposed. In section 2.1 we start by defining and contextualizing the General Purpose Operating System (VRP), since the Last Mile Delivery problem can be represented by a VRP. 2.2 is dedicated to the combination of the districting problem with the VRP, and it is subdivided into two subsections, 2.2.1 discusses hierarchical districting, an approach of special interest for this dissertation, and 2.2.2 discusses two-phase methods for solving the Clustered VRP. Section 2.3 delves deeper into the use of machine learning and data mining in solving VRP, with a particular focus on approaches where one seeks to build routes by incorporating the experienced knowledge of the driver.

2.1 Vehicle Routing Problem

From a formal point of view the last mile delivery problem can be looked at as the Vechicle Routing Problem, or VRP. The VRP was introduced for the first time in 1959 by Dantzig and Ramser. It was later generalized by Clarke and Wright to a standard problem in the field of logistics and transportation: how should one serve a set of customers, with a certain location, from a common depot (Braekers et al., 2016).

Essentially, VRP is a combinatorial optimization problem, where one seeks to determine a set of routes that serve a set of customers, while minimizing the total cost. The problem is solved under three assumptions: (i) all routes visit the depot; (ii) each customer is served once and only once, and (iii) the vehicle capacity cannot be exceeded (Demir et al., 2022).

The classic vehicle routing problem can be defined mathematically as it follows. Denote graph $G = (V, E)$. G includes a set of nodes $V = \{0, 1, \ldots, N\}$, where 0 represents the the depot, and a set of arcs $E = \{i,j : j,i \in V, i \neq j\}$. Every node,

or customer, $i \in V \setminus \{0\}$ is associated with a certain demand d_i. Each arc (i, j) is characterized by specific cost, c_{ij}, in case i isn't connected to j, we establish c_{ij} as infinite. Define $J = \{J_0, J_1, \ldots, J_m\}$ as the set of vehicles available in the depot, such that $|J| = K$. A possible solution for the VRP is a set of oriented circuits $T = \{T_0, \ldots, T_m\}$ in graph G (Demir et al., 2022).

Regarding the mathematical formulation of the VRP, we can adopt the one presented in Toth and Vigo (2002), defining x_{ij} as binary variable that takes the value 1 if, and only if, the vehicle visits customer j immediately after visiting i and C as the maximum capacity of every vehicle, we have,

$$\min \quad \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \tag{2.1}$$

$$\text{s.t} \quad \sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V \setminus \{0\} \tag{2.2}$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V \setminus \{0\} \tag{2.3}$$

$$\sum_{i \in V \setminus \{0\}} x_{i0} = K \tag{2.4}$$

$$\sum_{j \in V \setminus \{0\}} x_{0j} = K \tag{2.5}$$

$$\sum_{i \notin V} \sum_{j \in V} x_{ij} \geq r(S) \qquad \forall S \subseteq V \setminus \{0\}, S \neq \emptyset \tag{2.6}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in V \tag{2.7}$$

Constraints (2.2-2.3) state that only one arc enters and leaves a vertex (e.g., a customer is visited once and once only). (2.4-2.5) ensure that the number of vehicles leaving and entering the depot is the same. $r(S)$ is the minimum number of vehicles needed to serve all customers $i \in V$, and constraint (2.6) imposes connection in routes and that each route must not exceed the capacity of the vehicle. Constraint (2.7) ensures that the decision variable is binary.

The vehicle routing problem is a NP-hard problem that generalizes the travelling salesman problem Travelling Salesman Problem (TSP), in which one attempts to obtain the minimum cost of an Hamiltonian circuit, by establishing $K = 1$. In that sense, the TSP can be considered an instance, or a variation, of the VRP, where there is only one available vehicle at the depot.

2.1.1 VRP variants

The specificities found in the daily operations of delivery companies motivate the emergence of variants to the VRP problem. (Goel and Gruhn, 2008; Konstantakopoulos et al., 2022). Table 2.1 provides an overview of the different variants of VRP in terms of their formulation, as well as examples of their appearance in the literature.

Table 2.1. Summary of different variants of the VRP and examples.

Variant	Summary	Example
Capacitated Vehicle Routing Problem (CVRP)	Introduces a capacity constraint (e.g., the load of a vehicle J_i cannot be greater than a maximum value Q), while considering that the whole fleet has the same Q value.	Xiao et al., 2012
Heterogeneous Fleet Vehicle Routing Problem (HFVRP)	Introduces a capacity constraint (e.g., the load of a vehicle J_i cannot be greater than a maximum value Q_i), while considering that the fleet has different Q_i values.	Li et al., 2007
Open Vehicle Routing Problem (OVRP)	Considers that the return to the depot, after completing the route, is not mandatory.	Zachariadis and Kiranoudis, 2010
2D-VRP	Introduces two-dimensional loading constraints, ensuring it is possible to load the packages into the vehicles.	Zachariadis et al., 2016
3D-VRP	Introduces three-dimensional loading constraints, ensuring it is possible to load the packages into the vehicles.	Zachariadis et al., 2016
Truck and Trailer Vehicle Routing Problem (TTVRP)	Addresses the case where certain customers can be be served by a vehicle with a trailer, while others cannot.	Chao, 2002
Multi Trip Vehicle Routing Problem (MTVRP)	Considers the use of vehicles with reduced capacity, where multiple depot - customer - depot trips are required in the work shift.	Cattaruzza et al., 2014

To be continued

Table 2.1 (continued)

Variant	Summary	Example
Vehicle Routing Problem with Time Windows (VRPTW)	Introduces time constraints on the service, e.g., the case in which a customer i must be served in the time interval $[e_j, e_k]$. There are two different constraint typologies: (i) hard, in case it is required to wait if the vehicle arrives before e_j, and the vehicle isn't allowed to arrive after e_k ; (ii) soft, if the failure to comply with time windows is allowed but implies a cost.	Kim et al., 2006
Consistent Vehicle Routing Problem (ConVRP)	Aims to construct consistent routes, through the minimization of the number of different drivers, a client is served by.	Xu and Cai, 2018
Vehicle Routing Problem with backhauls (VRPPB)	Considers the case of package returns from customers.	Toth and Vigo, 1997
VRP with simultaneous pickups and deliveries	Consider the case where a route included deliveries and pickups.	Subramanian et al., 2010
Green VRP	The objective function combines the minimization of transportation costs and CO2 emissions.	da Costa et al., 2018
Electric Vehicle Routing Problem (EVRP)	Studies the case where the vehicles used are electrical.	Montoya et al., 2017
Hybrid Vehicle Routing Problem (HVRP)	Studies the case where the vehicles used are hybrid.	Yu et al., 2017
Time-Dependent VRP	The time of departure is the independent variable of a function, in order to increase the accuracy of delivery time	Donati et al., 2008

To be continued

Table 2.1 (continued)

Variant	Summary	Example
Dynamic Vehicle Routing Problem (DVRP) and Stochastic Vehicle Routing Problem (SVRP)	Considers that changes can be made while performing the route, due to the occurrence of unpredictable unpredictable events.	Montemanni et al., 2005; Bertsimas, 1992
Multi Depot Vehicle Routing Problem (MDVRP)	When there is more than one depot	Ho et al., 2008
Split Delivery Vehicle Routing Problem (SDVRP))	The restriction that each customer must be visited once and only once is relaxed and the demand can be splited and served by different vehicles.	Archetti et al., 2006
Periodic Vehicle Routing Problem (PVRP)	Construction of routes for a time horizon from days to weeks.	Norouzi et al., 2015

Needless to say, when working with real life data, the problems faced by a distribution company are often a combination of those presented in table 2.1. For instance, the constraints related to the vehicle's capacity and restrictions regarding time windows may coexist, and then we are dealing with both a VRPTW and a CVRP (Goel and Gruhn, 2008; Konstantakopoulos et al., 2022).

2.1.2 Approaches to solve the VRP

Since the VRP is a problem that addresses increasingly complex constraints: the increasing number of customers to serve and the growth of the vehicles fleet; solving it requires advanced algorithms. On the other hand, being an NP-hard complexity problem, the use of exact methods is frequently inadequate (Tan and Yeh, 2021).

Approaches to solve the VRP typically range from exact methods to heuristics and meta-heuristics; the last two being the ones used the most in problems with a higher number of constraints. While exact methods return optimal solutions, heuristics (and meta-heuristics) produce nearly optimal solutions (Tan and Yeh, 2021).

Using the taxonomy presented in Konstantakopoulos et al., 2022, heuristic methods can be subdivided into three categories: constructive algorithms, improvement algorithms and two-phase algorithms; whereas meta-heuristics can be categorized as population search algorithms and local search algorithms.

The constructive methods are the simplest, but the least efficient, the most well known example is Clarke and Wright's algorithm, also known as the method of savings (Clarke and Wright, 1964). In this genre of methods the solution is built gradually, by applying a greedy logic, e.g., at each iteration the locally optimal choice is made, in an attempt of, in the end, reaching the global optimum.

2.2 Distring and VRP

In the districting problem, or zoning / territorial design problem, one aims to divide a geographical area into a set of districts, with a given objective while complying with certain certain constraints. Usually, the constraints include three criteria: balance, contiguity and compactness. Balance guarantees fairness between districts relative to given parameter, for instance, producing districts with similar expected sales, with the same workload or the same number of inhabitants. Contiguity is a topographical property, in a contiguous district, it is possible to travel between every point within it, without having to cross another district. On the other hand, a compact district has a roughly round shape, has no holes and no distortions (Kalcsics and Ríos-Mercado, 2019).

Districting applications are diverse: it is used to design electoral systems (Ricca et al., 2013), in distribution services (Jarrah and Bard, 2012), in the assignment of students to schools Ferland and Guenette 1990, when planning of commercial territories (Salazar-Aguilar et al., 2011), in health services management (Darmian et al., 2021a), when planning emergency responses (Mayorga et al., 2013) and in solid waste collection systems (Darmian et al., 2021b). Broadly, in all of its applications, the objective of districting problem is to enable operations to be carried out efficiently within the boundaries of a district, assuming that it is easier to organize operations within a district, rather then in the territory as a whole.

When applied in the context of distribution or delivery services, districting involves assigning customers to a contiguous, compact and balanced district. In this context, the balancing the districts must, at least, ensure that the workload does not exceed the driver's working hours limit.

The advantages of partitioning the territory into districts, in the context of delivery companies, are rooted in different sources: on the one hand, grouping clients arises as a consequence of resources being limited, and so, serving a demand territory

as a whole (e.g., one vehicle and one driver for the whole region) is impracticable either by the volume of the demand or its geographical distribution (Expósito-Izquierdo et al., 2016); on the other hand, the specific topography of a region can motivate its partition, for instance, when two areas are separated by a river, it is useful for them to be considered different districts, since crossing a river can be time consuming; another aspect in favor of districting is allowing working the VRP at the cluster level, reducing the problem size and the computational effort for solving it; on another note, districting potentiates regional consistency, e.g., having drivers consistently assigned to the same area of service, which is shown to be important to route efficiency, in urban contexts (Holland et al., 2017; Vidal et al., 2020).

2.2.1 Hierarchical districting

A specific category of districting problems is the hierarchical or multilevel districting, in this type of problem, districts have different hierarchies, there are *macro* districts, containing *micro* districts, those made of *nano* districts (note that the use of three levels is merely illustrative).

In Darmian et al., 2021a the hierarchical districting approach is applied in the partition of territory for healthcare services. The healthcare services have a three levels structure: Level I includes specialty and sub specialty medical or paramedical services; level II concerns physicians and nurses in clinics and hospitals; and level III consists of emergency and treatment enters. A coarsening procedure is proposed, in order to create graphs of the different hierarchical levels. Essentially, adjacent nodes are cast in order to create graphs of lower dimension in the first and second level. For the third level of districting, the upper level solution is mapped in the original graph.

The use of multilevel districting for solving VRP has little to no presence in literature. However, it is intuitively suggested in the data structure of the challenged proposed by Amazon (ama, 2021), that will be further analysed in section 2.3. Briefly, the challenge involve developing solutions for the last mile delivery problem, using the tacit knowledge of drives and, in fact, from the submitted solutions, the ones that worked/considered the hierarchical inherent to the structure of the geographical service area outperformed the ones that didn't.

Using multilevel districting in the context of vehicle routing can be specially relevant when demand is subject to high variability. Thus, in high demand periods, a driver can be assigned to one, or a few, last level districts (e.g., the smaller ones) and, on the other hand, during low demand periods a driver can be responsible for serving a whole district of higher level (e.g., the ones with bigger area).

The closest to this approach is the proposal of Özdamar and Demir, 2012 for

vehicle routing in large-scale post-disaster distribution and evacuation activities. It proposes a multilevel clustering algorithm: the nodes are aggregated in K demand clusters, using K-means algorithm. The level 0 problem aggregates the original graph in K clusters. The solution of the level 0 is a set of routes (and quantity per route) that visit depots, hospitals and demand clusters. In the next level, each cluster is disaggregate, forming its own sub graph, and the vehicle routing problem is solved intra-cluster. This aggregation - disaggregation approach is repeated until the cluster size is suitable.

Given the lack of research on this topic, in this dissertation one aims to explore the use of a hierarchical districting structure to solve the VRP.

2.2.2 Clustered VRP

After partitioning the territory, the VRP can be interpreted as a Clustered VRP (CluVRP). The CluVRP is a generalization of the CVRP, where the customers are aggregated in clusters, making it compulsory to visit all the customers within a cluster, before moving to the next cluster. This variant is relevant when aiming the construct compact routes, where neighbors are visited successively. An example of this is routing for the collection of solid waste, or even, when a client prefers being served in grouped (with some sort of consistency), for instance, in elderly transportation, where the customer favors being driven with other local community members (Battarra et al., 2014).

To this dissertation, a specific approach to solve the CluVRP is relevant: the two level methods. In this type of approach, the VRP is addressed at the cluster and the customer level. Intuitively, one can identify convergence points between this approach and hierarchical districting, in the two level methods, one considers that the problem has two hierarchical levels, the cluster (a group of customers) and the customer.

Expósito-Izquierdo et al., 2016 presents an hierarchical approach to solve the CluVRP: it starts by constructing the inter-cluster routes (High Level Routing Problem, HRP) then, it constructs the intra-cluster routes (Low Level Routing Problem, LRP). It considers that the HRP can be approximated to a CVRP, where the customers to visit are clusters (represented by their virtual center) and uses the Record-to-Record algorithm to solve it. The LRP, on the other hand, is formulated as a Shortest Hamiltonian Path Problem (SHPP), solving it using the heuristic algorithm Lin-Kerninghan. The authors find it relevant to consider the result of the High Level Routing problem, when solving the low level. To handle this, the SHPP is solved iteratively, following the inter cluster sequence and the last node visited in the previous cluster and the first node visited in the next cluster are included in the

SHPP problem.

The same concept is applied by Defryn and Sörensen, 2017, when they use the Variable Neighborhood Search (VNS) algorithm. The CluVRP is, initially, solved at the cluster level, using the VNS. The result at the cluster level is used as an input for the VNS at the customer level. In this work, they introduce a new type of Clu VRP, the Clu VRP with weak constraints, where it is mandatory for customers at the same cluster to be served by the same client, however customers from different clusters can be served in any order (e.g., it isn't mandatory to visit all the customers within a cluster before heading to the next one).

Pop et al., 2017 also solves the CluVRP considering two subproblems. The (global) inter-cluster sequence is determined using a genetic algorithm. Then, global the route is transformed into a TSP by introducing an artificial cost M on all inter-cluster edges. The TSP is solved using the Concorde TSP Solver.

One of the questions that arises when using two-level methods is how to determine the distances between clusters, the proximity measures mostly used in literature are: the simple link (smallest arc connecting two clusters), the complete link (largest arc connecting two clusters), the average link (average of the distance of all the arcs connecting two clusters), the centroid (Euclidean distance between the centroids of the clusters) and the Ward connection (the increase in variance resulting from merging two clusters) (Barreto et al., 2007).

2.3 Machine learning and data driven methods as an approach to solve the VRP or the TSP

Due to the problems encountered when solving the VRP using purely analytical approaches (like unrealistic problem modeling, difficulties in parameter estimation and impractical resolution algorithms), the use of hybrid methods, combining machine learning (ML) tools with conventional optimization techniques, has been the subject of emerging interest (Bai et al., 2021).

In the research carried in Bai et al., 2021, two areas of application of ML and data analysis to the VRP problem are distinguished: ML to support VRP modeling and ML to support VRP resolution.

The main purpose of ML used to support VRP modeling is to deal with the uncertainty associated with VRP data, this includes stochastic programming [1],

[1] Stochastic programming assumes represents uncertain data by random variables with known probabilistic distributions.

robust optimization [2], chance constrained programming [3] and forecasting [4] (Bai et al., 2021).

When it comes to ML to support of VRP solving, it can be used to guide decomposition strategies [5], apply learning-assisted perturbation heuristics, learn evaluation functions, reducing computational complexity, and in learning how to construct VRP solutions Bai et al. (2021).

Another type of application of ML/data-driven methods, that is yet to explore, is the use of these methods to infer drivers' decision-making. In this sort of approach, it is understood that experienced drivers are able to plan better delivery routes than conventional optimization tools, so the tacit knowledge of drivers should be incorporated when constructing routes. This type of application is in line with the work this dissertation tries to develop, therefore, deserving special attention.

In 2021, Amazon.com, Inc. proposed a challenge in which it encouraged participants to develop learning techniques to solve the VRP, using historical data on routes actually taken by experienced drivers (ama, 2021).

Table 2 presents a summary of examples identified in the literature where the driver experience/knowledge is incorporated in route construction. Due to the scarcity of of examples, most of the articles referenced were submitted in response to the Amazon Routing Challenge, these cases are identified in the in the column Type.

Orgaz et al., 2015 developed a methodology combining Case Based Reasoning (CBR) with the genetic algorithm. The CBR is used to learn the usual service regions of each driver. The result of the CBR serves as an input for the genetic algorithm, used to solve a TSP, minimizing the travelled distance for each tuple $(driver, day)$. Because the authors work with real life data, they also develop a module for pre processing and correcting noisy data.

Li et al., 2018 apply a data driven algorithm to solve a Split Delivery Routing Problem with 3D Loading Constraints (3D-SDVRP). For the VRP part, a multinomial transition matrix is learned from historical data. Each cell on the matrix depicts the probability of a vehicle going from i to j. For the actual construction of routes, feasible solutions are extracted from the matrix, using sampling.

Chen et al., 2021 use the Case-Based Reasoning approach to solve a VRPTW, using a multi-objective integer programming formulation. The objective functions

[2]Robust optimization assumes that the probabilistic distribution of uncertain data can be unknown, but the values range is knowsn.

[3]In chance constraint programming, constraints are satisfied up to a certain level, measured probabilistically, without being fully met

[4]Forecasting in VRP modeling can be done at two levels: forecast uncertain elements of the problem (such as demand or service time) or fine-tuning hyper-parameters

[5]When the problem is decomposed into sub-problems (the case of Cluster First, Route Second), ML is used in the clustering phase (K-means, unsupervised clustering, among others).

aims to both maximize the number of historical client chains (consecutive sequences of clients in the historical data) and minimize the cost of the solution. The authors intend to store past route experience, and reuse it in future solutions. There are four steps to the methodology: the retrieval, where the algorithm searches for the most similar routes in the case base to the new problem; the reuse, when the customers that don't exist in the new problem are removed and customers that aren't in the case are inserted; the revise, where it is checked if the capacity constraints are complied; and, the retain, when new route is stored in the case base.

Canoy et al., 2022 uses a two phases approach to solve a TSP problem. First, the authors treat the problem at the zone level and then at the client level. Through the frequency of transitions between zones, the driver's preferences are modelled as a Markov chain, and represented by a probability transition matrix. The cost function combines the transition matrix with a distance matrix. To determine the weight of each matrix in the cost function, a structured prediction is used. The zone sequence is obtained by solving the TSP, that later is used for solving the TSP at the client level (violations of the sequence are penalized).

Wu et al., 2022 try to predict the routes performed by drivers, they develop a two phase approach: training and inference. In the training phase, they develop a Prediction by Partial Matching (PPM) model, trained to incorporate driver's preferences for zone sequencing. At the inference phase, they determine the zone sequence, using the PPM model and a Rollout algorithm. Within each zone, the stops (or customers), are sequenced with an LKH TSP Solver. To include the zone sequence in stop sequencing, the authors introduced an artificial stop in each zone, representing the next zone in the zone sequence.

Özarık et al., 2022 work the LMD as a reverse optimization problem. The authors develop a two phase algorithm called Pool and Select. They take advantage of the hierarchical structure of data and define macro, micro and nano zones. Then, they determine the transition frequency between different zones and use them in the pooling phase to generated candidate sequences. They then construct a regression model Lasso that using statistics on the candidate sequence, predicts its evaluation in terms of levenshtein distance. The sequence with higher evaluation is selected.

Cook and Held, 2022, the winners of the Amazon challenge, developed a penalty based local search algorithm. The objective function of the local search is the minimization of the travelled time in a route, with the following penalties: the number of times there is zone crossing (e.g., entering a new zone), violating the soft constraint ensuring that certain pairs of routes are visited one after the other, violating the soft constraint ensuring precedences seen in the historical data.

In van Beek, 2022 a reverse optimization model is used, using historical data as

an input, in order to determine drivers' preferences when sequencing routes. The author considers that the driven routes are solutions to a TSP, with an unknown coefficient in the objective function, representing the preference a driver has for travelling a certain arc.

From the analysed applications, some aspects should be noted:

1. Dividing the last mile delivery in levels (zones and customers, for instance) is beneficial in different aspects, first the driver learns at the zone level, because service zones are repeated daily, in contrast to customers, and, on the other hand, because zones appear several times in the data, it is more efficient for an algorithm to learn preferences in terms of zone transitions;

2. Methods to handle with customers and zones that aren't in the training sample should be developed;

3. In two phases problems, there are different ways to calculate the distance between zones, determining the centroid of each zone and using the haversine or euclidean distance, or considering the distance between two zones, the size of the smallest edge connecting them.

Table 2.2. Examples in literature where data-based methods are used.

Reference	Problem	Methodology	Instance Type	Objective
(Orgaz et al., 2015)	TSP	Case Based Reasoning to identify the usual zones of each driver, followed by a genetic algorithm to construct minimum distance routes	Real life data	Minimize travelled distance
(Xu and Cai, 2018)	3D-SDVRP	Construction of a multinomial transition matrix using historical data. Routes are designed using the transition matrix.	Real life data	Using experienced past knowledge to construct routes

To be continued

Table 2.2 (continued)

Reference	Problem	Methodology	Instance Type	Objective
(Chen et al., 2021)	VRPTW	Case Based Reasoning using a multi objective integer programming formulation.	Real life data	Maximize the number of historical client chains and minimize the cost of the solution
(Ghosh et al., 2021)	TSP	Two phased methodology: starts by determining the inter zone sequence, followed by the intra zone sequence.	Real life data. *Amazon Routing Challenged*	Predict the route a driver will take
(Chu et al., 2021)	VRP	Simulated Annealing in which the objective function coefficient (travelled time going from customer i to j) is predicted using the methodology smart predict-then-optimize.	Generated instances.	Minimize the total travel time.
(Mandi et al., 2021)	CVRP	Uses a neural network to determine the transition probability between arcs, then a CVRP formulation is solved, where the objective function is maximizing the summed transition probability.	Real life data.	Maximize transition probability.

To be continued

Table 2.2 (continued)

Reference	Problem	Methodology	Instance Type	Objective
(Chen et al., 2021)	CVRP	Uses an inverse optimization formulation to derive a cost matrix, using past data.	Real life data and generated instances.	Constructing routes based on past data.
(Canoy et al., 2022)	TSP	Two phase approach: solves a TSP on the zone level, considering drivers preference; then solves a TSP at the client level, intra-zone.	Real life data. *Amazon Routing Challenge*	Predict the actual route a driver carries.
(Wu et al., 2022)	TSP	Two phase approach: trains a Prection by Partial Matching (PPM) to predict zone sequence and then uses a LKH TSP solver to determine the sequence of clients, within a zone.	Real life data. *Amazon Routing Challenge*	Predict the actual route a driver carries.
(Özarık et al., 2022)	TSP	Uses a reverse optimization perspective and solves the problem with a Pool and Select algorithm.	Real life data. *Amazon Routing Challenge*	Predict the actual route a driver carries.

To be continued

Table 2.2 (continued)

Reference	Problem	Methodology	Instance Type	Objective
(Cook and Held, 2022)	TSP	Local search with penalties.	Real life data. *Amazon Routing Challenge*	Predict the actual route a driver carries.
(van Beek, 2022)	TSP	Reverse optimization to determine drivers' preference in travelling a certain arc.	Real life data. *Amazon Routing Challenge*	Predict the actual route a driver carries.

2.4 Final considerations

Having completed this literature review, it is possible to outline and contextualise the work to carried in more detail. A real world VRP problem will be tackled, firstly, a multi level districting problem will be solved. One of the levels of districting will represent the route level, thus, the assignment of a client to a vehicle/driver will be automated, e.g., if it belongs to a given district, it is assigned to a certain vehicle and driver. With this the VRP is converted into a TSP. To solve the TSP, one aims to incorporate the tacit knowledge of drivers. In this regard, inspiration will be taken from the machine learning and/or data driven methods already used in this area and with the same kind of purpose. purpose.

Chapter 3

Dataset description

In this chapter, the data set used will be described. A company provided a set of real life delivery requests, as well as their actual driven routes. In 3.1 the data sources are detailed and a description of the attributes is given. Section 3.2 presents an overview and an initial analysis of the data.

3.1 Data source and attributes description

The company provided two .xlsx files with data regarding performed routes within Porto's district. One of the files concerns the time period from 01/07/2021 to 30/09/2021, the other refers to the period between 01/10/2021 and 31/12/2021. The two files have the same data format and were merged. The data is in tabular format and, briefly, the data set contains information on a route and package level. Table 3.1 presents a description of every attribute on the data set.

For better understanding of the data, a prior definition of concepts is crucial. In what regards the present work, a package is defined as a specific parcel delivered to a client, it must be noted that multiple packages can be delivered to the same client, on the same day. A stop, on the other hand, denotes the moment a driver parks and delivers all the packages assigned to that stop; for instance, if there are multiple packages for delivery in the same building, the driver will only *stop* once, even though the number the number of packages for delivery in that stop is bigger than one. Also, the circuit_code attribute is an internal code that denotes a specific route type, the code is repeated throughout the days, and is related to the geographical area the route serves. Finally, a route is defined by a set of stops, with the same date and circuit_code values, ordered by their delivery_time. Table 3.2 exhibits a snippet of the data set with an example of what would be a route, in that case, defined as $\{A, B, C, D\}$.

Table 3.1. Description of the attributes on the data set.

Attribute	Description
package_id	Unique identifier of each package
date	The date of the delivery in a DDMMYYYY format
address_1	The first line of delivery address
address_2	The second line of the delivery address
county	The county of the delivery
postal_code	The postal code of the delivery
circuit_code	Identifier of a route type
performance	Categorical variable denoting the performance of the route (good, acceptable or unacceptable)
productivity	Categorical variable denoting the productivity of the route (good, acceptable or unacceptable)
latitude	The latitude of the delivery's address
longitude	The longitude of the delivery's address
last_incidence	Categorical variable describing the last type of occurrence on the address (there are 27 different types of incidences in the data set)
service	Categorical variable denoting the delivery time constraint on some packages (A, M or D)
weight	Weight of the package
delivery_time	The time the package was delivered, in the format HHMMSS

3.2 Data overview

The data set acquired has 1344967 records, which correspond to 9646 different routes. On average, a route deals with the delivery of 139 different packages. Table 3.3 presents relevant statistics on route sizes (in number of packages). There are 459828 different client locations, identified on the data set. Figure 3.1 presents the spacial distribution of customers locations and the demand for the study period (demand as number of packages delivered to the specific location). Observing figure 3.1, it is easily noted that the spacial distribution of demand goes beyond Porto's district. This points to possible errors on the geographical coordinates of the data set, since it would be highly unlikely for a client in Lisbon to be served by the depot in Porto. Further analysis on this is required. Moreover, 28% of the deliveries in data set are within Porto's city bounding box.

As it was already stated, the circuit_code attribute relates to a geographical zone, e.g., routes with the same circuit_code will serve customers within a somewhat regular geographical space. Figure 3.2 shows the geographical distribution of customers, in two different days, each color represents a different circuit. Comparing the two images, it seems that circuits are geographically permanent over time, especially

Table 3.2. Example of a route on the data set.

stop_id	date	circuit_code	delivery_time
A	20210729	LBOA	91509
B	20210729	LBOA	91541
C	20210729	LBOA	92352
D	20210729	LBOA	93046

Table 3.3. Statistics on route size, in number of packages.

mean	median	min	max
139.4	136.0	1.0	1036.0

in areas with high demand (like Porto and its surroundings). While in zones with lower demand density, like Valongo or Ermesinde, circuits are sort of flexible, e.g., the same area, in different days, can be served by different circuits.

Regarding the last_incidence variable, there are 27 different categories in the data set, but the majority of the data, $\approx 76\%$, has the category POD. POD means the parcel was correctly delivered.

There are three different categories for the service attribute. A, meaning the parcel must be delivered until 10:00; M, defining the maximum delivery time as 14:00; and, D, if the parcel can be delivered until 18:00. Figure 3.3 presents a bar plot with the frequency of occurrence of each category. The mode is D, meaning that most deliveries can be delivered until 18:00. Only a small percentage of records, $\approx 0.18\%$, have the most restrictive time window, A.

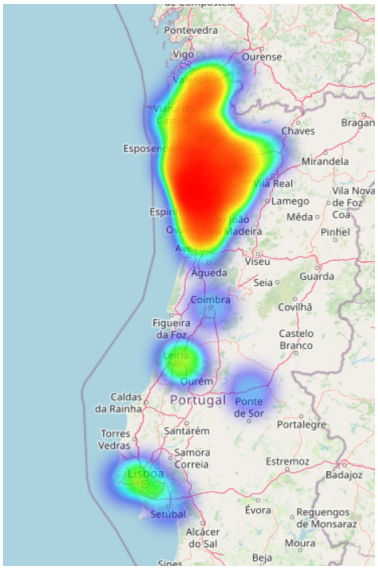
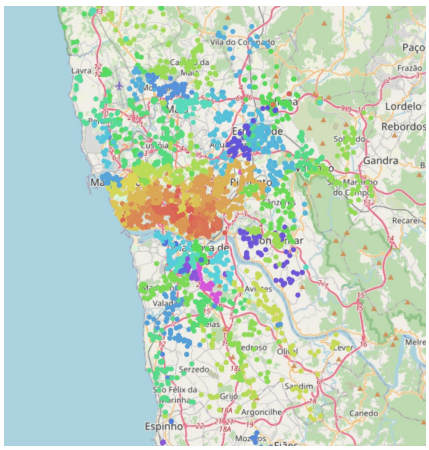
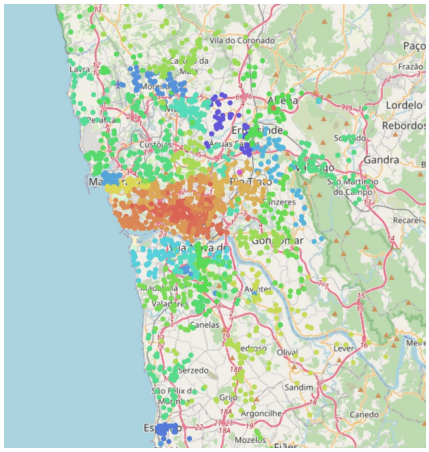


Figure 3.1. Demand density heatmap.



(a) Customers location on 03/12/2021.



(b) Customers location on 14/11/2021.

Figure 3.2. Customers location, by circuit, each color represents a different circuit.

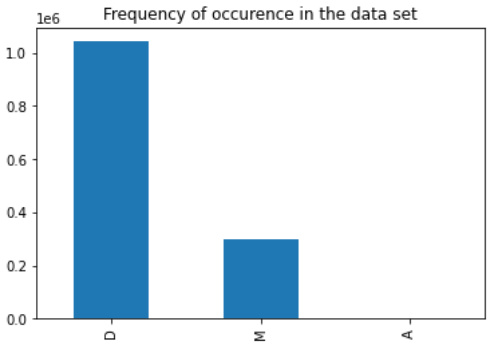


Figure 3.3. Bar plot with frequency of different category's occurrence.

Chapter 4

Data cleaning

In this chapter the preparation performed on the raw data will be detailed. In section 4.1 the framework used is described, section 4.2 details how were missing values treated, the handling of outliers is described in 4.3, while in section 4.4 the process of aggregating packages into stops is explained. The result of the raw data preparation is summarized in 4.5.

4.1 Framework used

Brownlee (2020) states that a data cleaning framework should involve identifying outliers, removing attributes without variance and imputing missing values. Before the actual definition of the data cleaning process, we must delineate its expected output. Because we are interested in analysing a route as a sequence of *stops*, the data cleaning process should return a collection of routes, without outliers nor missing values in the variables *delivery_time*, *latitude*, *longitude*, *weight*, *service*. Since on the original data set there isn't data on the stop level (only at the package level), the process should include an aggregation step, where packages are aggregated in a given stop.

4.2 Handling missing values

In this work, missing values were treated differently depending on the attribute at loss. This is explained by the specific structure of data, in the sense that there are attributes that can jeopardize data when missing, while others have no effect on data.

We will start by analysing the number and distribution of missing values on the data set. Figure 4.1 shows the distribution of missing values, throughout the data set. Attribute *address_2* has highest number of missing values, but one should note that

as it represents the second line of a given address, it isn't needed if the full address is stated on the *address_1* attribute. considering this, there is no need for imputation in missing values on *address_2*. On the other hand, attributes *package_id*, *date*, *circuit_code* and *weight* have zero missing values.

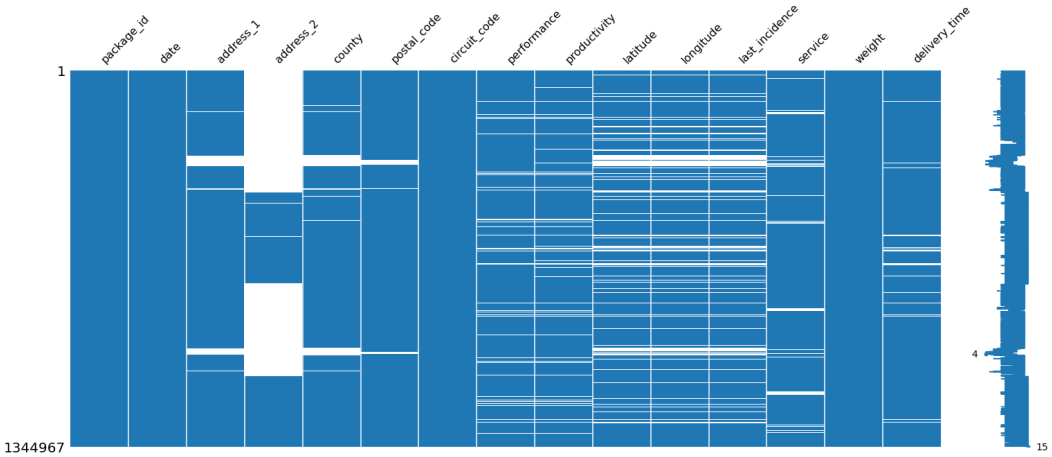


Figure 4.1. Missing values matrix, each white line represents a missing record.

Furthermore, table 4.1 presents the percentage of missing values (in total of records) for each attribute. Excluding *address_2*, that as stated previously doesn't actual represent a record at loss, the percentage of missing values, for the rest of the attributes, is below 20%.

Table 4.1. Percentage of missing values (in total of records), by attribute.

Attribute	**% of missing values (%)**
address_1	4.9
address_2	57.2
county	5.5
postal_code	2.0
performance	8.7
productivity	10.1
latitude	15.5
longitude	15.5
last_incidence	15.5
service	6.0
delivery_time	2.8

Before discussing the imputation method used for each attribute, one should note that there is redundancy on the dataset, in other words, both the geographical coordinates (given by latitude and longitude) and the address depict the same

information, so only the geographical coordinates will be imputed when missing.

In that sense and because there isn't an absolute overlap in the missing values on geographical coordinates and addresses, the addresses were used to compute latitude and longitude, through geocoding. The process of imputation is described in algorithm 4.2. It starts by computing the full address as the concatenation of every address attribute (*address_1*, *address_2*, *county* and *postal_code*). Then, the latitude and longitude are geocoded, using the Bing API (Microsoft et al., 2022). If the geocoding retrieves latitude and longitude values, they are returned and imputed to the missing values. Otherwise, when the geocoding fails, the whole process is repeated but with the full address as the concatenation of *address_1* and *postal_code*, because usually the geocoding fails due to incorrect data in the *address_2* and *county* (people, sometimes, use this data entries to register notes to the driver, like a phone number to call if the delivery fails).

Algorithm 4.1 Imputation of missing values for geographical coordinates

```
1:  Input: address_1, address_2, county, postal_code
2:  Output: lat and lon
3:
4:  procedure IMPUTATION(address_1, address_2, county, postal_code, delivery_time)
5:      full_address ← address_1||address_2||county||postal_code
6:      lat, lon ← geocode(full_address)
7:      if lat ≠ None and lon ≠ None then
8:          return lat, lon
9:      else
10:         full_address ← address_1||postal_code
11:         lat, lon ← geocode(full_address)
12:         if lat ≠ None and lon ≠ None then
13:             return lat, lon
14:         else
15:             return None, None
16:         end if
17:     end if
18: end procedure
```

On the other hand, it was indicated by the company that rows with missing values on the service attribute have the less restricting time window (e.g., D). Thus, all the missing values in the attribute service were imputed with D.

The attributes performance and productivity weren't used in the project, so its missing values weren't treated.

Moreover, for the delivery_time attribute it was noticed that the all the missing values occurred within the same routes. E.g., when a row is null in the delivery_time attribute, all the rows with the same instance of the tuple (date, circuit_code) were also null. Because the delivery_time is crucial to determine the sequence of a performed route, all the routes with all nulls in delivery_time were considered useless, therefore, deleted from the dataset.

4.3 Handling outliers

Given the structure of the dataset, outliers can only be found on the geographical coordinates. Different analysis can be conducted in order to handle outliers: outliers at the dataset level or outliers at the route level. As Zimek and Filzmoser state, outliers detection is a somewhat subjective procedure; one needs to understand what is "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data" (Barnett et al., 1994).

Considering this, two *types* of outliers are defined for the dataset:

1. A outlier on the dataset level, if the geographical coordinates of a record are inconsistent with the remainder of the dataset;

2. A outlier on a route level, if the geographical coordinates of a record are inconsistent with the remainder of the route data.

Table 4.2 presents the statistics on the *latitude* and *longitude* attributes of the dataset. If one were to use the interquartile range as a rule of thumb for outlier detection, then an outlier would be defined as any point below $IQ1 - 1.5IQR$ and above $IQ3 + 1.5IQR$ [1]. For this dataset, it would result in 174217 outliers ($\approx 12.9\%$ of records) in terms of latitude and 57877 outliers ($\approx 4.3\%$ of records) related to longitude. Combining the two, it totals to 193116 locations identified as outliers, representing $\approx 14.4\%$ of the dataset and meaning that there are 5989 routes with at least one outlier.

Table 4.2. Statistics on data distribution of latitude and longitude.

Statistic	Latitude	Longitude
Mean	41.1163	-8.6045
Standard deviation	0.2303	0.3551
IQ1	41.1421	-8.6547
Median	41.1777	-8.6191
IQ3	41.2285	-8.5694

Figure 4.2 represents the borders of the area without outliers, if they were identified by interquartile range. Looking at figure 4.2 and considering that 14.4% of the locations on the dataset are out of the bounds in blue, it seems that this method is incorrectly classifying data as "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data" (Barnett

[1] Any point with latitude below 41.0125 or above 41.3581 or longitude above -8.7828 or above -8.4415.

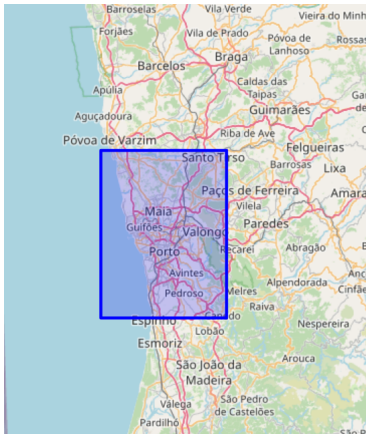


Figure 4.2. Borders of the area without outliers.

et al., 1994). This is easily proved, for instance, there is a circuit called LPOV, that consistently serves the area of Póvoa de Varzim, but Póvoa de Varzim is outside of the bounds, in that sense, considering the points on LPOV outliers is inaccurate.

Thus, instead of using a traditional methods for outliers detection, they were defined by district boundaries. Every point north and west of Porto's district and south of the district of Aveiro were considered outliers (at the dataset level).

The analysis of outliers at the route level should also be conducted, e.g., a point can be an outlier within a certain route, without being an outlier on the whole dataset. Take the section of a route as an example, $\{p_1, p_2, p_3\}$, with p_i representing the geographical location of a customer and i the moment the customer is served. If p_1 and p_3 are very near from one another, while p_2 is far away from the other two; it is heavily unlikely for p_2 to be served in between p_1 and p_2. Naturally, defining numerically what it is to be $near$ and far is complex, so, for simplification purposes, it is defined that a point is an intra-route outlier if all the following constraints are true (the same happens if, instead of latitude, it is longitude):

$$\|lat_{p_1} - lat_{p_2}| \geq 1 \tag{4.1}$$

$$\|lat_{p_2} - lat_{p_3}| \geq 1 \tag{4.2}$$

$$\|lat_{p_1} - lat_{p_3}| < 1 \tag{4.3}$$

After outliers are detected, they must be handled. In this case, because the data is time sequenced, the entry on the row above is imputed to the outlier.

4.4 Aggregating packages to a stop

Although the data is given at the package level, we are interested in the stop level. Thus, it is important to aggregate the packages in a common stop. A *stop* should be interpreted as a moment when a driver parks and serves a given number of *packages*. In the dataset, different instances where aggregation is possible are found:

1. When two packages have the same geographical coordinates and are delivered at the same time.
2. When two packages have the same geographical coordinates and are delivered at sequential times.
3. When two packages have different geographical coordinates but are delivered at the same time.

Defining $I_s = \{i_1, \ldots, i_n\}$ as the set of packages that belong to the stop s, then one can establish the weigh load of s, q_s, as the sum of the weight of each package in I_s. On the other hand, the time window for the deliveries at the stop s, tw_s, is defined by the most restrictive time window in the set of packages I_s. The delivery_time at a stop s is, then, defined as the latest delivery time in the set of packages I_s. Finally, one must define the stop by its the geographical coordinates (lon_s, lat_s), in order to do so, the mean value of the geographical coordinates within a given route is computed and, then, the coordinates at s are established as the value of (lon_I, lat_I), while I represents the package nearest to the mean value.

Algorithm 4.2 Aggregation of packages in a stop.

```
1:  Input: route = {latitude, longitude, delivery_time, service, weight} at package level
2:  Output: route = {latitude, longitude, delivery_time, service, weight} at stop level
3:
4:  procedure GROUPING DATA(Dataset D)
5:      for d in D do
6:      end for
7:  end procedure
8:  procedure AGGREGATION(route at package level)
9:      i ← 0
10:     for i ∈ {1, ..., len(route)} do
11:         if lat_i = lat_{i+1} and lon_i = lon_{i+1} then
12:             delivery_time_i ← delivery_time_{i+1}
13:             service_i ← min(service_i, service_{i+1})
14:             weight_i ← weight_i + weight_{i+1}
15:         else if  then
16:         end if
17:     end for
18: end procedure
```

4.5 Chapter's findings

After the steps described in sections 4.2, 4.3 and 4.4, the obtained dataset has a total of 542908 rows, which translates to 8524 different routes.

Chapter 5

Districting

In this chapter, the process of district creation is detailed. It starts with section 5.1, in which the districting problem is mathematically formulated; in section 5.2 the iterative solution algorithm used is further explained. Then section 5.3 details the real life data used as an input to the model, how it was adapted to fit it, and the parameters estimation of the model.

5.1 Problem formulation

As it was already stated, the goal of the district phase of the project is the hierarchical partition of a service territory. In order to do so, one must define a recursive method for districting, in that sense, a top down approach will be applied, similar to the one used in divisive hierarchical clustering (Madhulatha, 2012). In this type of approach, we start by defining the whole service region as a district and, iteratively, partitioning it into more districts. In this work, three different levels of districts will be defined: nano districts, micro districts and macro districts. Intuitively, one can define a macro district as a set of micro districts, and, a micro district as a set of nano districts.

The formulation used as a base for the districting problem is the one presented in Fellus (2022). The mathematical model will be applied recursively, in order to achieve the hierarchical structure. Figure 5.1 describes the recursive algorithm developed. The algorithm starts by solving the problem for $level = 0$, retrieving K_{macro} macro contiguous districts. The optimization problem is solved for each $k \in K_{macro}$ macro districts ($level = 1$), resulting in K_{micro} districts. Again, the model is solved for each $k \in K_{micro}$ districts ($level = 2$), producing K_{nano} districts, and ending the algorithm.

Denoting $C = \{1, \ldots, |C|\}$ as the set of customers, a customer is represented by $c \in C$ and has an associated demand q_c and service time t_c. One establishes

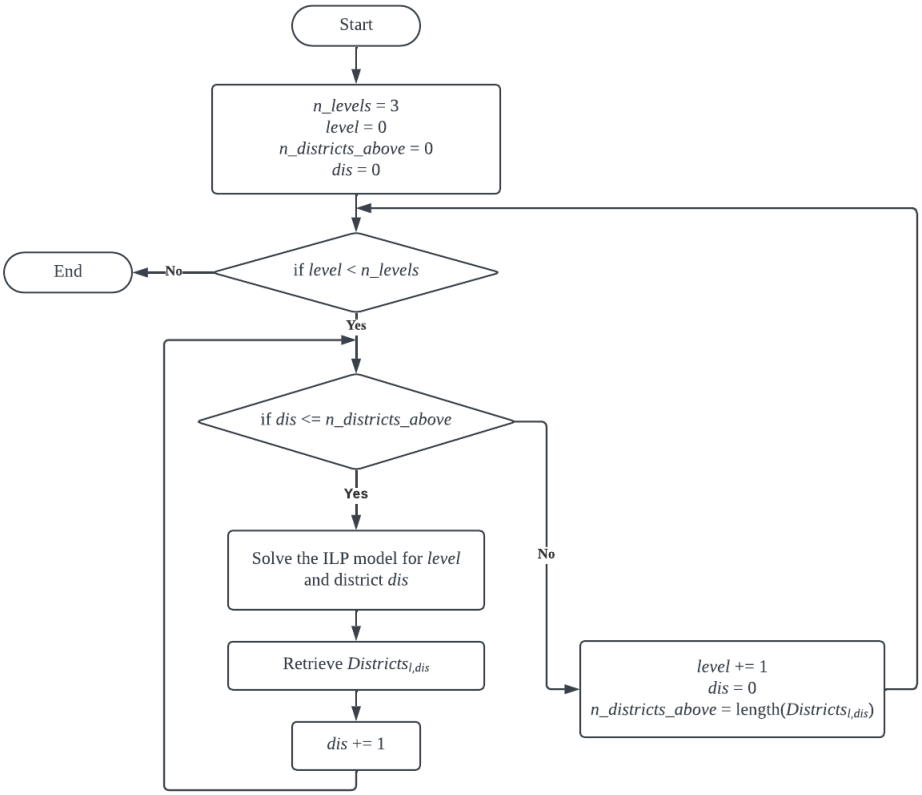


Figure 5.1. Flowchart describing the recursive algorithm that achieves the hierarchical structure.

$B = \{1, \ldots, n\}$ as a set of basic units (A service region consists of n basic units of equal dimension). The decision variable is binary and defined by equation 5.1. Note that when $x_{ii} = 1$, defines i as the center of district i.

$$x_{bi} = \begin{cases} 1, & \text{if BU } b \text{ is assigned to BU represented by center } i \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

The model aims to balance the workload between different districts, so one defines $\varphi_i^{(Q)}$ as the relative workload of a district weight wise and $\varphi_i^{(T)}$ as the relative workload of a district time wise. $\varphi_i^{(Q)}$ is computed as stated in equation 5.2, with Q_b representing the total weight of the parcels delivered in BU b and Q_{MAX} the total parcel weight allowed in a district.

$$\varphi_i^{(Q)} = \frac{\sum_{b=1}^{n} Q_b x_{bi}}{Q_{MAX}} \tag{5.2}$$

Moreover, $\varphi_i^{(T)}$ has three factors: the travelled time from the depot to the district

center, the approximation of the travelled time inside each district and the service time for each client. It is obtained with the equation 5.3, where $cost_{depot,i}$ is the distance from the depot to the BU i, v_L the average road speed inter-district, β_2 a parameter related to the number of clients in the area[1], n_b represents the number of clients within basic unit b, A_b the area of the BU, and S_b the sum of the service time of the clients in the BU.

$$\varphi_i^{(T)} = \frac{\frac{2cost_{depot,i}x_{ii}}{v_L} + \frac{\sum_{b=1}^{n}\beta_2(\sqrt{n_b A_b x_{bi}})}{v_U} + \sum_{b=1}^{n} S_b x_{bi}}{T_{MAX}} \tag{5.3}$$

Regarding the workload factors, two complementary decision variables must be introduced, y_i, defined by equation 5.4, and z_i, defined by equation 5.5. The decision variables establish which workload factor is more restrictive, weight or time, e.g., if $y_i = 1$ the relative workload weight wise is tighter than the relative workload time wise.

$$y_i = \begin{cases} 1, & \text{if } \varphi_i^{(Q)} \geq \varphi_i^{(T)} \text{and } \varphi_i^{(Q)} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.4}$$

$$z_i = \begin{cases} 1, & \text{if } \varphi_i^{(T)} \geq \varphi_i^{(Q)} \text{and } \varphi_i^{(T)} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.5}$$

The model is formulated as follows:

[1]The approximated the value of β_2, depending on the number of clients in the area is presented in Lei et al. (2015).

$$\min \quad \sum_{b=1}^{n}\sum_{i=1}^{n} cost_{bi}x_{bi} \tag{5.6}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{bi} = 1 \qquad \forall b \in B \tag{5.7}$$

$$x_{bi} \leq x_{ii} \qquad \forall b \in B, \forall i \in B \tag{5.8}$$

$$\varphi_i^{(Q)} \leq 1 \qquad \forall i \in B \tag{5.9}$$

$$\varphi_i^{(T)} \leq 1 \qquad \forall i \in B \tag{5.10}$$

$$y_i \geq \varphi_i^{(Q)} - \varphi_i^{(T)} \qquad \forall i \in B \tag{5.11}$$

$$y_i \leq x_{ii} - (\varphi_i^{(Q)} - \varphi_i^{(T)}) \qquad \forall i \in B \tag{5.12}$$

$$y_i + x_i = x_{ii} \qquad \forall i \in B \tag{5.13}$$

$$\varphi_i^{(Q)} \geq (1-\varepsilon)y_i \qquad \forall i \in B \tag{5.14}$$

$$\varphi_i^{(T)} \geq (1-\varepsilon)z_i \qquad \forall i \in B \tag{5.15}$$

$$\sum_{b \in \bigcup_{b' \in S} ADJ_{b'} \setminus S} x_{bi} - \sum_{b \in B} x_{bi} \geq 1 - |S| \quad i \in B, S \subseteq B \setminus (i \cup ADJ_i), S \neq \emptyset \tag{5.16}$$

$$x_{bi} \in \{0,1\} \qquad \forall b, i \in B \tag{5.17}$$

$$y_i \in \{0,1\} \qquad \forall i \in B \tag{5.18}$$

$$z_i \in \{0,1\} \qquad \forall i \in B \tag{5.19}$$

The objective function (5.6) minimizes the cost of assigning a BU to a district, in other words, it minimizes the distance from a BU center to all the BUs assigned to that given district/center. Constraint (5.7) ensures that each BU must be assigned to a district. (5.8) makes sure that a BU $b \in B$ is assigned to a district $i \in B$, if and only if, i has been selected has a district center. Constraints (5.9) and (5.10) establish an upper limit on the size of the districts, e.g., regarding the quantity served and the time needed to perform the service, within a district. (5.11), (5.12), (5.13) are logical constraints needed to define the decision variables y_i and z_i. Constraints (5.14) and (5.15) constitute the lower limit on the workload factors of each district. (5.16) ensures the connectivity of the solution, assuring that territorial design is contiguous. Finally, (5.17) and (5.18) insure that the decision variables are binary.

Effectively, the number of connectivity constraints (5.16) grows exponentially, depending on n (the number of BUs), in that sense, solving the problem with all the connectivity constraints is not viable. And so, the iterative solution algorithm developed in Fellus Fellus (2022) is used.

5.2 Iterative solution algorithm

As already explained in section 5.1, an iterative solution algorithm will be used, as developed in ... The algorithm used is represented in figure 5.2. It starts by solving the formulation presented in (5.6)-(5.19), with the connectivity constraint (5.16) relaxed. Afterwards, the connectivity of the obtained solution is checked, if no constraint is violated, a solution for the optimization problem was found, and the algorithm ends. Otherwise, e.g., when the solution is contiguous (fails the connectivity check), the violated constraints are identified and added to the model. Next, the model is solved with the new constraints and the process loops back.

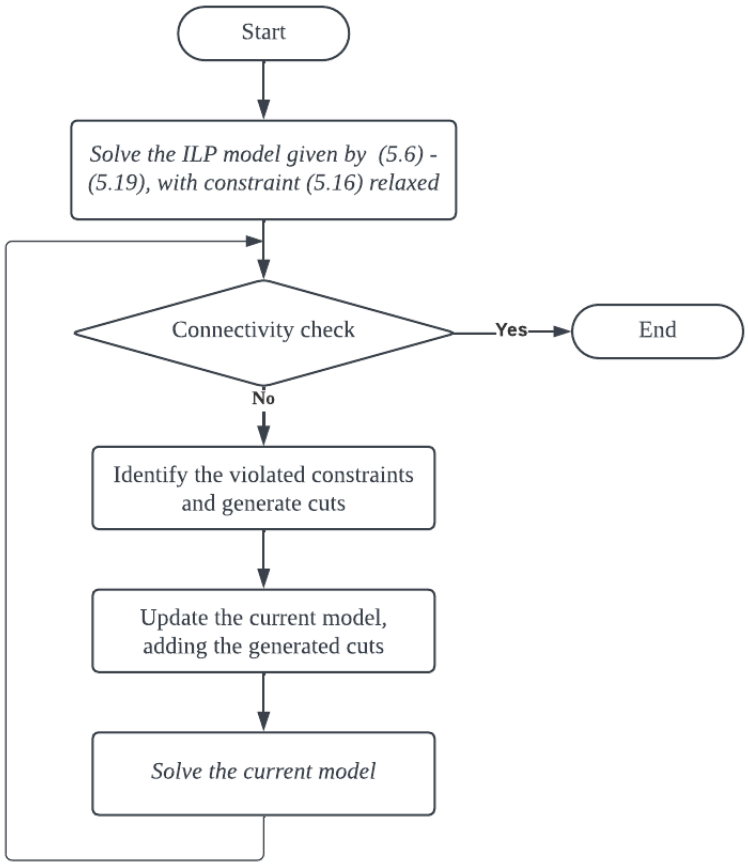


Figure 5.2. Fluxogram describing the iterative solution algorithm. Adapted from Fellus (2022)

It is necessary to explain what a contiguous solution is, in the context of the problem. As each BU $b \in B$ defined by a rectangle, it is established that a solution is connected, if and only if, every BU $b \in B$ assigned to a district $i \in B$ share, at least, one edge. Figure 5.3 presents an example of what an unconnected solution

would look like, each square represents a BU $b \in B$ and their colour indicates the district they are assigned to. This violates the connectivity constraints (e.g., isn't contiguous/connected) because the BUs 1 and 5 are assigned to the same district, but don't share an edge; the same happens for BUs 2 and 4.

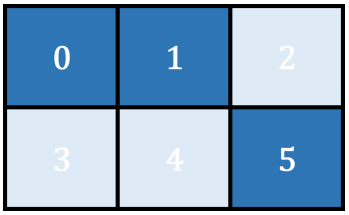


Figure 5.3. Example of an unconnected solution.

5.3 Real life data instance

In this section, the data used as an input for the districting problem is described. Beyond this, the transformations performed on data and the rationale behind the parameters values' decision is outlined.

First, table 5.1 provides a summary on the inputs needed to solve the iterative solution algorithm presented in 5.2. Note that, in this context a customer is represented by a stop, concept already explained in chapter 3.

Table 5.1. Model inputs.

Input	Description
$C = \{1, \ldots, \lvert C \rvert\}$	Set of customers, $c \in C$ represents a single customer
q_c	Quantity, in weight, required for customer c
s_c	Service time, in seconds, required for customer c
(x_c, y_c)	Coordinates of the location of customer c
R	Service region
$dim_{R,hor}$	Dimension of the service region, in the x axis
$dim_{R,ver}$	Dimension of the service region, in the y axis
$B = \{1, \ldots, n\}$	Set of basic units, $b, i \in B$ identifies a single BU
$C_b = \{c \in C : c$ in the border of BU $b\}$	Set of customers in BU b
Q_b	Total quantity, in weight, required by customers in BU b
S_b	Total service time, in seconds, required by customers in BU b
(x_{depot}, y_{depot})	Depot's coordinates i
$cost_{b,i}$	Cost of assigning BU b to district i
$cost_{depot,i}$	Distance between the depot and i

The instance used for the districting problem will be a weekly sample of the dataset, with median demand. Figure 5.9 shows a bar plot with the different demand levels, in terms of number of stops per week. Red line represents the week with maximum demand value (a total 30622 stops), yellow line emphasis the week with median demand (19508 stops) and the green line highlights the week with minimum demand (8972 stops). A weekly basis sample (as opposed to a daily analysis) is chosen in order to absorb the daily variability of locations.

The median demand week, week 28, includes a total of 300 routes, e.g., 300 unique occurrences of the tuple (date, circuit_id), distributed by 5 days.

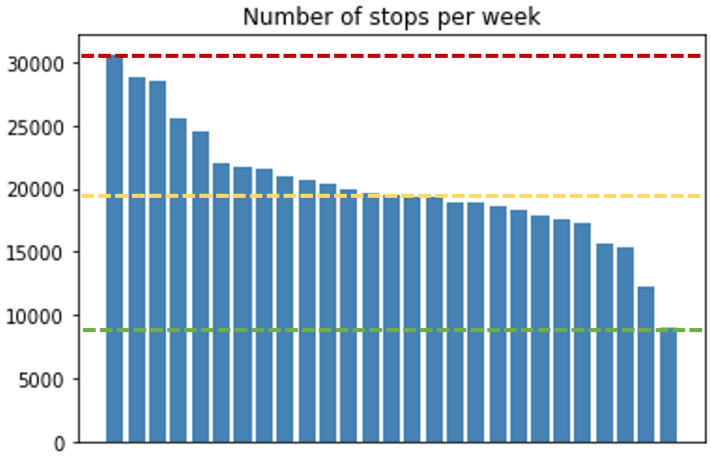


Figure 5.4. Bar plot with the number of stops per week, red line highlights maximum demand week, yellow line the median demand week and green line the minimum demand week.

Note that, in the dataset all the coordinates are in geographical coordinates system, but the districting problem involves Cartesian operations, thus, all the coordinates were converted to Universal Transverse Mercator (UTM) coordinates, using the library Bieniek (2022) in Spyder environment.

5.3.1 Service area

The definition of the service area, R, that later will be partitioned into districts is crucial. R is a region in the 2D space defined by a bounding box, $R = \{x_R \in [x_{min}, x_{max}] \ \wedge \ y_R \in [y_{min}, y_{max}]\}$. The values of x_{min}, x_{max}, y_{min} and y_{max} should be carefully chosen, taking into account that the model aims to minimize compactness while keeping the districts balanced. In that sense, the demand should be sort of spatially distributed within R. Figure 5.5 depicts the location of the points in the 1st and 99th percentiles of the dataset, showing the importance of selecting the bounding box, in order to get feasible solutions from the model. If one were to define

R as a bounding box that could fit every location point on the dataset, it would be impossible to retrieve balanced districts, since most of the demand is located in the area in dark blue.

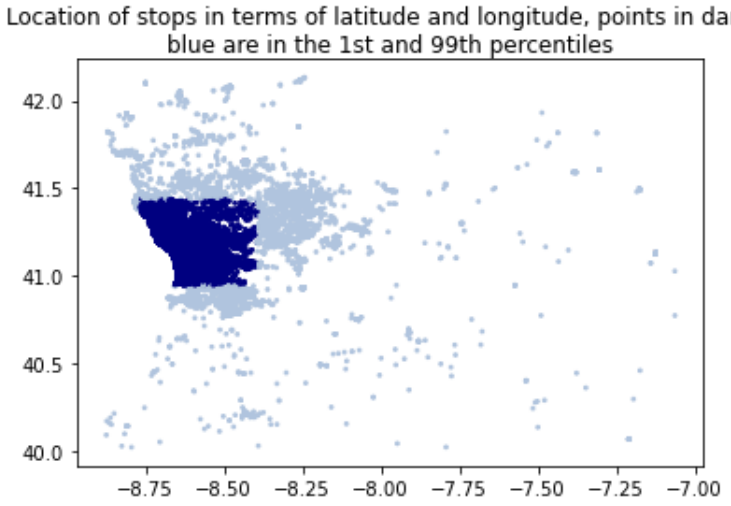


Figure 5.5. Location of each stop in the dataset in terms of longitude and latitude, the darker shade represents values within the 1st and 99th percentiles of data distribution.

Thus, the service area is established as the rectangle that delimits the points in dark blue, plotted in 5.5. In Universal Transverse Mercator (UTM) coordinates, this corresponds to having $R = \{x_R \in [519507, 552007] \wedge y_R \in [4532891, 4587891]\}$.

Note that, for the further levels of the districting problem (e.g., for the creation of micro and nano districts) R is a determined by the prior level of districting.

5.3.2 Basic units' dimension

It is also mandatory to define the size of the basic units, for each level of the districting problem. The basic unit is the smallest unit in the problem, e.g., a district is a set of BUs. The size of the problem will be dependent on the dimension of each BU, if the chosen dimension is too small the problem will be too complex.

Other than that, one take into consideration that, for the third level of districting, the size of each BU should be, at least, smaller that the average route size. This is explained because the goal of the whole project is to create districts that can be used to infer preferences of drivers when driving a route, so, in order to know the usual traversing between nano-districts, the nano-district needs to be smaller that the route (the average route size is $\approx$ 8.7 km2).

These two limitations should be considered when defining the BUs dimension.

For the first level of districting, a BU is defined as 2500 x 2500 square, meaning that an area of, approximately, 6.25 km2 of clients, or demand, is aggregated,

and producing a problem with 286 basic units. Figure 5.6 depicts a graphical representation of region R and the BUs.

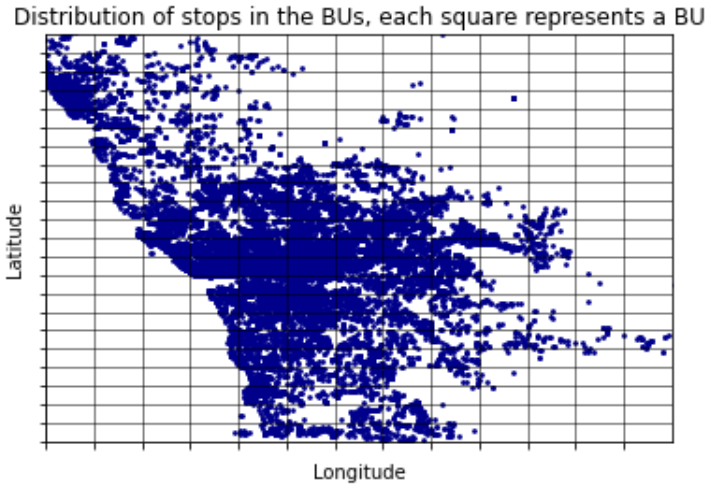


Figure 5.6. Distribution of the stops in the BUs, each square is a BU.

In the further levels of districting, the dimension of the BU must be a divisor of 2500. For the second level, it is set as 1250 ($\approx$ 1.56 km2). And, for the third level, the dimension of the BU is 625 ($\approx$ 0.39 km2).

5.3.3 Artificial district

As it was already stated, it is important for the demand to be somewhat distributed through R, but that isn't the case when looking at figure 5.6, since there are multiple empty BUs. For example, the empty basic units on the left are located in the ocean, so, naturally, they shouldn't be included in the model. Thus, it is important to define an artificial district that gathers the BUs without or almost without demand. Figure 5.7 exhibits the BUs that belong to the artificial district. The artificial district, like the other districts, is defined by a BU (the center of the artificial district). So, in other to add it to the model, a new constraint must be introduced. Defining BU_{art} as the set of BUs that belong to the artificial district and i_{art} as the center of the artificial district (that can be any BU in BU_{art}), one can set:

$$x_{bi_{art}} = 1, \forall b \in BU_{art} \tag{5.20}$$

5.3.4 Customers, service time and demand

A customer $c \in C$ represents a stop where a driver must deliver a parcel. The location of a customer (or service point) is denoted by its coordinates (x_c, y_c).

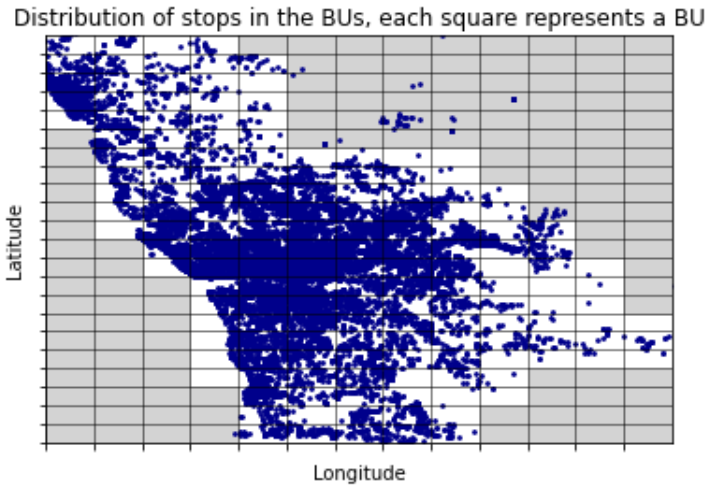


Figure 5.7. Distribution of the stops in the BUs, each square is a BU, the squares belong to the artificial district.

Each customer is also associated with a service time s_c, representing the handling time required for a driver to physically deliver a parcel once he gets to location, meaning that the service time does not take into account the travelled time.

Because service time data isn't explicit on the data set, one must infer it. Considering that the time of each delivery is given, then, the service time could be defined as the time between the delivery time of two consecutive stops, minus the travelled time required to go from the first to the second stop. The following equation summarizes it:

$$serv_c = delivery\_time_c - (delivery\_time_{c-1} + trav\_time_{c-1,c}) \tag{5.21}$$

where $delivery\_time_c$ is the time when customer c is served, $delivery\_time_{c-1}$ the time customer $c-1$ was served and $trav\_time_{c-1,c}$ the driving time going from customer $c-1$ to customer c. To calculate driving time the package OSMnx was used (Boeing, 2017). The service time at each stop, in the week in analysis, has a mean value of 292.6 seconds and standard deviation of 293.8 seconds. Figure 5.8 shows the distribution of the service time, the majority of data sits in the bin from 60 to 144 seconds.

A customer, or stop, c is also defined by the demand required, d_c. In the data set the demand is depicted as weight, given in grams. Figure 5.9 represents the distribution of demand, in grams, in the stops. The value of d_c mostly sits between 0 and 69.93 grams, having a mean value of 7.7 grams with a standard deviation of 25.7 grams.

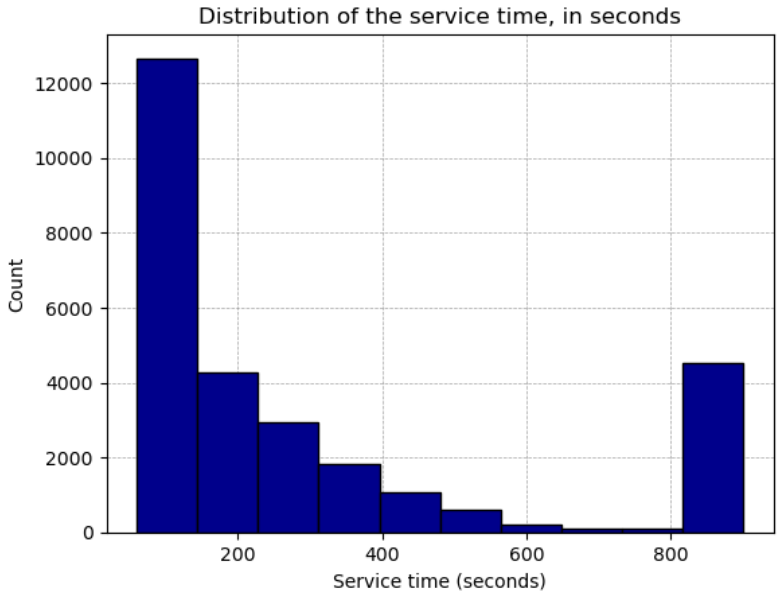


Figure 5.8. Distribution of the service time of stops, in the week in analysis.

5.3.5 Maximum workload in time and demand

One of the parameters of the model is the maximum workload of a district, T_{MAX}, its value is different for each level of districting. On the macro level, one considers all the workload of the sample, and so, assuming each route is assigned to a single driver, the total workload is the number of routes in the time period multiplied by the number of hours of a work shift. Assuming the work shift in Portugal as 8 hours and using the median number of routes per week, 300, the whole workload represented in the weekly sample can be established as $300 \cdot 8 \cdot 3600$ seconds. If one were to set $T_{MAX} = 300 \cdot 8 \cdot 3600$, it was assuming that the maximum workload per district was equal to the workload of the whole service region, which would turn the constraint on workload limit trivial. So, for the macro level, one defines $T_{MAX} = (300 \cdot 8 \cdot 3600)/2$, aiming at the creation of at least two macro districts. Two was used as a divisor because using a bigger number resulted in an unfeasible model.

On the micro level, one wants each district to represent an area covered by a single driver, so, intuitively, T_{MAX} is established as $5 \cdot 8 \cdot 3600$ seconds, e.g., the length of a driver's work shift multiplied by the number of days in the sample.

Furthermore, on the nano level, the districts should be smaller than the average route size, meaning that one can set T_{MAX} as the T_{MAX} on the micro level, divided by a number that produces feasible solutions. In that sense, T_{MAX} was established as $(5 \cdot 8 \cdot 3600)/5$, aiming to divide the district on the micro level in 5 different zones.

Likewise, we need to define the maximum weight capacity a vehicle, although

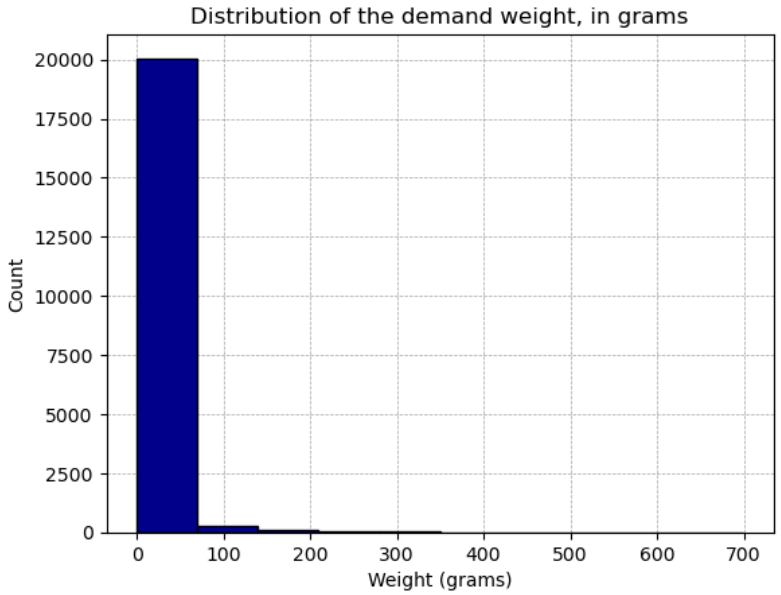


Figure 5.9. Distribution of the demand at the stops, in the week in analysis.

we don't have that sort of information, it is possible to estimate it. Because we are dealing with last mile delivery, typically the vehicle used is a light goods vehicle, with a maximum capacity of 1200 kg. Thus, we denote $Q_{MAX} = 1200 \cdot num\_days$.

5.3.6 Other parameters

Given the model in question, other parameters such as, β_2 and ε, are required to define. Applegate et al. (2006) presents an estimate on the value of β_2 as a function of n, the number of customers within a given area, for a range of n from 100 to 2000. Because β_2 relates to n, it will be different for macro, micro and nano districts, because the areas of each are also different (and consecutively the number of customers within those).

The service region R has an area of $1006.25\,\text{km}^2$ and serves 28283 customers weekly (≈ 5142 daily). An approximate value for the area of the districts, depending on the level, can be obtained. For the macro level, considering the constraint on the workload in time, it is expected for the model to return two macro districts, in that case, the area of each is expected to be $\frac{1006.25}{2} = 503.13\,\text{km}^2$ and the daily number of customers within each, n, 2571. The macro districts are expected to have a workload of $(300 \cdot 8 \cdot 3600)/2$, thus, the number of micro districts in each macro district is expected to be $T_{MAX,macro}/T_{MAX,micro}$, approximately 28. Therefore, the expected area of each micro district is $\frac{503.13}{27} = 18.63\,\text{km}^2$ and the expected number of clients per district is 95. Using the same rationale, the expected area of each nano district is $3.73\,\text{km}^2$ and the number of clients is 19.

Table 5.2. District level, dimension BU, number of clients per BU, β_2.

District level	Approx area (km^2)	Approx n	β_2
Macro	503.13	2571	0.7256264
Micro	18.63	95	0.7764689
Nano	3.73	19	0.8584265

On the other hand, ε acts as a minimum workload constraint, for instance, setting $\varepsilon = 0.4$, ensures that each district should hold, at least, 60% $(1 - 0.4)$ of the maximum workload (in terms of time or demand). The values for ε vary depending on the districting level, table 5.3 shows the values defined.

Table 5.3. District level and ε.

District level	ε
Macro	0.5
Micro	0.35
Nano	0.30

5.4 Model output to district representation

5.4.1 Notation

The output of the model (5.6)-(5.19) is the value of the decision variables x_{bi}, further explanation is needed to understand how that leads to the representation of each district. As an example use the solution defined by (5.22)-(5.24), where BUs 0, 2 and 9 are the centers of the 3 districts created:

$$x_{0,0} = 1 \wedge \ x_{1,0} = 1 \wedge \ x_{4,0} = 1 \tag{5.22}$$

$$x_{2,2} = 1 \wedge \ x_{3,2} = 1 \wedge \ x_{6,2} = 1 \wedge \ x_{7,2} = 1 \wedge x_{11,2} = 1 \tag{5.23}$$

$$x_{9,9} = 1 \wedge \ x_{5,9} = 1 \wedge \ x_{8,9} = 1 \wedge \ x_{10,9} = 1 \tag{5.24}$$

The region R of this solution is defined by the coordinates $R = \{x_R \in [x_0, x_1] \ \wedge \ y_R \in [y_0, y_1]\}$ and each BU is a square with a certain dim_{BU} (the side length of the BU). Figure 5.10 shows a visual representation of the solution described.

Considering this, one can define each district of the solution as a polygon. In order to do so, one must define each district as a set of BUs, resulting in $Dis_0 = \{0, 1, 2\}$, $Dis_2 = \{2, 3, 6, 7, 11\}$ and $Dis_5 = \{5, 8, 9, 10\}$. Then, each district is treated

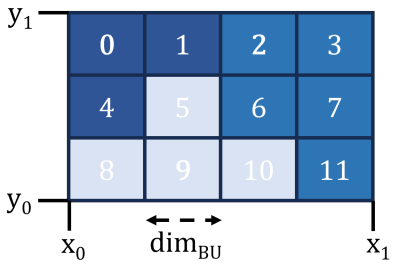


Figure 5.10. Visual example of a solution.

independently, and the the vertices of each BU are identified, as shown in figure 5.11. To compute the vertices' coordinates, procedure in 5.2 is followed. The algorithm takes as input the BU index, b, the borders of region R, x_0, x_1, y_0, y_1, and the side length of each BU, dim_{BU}. Then it computes the row and column number of each BU, for instance, BU $b = 0$, sits in row 0 and column 0, the column number increases left to right and the row number increases from top to bottom. Then, the row and column number are used to determine the coordinates of the four vertices of the BU, A (top left), B (top right), C (bottom left) and D (bottom right). Then, for each district there is a set of vertices, V_i, representing the vertices of each BU b assigned to district i.

Algorithm 5.1 Computation of the coordinates of each vertex in a BU

Input: $b,\ x_0,\ x_1,\ y_0,\ y_1,\ dim_{BU}$
Output: $(x_A, y_A),\ (x_B, y_B),\ (x_C, y_C),\ (x_D, y_D)$

procedure GET COORDINATES($(x_A, y_A),\ (x_B, y_B),\ (x_C, y_C),\ (x_D, y_D)$)
    $num\_bus\_hor \leftarrow (x_1 - x_0)/(dim_{BU})$
    $row\_number \leftarrow b//num\_bus\_hor$ ▷ Integer division
    $col\_number \leftarrow b \mod num\_bus\_hor$ ▷ Modulus division
    $x_A \leftarrow (x_0 + col\_number * dim_{BU})$
    $y_A \leftarrow (y_1 - row\_number * dim_{BU})$
    $x_B \leftarrow (x_0 + col\_number * dim_{BU} + dim_{BU})$
    $y_B \leftarrow (y_1 - row\_number * dim_{BU})$
    $x_C \leftarrow (x_0 + col\_number * dim_{BU})$
    $y_C \leftarrow (y_1 - row\_number * dim_{BU} - dim_{BU})$
    $x_D \leftarrow (x_0 + col\_number * dim_{BU} + dim_{BU})$
    $y_D \leftarrow (y_1 - row\_number * dim_{BU} - dim_{BU})$
    return $(x_A, y_A), (x_B, y_B), (x_C, y_C), (x_D, y_D)$
end procedure

Using the set V_i, the district can be defined by a polygon (e.g., an ordered set of vertices). The polygon is a subset of V_i, containing only the vertices with an even number of incident edges. For the example given, the three polygons (representing the three districts) are defined by the set of vertices in figure 5.12.

For the polygons to be correctly defined, it is mandatory that the vertices are correctly ordered, figure 5.13 uses district with center $i = 2$ as an example of different ways to order the polygon's vertices. In fact, 5.13a presents the only *correct* way to

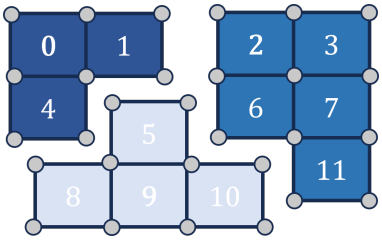


Figure 5.11. Vertices of the BUs in each district.

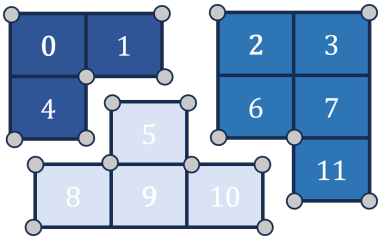


Figure 5.12. Vertices of each polygon, representing each district.

order the set of vertices, because, since the districts are composed of squares (e.g., the BUs), every polygon must be rectilinear.

(a) Vertices correctly ordered. **(b)** Vertices incorrectly ordered.

Figure 5.13. Examples of correct and incorrect orders for the set of vertices of the polygon.

In order to achieve this, algorithm 5.2 was developed. The algorithm retrieves every edge on the polygon. It uses the set of vertices of the polygon. It starts by grouping the vertices by their y coordinate and ordering them, within each group, by x coordinate. In each group there will be an even number of vertices (because there are no degenerate vertices) and every two vertices form an edge (that is inserted in the set of edges). The same process is repeated but for groups with the same x coordinates. The algorithm retrieves the set of edges of the polygon. Every vertex has two incident edges, so going from edge to edge (with common vertex) correctly orders the vertices.

5.4.2 Achieving the recursive approach

To achieve the recursive approach, some issues must be handled, because the model was constructed having it in consideration.

For starters, after the first level of districting, it is unlikely for the service area (or

Algorithm 5.2 Ordering a set of vertices to form a rectilinear polygon

Input: $Vertex_p$
Output: ordered $Vertex_p$

procedure ORDER VERTICES($Vertex_p$)
    G_y = Group of vertices with y coordinate y ▷ Creates a group that groups the vertices with the same y coordinate
    $G_y = Order(G_y)$ ▷ Orders each group by x value
    $edges \leftarrow \emptyset$
    for $(i = 0;\ i < len(G_y) - 1;\ i = i + 2)$ **do**
        $edges\ \cup\ (G_y[i], G_y[i+1])$
    end for
    G_x = Group of vertices with y coordinate x ▷ Creates a group that groups the vertices with the same x coordinate
    $G_x = Order(G_x)$ ▷ Orders each group by y value
    $edges = \emptyset$
    for $(i = 0;\ i < len(G_x) - 1;\ i = i + 2)$ **do**
        $edges\ \cup\ (G_x[i], G_x[i+1])$
    end for
end procedure
return *edges*

region R) to remain rectangular, in 5.10, for instance, none of the districts created are rectangular. Thus, for the model to handle the different shapes designed, a procedure must be followed.

First, x_0, y_0, x_1 and y_1 (minimum and maximum x and y values of the district) are identified. R is then defined as the area within x_0 and x_1 and y_0 and y_1. In figure 5.14 R is the area in yellow, for district 9. It is trivial to state that, although the BUs 4 and 6 aren't assigned to district 9, they still lay in R. Therefore, a *forbidden area* must be defined, the *forbidden area* is an area inside R, but outside of the district's bound.

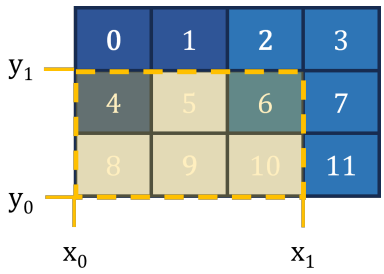


Figure 5.14. Representation of R (in yellow) for the next level of districting.

The *forbidden area* is, then, defined by the bounds of the BUs inside R, but outside the district's bound. Figure 5.15 represents this. Later, every BU in the *forbidden area* will be assigned to the artificial district of the instance.

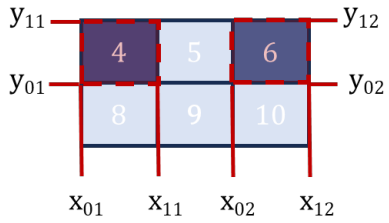


Figure 5.15. Coordinates of the BUs in the forbidden area.

5.5 Result Analysis

5.5.1 Macro Districts

At the macro districts level, the most restrictive constraint was the maximum workload in terms of time, defining the minimum demand per district as $(1 - 0.5) \times T_{MAX}$.

The model created six different districts, one of those being the artificial district. Table 5.4 summarizes the result on the macro level, district with center 5 is the artificial district, thus $\varphi_5^{(T)}$ is 0. District 120 has the higher $\varphi_i^{(T)}$, meaning that the demand level is greater. As expected, the model produced balanced districts, with weekly working hours from 603 to 918. The number of BUs within each districts, allows inferring the area of each, e.g., the more BUs a district has, the bigger its area.

There isn't an exact correlation between the number of basic units per district and its workload. For instance, the district with the lower number of BUs (and consequently the one with a smaller area), district 120, is, at the same time, the district with the higher workload. This is indicates that the district is located in a higher demand area, like central Porto.

Table 5.4. Macro level solution.

District Center	$\varphi_i^{(T)}$	Workload in time (hours)	Number of BUs
5*	0	0	124
41	0.50315	603.78	38
120	0.76549	918.59	21
124	0.51701	620.41	26
192	0.50878	610.54	38
239	0.55037	660.44	49

Figure 5.16 shows the location of the macro districts on the map (the artificial district is not included). The highest demand district is located in the coastline of Porto's municipality.

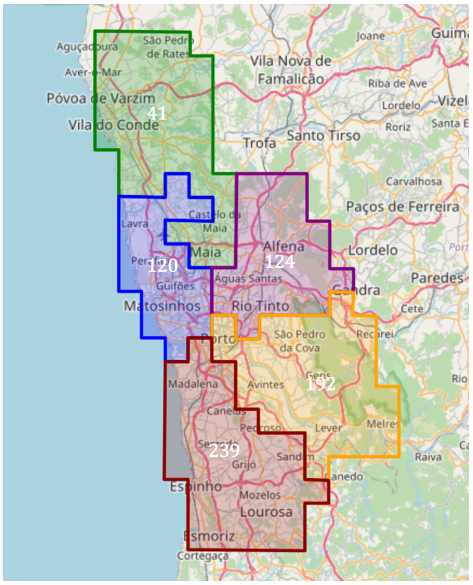


Figure 5.16. Macro districts, as polygons, each color represents a different macro district, the numbers identify the district center.

Analysing it, some notes should be taken, regarding the specific geographical features of the territory. There is a river separating two major cities in region R, Porto and Gaia. Because crossing the river is too time consuming, it is important to have the two sides of the river in different districts (specially the micro districts), but this doesn't happen in the solution provided. One could state that, if the driving distance was used, instead of euclidean, the problem of splitting river sides would be solved. But, in fact, that isn't the case.

Due to the grid structure, where a region R is divided in equal squares (the BUs), there are BUs, e.g., the smallest unit of the problem, that contain both sides of the river. Figure 5.17 exemplifies this. Because neither using driving distance, nor using a multiplier for the cost of assignment between different sides of the river, handles this issue; the districts will be post-processed, manually changing them, to impose the division by the river. This will be later explained in section 5.5.2, since the post processing will be done at the micro district level.

5.5.2 Micro Districts

Using the framework in 5.1, the districting problem is then applied for each macro district, resulting in 49 micro districts. Appendix A shows the representation of the created micro districts. Table 5.5 exhibits the number of micro districts per macro district. Every macro district origins nine micro districts.

Figure 5.18 captures the workload, in time, of the different micro districts, on the x axis the macro district containing them can be checked. The workload varies

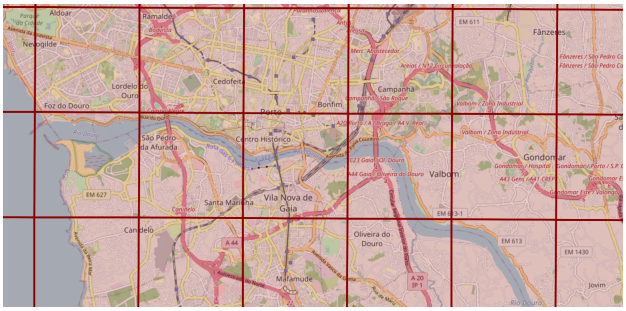


Figure 5.17. Examples of BUs that contain two sides of the river.

Table 5.5. Measures of central tendency and dispersion for the workload, in hours, per route.

Macro district	# Micro districts
41	9
120	9
124	9
192	9
239	9

from 0.600 to 0.986; meaning that the working hours, per district, are expected to range from 4.8 to 7.9 hours. As expected, the micro districts contained in macro district 120 have a higher workload factor, this because macro district 120 has the greater workload factor among all the macro districts.

Adjustments related to the topography of the region

As already stated, the geographical region R contains a river and, in a districting for VRP context, the river should serve as a physical barrier, when constructing routes. In other words, a route shouldn't serve customers in different sides of the river.

The micro districts represent a route, meaning that all the packages within the area of a micro district are served by the same route. Thus, a micro district shouldn't contain different river sides. However, this isn't the case for the obtained solution, as figure 5.19 shows.

Because of this, adjustments are needed to be carried. To readjust the districts, the river is used as a border for the micro districts. Figure 5.20 shows this, the line in blue is used as a frontier, and will serve as an edge of the district. On the right, the districts created by the frontier are shown.

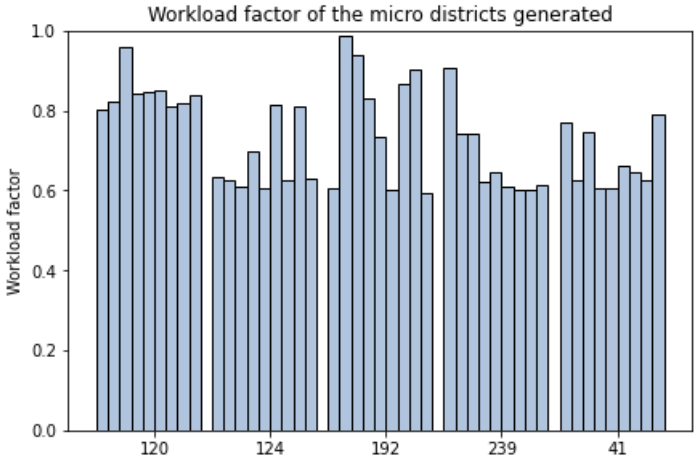


Figure 5.18. Workload in time of the micro districts, depending on the macro district they belong to.

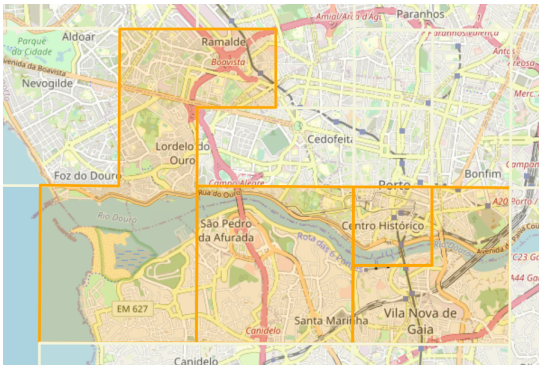


Figure 5.19. Micro districts with both sides of the river, in orange.

5.5.3 Nano districts

At the nano level, the most restrictive constraint for the workload is time. The model produced 257 different nano districts. Appendix B shows the representation of the created micro districts. On average, a micro district is splitted into 5.7 nano districts ($\sigma = 1.99$). The minimum number of nano districts per micro district is 2 and the maximum number is 10.

In terms of the workload factor, it ranges from the minimum 0.7 up to 0.99. The mean value is 0.75 with a standard deviation of 0.11. As already stated, T_{MAX} for the nano district is $\frac{8 \times 3600 \times 5}{6}$, the weekly working time per micro district divided by 6. Considering this, the weekly working time in the nano districts is on average $0.75 \times T_{MAX}$, e.g., 5 hours.

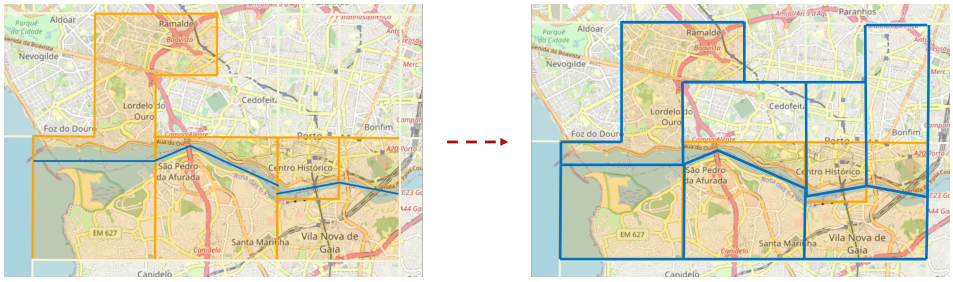


Figure 5.20. Micro districts with the river as border.

5.5.4 Approximation to road network

As shown in the visual representation of the districts created, the district edges don't fit the road network. It would be valuable for them to fit, as it would make the solution more practical. Although this element wasn't fully developed, due to time limitations, tests were carried on how to approximate the districts (or polygons) edges, to edges on the road network.

The pseudocode for the test procedure developed for the approximation to road network is available in 5.3. It takes as an input P, the set of polygons representing the districts and retrieves P_{road} the set of polygons/districts in the road network. Each polygon in P is written as a set of ordered vertices.

The algorithm starts by iterating over all the polygons in P. For each polygon, its bounding box is created, using the function osmnx.graph.graph_from_bbox (Boeing, 2017), a buffer of 200 meters is used. Then, each edge on the polygon is interpolated, retrieving 10 ($n = 10$) points. The nearest edge to each interpolated point is identified using the function osmnx.distance.nearest_edges (Boeing, 2017). The nearest edge to the point is added to the set P_{road}. The output is a set of road edges.

As already pointed, this is only a test procedure, e.g., it does not work perfectly and should be improved in future research. The result of this approximation is shown in figure 5.21, as it can be noted the solution requires improvements as there are gaps between polygons, as well as overlaps.

5.5.5 Daily analysis and comparison with company's solution

As already explained, the goal for the micro districts is for them to represent a route performed by a single driver (and a single vehicle), so the workload for a micro district should be smaller than eight hours. The model was parameterized to handle this (by defining $T_{MAX} = 8 \times 5.5 \times 3600$). But because weekly data was used, it is important to check if it matches to the daily workload.

To do so, one should use the daily dataset of performed deliveries and calculate

Algorithm 5.3 Approximating polygons to road network

```
1: Input: P
2: Output: P_road
3:
4: procedure To road network(P)
5:     for polygon ∈ P do
6:         north ← max(polygon_lat)
7:         south ← min(polygon_lat)
8:         east ← max(polygon_lon)
9:         west ← min(polygon_lon)
10:        G ← graph_from_bbox(north, south, east, west)
11:        for edge ∈ polygon do
12:            points ← interpolate(edge, n = 10)
13:            for p ∈ points do
14:                nearest_edge ← nearest_edge(G, p)
15:                P_road.append(nearest_edge)
16:            end for
17:        end for
18:    end for
19: end procedure
```

the workload if the districting solution was used. In that case, each customer is assigned to a district, depending on its geographical location. All the customers within a district will be served by the same driver, forming a route.

To estimate the workload per district in the districting solution, one should consider the travelled time from the depot to the district center, the routing time inside the district and the total service time off all the customers within the district. Equation 5.25 can be used to calculate the workload for district i, $cost_{depot,i}$ represents the distance from the depot to the district center, n_i is the number of clients in the district, A_i is the area of the district, S_i the sum of the service time for all customers in a district. Thus, $workload_i$, is the working shift for a district, on a given day.

$$Workload_i = \frac{2cost_{depot,i}}{v_L} + \frac{\beta_2(\sqrt{n_i A_i})}{v_U} + S_i \tag{5.25}$$

Despite the fact the company doesn't use a district solution, one can make a comparison by interpreting the different circuits as micro districts, since a circuit is performed by a single driver (and vehicle). Therefore, one can compute the convex hull of the stops in a circuit to determine the approximate structure of the circuit (if it was a district). Figure 5.22 presents the procedure for the circuit LABO, on day 03/08/2021, the area in blue is the convex hull, e.g., the smallest convex polygon that encloses all the stops of the circuit. This polygon can be compared to the micro districts generated by the model. And so, it is possible to compute the workload, for the company's solution, using a similar approach to **??**, with c defining a circuit and $center_c$ the center of c, n_c the number of stops in circuit c, A_c the are of the convex hull defined by the stops in c and S_c the total service time of all the stops in c.

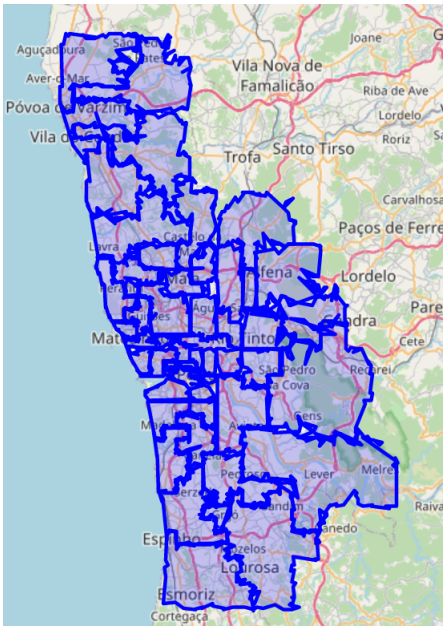


Figure 5.21. Results of the test procedure for district approximation to the road network.

$$Workload_c = \frac{2dist_{depot,center_c}}{v_L} + \frac{\beta_2(\sqrt{n_c A_c})}{v_U} + S_c \tag{5.26}$$

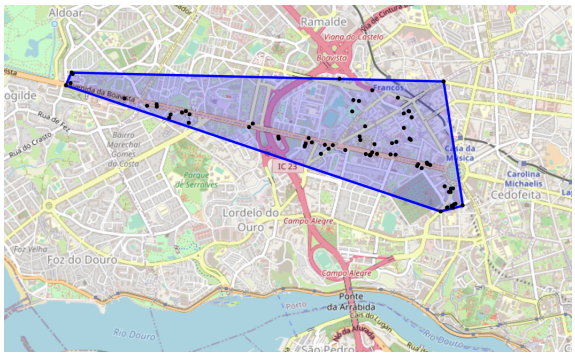


Figure 5.22. The convex hull for all the stops in the circuit LABO, on 03/08/2021.

Figure 5.23 shows the area each circuit covers, on day 03/08/2021, each color represents a different circuit, there is an overlap between the geographical areas of different circuits, specially in central Porto area.

To compare the two solutions (the districting and the company's), one can set the service time per stop as the median service time, 311.04 seconds and use equations 5.25 and 5.26 to compare the workload for all the data in the dataset. Table 5.6 shows measures of central tendency and dispersion for the workload, in hours, per route.

For a six month sample (131 days), the districting solution needs a total of 6282

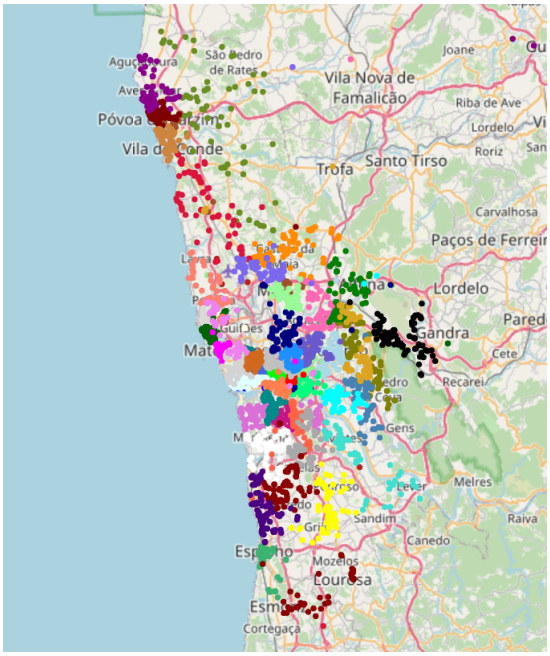


Figure 5.23. The area covered by each circuit, on 03/08/2021, and the demand level on each circuit, each color represents a different circuit.

routes (e.g., 6282 vehicles and drivers), while the current solution uses 7916 routes (+1634 in a six month period, or +12.5 routes, daily). On average, a route in the districting solution takes 7.046 working hour, while in the current solution it takes 6.174 hours. For the district solution 50 % of the routes take less than 6.859 hours, on the other hand, in the current solution in 50 % of the routes, a working shift is lower than 6.540 hours.

Table 5.6. Measures of central tendency and dispersion for the workload, in hours, per route.

Measure	**Districting Solution** ($n = 6282$)	**Current Solution** ($n = 7916$)
Mean	7.046	6.174
Median	6.859	6.540
Max	20.329	16.953
Min	0.512	0.157
1st quartile	5.384	5.222
3rd quartile	8.494	7.577
Range	19.817	16.797
IQR	2.354	2.355
Std	2.394	2.257

Figure 5.24 shows the boxplot representing the workload distribution for the districting and the current solution. The values between the first and third quartile

are similar, but the range is greater for the districting solution, both have upper and lower outliers, indicating days with a higher demand level (like days before Christmas or black friday) and days with lower demand level (like public holidays where only a couple of routes are performed).

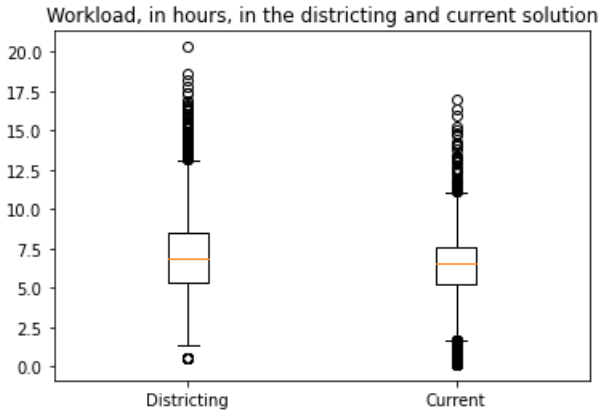


Figure 5.24. Boxplot with the workload, in hours, in the districting and current solution.

Another relevant statistic is the number of routes that need more than an 8 hour shift. In the districting solution, there are 1980 routes with a workload level above 8 hours, while on the company's solution there are 1388, a difference of 592.

After analysing this results, it seems that the current solution used by the company is adapted daily, meaning that, the circuits are designed to fit the realized demand, e.g., in high demand days, there are more circuits covering a smaller area; and on lower demand days the opposite. This allows keeping the working hours' variability lower.

However, the company's solution uses daily approximately 12.5 more routes than the districting solution. This represents a cost of using more 12.5 vehicles and 12.5 drivers. And, with this difference, the company's solution is only able to reduce the number of daily overworked routes in 4.5. In fact, the districting solution was designed for a median demand week, if a minimum (or maximum) was used, the solution would be different. Considering this, one can state that the districting solution is more adequate for median demand weeks (as it was modeled for it), and in other circumstances, the solution should be adjusted.

To understand the performance of the districting solution for the type of days, in terms of demand, it was modelled for, e.g., days with a demand level near the median; the analysis was also carried using a sample with only data from weeks with a demand level (in terms of number of stops) between the first and third quartile. Figure 5.25 points out the weeks used to carry this analisys, the bars in medium blue represent demand levels within the first and third quartile, while the bar in dark

blue represents the median demand (e.g., the week used to design the districting solution).

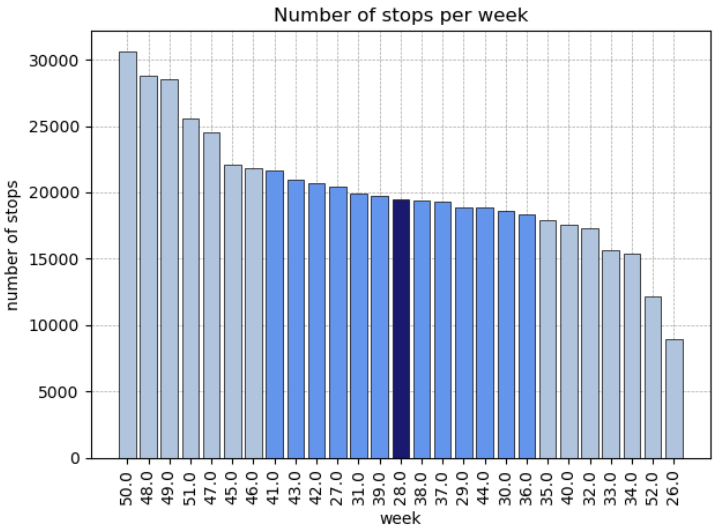


Figure 5.25. Bar plot with the number of stops per week, bars in medium blue are within the first and third quartile, bar in dark blue is the week with median demand.

Using the daily data from the weeks signaled in figure 5.25, the workload, or the number of hours needed to serve the customers, can be calculated for the districting solution, and for the company's solution. The box plot in figure 5.26 and table 5.7 can be used two compare the two approaches, when the volume of demand is median or nearly median. The number of days in the sample is 64 and, to cope with demand, the districting solution uses a total of 3136 routes, while the current solution uses 3672. Moreover, the total working time in the districting solution is 21440.6 h, and in the current solution it is 23371.5 h, which represents a difference of 1930 hours (in a 64 day period). This indicates the districting solution as a better solution.

On the other hand, the variation coefficient ($\frac{\mu}{\sigma} \cdot 100$) is 29.5% in the districting solution and 36.3% in the current solution, which points that the districting solution created more balanced routes (less variability).

On average a route in the districting solution takes 6.837 hours, while on the current solution it takes 6.365 hours (28 minutes less). There are 778 routes in the current solution with a working shift above the 8 hours limit, versus 883 routes in the districting solution.

Comparing the number of stops per route, in the two approaches; in the districting solution, a route (or a district) serves, on average, 71 stops (with $\sigma = 22$), while on the current solution it serves on average 63 (with $\sigma = 24.8$).

Overall, all this parameters point to the districting solution as a better solution, allowing a daily decrease in the number of vehicles and drivers of 8.4.

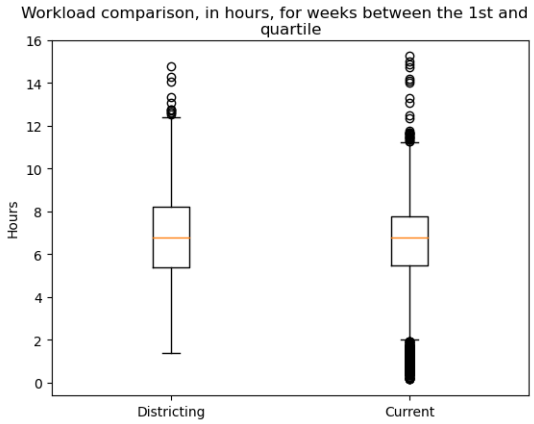


Figure 5.26. Box plot with the workload, in hours, in the districting and current solution, using a sample with only days from weeks in between the 1st and 3rd quartile of demand.

Table 5.7. Measures of central tendency and dispersion for the workload, in hours, per route (sample between 1st and 3rd quartile).

Measure	**Districting Solution** ($n = 3136$)	**Current Solution** ($n = 3672$)
Mean	6.837	6.365
Median	6.779	6.795
Max	14.790	15.257
Min	1.406	0.156
1st quartile	5.383	5.477
3rd quartile	8.234	7.788
Range	13.384	15.102
IQR	2.851	2.311
Std	2.023	2.308

However, this solution isn't as adequate for higher, or lower, demand levels as it as already pointed out. Looking at figure 5.24, most of the outliers are above the upper limit of the distribution of working hours per district (visible in figure 5.24). This indicates that the problem with adjusting the solution to days with demand levels distanced from the median demand level is more relevant on higher demand days (compared to lower demand days).

The hierarchical structure of the districting problem is useful to handle this type of variability. In fact, in higher demand level days, a driver/vehicle can be assigned to one or two nano districts instead of a whole micro district. This type of adjustment can be done by planners but could be also a topic of further development.

A visual tool, shown in figure 5.27, was also developed, in order to help planners performing adjustments in high demand days. With this, a planner can realize which micro districts are critical (in red), e.g., districts with working time above the 8 hours limit; which have a average demand level, with a working time between 6 and

8 hours; and which have low demand, with a working time below 6. When clicking in the micro district, a pop up with a estimation of the working time in the district appears. It is also possible to use the layer control, to visual the region at the nano district level and analyse the demand level (in working times) per nano district. Using this, a planner can make a decision supported by data, to readjust a micro district for a certain realized demand.

The data used for plotting figure 5.27, represents a day within the maximum demand week. As expected, most of the micro districts are overworked (they exceed the 8 hours limit), because they are designed for a median demand week. When clicking in macro district BE, the planner can check that the estimated workload for the district is 7.2 hours. The planner can also check, the workload in nano districtcs and, for instance, divide an high demand micro district in two.

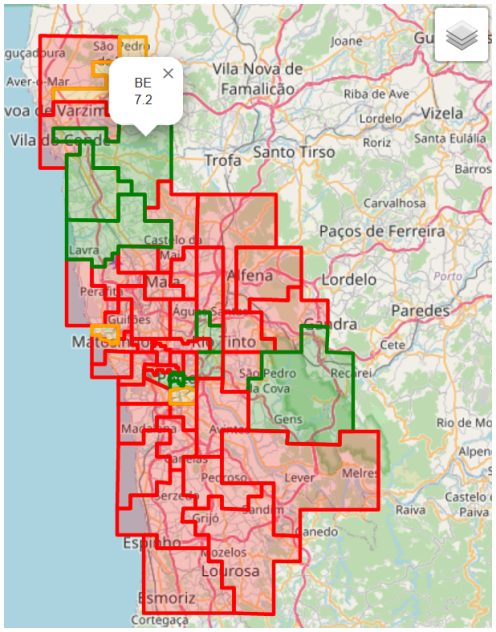


Figure 5.27. Visual tool with the districts and its demand level, red if higher than 8 hours, orange if between 6 and 8 hours and green if below 6 hours..

Chapter 6

Nano districts sequencing

This chapter focuses on the procedure developed for the sequencing of nano districts. Section 6.1 depicts the notation used, section 6.2 describes what are transition probabilities and how are they relevant and section 6.3 describes the procedure and states the mathematical formulation of the problem.

6.1 Formalization

First, one must define the objective and data structures. Formally, there is a set of stops S to serve on a given day, t. Each stop $s \in S$ sits within a micro district m. $S_m^t = \{0, 1...|S_m^t|\}$ is a set defining all the stops in m, in a day t, 0 represents the depot. N_m^t defines the set of unique nano zones to serve in micro district m, on day t. Then, one defines $\mathcal{X}_m^t$ as all the feasible sequences for N_m^t, visiting all nano districts once only.

For learning preferences from past data, one uses the dataset $\mathcal{H} = \{x^t\}_{t=1}^{t=T}$, corresponding to all the route effectively performed, in terms of nano zones.

6.2 Transition Probabilities

To model the preference of drivers when sequencing the nano districts one uses transitions probabilities, assigning probabilities to all the arcs in the network. Formally we estimate $Pr(i|j)$, denoting the probability of the next stop being i, conditional on the current stop j.

6.3 Maximum likelihood routing

To sequence the nano districts (or constructing the routes), one aims to search for the most *likely* routing solution. Formally, this means,

$$\max_{x \in \mathcal{X}_m^t} \prod_{(i \to j) \in x} Pr(i|j) \cdot x_{ij} \tag{6.1}$$

To transform the product in the objective function into a sum, log probabilities are considered in the objective function, to minimize the following:

$$\min_{x \in \mathcal{X}_m^t} \sum - logPr(i|j) \cdot x_{ij} \tag{6.2}$$

Using statistics theory, $Pr(i|j)$ can be obtained by the following,

$$Pr(i|j) = \frac{Pr(i \to j)}{Pr(j)} \tag{6.3}$$

Using the dataset $\mathcal{H}$, $Pr(i \to j)$ is the frequency of a transition $i \to j$, e.g., the number of times, in all the routes in $\mathcal{H}$, the driver goes from nano zone i to nano zone j. On the other hand, $Pr(j)$ is the frequency of being in nano zone j.

Then, the problem can be modeled as a TSP, with 6.2 as an objective function, as follows,

$$x_{ij} = \begin{cases} 1, & \text{if the path goes from stop ito stop j} \\ 0, & \text{otherwise} \end{cases} \tag{6.4}$$

$$\min_{x \in \mathcal{X}_m^t} \quad \sum - logPr(i|j) \cdot x_{ij} \tag{6.5}$$

$$\text{s.t.} \quad \sum_{i=1, i \neq j}^{n} x_{ij} = 1 \qquad \forall j \in N \tag{6.6}$$

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1 \qquad \forall i \in N \tag{6.7}$$

$$x_{ij} \in \{0, 1\} \tag{6.8}$$

Figure 6.3 shows the routing process. In order to construct the routes to serve all the stops on a given day t, the process takes the micro and nano districts and all the stops in t an input. Then, each stop is assigned to the micro and nano district it belongs to, depending on its geographical location. All the stops within micro district belong to the same route, e.g., they define S_m^t, then, it is possible to define the set of unique nano zones, N_m^t the route needs to serve.

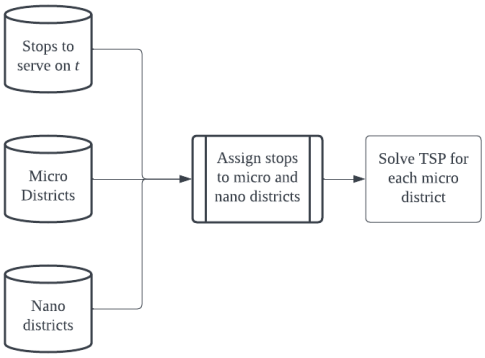


Figure 6.1. Flowchart of the routing process.

	depot	IN	IQ	IP	IR	IO
depot	1000	7.091	6.734	4.747	5.779	6.958
IN	5.675	0.674	2.655	1.941	3.175	7.618
IQ	3.403	2.496	0.633	4.022	5.854	1.816
IP	5.519	1.871	2.931	0.659	2.653	3.833
IR	3.653	3.534	5.749	2.771	0.546	5.932
IO	4.417	7.754	2.533	4.016	6.144	0.285

Figure 6.2. Cost matrix, each entry states the probability of going to i, coditional on being in j.

Table 6.1 shows exemplifies this, t and m are the tuple that identify a routing instance, S_m^t is the set of stops to serve, each stop is assigned to a nano district, and N_m^t is the set of unique nano districts to serve in (t, m), as well as the depot. The TSP solution retrieves the maximum likelihood sequence for N_m^t.

Table 6.1. Example: S_m^t and N_m^t.

t	m	S_m^t	N_m^t
20211230	AZ	$\{0, 1, 2, \ldots, 69\}$	{depot, IN, IQ, IP, IR, IO}

The log probabilities matrix of this routing instance are shown in figure 6.2, the rows represent the origin and the destination, the entry with $row = IN$ and $column = IN$, represents the log probability of visiting the depot next, condition on being at nano zone IN.

To solve the instance of the TSP, the Lin-Kernighan heuristic is used, using the package *lkh* (Hudson, 2022) in Spyder environment. The solution to the TSP is the

following nano zone sequence $\{depot, IO, IN, IQ, IR\}$, with an objective function of 18.59 (the summed log probabilities), the result is also represented in figure .

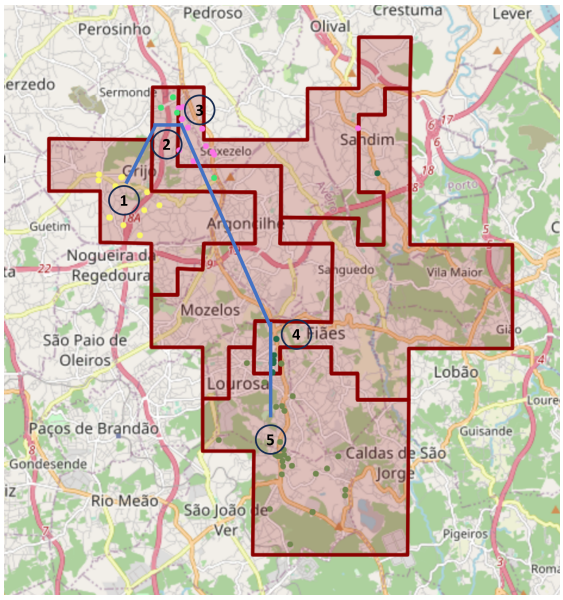


Figure 6.3. Sequence solution for TSP instance, $t = 20211230$ and $m = AZ$.

6.4 Comparing with distance based TSP

For testing purposes, it is possible to compare the transition probabilities based TSP with a distance based TSP. Using a test sample $\mathcal{H}_{test} = \{x^t\}^{t \in T}$, with T being the set of days in the testing set.

Using the transitions probabilities based TSP, the cost function in the objective function is 6.5; on the other hand, in the distance based, the cost function is the euclidean distance between zones. To compute distance between zones, the zone center must be defined. The zone center is nearest stop to the zone centroid.

In fact, in the transition probabilities based TSP it isn't expected for the distance to be minimized. On average the total distance of the transition probabilities based TSP solution is 11.1% higher, when compared with the distance based TSP. Figure 6.4 shows the histogram representing the distribution of percentage difference between transition based TSP and distance based TSP.

As expected, the distance in the transition based TSP is almost always greater (at minimum it is equal). When deciding over a route sequence, a driver tends to consider different parameters beyond minimizing distance, he considers traffic, parking conditions, customers preferences, among others. Thus, learning drivers' preferences does not translate into learning the minimum distance.

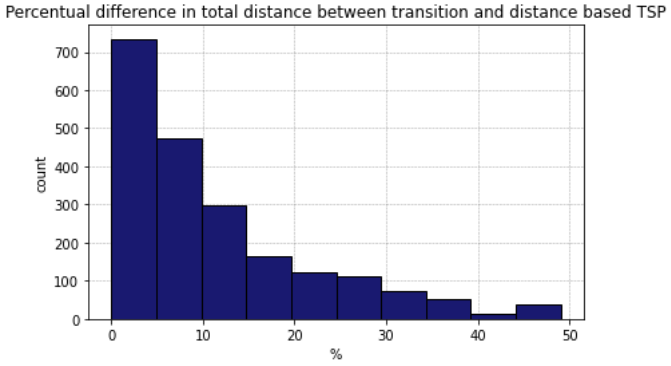


Figure 6.4. Difference (in percentage) between the total distance of the transition based TSP solution and distance based TSP solution.

Chapter 7

Conclusion

7.1 Final conclusions

This dissertation aims to fill a gap in the literature, by developing a novel approach for the last mile problem, trying to achieve consistency and inferring preferences from drivers. The dissertation uses the case study of a parcel delivery company operation in Portugal, using a dataset with data from performed deliveries in Porto's district.

The literature review highlighted that drivers tend learn on the zone level, rather than on the customer level. To create this learning zones, a recursive districting problem was modeled, with three hierarchical levels of districting (macro, micro and nano districts).

The micro districts represent a route and they are modelled so that each micro district has a demand level from 5.6 to 8 hours, considering a day with median demand. Comparing the districting solution to the company's approach for the package-route assignment, the districting solution designs more balanced routes (the variability coefficient for the workload per route is lower in the districting solution), using a lower number of drivers and vehicles. Moreover, using the districting solution allows achieving *consistency*, by having drivers repeatedly assigned to the same micro district.

One disadvantage of the districting solution (vs. company's approach) is not adjusting to high demand level days, in this type of days, most of the districts become overworked. This is explained because the problem was modelled using data from a median demand week and the solution is stationary, there will always be a maximum of 45 routes per day (because there are 45 micro districts). To handle this downside the districting hierarchical structure can be taken advantage of, in fact, in higher demand days, a driver can be assigned to one or two nano districts, instead of a whole micro district; this readjustments must be carried by planners

and a visual support tool was developed to help the procedure.

To incorporate driver's knowledge and identify the *natural* sequence of nano zones, the transition probabilities between nano zones were determinate. Then, a TSP is solved, with the objective function of minimizing log transition probabilities. Thus, the natural sequence of zones is *inferred*.

7.2 Limitations and future research

The main limitation on this work relates to the use of real life data, the data cleaning process took a lot of time that could have been use for the actual development of the solution. On the other hand, it would be more accurate to use road distance, instead of euclidean, in the district design phase; but obtaining all the distances was not possible, as it was time/memory consuming. Moreover, the mathematical model was originally developed for computer generated instances, with equally space distributed demand, but the real life data is far from that. In the real life data the geographical distribution of demand is highly asymmetrical (there are high demand focus like central Porto, and further areas with little to no variables), thus, it was needed to adjust the model, making the parameterization more complex and causing the need for more iterations, until a feasible solution is reached. The combination of this aspects made the districting phase take much more time than what it was initially expected.

Regarding future research, for the districting it would be important to develop a proper method for the readjustment of districts for high fluctuations in demand and the approximation of the district polygons to the road network, e.g., turning the polygons' edges into streets/roads on the road network. On the other hand, the impact of using road distance (instead of euclidean) could be studied. Furthermore, the use of different shape polygons, instead of rectangles, as basic units should be explored, using rectangles limits the constraint on connectivity, since a rectangle, at max, has 4 adjacent rectangles; if an hexagon was used as a basic unit shape the feasible solution set would increase, possibly retrieving better solutions.

Because the districting phase took more time than expected, the sequencing procedure wasn't as developed as desired. Considering this, future research should focus on how to improve the procedure, by including distance between zones in the cost function of the transition based TSP, or using more complex methods like deep learning, to include the different attributes on the dataset. One could also test using multi level transition probabilities, e.g., instead of estimating $Pr(i|j)$, estimating $Pr(i|j,k)$, storing more information about the performed sequences.

Appendix A

Micro districts created

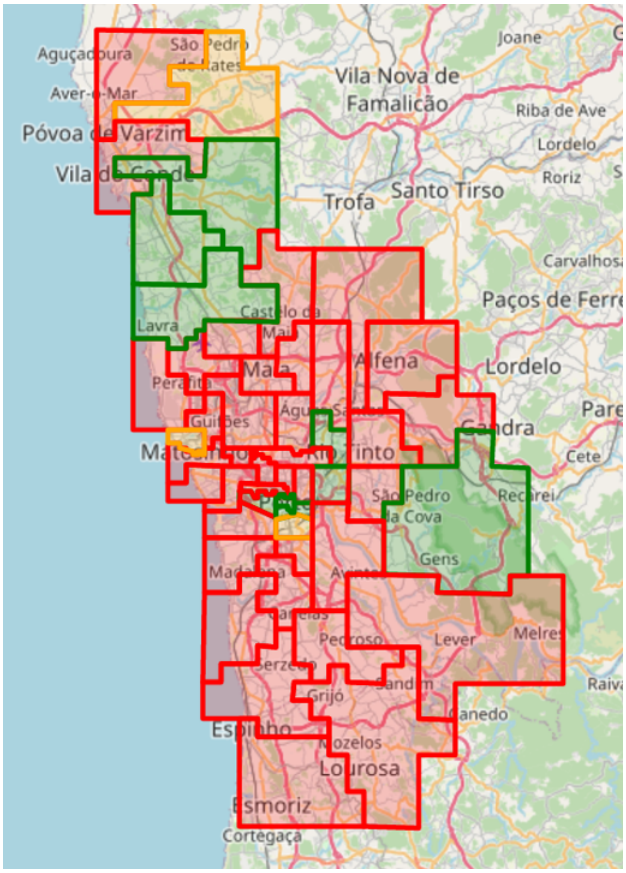


Figure A.1. Micro districts created

Appendix B

Nano districts created

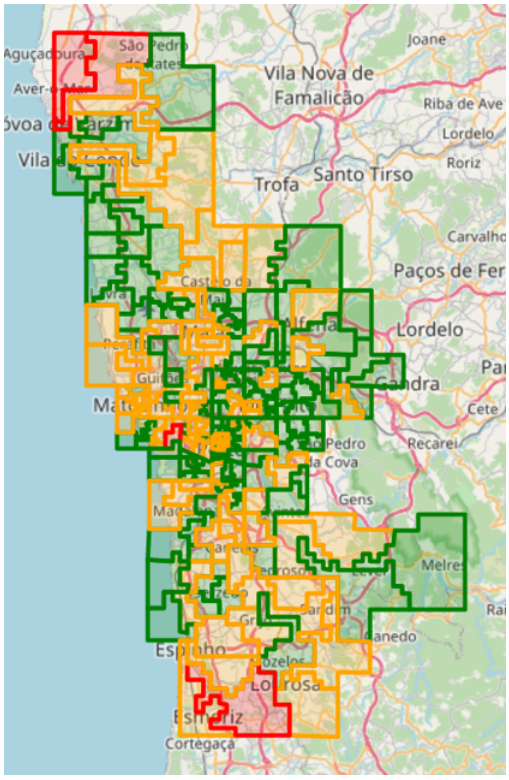


Figure B.1. Nano districts created

Bibliography

(2021). Amazon last-mile routing research challenge | supported by the mit center for transportation logistics.

Applegate, D. L., Bixby, R. E., Chvatál, V., and Cook, W. J. (2006). *The Traveling Salesman Problem: A Computational Study*. Princeton University Press.

Archetti, C., Speranza, M. G., and Hertz, A. (2006). A tabu search algorithm for the split delivery vehicle routing problem. *https://doi.org/10.1287/trsc.1040.0103*, 40:64–73.

Bai, R., Chen, X., Chen, Z.-L., Cui, T., Gong, S., He, W., Jiang, X., Jin, H., Jin, J., Kendall, G., Li, J., Lu, Z., Ren, J., Weng, P., Xue, N., and Zhang, H. (2021). Analytics and machine learning in vehicle routing research. *International Journal of Production Research*.

Barnett, V., Lewis, T., et al. (1994). *Outliers in statistical data*, volume 3. Wiley New York.

Barreto, S., Ferreira, C., Paixão, J., and Santos, B. S. (2007). Using clustering analysis in a capacitated location-routing problem. *European Journal of Operational Research*, 179:968–977.

Battarra, M., Erdoğan, G., and Vigo, D. (2014). Exact algorithms for the clustered vehicle routing problem. *Operations Research*, 62:58–71.

Bertsimas, D. J. (1992). A vehicle routing problem with stochastic demand. *https://doi.org/10.1287/opre.40.3.574*, 40:574–585.

Bieniek, T. (2022). utm.

Boeing, G. (2017). Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems*, 65:126–139.

Braekers, K., Ramaekers, K., and Nieuwenhuyse, I. V. (2016). The vehicle routing problem: State of the art classification and review. *Computers and Industrial Engineering*, 99:300–313.

Brownlee, J. (2020). *Data Preparation for Machine Learning - Data Cleaning, Feature Selection, and Data.* machine learning mastery.

Caixeta, M. C. B. F. and Fabricio, M. F. (2018). Métodos e instrumentos de apoio ao codesign no processo de projeto de edifícios. *Ambiente Construído*, 18:111–131.

Canoy, R., Bucarey, V., Molenbruch, Y., Mulamba, M., Mandi, J., and Guns, T. (2022). Probability estimation and structured output prediction for learning preferences in last mile delivery.

Cattaruzza, D., Absi, N., Feillet, D., and Vidal, T. (2014). A memetic algorithm for the multi trip vehicle routing problem. *European Journal of Operational Research*, 236:833–848.

Chao, I. M. (2002). A tabu search method for the truck and trailer routing problem. *Computers Operations Research*, 29:33–51.

Chen, L., Chen, Y., and Langevin, A. (2021). An inverse optimization approach for a capacitated vehicle routing problem. *European Journal of Operational Research*, 295:1087–1098.

Chu, H., Zhang, W., Bai, P., and Chen, Y. (2021). Data-driven optimization for last-mile delivery. *Complex and Intelligent Systems.*

Clarke, G. and Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *https://doi.org/10.1287/opre.12.4.568*, 12:568–581.

Cook, W. and Held, S. (2022). Constrained local search for last-mile routing. pages 1–19.

Cramer-Flood, E. (2022). Worldwide ecommerce forecast update 2022.

da Costa, P. R. O., Mauceri, S., Carroll, P., and Pallonetto, F. (2018). A genetic algorithm for a green vehicle routing problem. *Electronic Notes in Discrete Mathematics*, 64:65–74.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6:80–91. doi: 10.1287/mnsc.6.1.80.

Darmian, S. M., Fattahi, M., and Keyvanshokooh, E. (2021a). An optimization-based approach for the healthcare districting under uncertainty. *Computers Operations Research*, 135:105425.

Darmian, S. M., Fattahi, M., and Keyvanshokooh, E. (2021b). An optimization-based approach for the healthcare districting under uncertainty. *Computers and Operations Research*, 135.

Defryn, C. and Sörensen, K. (2017). A fast two-level variable neighborhood search for the clustered vehicle routing problem. *Computers and Operations Research*, 83:78–94.

Demir, E., Syntetos, A., and Woensel, T. V. (2022). Last mile logistics: Research trends and needs. *IMA Journal of Management Mathematics*, 33:549–561.

Donati, A. V., Montemanni, R., Casagrande, N., Rizzoli, A. E., and Gambardella, L. M. (2008). Time dependent vehicle routing problem with a multi ant colony system. *European Journal of Operational Research*, 185:1174–1191.

Expósito-Izquierdo, C., Rossi, A., and Sevaux, M. (2016). A two-level solution approach to solve the clustered capacitated vehicle routing problem. *Computers and Industrial Engineering*, 91:274–289.

Fellus, L. (2022). *Districting Optimization for Last-Mile Routing in Urban Contexts*. PhD thesis.

Ferland, J. A. and Guenette, G. (1990). Decision support system for the school districting problem. *https://doi.org/10.1287/opre.38.1.15*, 38:15–21.

Ghosh, M., Mahes, R., Maragno, D., and Kuiper, A. (2021). Learn global and optimize local: A data-driven methodology for last-mile routing. pages 1–29.

Goel, A. and Gruhn, V. (2008). A general vehicle routing problem. *European Journal of Operational Research*, 191:650–660.

Ho, W., Ho, G. T., Ji, P., and Lau, H. C. (2008). A hybrid genetic algorithm for the multi-depot vehicle routing problem. *Engineering Applications of Artificial Intelligence*, 21:548–557.

Holland, C., Levis, J., Nuggehalli, R., Santilli, B., and Winters, J. (2017). Ups optimizes delivery routes. *Interfaces*, 47:8–23.

Hudson, B. (2022). lkh.

Jarrah, A. I. and Bard, J. F. (2012). Large-scale pickup and delivery work area design. *Computers Operations Research*, 39:3102–3118.

Kalcsics, J. and Ríos-Mercado, R. Z. (2019). Districting problems. *Location Science*, pages 705–743.

Kim, B. I., Kim, S., and Sahoo, S. (2006). Waste collection vehicle routing problem with time windows. *Computers Operations Research*, 33:3624–3642.

Konstantakopoulos, G. D., Gayialis, S. P., and Kechagias, E. P. (2022). Vehicle routing problem and related algorithms for logistics distribution: a literature review and classification. *Operational Research*, 22:2033–2062.

Lei, H., Laporte, G., Liu, Y., and Zhang, T. (2015). Dynamic design of sales territories. *Computers and Operations Research*, 56:84–92.

Li, F., Golden, B., and Wasil, E. (2007). A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Computers Operations Research*, 34:2734–2742.

Li, X., Yuan, M., Chen, D., Yao, J., and Zeng, J. (2018). A data-driven three-layer algorithm for split delivery vehicle routing problem with 3d container loading constraint. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 528–536.

Madhulatha, T. S. (2012). An overview on clustering methods. *IOSR Journal of Engineering*, 2.

Mandi, J., Canoy, R., Bucarey, V., and Guns, T. (2021). Data driven vrp: A neural network model to learn hidden preferences for vrp.

Mayorga, M. E., Bandara, D., and McLay, L. A. (2013). Districting and dispatching policies for emergency medical service systems to improve patient survival. *http://dx.doi.org/10.1080/19488300.2012.762437*, 3:39–56.

Microsoft, Brundritt, R., Munk, S., Lindeman, E., and French, C. (2022). Find a location by address - bing maps.

Montemanni, R., Gambardella, L. M., Rizzoli, A. E., and Donati, A. V. (2005). Ant colony system for a dynamic vehicle routing problem. *Journal of Combinatorial Optimization*, 10:327–343.

Montoya, A., Guéret, C., Mendoza, J. E., and Villegas, J. G. (2017). The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological*, 103:87–110.

Norouzi, N., Sadegh-Amalnick, M., and Alinaghiyan, M. (2015). Evaluating of the particle swarm optimization in a periodic vehicle routing problem. *Measurement*, 62:162–169.

Orgaz, G. B., Barrero, D. F., R-Moreno, M. D., and Camacho, D. (2015). Acquisition of business intelligence from human experience in route planning. *Enterprise Information Systems*, 9:303–323.

Pop, P. C., Fuksz, L., Marc, A. H., and Sabo, C. (2017). A novel two-level optimization approach for clustered vehicle routing problem.

Ricca, F., Scozzari, A., and Simeone, B. (2013). Political districting: From classical models to recent approaches. *Annals of Operations Research*, 204:271–299.

Salazar-Aguilar, M. A., Ríos-Mercado, R. Z., and González-Velarde, J. L. (2011). A bi-objective programming model for designing compact and balanced territories in commercial districting. *Transportation Research Part C: Emerging Technologies*, 19:885–895.

Subramanian, A., Drummond, L. M., Bentes, C., Ochi, L. S., and Farias, R. (2010). A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers Operations Research*, 37:1899–1911.

Tan, S. Y. and Yeh, W. C. (2021). The vehicle routing problem: State-of-the-art classification and review. *Applied Sciences (Switzerland)*, 11.

Toth, P. and Vigo, D. (1997). An exact algorithm for the vehicle routing problem with backhauls. *https://doi.org/10.1287/trsc.31.4.372*, 31:372–385.

Toth, P. and Vigo, D. (2002). An overview of vehicle routing problems. pages 1–26.

van Beek, P. (2022). Learning drivers' preferences in delivery route planning.

Vidal, T., Laporte, G., and Matl, P. (2020). A concise guide to existing and emerging vehicle routing problem variants. *European Journal of Operational Research*, 286:401–416.

Wu, C., Song, Y., March, V., and Duthie, E. (2022). Learning from drivers to tackle the amazon last mile routing research challenge. pages 1–12.

Xiao, Y., Zhao, Q., Kaku, I., and Xu, Y. (2012). Development of a fuel consumption optimization model for the capacitated vehicle routing problem. *Computers Operations Research*, 39:1419–1431.

Xu, Z. and Cai, Y. (2018). Variable neighborhood search for consistent vehicle routing problem. *Expert Systems with Applications*, 113:66–76.

Yu, V. F., Redi, A. A., Hidayat, Y. A., and Wibowo, O. J. (2017). A simulated annealing heuristic for the hybrid vehicle routing problem. *Applied Soft Computing*, 53:119–132.

Zachariadis, E. E. and Kiranoudis, C. T. (2010). An open vehicle routing problem metaheuristic for examining wide solution neighborhoods. *Computers Operations Research*, 37:712–723.

Zachariadis, E. E., Tarantilis, C. D., and Kiranoudis, C. T. (2016). The vehicle routing problem with simultaneous pick-ups and deliveries and two-dimensional loading constraints. *European Journal of Operational Research*, 251:369–386.

Zimek, A. and Filzmoser, P. (2018). There and back again: Outlier detection between statistical reasoning and data mining algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8.

Özarık, S. S., da Costa, P., and Florio, A. M. (2022). Machine learning for data-driven last-mile delivery optimization. *SSRN Electronic Journal*.

Özdamar, L. and Demir, O. (2012). A hierarchical clustering and routing procedure for large scale disaster relief logistics planning. *Transportation Research Part E: Logistics and Transportation Review*, 48:591–602.