FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Sectorization for last-mile delivery

Gustavo Macedo Torres

Mestrado em Engenharia Informática e Computação

Supervisor: Tânia Fontes

Second Supervisor: José Soeiro Ferreira

July 31, 2022

Sectorization for last-mile delivery

Gustavo Macedo Torres

Mestrado em Engenharia Informática e Computação

July 31, 2022

Abstract

The world is increasingly connected, which leads to the growth of logistics markets to meet our needs for the distribution of goods in the quantity and time demanded. Last-mile logistic companies are responsible for ensuring that their clients receive the goods in a timely and cost/resource-effective way. Driver's areas of responsibility, also called sectors, must be designated, on which parcels are organized and delivered.

The geographic space tends to be of large dimension, and many parcels are delivered daily. Sectors must be robust to adapt to the daily demand while optimizing the company's criteria, from compactness of the sectors, packages delivered, the balance of work and sectors' dissimilarity on different days.

This dissertation considers a multi-objective Sectorization and Routing problem over several time frames, in which small geographic areas, postal codes, are allocated to define the driver's sectors, on where the delivery route is defined. There are many criteria to optimize simultaneously, and the company's decision-maker should be able to choose the best fit allocation.

The problem is modeled, allowing for the generation of problem-specific data, used as input for the proposed solution, based on a real-world case. Sectorization with routing heuristics is employed to solve the definition of the sectors in a two-phase approach, where the allocation is made in the first stage, followed by the daily generation. A Preference-Inspired Evolutionary Algorithm with Goal vectors using Mating Restriction, PICEA-g-mr, is developed to solve the multi-objective optimization problem, together with representation-specific genetic operators. A visualization tool for helping the decision-maker choose the best fit allocation is described.

The algorithm is tested against large-scale real-world-based data, and its behaviour is analysed. The results confirm that the proposed solution can solve the problem under study and generate a set of feasible good-quality allocations from which the decision-maker can choose using the developed visualization tool.

Keywords: Sectorization, Multi-Objective Optimization, Last-mile Logistics, Routing, Mating-Restriction, Co-Evolutionary Algorithm

Resumo

O mundo encontra-se cada vez mais conectado, o que origina um crescimento dos mercados logísticos de modo a preencher os nossos requisitos de distribuição de bens na quantidade e tempo exigidos. Empresas de entrega last-mile são responsáveis por garantir que os seus clientes recebem os bens de forma rápida e eficiente, a nível de custo e recursos. Áreas de responsabilidade dos condutores, também denominadas de setores, devem ser definidas, onde as entregas serão realizadas.

O espaço geográfico tende a ser de grandes dimensões e muitos pacotes são entregues diariamente. Os setores devem ser robustos para se adaptarem à procura diária, enquanto otimizam os objetivos definidos pela empresa, desde a compacidade dos setores, os pacotes entregues, o balanceamento do trabalho e a dissimilaridade em diferentes dias.

Esta dissertação considera um problema Multi-Objetivo de Sectorização com Roteamento em vários intervalos de tempo, em que pequenas áreas geográficas, códigos postais, são alocados de modo a definir as áreas de responsabilidade dos condutores, onde as rotas são criadas. Existe um conjunto de objetivos a minimizar em simultâneo, e o decisor da empresa deve ser capaz de escolher a alocação que melhor serve a empresa.

O problema é modelado, permitindo a geração de dados específicos para o problema, que são usados como input para a solução proposta, baseado num caso do mundo real. Sectorização com heurísticas de roteamento são aplicadas para resolver a definição dos setores, em duas fases. A alocação é feita inicialmente, seguida da geração para cada dia. O algoritmo Preference-Inspired Evolutionary Algorithm with Goal vectors using Mating Restriction, PICEA-g-mr, é desenvolvido para resolver o problema de otimização multi-objetivo, em conjunto com operadores genéticos específicos da representação. É descrevida uma ferramenta de visualização para ajudar o decisor a escolher a melhor alocação.

O algoritmo é testado em dados de grande dimensão representado um caso do mundo real, e o seu comportamento analisado. Os resultados confirmam que a solução proposta é eficaz em resolver o problema em estudo e gerar um conjunto de soluções admissíveis com boa qualidade, das quais o decisor pode escolher, com recurso à ferramenta de visualização.

Palavras-chave: Sectorização, Optimização Multi-objetivo, Logística last-mile, Roteamento, Mating-Restriction, Algoritmo Co-Evolucionário

Acknowledgments

Firstly, I would like to thank my supervisors, Researcher Tânia Fontes and Professor José Soeiro Ferreira, who provided me with guidance throughout this dissertation and whose feedback contributed to the success of this work. Additionally, I would like to thank Professor Ana Maria Rodrigues for the exceptional and tireless support, who followed this work closely throughout the whole process. Moreover, a thanks INESC TEC, especially to the project researchers, Joel Ribeiro, Pedro Rocha and Elif Ozturk, who provided invaluable advice and insights on each step of this work.

To my mother, I give my biggest and most hearth-warming word of appreciation, to the person who never left my side, who provided me with all the tools and lessons to be who and where I am today. To the person who always loved me unconditionally and inspires me. To my father, I give the heartiest thanks for always being present and always believing in me. For making me think about my decisions but always trusting and supporting me on every step I gave. I also leave a word of appreciation to my grandparents for all the love and support throughout my life. – Thank you for everything.

To my lovely girlfriend, Nádia, I give a touching thanks, for always reminding me that there is more to this than work. For never leaving my side, for all the uplifting moments and bringing joy into my life.

Lastly, to all my friends and people that made part of my academic journey, from the day it began, for all the help they gave me and for all the moments of fun and laughter.

To all the people who made these incredible 5 years.

Gustavo Torres

This work is financed by National Funds through the FCT—Fundacão para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within the projects e-LOG (EXPL/ECI-TRA/0679/2021).
This work is financed by the ERDF - European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project POCI-01-0145-FEDER-031671.

“Everything not saved,
will be lost.”

Nintendo "Quit screen"

Contents

List of Figures

List of Tables

Abbreviations

ALVS	Adaptive Large Neighborhood Search
AMP	Área Metropolitana do Porto
BU	Basic Unit
B2B	Business-to-Business
B2C	Business-to-Client
CC	Capacitated Clustering
CL	Candidate List
CP4	4-digit Postal Code
CP7	7-digit Postal Code
CSV	Comma-Separated-Values
CTDP	Commercial Territory Design Problem
CV	Coefficient of Variation
CVRP	Capacitated Vehicle Routing Problem
DT	Delaunay Triangulation
GRASP	Greddy Randomized Adaptive Search Procedure
GLS	Guided Local Search
HFVRP	Heterogeneous Fleet Vehicle Routing Problem
HHC	Home Health Care
HypE	Hypevolume-Indicator Estimation Algorithm
IP	Integer Programming
ILP	Integer Linear Programming
INE	Instituto Nacional de Estatística
KPI	Key Performance Indicator
MaOP	Many-Objective Problem
MIP	Mixed Integer Programming
MILP	Mixed Integer Linear Programming
MOEA	Multi-Objective Evolutionary Algorithm
MOP	Multi-Objective Problem
NSGA-II	Non-dominated Sorting Genetic Algorithm II
PICEA-g-mr	Preference-Inspired Evolutionary Algorithm with Goal vectors using Mating Restriction
RCL	Reduced Candidate List
SCVRP	Stochastic Capacitated Vehicle Routing Problem
TB	Tabu-Search
TDP	Territory Design Problem
VRP	Vehicle Routing Problem

Nomenclature

Ω	Solution Space
$b^k_{P_i}$	Set of BUs that belong to border level-k of P_i
b_d	Number of deliveries in BU b
$b.t$	Demographic type of BU b, urban or rural
$b.x$	x-coordinate of BU b
$b.y$	y-coordinate of BU b
$B_{i,j}$	Bézier Endpoint j for criteria i
$B^c_{i,j}$	Bézier Control point j for criteria i
C	List of Active Clients for each Working Day
C_d	Set of Active Clients for Day d
c_i	Active Client i
c_p	Number of packages for client c
$c(P_i)$	Center of P_i
$d_{i,j}$	Distance between i and j
D	Number of Working Days
$extra_t$	Extra working time
$f_i(x)$	Objective value of criteria i
I	Instance/Individual, Global Allocation and Allocations for the D days
L	Number of Sectors
M	Number of criteria
n_g	Number of individuals that dominate goal vector g
N_x	Neighborhood of BU x
P_i	Sector/Partition i, set of Basic Units
P^d_i	Sector/Partition i for Day d, set of Basic Units
$P_{i,s}$	Subsector s of P_i
PF	Set of Solutions in the Pareto-Front
X^d	Set of Sectors for Day d (daily allocation)
R	Reference Point
R_{earth}	Radius of Earth
S	Solution of the Problem, Set of Instances that represent the PF approximation
S_p	Selection Pressure
T	Tour
t_d	Average delivery time
T_d	Number of deliveries in tour T
t_{opt}	Optimal working time
V	Set of All Basic Units
$x.s$	Sector of BU x
X	Set of Sectors (global allocation)
w	Depot/Warehouse

Chapter 1

Introduction

Logistics processes are a fundamental part of our lives, either directly or indirectly. At some point, we all had some kind of contact with them. The packages we received via express mail had to go through some logistics planning defining how the cargo should be transported. The groceries we buy on the market had to be shipped, delivered, and managed by a logistics company. Temperature and time-sensitive items, like organ transplants or vaccines, also have to go through logistics planning to find the routes that allow the safe delivery of those items. There are endless examples in the world we are part of, and they are an essential part of our daily lives, keeping the world connected and ensuring we have access to what the world has to offer.

All these services use logistic processes to manage the shipment and delivery of items to customers. As the primary and secondary (manufacturing) sectors of activity grow, so does the demand for logistic companies to distribute goods. Moreover, as the e-commerce and online shopping market grows, with an estimated growth of 50% in the next four years [12], so does the need for logistic processes. In 2021, the market estimation was at 352.4 billion euros [17], and the expectation is to have an annual growth of 4.7% until 2024 [16]. In Europe, the usage of road freight increased 76.5% in 2018, compared to rail transportation, making use of 1,764 billion kilometers of roads as of 2019 [17].

Logistic processes in last-mile delivery are responsible for managing the inventory, how it is processed and how items are shipped and ensuring that goods are delivered to their location in a timely and cost/resource-effective fashion. As little as a 5% increase in performance metrics can have a massive impact on the company's financial balance, as they survive on minimal profit margins [30]. Therefore, delivery planning is paramount for logistic companies to be ahead of their competition.

The demand can be fixed for each day, with minor variations over time; for example, a set of restaurants requests groceries daily. A logistic transportation plan can be defined in this situation, and, no significant changes occur between different days. As such, significant events are not frequent; following the previous example, a new restaurant appears, or a restaurant changes supplier.

There is no need to have daily planning of the logistic process. The same planning can be used daily and redone once a change occurs, which probably, will not significantly affect the previous logistic process. However, this situation is not the most frequent.

On the other hand, the demand can be dynamic and vary from day to day; for example, post-mail services know the set of letter and their location shortly before their delivery, on the previous or even on the same day; and must plan accordingly how to deliver them. This situation is the most typical for real-world cases and generally deals with many customers to serve. In many cases, geographic zones are defined by using geographic and topological structures, such as city districts, main streets or address attributes, to name a few. These zones allow for organizing deliveries per zone. Drivers are then responsible for those geographic zones. The following leads to one-to-one relationships between the driver and the zone. The driver gets familiarized with the zone, increasing its performance and the client's satisfaction with the driver. However, this can diverge from the optimal use of resources and changes in demand, so some companies redesign zones periodically [36].

This dissertation deals with logistics processes in the last-mile delivery context. Last-mile delivery is a complex process that involves a vast set of problems, such as, the ones related to vehicle routing, allocation of packages, resource management and definition of the geographic areas. An efficient and adequate management of all these problems and their dependencies between each other is crucial for the company's success.

The problem of serving the goods to the clients by the logistic companies can be seen as a variation of the Vehicle Routing Problem (VRP), where an allocation of items to vehicles must be made and their subsequent routes, while minimizing and objective function, such as, cost of transportation, CO2 emissions or client dissatisfaction. Logistic transports deal with delivery problems, where vehicles have different capacities and items have their volume and weight. In last-mile delivery problems, there is a dynamic demand, a large number of goods to be shipped, and their location is usually known within a short time interval to the delivery execution [24]. Due to the dynamic demand nature and context, there is the need to have a good definition of geographic zones and easy adaptation of those to the daily demand.

Defining a geographic area to drivers can be seen as a Sectorization problem. Not only is it widely used in the logistic processes for grouping the packages, but it also allows for reducing a large search space problem into smaller instances, making it easier to solve and in a faster time. The partitioning elements are also known as sectors, districts, or zones. Sectors consist of basic units, indivisible units that must be considered as a whole and can be of the form of points, lines, or polygons. Basic units are grouped, forming sectors so that they have similarities between each other. This contrasts with clustering, where similar points are also grouped, but the difference between clusters is maximized, whereas, in sectorization, sectors tend to have similar characteristics [10]. For example, the distribution of stores across a territory can be seen as sectoring the territory, where possible clients are distributed to balance each sector and allocate a store for each sector so that each client can be served the best way. Sectorization is also called Districting or Territory Design.

The space division into sectors follows a set of restrictions and can be problem-dependent. As such restrictions, sectors should share the same amount of work; for example, the number of items should be approximately the same for all sectors. In many applications, a path between two entities in the same sector should be contained in the sector. Also, the measurement of compactness is an essential factor that defines how a sector is compacted and can affect its general shape. Moreover, routes are an important part of the problem, allowing to measure important metrics for a last-mile logistics company, such as the number of packages delivered, the route cost and many others.

Several metrics to consider when designing a space partition tend to conflict. The partition design can follow very different paths depending on the decision-maker's interests. As such, the decision-maker should be able to choose the partition that is best for the company. Multi-Objective Optimization deals with problems that have multiple objectives to optimize at the same time.

1.1 Context and Motivation

Logistics processes are part of our daily life in the most diverse manners. It is a growing and competitive market, where companies look to improve and take advantage of every detail to keep up and stay ahead of their competitors. Planning the delivery processes, the definition of routes, vehicle allocation, and item allocation are not straightforward tasks and impact the business's financial and energetic costs, human resources, and, client satisfaction, for instance.

Last-mile delivery problems consider a dynamic demand and a large number of items to be delivered every day. Routes between clients should be defined and vehicles allocated, maximizing the business interests. Finding a good allocation, organizing packages and shipping them into the vehicles must be made quickly. Many times, these are highly manual processes that depend on the experience and knowledge of the employees. With the growing market dynamic, ever-changing world and challenges, from the pandemic to the ongoing war, and, limited human resources, there is the need to have automated and algorithms tools to generate and validate the methods employed by the company. Only then, can the last-mile logistics business be successful.

The geographic area where logistics companies act tends to be of considerable dimension and the number of vehicles to meet the demand is also high. The partition of this space is not an easy procedure and there are several variables to take into account, that it is hard for the decision-maker to define and choose the partition that maximizes the company's interests.

Therefore, the relevance and importance of the logistics processes and the complexity of the problems associated to this thematic constitute the main motivation of this work.

1.2 Scope and Objectives

In this dissertation, a last-mile logistic problem is studied. Sectorization is employed to deal with the determination of drivers' areas of responsibility. The problem at hand is one of large scale, and sectorization aims to produce good quality allocations in a feasible time and represent the business logic.

Adequate criteria for the problem are developed to represent the context of last-mile delivery well. In order to allow the decision-maker to choose the allocation whose criteria best suits the interests, a multi-optimization technique is employed to return the set of allocations from which the decision-maker can choose. Furthermore, a visualization tool to aid the process of choosing the best-fit allocation is essential.

Ultimately, this work intends to design a decision-making tool to help last-mile logistics companies define their sectors for service demand. The tool aims to partition the geographic space and delivery locations into sectors that correspond to the company's needs. Sectors must be balanced in the amount of work, compact and contiguous, allowing for a definition of good quality routes within each sector and a good quality solution overall. The defining routes serve as a means of validating the proposed method.

Such implementation proposes to achieve the following *requirements* (Rs):

- **R1: Provide a framework to aid and support the decision-maker,** so that the decision maker can make a thoughtful decision, thus, optimizing the service.
- **R2: Produce a set of possible partitions,** so that the decision-maker can be presented with the possible allocations that could fit his interests.
- **R3: Produce robust and adequate partitions,** allowing for a balanced and easy management/distribution of the deliveries by their postal codes and consequent distribution by the drivers' zones.

1.3 Main Hypothesis and Research Question

From the *requirements* proposed, the following hypotheses can be formulated:

> "Sectorization applied to the logistic business logic provides a useful tool, by, (i) dividing a complex problem into smaller, easier-to-manage problems, (ii), providing a good quality representation of the drivers' areas of responsibility."

> "Multi-objective Optimization applied to the logistic business logic provides a valuable tool, by, (i) allowing for optimizing multiple criteria simultaneously, (ii), aiding the decision-maker to choose the best-fit allocation."

To validate the proposed hypothesis and to guide this work, the following research questions (RQs) were identified:

RQ1: Can Sectorization contribute to better dealing with the last-mile logistics business logic?

The daily process of managing parcels is a demanding task. Identifying key aspects of the process where sectorization can be applied and transposed to real situations is vital to optimizing

business logistics. How can sectorization be used to define geographic areas?

RQ2: Is Multi-Objective Optimization adequate for optimizing multiple criteria in the last-mile delivery business?

There are several criteria that last-mile delivery companies consider when designing geographic areas. These companies work on large-scale instances. How can Multi-Objective Optimization be integrated?

RQ3: How can a decision-maker decide on the best geographic partition?

The decision-maker can choose a partition from a multitude of possible geographic partitions. Is the decision-maker aware of the decisions and understands what differs between them?

1.4 Document Structure

This document is composed of eight chapters, structured as follows:

- Chapter 1, **Introduction**, introduces the context and motivation for this work, the problem under study, and the general goals of this dissertation.
- Chapter 2, **State of the art**, performs a literature review on Vehicle Routing Zones, Sectorization and Multi-objective Optimization in the last-mile context, relevant for this work.
- Chapter 3, **Last-mile delivery**, gives a general description of the problem based on a real-world case, formalizes the problem and exposes the expected results.
- Chapter 4, **Data generation**, explains the thought process behind data generation and its implementation.
- Chapter 5, **Handling the Problem**, explains the approach to the problem, stating the methods and techniques considered.
- Chapter 6, **Implementation**, describes in-depth the implementation of the solution for the problem.
- Chapter 7, **Results**, exposes and explains the results achieved by the proposed solution.
- Chapter 8, **Conclusion**, summarizes this work, exposes open questions and suggests future work.

Chapter 2

State of the art

The chapter introduces relevant literature, as well as some crucial concepts and definitions, about Vehicle Routing Zones, Sectorization and Multi-Objective Optimization in the context of this work.

2.1 Vehicle Routing Zones

The Vehicle Routing Zones (VRZ) problem is closely related to the Capacitated Vehicle Routing Problem (CVRP). The last-mile delivery and pickup can be formulated as solving a CVRP. Vehicles are responsible for delivering a set of goods and collecting others. Thus, optimized routes must be defined to accommodate all items, reduce costs and increase satisfaction.

At a planning level, companies are interested in defining vehicle routing zones, allowing for drivers to specialize on that region and get to know its characteristics and its clients, allowing for a better route management and handling of unexpected problems, such as accidents, construction work, traffic, etc. The definition of VRZ is often applied when we are dealing with a dynamic and stochastic distribution of clients. In such situations, some elements of the problem are random and only known after the allocation and routes must be defined to serve all the client. VRZ play an important role because they allow the definition of zones, such that, the expected demand is approximately the same; thus, balancing the expected work. This problem is called the Stochastic CVRP (SCVRP).

In order to understand what makes a good VRZ, one must understand what is a good route for the VRP. A good route for the VRP is only as good as the other routes are. The measurement of a route quality, on its own, is meaningless in the whole solution of the VRP, so in order to analyse the characteristics of good routes, one must analyse a good VRP solution. In the work of [3], the authors evaluate what differentiates a near-optimal solution and a non-optimal solution for the CVRP. In this work, the authors make the following statement:

> "...the four most characteristic properties of near-optimal VRP solutions are narrow, compact routes with few intersections and short outgoing edges from the depot."

In this work, the goal is to use the analysis to design better problem-dependent heuristics and it is done so for Guided Local Search (GLS) [3]. The authors in [41] make a comment on the previous work, analysing the correlation between different characteristics in order to produce more meaningful data and heuristics.

Quote 2.1 provides valuable insight on how a VRZ should be defined for the CVRP. Route, and, consequently, their zones, should be narrow, with small width with relation to length. Routes should be compact, with points near each other. Routes should have few intersection, so that vehicles do not cross and enter inside another vehicle's area. Routes should have small outgoing edges from the depot, so that, vehicles can start their service as soon as possible. Intuitively, VRZ should be defined with the same characteristics as routes.

For the SCVRP, VRZ are important as they allow for a good strategic planning, optimize management of packages and reduce the effort of defining vehicle routes and generating a daily planning. However, many times the vehicle routes are not defined using an algorithmic approach. In some real-world scenarios, routes are defined by the drivers. For this to happen, routes must comply with "visual attractiveness". On the one hand, it allows for an easier adoption by the company and planning of the routes by the drivers. On the other hand, it deviates from the optimal solution [48]. In this work, the authors acknowledge that humans find non-intersecting and compact routes more attractive. These are some characteristics expressed in quote 2.1. In [44], the authors investigate the human ability to make Travel Salesperson Problem (TSP) routes and state that humans find optimal solutions when the routes are simple (approximate the convex hull); as the routes get more complex (disruptions to the convex hull analogy), humans generally find sub-optimal routes.

The concept of VRZ for a large-scale SCVRP in the distribution context is studied in [45]. In this work, the authors represent a continuous distribution function of work (deliveries and pickups) into an radius-angle space and produce square shaped zones, that map into wedge-shape zones in the "xy" space. The zones are defined based on the equilibrium of two types of repulsive forces. The depot is located at a corner of the geographic space.

The authors in [34] study a VRP for courier companies, in which they introduce the concept of "microzones", small geographic regions that have a set parcels to be delivered. Each microzone is associated with a vehicle, responsible for those microzones. A preferred assignment of microzones is preferred, known as *tactical planning*, but, depending on each day demand's, the plan is modified, known as *operational planning*. The deviation between the two plans, the imbalance, and, transportation costs are minimized.

The same concept of *tactical planning* and *operational planning* is also used in [5]. The idea of the two concepts is to allow consistency in VRZ created. A two-phase approach is implemented, on the *tactical level* the authors use Sectorization to define the tactical planning and on the *operational level* the assignment is adapted to each day's demand, called Resectorization. The IP models were solved using MIP.

2.2 Sectorization

Sectorization is the main focus of this project as a means of improving the quality and computational time of good solutions for the allocation of vehicles to routes in last-mile logistic processes.

Sectorization groups similar elements, also known as basic units, into sectors, such that each sector has similar characteristics. Basic units can be in the form of points, lines, or polygons. Each basic unit generally has an activity level associated, representing the amount of work needed for that basic unit; for instance, the number of clients, the number of voters, time needed to serve, to name a few.

Sectors can also be called districts, territories, or zones and result from grouping a set of basic units. Sectors are disjoint, and their union must result in the set of all basic units.

For each sector, one can define a central point as the center of the sector. This center can either be a real point from the data with geographic meaning as it is used on Location-Allocation Problems [55], LAP, where a point for each sector must be chosen as the distribution point for that sector. On the other hand, centers can be artificial points, meaning that either the point is a real point from the dataset, but no actual action is taken on that point. For example, no facility is allocated there, as in Territory Design Problem (TDP) applied to commercial districting (CTDP) [50] or the point does not belong to the dataset but has a purely geometric meaning, like the mean coordinates of the basic units of that sector. Central points are especially relevant when they are part of the solution, like in LAP, or when they are used for the compactness measurement calculation. In these cases, the solution's quality depends on the calculation of the central point. A poor allocation of the central point can deteriorate the solution quality even when the sector is well-formed.

For each sector, one can define a central point as the center of the sector. This center can either be an actual basic unit or an artificial point, with or without geographic and geometric meaning. In LAP, as in [55], a basic unit for each sector must be chosen as the distribution point for that sector. On the other, hand, a basic unit can be chosen to define the center of the sector, but no action is taken on that point, as a in Territory Design Problem (TDP) applied to commercial districting (CTDP) [50]. Moreover, in other situations, the point can be purely artificial and have only a geometric meaning, like the mean coordinates of the basic units of that sector. Central points are especially relevant when they are part of the solution, like in LAP, or when they are used for the sector's quality measurements, such as the Compactness calculation. In these cases, the solution's quality depends on the calculation of the central point. A poor allocation of the central point can deteriorate the solution quality even when the sector is well-formed. Some authors opt for not using any definition of central point for their solutions, mainly as it reduces the need to recompute the center on an update on the sector, as in [49]. An illustrative example of Sectorization from this work can be seen in Figure 2.1.

The set of all sectors and its information results in the Sectorization Plan, consisting of the definition of each sector and information on the sectors, depending on the problem. For example, the central point location for LAP or sector's routes for the routing problems. Other terms for this

are Districting Plan, Territory Layout, or Solution/Instance.

In some situations, the number of sectors is already defined [50]; in other situations, the goal is to minimize the number of sectors capable of fulfilling the constraints [40].

Sectors are subject to several constraints. As mentioned earlier, sectors must be disjoint, and their union results in the set of all points. Let V be the set of all basic units, and P_i represent the set of basic units comprised in the sector of index i:

$$P_i \cap P_j = \emptyset \tag{2.1}$$

$$\bigcup_{i=1}^{n} P_i = V \tag{2.2}$$

Moreover, each sector must comply with some criteria or restrictions that define the way sectors are defined. Balance is one restriction that states that the amount of work between each sector must be the same. For example, each sector must have the same amount of basic units as any other. Another restriction is Contiguity, which states that for every pair of basic units within the same sector, there must be a path between them without leaving the district. The last criteria is Compactness, which defines the way sectors are compacted, such that basic units within the same district should be close to each other. However, there is no formal or universal definition for compactness. These criteria will be further detailed below.

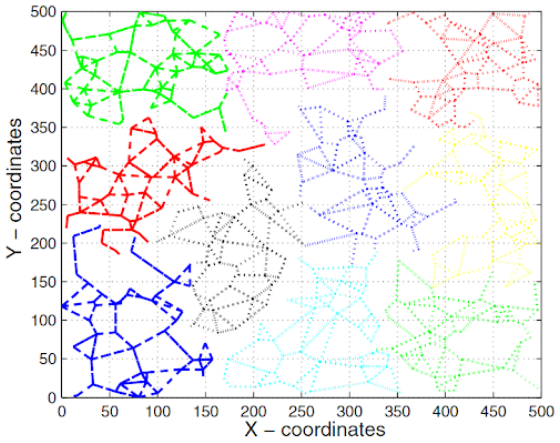


Figure 2.1: Solution for an instance of CTDP [49]

Sectorization had its first uses in political districting as a means of redistricting a geographic area into districts in order to avoid gerrymandering, such that every vote counts the same and no party can take advantage of the district design [31]. Political districting aims to segment a governmental region into districts capable of electing politicians for the government. It must be done so that every vote counts the same. Political districting has been widely studied over the years [20] [22] [27] [9].

Another use of sectorization, as seen in the literature, is Commercial Territory Design. In this situation, the territory must be segmented into sectors of an equal amount of work. There is an associated salesperson for each sector where the goal is to maximize the profit for the available salespersons. Sectors must be compact, contiguous, and non-intersecting for an efficient allocation of the salespersons. This problem is studied in [50] [49] [38].

We combine Police Districting (PD) and Home Health Care (HHC) in the same category, in the sense, that there is the need to have a short travel time between several geographic points within the same sector; for example, incident calls or home service locations, but also to have a short response time in case of an unexpected emergency. The PD is studied in [11] and the HHC in [6] [7] [40].

Location-Allocation Problems are also a family of problems where, apart from designing the sectors, it is important to determine a main facility (basic unit) within the sector to serve the sector [55].

Other problems can be derived and solved with resource to Sectorization as Waste Collection [47], distribution and routing problem such as in [2] [37] [25] [5].

2.2.1 Balance

Each basic unit has some amount of work defined, also known as activity. Some examples of activity are the number of packages, volume of packages, time demand, product demand, etc. The activity definition generally depends on the context of the problem one is trying to solve. Moreover, each basic unit can have several dimensions of activity; it is not constrained to only one. Let $w_a(P_i)$ represent the amount of work of activity a for a sector i, such that:

$$w_a(P_i) = \sum_{b \in P_i} w_a(b) \tag{2.3}$$

Balance states that each sector should have the same amount of work on every dimension, avoiding situations where some sectors are responsible for a large part of the work on the solution. For example, some vehicles need a large amount of time to complete the delivery process, whereas others need a small portion of that time. Balance allows for a fair and even distribution of the workload across all sectors. Without loss of generality, consider a single level of activity. In the optimal scenario, sectors are perfectly balanced, such that the average activity $\mu = \frac{w(X)}{T}$, where T represents the number of sectors, $w(X)$ is the balance for the whole allocation of sectors X. Perfectly balance sectors are often infeasible, since most of the times, no allocation yields the same amount of work across all activity levels. As such, a definition that considers the deviation from the optimal value is preferred. For heterogeneous sectors, the balance constraints cannot apply fully and depend on each vehicle's characteristics.

In the most common case, there is a user-defined threshold, τ, for how much a deviation is allowed. Note that the bigger the threshold, the more unbalanced sectors are allowed to be. For cases where there is no violation, the balance metric is 0; otherwise, a positive value is charged, which can be normalized by the average balance μ [9] [50] [49].

$$bal_{\tau}(P_i) = \frac{max\{w(P_i) - (1+\tau) \times \mu, (1-\tau) \times \mu - w(P_i), 0\}}{\mu} \tag{2.4}$$

In linear programming approaches, the equation 2.4 is enforced on each sector, such that each sector must be within the threshold [52] [55].

$$|w(P_i) - \mu| \leq \mu \times \tau \tag{2.5}$$

The balance of the solution, taking into account all the sectors, can be of several forms. One possible form is to take the maximum of all sectors' balance, such that the objective is to minimize the maximum workload of all sectors:

$$bal_{max} = max_{P_i \in X} bal(P_i) \tag{2.6}$$

The formulation in 2.6 has the problem that only one district is considered and the rest are not taken into consideration. For example, if all districts are highly unbalanced or just one is highly unbalanced, there is no distinction following the given formulation. As such, a different formulations can be chosen, as taking the sum of the measurements of the balance constraints [50] or the average of the balance measurements:

$$bal_{sum} = \sum_{P_i \in X} bal(P_i) \tag{2.7}$$

$$bal_{avg} = \frac{bal_{sum}}{T} \tag{2.8}$$

The problem with the two formulations 2.7 and 2.8 is that effect of highly unbalanced sectors can be reduced by balanced ones. Thus, in order to tackle the problems in both formulations, a combination of both has been proposed [10], with a user-defined parameter α:

$$bal_{cc,\alpha} = \alpha \times bal_{avg} + (1-\alpha) \times bal_{max} \tag{2.9}$$

Another possible approach is to use the Coefficient of Variation (CV) in order to mitigate the effect of good balanced sector on poor balanced sectors [47]. CV is defined as the division of the standard deviation by the mean of the values. A good balance measurement as CV close to 0, otherwise the effect of unbalanced sectors will be measured. It compares values of different sectors and calculates the dispersion of values.

$$bal_{cv} = \frac{\sqrt{\frac{1}{T-1} \times \sum_{P_i \in X} (bal(P_i) - \mu)^2}}{\mu} \tag{2.10}$$

For multiple dimensions of activity, the most common approach is to use the unweighted sum of all values of the balance for each activity [49]:

$$bal_\tau = \sum_{j \in a} bal_\tau^j \tag{2.11}$$

where bal^j is the balance formulation chosen, with respect to the activity dimension j. A parameter β_τ^j can be defined to give more weight to some more critical or less flexible activities. For this formulation, it is recommended a normalization of each activity level.

For a logistic districting problem with pickups and deliveries [25], the authors formulated the multi-activity balance constraints where they transform each of the dimensions of activities into a shared one, more specifically, the number of pickups was transformed into the expected time to fulfill all the pickups, the same logic for the deliveries and the distance from the depot to the furthest point to serve was transformed into the expected time to reach that point. In some sense, transforming activities of different contexts into the same dimension, allowing to be considered as the same.

2.2.2 Contiguity

Contiguity states that there must be a path between two points on the same sector that does not leave the sector. This is important as it keeps travel times low between clients in the same sector and allows to clearly define zones of actuation for each driver, such that no driver has to pass through other drivers' zones.

This property can also be called Connectivity. It can be seen as every basic unit on sector i must be connected to a neighboring another basic unit of the same sector i. The relation of neighborhood is problem-dependent and is not always defined. For example, basic units can represent city blocks, and the neighborhood is defined by edges representing blocks adjacent to each other. In other cases, basic units are points laid out in the geographic space, and no explicit neighborhood is given. It is impractical and meaningless to consider a fully connected graph in such cases.

In work [34], a proximity graph approach is implemented to derive the neighborhood of the basic units. A Delaunay Triangulation (DT) is proposed, which comes from the Voronoi Diagrams. Neighborhood is defined on basic units that share an edge in the DT.

In [10], several other proximity graphs are mentioned. From the DT, several other proximity graphs can be derived and used to represent the neighborhood of the basic units, such as the Gabriel Graph, Relative Neighborhood Graph, Urquhart Graph, Haugland Graph, which are all subgraphs of the DT.

Another possible approach is to consider Contiguity not as a neighbouring relationship and a restriction but as a metric. For example, the intersection of the convex hull of the sectors, as done in [36]. The idea is that in a continuous sector, its convex hull must not intersect another sector's convex hull. It is possible to measure the area intersected.

In some works, contiguity is not enforced directly but rather enforced through the compactness criteria, such that a compact sector tends to be continuous.

On some iterative construction approaches, sectors' contiguity is ensured as an invariant of the construction procedure [49]. In each iteration of the construction and search process, the contiguity criteria is always fulfilled.

2.2.3 Compactness

Compactness is a way of measuring dispersion within a territory. There is no formal or general definition or formulation, and many attempts have been proposed over the years. Some definitions are focused on the shape of the sector, whereas others take into account the area of the sector. In contrast, others consider the perimeter of the sector and some on the dispersion of the basic units on the sector. All of these definitions have some weak and strong aspects.

The first problem arises from the object the Compactness metric is measuring. If the metric measures concerning the individual sector, the overall solution quality can be degraded because of a single sector. Consider a given metric and a sector with a particular distribution of basic units. The metric can yield a poor quality due to the distribution of basic units. On the other hand, another metric measures concerning every sector, the quality measure of some sectors can be minimized by the quality of the others. The problem is similar to the one described for the Balance in 2.2.1. A combination of several metrics and ways of calculating the global Compactness metric can be preferred to tackle these problems and get a more general formulation of Compactness across several dimensions, for example, perimeter, shape, and area, using maximum and average values.

The issue of combining different compactness measurements in a naive way is that different measurements can have different output spaces. Consider a shape-based metric. In this situation, the metric will output a value between 0 and 1 depending on similar the sector is to the shape. The metrics outputting a value between 0 and 1 have the advantage that they are easy to understand and interpreter in a wide range of situations. However, in a perimeter-based metric or an area-based, the output is not bounded to 0 and 1. A normalization factor does not necessarily bound the metric to 0 and 1; thus, it can be hard to understand, compare and combine metrics with one another. A naive way of combining several metrics with a weighted average can not lead to the expected results.

As such, it is difficult or impossible to define a comprehensive and meaningful metric that considers all the dimensions for compactness. The naive approach many times shows up infeasible for the problem. However, one relevant consideration is that a Compactness metric should be easy to understand and explain as it allows for easier adoption and comprehensibility by the logistics company, transmitting a more trustworthy feeling.

Another relevant aspect for the logistic companies and drivers is a clear definition of sectors such that they are visually sound, easy to identify and comprehend. This is especially important for drivers as they are the ones that will be doing the work on the designated areas, and, as such, each driver's zone must be intuitive to the driver. In other words, a sector must be simple. This might come at the cost of not defining the optimal set of routes for the optimal set of customers; however, the visual attractiveness of sectors allows for a more practical implementation of routes, easier adoption from drivers and logistic companies without necessarily worsening the solution quality [48].

Butsch [10] does a comprehensive study on compactness and its definitions through the literature and defines it as the following:

> "a district is compact if it is nearly round-shaped or square, undistorted, without holes, and has a smooth boundary"

however many of the compactness metrics definitions come from the political districting problem or the service/sales territory design, and few focus on routing problems. In this section, a more routing focus approach is conducted.

2.2.3.1 Sum of distance

A compactness metric as defined in [5], it considers the sum of the distances of all basic units within each sector to the center sector's center.

$$comp_d(P_i) = \sum_{b \in P_i} d_{b,c_i} \tag{2.12}$$

A compact sector will have this metric as close as possible to 0.

2.2.3.2 Maximal Distance

The maximal distance metric considers the maximal distance from two basic units on the same sector or the maximal distance from a basic unit to the center of the sector.

$$comp_{md}(P_i) = max_{j,k \in P_i} d_{j,k} \tag{2.13}$$

As such, the compactness of the solution is given as the maximum compactness of each sector:

$$comp_{md} = max_{P_i \in X} comp_{md}(P_i) \tag{2.14}$$

This metric is applied to the Sales Districting Problem [50] [49]. Also, it is used for the home care services applied to the Home Health Care (HHC) in order to sector the geographic space into

sectors capable of being handled by a healthcare provider on time, allowing for a rapid response in a case of urgency [7]. The intuition of this method is that time-expensive routes have basic units that are far away from each other and spread out, and this is minimized by the compactness metric.

One possible variation of this metric is to use the driving/walking duration between two basic units in the same sector, such that the maximum driving/walking duration between two basic units is bounded from above or minimized. For that, the driving duration between two basic units must be known [40] or estimated.

2.2.3.3 Perimeter metric

Perimeter-based measurements take advantage of the sectors' perimeter to evaluate the compactness metric. The intuition of this metric is that a compact sector tends to have a small perimeter; thus, minimizing the perimeter, the sector will be compact.

The most intuitive definition for a perimeter-based metric is considering the sum of all sector's perimeters:

$$comp_p = \sum_{S_i \in X} perimeter(S_i) \tag{2.15}$$

where $perimeter(S_i)$ is the perimeter of sector S_i. The problem of a perimeter-based metric is how the perimeter is calculated. If the boundary is not smooth and irregular, the perimeter will have a high value. On the other hand, if the perimeter is calculated on the convex hull of the sector, the perimeter will be lower. Note that the convex hull of the sectors can intersect each other.

A perimeter-based compactness metric is used to solve the Vehicle Routing, and Districting Problem with Stochastic Customers [37], where the authors compute the mean normalized perimeter of the districts and claim it produces visually compact districts.

$$comp_p = \frac{\sum_{P_i \in X} perimeter(P_i) - perimeter(X)}{2 \times perimeter(X) \times T} \tag{2.16}$$

where $perimeter(X)$ is the perimeter of the entire territory.

The same formulation is adapted to the design of districts and subdistricts for multi-vehicles with dynamic clients with multi-depots [38]. In this work, subdistricts are considered within each district, such that a driver serves all clients in its district. However, he only serves each subdistrict at each time frame. It follows the following formulation:

$$comp_p = \frac{\sum_{P_i \in X} (\sum_{P_{i,s} \in P_i} perimeter(P_{i,s}) - perimeter(X))}{2 \times perimeter(X) \sum_{P_i \in X} |P_i|} \tag{2.17}$$

where $P_{i,s}$ is the subdistrict s of P_i and $|P_i|$ is the amount of subdistricts in P_i.

2.2.3.4 Route Length

The idea behind this metric is that a compact route will produce a route with a low cost (distance). The definition of the optimal route for several points is a hard problem in itself. As such, it might be more feasible to use an approximation of the route length as a metric for the compactness.

For a last-mile distribution problem, where the goal is to design distribution goals, the authors in [2] define an approximation function for the route length based on the deviation of the basic units. However, the compactness is not evaluated directly on the proposed solution, using any definition.

$$comp_{rl}(P_i) = 2.791 \times \sqrt{n \times \sigma_x' \times \sigma_y'} + 0.2669 \times \sqrt{\frac{\sigma_x \times \sigma_y}{\overline{d_x} \times \overline{d_y}} \times n \times A_s} \tag{2.18}$$

where $\overline{d_x}$ / $\overline{d_y}$ is the average of the distances of the basic units to the horizontal / vertical center axis, σ_x / σ_y is the standard deviation of the basic units in the x / y axis, σ_x' / σ_x' is the standard deviation of the absolute distance of the basic units in the x / y axis, A_s is the area of the district and n is the number of basic units in the sector.

2.2.3.5 Activity Density

Another possible approach is to consider the activity to measure the density of each sector. The authors in [47] study a waste-collection problem where sectors, and, routes need to be defined; and, perceive compactness as a measurement of the concentration of activity:

$$comp_{act}(P_i) = \frac{\sum_{b \in P_i} w(b)}{d_{c_i,f_i}} \tag{2.19}$$

where d_{c_i,f_i} is the distance between the center of sector P_i, c_i, and the farthest basic unit from c_i, f_i. This metric aims to evaluate the activity density of the sector, avoiding spread out sectors with elongated "arms" coming out of the sector. The higher the value, the higher the compactness.

In this work, the authors use the Coefficient of Variation (CV) to measure the complete solution's compactness. The definition is the same as in 2.10. The better the compactness, the closer to 0 the measurement.

2.2.3.6 Radial Span

In some work, the radial span is used to compute some solutions' metrics. In the analyzed literature, no direct use of radial angles was defined as a compactness metric.

In [13] the concept of radial span is used in a Capacitated Clustering (CC) for solving a vehicle routing problem. The problem of CC appears related to the one of Sectorization; however, the idea behind Clustering is that points in a cluster must be similar, but clusters should be as different as possible from each other. The introduction of a capacity for each cluster ensures a sense of balance

criteria; however, there is no notion of contiguity, and clustering algorithms are used. In this work, the goal is to minimize the total square error of the K-means algorithm, defined as follows:

$$comp_{act}(P_i) = \sum_{P_i \in X} \sum_{b \in P_i} (\theta(b) - \theta(c_i))^2 \tag{2.20}$$

where $\theta(s)$ is the angle of the basic unit s in relation to the horizontal axis and c_i is the center of S_i.

On the work of [26], a reference to a greedy heuristic based on angles is done. Here, the authors mention a construction heuristic that considers the angles of the basic units and partitions the service region into wedge-shaped radial sectors of equal size.

2.2.4 Resectorization

Resectorization or redistricting is the term used when a solution is generated taking into account an initial solution. This concept is useful when one wants to improve an existing solution while keeping some degree of similarity between the original and final solution. Taking for example a company that already enforces sectors in their business logic, might want to improve their current allocation without suffering radical changes to their design. Another possibility is to define the operation plan from the tactical plan.

Resectorization implies a small modification of an original solution into a better solution, either by a small number of transfer of basic units to other sectors or even by adding or deleting a small number of sectors, such that there is some similarity between the two. It then produces an extra metric and constraint to take into account when building and searching for a solution. In the work [6], the authors mention the similarity between existing plan as a comparison criteria between two different solutions.

The authors in [9] use a similarity measure, which is based on the area that keeps belonging to the same sector:

$$sim_{area}(X) = 1 - \frac{\sum_{i=1}^{T} o_{P'_i}(X)}{A} \tag{2.21}$$

where P'_i is the original definition of sector i, $o_{P'_i}(x)$ is the area of the largest overlay of district P'_i in the new solution and A the total area of the geographic space.

The authors in [38], use the same logic to measure the dissimilarity between consecutive allocations in a dynamic design of sectors. In this scenario, there are several definition of sectors over time and they must be similar between each consecutive time stamp.

The redistricting measurement can also be used to compare the difference between the tactical and the operation plan. The operation plan should not be radically different from the tactical plan. Diverging from the previous definition, one based on the number of basic units that differ from one solution to another can also be considered, as in [34].

$$sim_{count}(X) = 1 - \sum_{i=1}^{T}(|P_i \cup P'_i| - |P_i \cap P'_i|) \tag{2.22}$$

The previous definition can be normalized by dividing the summation in 2.22 by the total number of basic units. Then, solutions which are at least ι close to the original partitioning are considered, with $\iota \in [0,1]$ [18].

The definition on 2.21 can only be used for polygon basic units or by applying some hull algorithm, has they have area as a property.

2.2.5 Implementations

Several implementations and methods have been proposed over the years but it is possible to divide those in two major classes: exact methods and approximation methods.

Exact methods are generally modeled as integer linear programming model and solved by and Integer Linear Programming (ILP). Exact methods have the advantage to find the optimal solution for an optimization problem, however, generally they take a considerable amount of time to solve the problem, making it infeasible for large-scale instances. Optimization problems are known to be NP-hard, thus, their complexity increases exponentially with the size of the problem, as such, exact methods tend to be used for small-scale instances. For the HHC problem, the authors in [6] propose ILP models for the problem and recall that for real-world instances heuristics approaches would be needed; also in [7] the authors use an ILP up to 100 basic units but do not supply the execution time. In [20] a political districting problem is approached, with up 48 basic units and taking from some minutes to several hours of execution time. The authors in [55], define a ILP model to solve the LAP for instances from 30 to 900 basic units, however no reference to the computational time is given. ILP is also used in [47] [52].

Approximation methods produce decent results in relative shorter time, however they do not guarantee an optimal solution. Is it preferred for larger instances where time is of the essence and a near-optimal solution is satisfactory. This methods are powerful in providing good solution in short time but lack the ability of guaranteeing an optimal solution in a finite time. Often, they are dependent on the implementation of the algorithm, on the definition of the local search procedures and heuristics and have a component of randomness. Many times, these algorithms end up stuck in a local optima with respect to the local neighborhoods defined. A global optimum is a local optimum with respect to all neighborhood structures [29]. It is very hard and time-consuming to define and explore every possible neighborhood. Thus, it is important to define meaningful and adequate heuristics and neighborhoods for the problem. There is no silver bullet for every optimization problem and even considering every problem, all algorithms and meta-heuristics perform equally well [59]. Over the year, several methods have been applied to Sectorization.

For the CTDP, a GRASP approach has been used in [50] [11] and with path-relinking on the best solutions found in [49]. It is also applied to a e-commerce logistics districting [2]. Tabu-Search (TB) has also been used for political districting in [9]. For a TDP for pickup and delivery operations, a GRASP and TB approach was implemented [26] and [25]. An Adaptive Large Neighbourhood Search (ALNS) was implemented to tackle the design of dynamic sales districting with a routing variant, where salespersons had to visit their clients on several time-frames [38] and to solve a routing problem with stochastic customers [37]. A Variable Neighborhood Tabu Search is also implemented in the work [34] to solve a courier routing problem. In the work [40] the authors propose a method based on a greedy heuristics. Column-generation is an heuristics also used to solve a relaxed version of the ILP model [27] and it is also used to generate good clusters in [36]. In [5], the authors implement an heuristic procedure to iteratively solve a relaxed ILP model. The work [2] combines a GRASP-like metaheuristics with an ILP. LAP can be divided into two phases, first as the location, where, the center of the sectors are determined (location), and only then, the sector is constructed by adding basic units (allocation) [51].

2.3 Multi-objective optimization

Single-objective can be represented in the following way:

$$min f(x), x \in \Omega \tag{2.23}$$

where f represents an objective function over the solution x from the solution space Ω.

In the previous examples, when dealing with several optimization criteria, the chosen method was to reduce the several criteria to a single function that would represent the objective function. One possibility is to define weights for each of the M criteria, giving more weight to more important ones and less to the less important and then apply an weighted sum, also known as scalarization:

$$f(x) = \sum_{i=1}^{M} w_i \times f_i(x), x \in \Omega \tag{2.24}$$

where, w_i represents the weight of criteria i and $f_i(x)$ represents the criteria measurement of solution x.

The key step in this process is the set up of the weight values. In most cases is it up to the decision-maker to define the preferences over the multiple criteria and even so, choosing the appropriate values is not straightforward. By specifying a set of weights we are leading the search process through a specific path and a small change in the weights can yield a completely different solution, it requires problem specific knowledge and parameter setting. Also, only one solution is

found at each algorithm execution, per the parameter configuration. Furthermore, the greater the number of weights, the harder it is to define preferences and the weight values.

The problems that deal directly with multiple objectives are known as Multi-Objective Problems (MOPs). In those problems, the ones that have a large number of objectives, more than three, are known as Many-Objective Problems (MaOPs). Multi-objective optimization can be represented as the following:

$$min[f_1(x), f_2(x), ..., f_n(x)], x \in \Omega \tag{2.25}$$

There are several problems that are MOPs by nature, and reducing it to single-optimization is restricting the solution space. Multiple real-world problems have the need to solve two or more objectives at the same time, which most of the times, objectives are in competition with each other, a decrease in one, represents an increase in another, thus, the solution for such problems is a set of solutions, also known as Pareto-front. For achieving such solutions, multi-objective optimization techniques must be employed.

2.3.1 Pareto-front

In order to understand the definition of Pareto-front, some definitions must first be presented. Consider a minimization problem, where all objectives need to be minimized.

Definition 2.3.1 (Pareto-dominance)**.** For two feasible solutions, x and y, it is said that x Pareto-dominates y ($x \preceq y$), if and only if, $\forall i \in 1, ..., M, f_i(x) \leq f_i(y) \wedge \exists i \in 1, ..., M, f_i(x) < f_i(y)$.

In other words, Pareto-dominance states that x dominates every solution y, whose criteria measurement are worse or equal to the ones of x, but there is at least one criteria that x outperforms that solution.

Definition 2.3.2 (Pareto-optimal)**.** A solution x is said to be Pareto-optimal if there are no solutions that Pareto-dominate x: $\nexists y \in \Omega, y \preceq x$

With this definition, we are ready to define the Pareto-front.

Definition 2.3.3 (Pareto-front)**.** The set of all feasible Pareto-optimal solution and its corresponding objective values: $PF = \{x \in \Omega : \nexists y \in \Omega, y \preceq x\}$

The concept of non-domination arises from the context of multiple objectives. When taking about solution that have multiple objectives to optimize, the definition for comparison and dominance has to contain all the objective values.

The goal of multi-objective optimization is to find the set of Pareto-optimal solutions that comprise the Pareto-front. Due to the population nature of the Pareto-front, many population evolutionary based algorithms have been studied over the years, also known as multi-objective evolutionary algorithms (MOEAs). The goal of these algorithm is to return a family of solutions

that represent an approximation of the Pareto-Front. In the next subsections, we explore the two most used and promising approaches to multi-optimization in the context of sectorization, NSGA-II and PICEA-g, and also some other approaches that have been studied.

2.3.2 NSGA-II

Multi-Objective Optimization using Non-dominated Sorting in Genetic-Algorithms was first mentioned in [54], also known as NSGA. However, it is its successor that became one of the most known and studied multi-optimization techniques, the NSGA-II. NSGA-II was introduced by [15] in order to address the issues and critics NSGA have been receiving over the years.

NSGA-II works on an elitist framework, which means that the non-dominated solutions are kept during the search process, either by an external archive, or, in this case, by keeping the parent population with the children one upon the pooling operation. It has a population of solutions and at each iteration of the algorithm, parents and children go through the non-dominated sorting procedure, which sorts the solution by domination front. The least dominated solutions are chosen by domination level until reaching the desired population size. If the solutions at domination level would increase the population size past its maximum, then a crowding-distance sorting would occur, in order to choose solution that better represent the entire search space and disregard solution that are near others (avoid crowded objective-space regions). The new population is used to generate the children for the new population. The children are generated by a genetic operator that recombines two solutions to generate an offspring and a mutation operator that induces a change over the solution. This is essential to keep exploitation and exploration of the search space.

The problem with Pareto-dominance based algorithms is the deterioration of the algorithms, as the number of non-dominated solutions increases [33] and there is a lack of comparability between solutions, as the majority of the solution tends be non-dominated. Thus, NSGA-II is said to produce good result up to three objectives.

In the context of sectorization, NSGA-II was been used in a water distribution network [61]. The authors formulated the problem as having 4 objectives to optimize, from water pressure, water age, transforming costs and number of sectors. The authors resort to some local operators that modify the solution, upon its generation, such as merging small sectors or optimizing connections between sectors.

The authors in [57] apply an hybrid approach to solve the political redistricting problem. The NSGA-II metaheuristics is applied together with a Variable Neighbourhood Search procedure, in order to increase the quality of the solution by exploiting the search space that solution represents.

In [23], NSGA-II is employed to solve a territory design problem. Here, the algorithm has several problem-specific improvements made in the initialization, crossover and mutation, to better deal with the land-use optimization and converge faster.

The authors in [32] apply NSGA-II to solve the allocation of earthquakes shelters, for two criteria, travel distance and number of shelters.

2.3.3 PICEA-g

The Preference-Inspired Co-Evolutionary Algorithm using Goal Vectors was first introduced by [58]. The idea behind the algorithm is to provide multiple artificial goals/preferences to the solution population, in order to provide comparison capability between the solutions, allowing to discriminate non-dominated solutions in an high-dimensional space. In their work, the authors do a study on different algorithms and conclude that PICEA-g makes a strong competitive performance against the other state-of-the art algorithms and is adequate for use in MaOPs.

PICEA-g works also on an elitist framework, where new populations are selected from a joined set of parents and offsprings, and an archive keep track of the best non-dominated solutions throughout the execution. There exists a population of solutions and goals, which is evolved during several generations, and the solutions are ranked according to a fitness evaluation which takes into account the dominated goals, as in formula 2.26. Goals also are ranked accordingly to a fitness evaluation that considers how much a goal is dominated, as in formula 2.27. For more information on this, readers are referenced to [58]. The bigger the population of goals, the better the comparison quality between solutions. After the population of solutions and goals have been pooled together with their offsprings, the cycle repeats itself.

$$F_s(x) = 0 + \sum_{g \in G_j | x \preceq g} \frac{1}{n_g} \tag{2.26}$$

$$F_g(g) = \frac{1}{1+\alpha} \tag{2.27}$$

$$\alpha = \begin{cases} 1 & \text{if } n_g = 0 \\ \frac{n_g - 1}{2N-1}, & \text{otherwise} \end{cases} \tag{2.28}$$

The authors in [39], solve a multi-objective stochastic districting and routing problem with resource to a PICEA-g with mating restriction (PICEA-g-mr) framework. In this variation, the authors introduce a mating restriction in order to guide the crossover of solution. The mating restriction is used to enhance the generation of new solution by the crossover of parents that share the similar features.

2.3.4 Other methods

In the work [28], an ε-Constraint Method is used to guide the search to obtain the Pareto front. The method is applied to a two criteria optimization problem of agricultural machinery service network. The ε-Constraint Method works by setting a set of constrains on the objective function, generally, upper bounds, and solving the problem minimizing one of the constraints. As such, there are needed a multitude of constraints for each of the objectives and it does not scale well for many-objective problems.

An approached based on an Hypevolume-Indicator Estimation Algorithm, HypE, is proposed in [60] to solve a urban design problem for neighbourhoods in China. HypE has also shown good results in many-optimization problems [4].

2.4 Summary

This chapter presents a review on the current work on Vehicle Routing Problem/Vehicle Routing Zones, Sectorization and Multi-Objective Optimization applied to the last-mile logistics context.

In section 2.1, a overview of the Vehicle Routing Zones is presented. VRZ are closely related to the VRP. In some practical and real-world scenarios, the definition of routes for the VRP is not always required straight ahead. Instead, business might be more interested in defining VRZ, which allow the specialization of the driver, increase consistency, client satisfaction and overall efficiency. To understand how VRZ should be defined, good characteristics of the CVRP routes are presented as well as characteristics for the zones that humans value and find relevant for the adoption of those zones.

In section 2.2, a review on the current work of sectorization is presented. Sectorization divides a geographic space into balance, continuous, and compact sectors. Balance aims to get sectors that have the same amount of work. Contiguity restrict sectors to connected sub-graphs. Compactness states that elements in the same sector must be close to each other. Each one of these three criteria is explained in more detail in a respective subsection, where possible definitions, examples and applications are presented.

Lastly, in section 2.3, a review on the relevant literature on the sectorization problems using Multi-Objective Optimization is conducted, presenting the most employed techniques to solve this problem.

Chapter 3

Last-mile delivery

In this chapter, a description of a last-mile delivery case study is given. It is based on a company providing meaningful information about its activities and procedures. In addition, the chapter presents a more formal and technical characterization of the associated problem to be addressed and identifies what is expected to be achieved with its resolution.

3.1 Real case study

The case study is deeply inspired by the activity of a last-mile parcel delivery company. The visit to the company provided fruitful knowledge and valuable insight into the business logic and inner processes. The information presented here is a starting point for understanding and modeling the problem. The real scenario takes place in the North of Portugal.

The last-mile delivery company in this study is divided into two departments, the Business-to-Business (B2B) and Business-to-Client (B2C). The B2B deals with the delivery of large packages in small number to large commercial entities, like supermarkets, shoppings, distribution centers and so on. The B2C deals mainly with the delivery of small packages to a large number of customers and is strongly connected to the e-commerce market. There are two main types of events, deliveries and pickups.

The logistics company receives the packages to deliver by night from several locations in a single depot, where all the processing is made. After that, packages are processed and manually organized by "geographic areas". Geographic areas are used to describe a set of postal codes that compose a partition of the geographic space. They are given the name of a main structure in those postal codes for easier identification by the employers, for example, the name of a city or the region's name. An allocation for the geographic areas has been defined and has been changed over time by the company to met with the company's demands.

Each package has an address to which it must be delivered, that usually includes the city, street name, door number and so on, however, it might come incomplete. The most important information for the logistics business is the postal-code, and this is used for organizing the packages per geographic area. As this process is done manually, using an identifiable name for the geographic areas is essential for an easy recognition of where the package should go. Each geographic area has a cage on which parcels are organized. Because of time to get used to changes to the organizational process, as it might affect efficiency, geographic areas are not frequently changed.

In the next stage, an assignment of parcels to vehicles/drivers is made, by a responsible person, within each geographic area, according to the postal codes. Drivers arrive for their shift and validate the assignment. They validate the assignment by looking into the deliveries and pickups, their location and distribution, understanding what can and cannot be done in a day's work. This is achievable as each driver operates in the similar region every day, so is familiar with the zone and knows the travel times to reach and return from that zone, the amount of time the driver has available for doing the work, and therefore can assess if the distribution is feasible. For example, if a driver identifies a far away isolated delivery that can impact the remaining deliveries, the driver can prioritize the work on other postal codes, proposing a transfer of the delivery to another driver.

Drivers can then organize and balance the workload between themselves, transferring delivery and pickup assignments to others to maximize service efficiency. The level of detail of the described transfer can go beyond the allocation of postal codes. It can be made in terms of regions, streets, or single assignments of events within a postal code, not restricted to the postal code. The decision is then passed into the responsible person who validates the new assignment into the system. This is of interest to everyone, as there are several Key Performance Indicators (KPIs) that track the performance of every person and service in the logistic process and everyone as a group. They are significant factor in financial bonuses for the whole team, and managers keep a close eye in order to guarantee client satisfaction.

After each driver has an agreed assignment, it is up to the driver to define the order on which he will make the deliveries and pickups based on the respective locations, but mainly focused on their postal codes. A driver tends to serve every client in the postal code before leaving it. There is no guarantee of the optimality of the route, and it is based on the geographic and empirical knowledge of each driver. At the end of the workday, if some parcels could not be delivered due to a "not at home" client or lack of time, thus, are delivered to a service point from where the client can pick up his package. Service points are external businesses that agreed to hold the packages for the client to pick up and are up to 3 kilometers away from the client.

There are about 80 drivers/vehicles available for the work in the North of Portugal, and generally, every one of them is working, independently of the delivery demand. The logistics company outsources vehicles and drivers, and the contract enforced does not allow for varying the number of drivers on a day-to-day basis as the subcontracted company needs to pay to its drivers. Each outsourced company is fully responsible for at least a geographic area, and no geographical area is divided by an outsourced entity.

The total freight comprises three types of vehicles; however, for B2C, only two are relevant:

the smaller size mainly used for urban areas and the medium size, used for either suburban or rural areas. Big size vehicles are used for B2B situations. All the vehicles start and end their journey on the depot, which is common for all vehicles.

Every day about 10000 packages are delivered by the drivers in the North of Portugal. The company distinguish between packages delivered and deliveries; a delivery is considered a stop where a package or several are delivered. It states that it does around 7000 to 8000 deliveries. Each package has a location associated, and once a package is delivered, the driver does not need to go back there anymore. The delivery characteristics are not homogeneous: the number of parcels, their weight and volume, and location. The company main focus is on the delivery of packages and not so much on pickups.

3.2 Problem background

From the description in 3.1 we can identify two levels of sectorization: (i) definition of the geographic areas and (ii) adaptation of the geographic areas to the daily demand. This study focus on the parcel delivery.

In the first level, postal codes are grouped forming geographic areas. The definition of geographic areas by postal codes is important for the company as it allows for easy identification of where the package belongs to. Postal codes are a partition of the geographic area into non-intersecting polygons, covering all the geographic areas; thus, it is possible to define a neighborhood relation based on postal codes that share edges between each other. Due to business logic, the allocation of postal codes to geographic areas is not frequently changed.

In the second level, the geographic areas are sectored to attribute areas of responsibility to drivers. This procedure is done on a daily basis, and generally, drivers have a similar assignment each time, making them familiar with the work area. There is some flexibility on the assignment, such that driver's sectors can contain deliveries/pickups from other geographic zones, allowing for a better balance of the workload across the drivers.

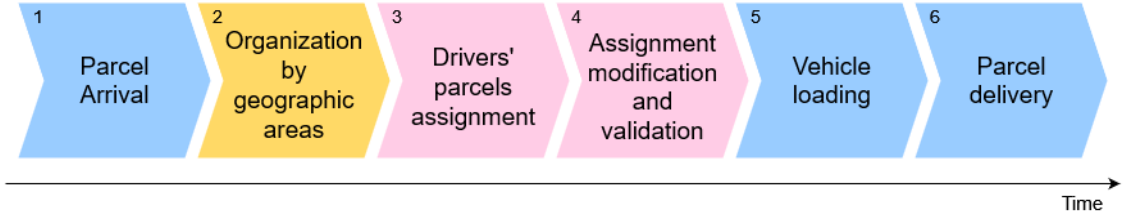


Figure 3.1: Logistics parcel management.

In figure 3.1 we can identify the different steps a parcel goes through from the moment it arrives to the last-mile company. In the first phase, the parcel arrives to the company. Then, it is organized by the geographic area its postal code corresponds to. Then, a responsible person does the assignment of postal codes to the drivers. Drivers validate and propose modification on

infeasible allocations, which is then validated by the responsible person. Finally, when there is an agreement, parcels are loaded into the vehicle and the delivery is executed.

The yellowish and pinkish colors represent the two different levels of sectorization manually enforced by the company. These two colors, represent two different levels where an algorithmic sectorization can be employed.

Understanding Postal Codes

To understand the problem and the underlying structure, a definition of the postal codes is presented.

Postal codes or zip codes are a string of digits and sometimes characters. This code are used worldwide to refer to specific locations and help couriers to identify the location of an address and sort packages accordingly. The structure of a postal code is country-dependent. For the purpose of this work will be used the Portuguese definition of postal codes.

In Portugal, postal codes were introduced in 1976. At the time, postal codes were consisted of four digits (CP4) and were mainly based on administrative geographical structures employed at the time, such as parish and counties, also known as *freguesias* and *municípios*.

Along the years, there was the need to expand and further divide the current CP4s in order to further specify the address and identify a specific place inside the CP4. So, in 1994, three more digits were added to the the current CP4s and were separated by an hyphen. This resulted in the seven digits postal codes (CP7), for example, 4049-001 for Porto City Hall.

Postal codes are organized following an hierarchical structure. The first number of the CP4 part corresponds to one of nine subdivisions of Portugal. Most of CP4s end with either a 0 or a 5, and generally refer to residential areas. However, entities with an high delivery demand can be supplied with their own CP4 and CP7, as it is the case with public facilities, large companies and so on. CP7 represent regions inside of a CP4.

The distribution of the postal codes is geographically and demographically dependent. Most of the times, a street is represented by a CP7, however in high residential areas, a street can be represented by several CP7. The other way around, even though, less common, can also occur, a CP7 can represent a set of streets where the volume of deliveries is small.

In Portugal there are a total of 750 CP4 and 197,834 CP7. In the Metropolitan Area of Porto there are 20,004 CP7.

3.3 Problem description

Last-mile delivery companies deliver parcels to a multitude of clients on each day. Drivers have an allocation of parcels that they are responsible for delivering during the working day. These companies make use of partitions to define drivers' areas of responsibility. These partitions are made of a set of postal codes. Postal codes exists as a means of helping the logistical and spatial organization of addresses, in the context of couriers and parcel delivery.

Drivers' areas of responsibility tend to stay similar on different days, because drivers get familiar with the area's characteristics, and, are able to make more informed decisions to benefit the delivery work and develop an healthy relationship with the customers. On different days, these areas are allowed some flexibility in order to adapt to different delivery demands.

The company, or a decision-maker, is responsible for defining this partition of the geographic space. A good definition is paramount for the company as it is one of the core components of the business logic. It directly affects the business success, a bad allocation can yield client dissatisfaction, drivers inability to fit his area of responsibility, unbalance of work between drivers, financial and ecological cost and so on. The number of partitions is known before each day and constant throughout the days.

From the case study, there are three main criteria that were identified as the most relevant. One of them is the similarity between allocations on different days, such that, different allocations must not differ too much from each other. Secondly, the efficiency of the allocation, as it is one of the most important metrics, because clients are expecting a delivery and value a company they can trust. Thirdly, logistic companies try to make partitions balanced between each other, so that drivers can have a fair and similar amount of work between each other.

The decision-maker has to consider a lot of variables in order to optimize the allocation of postal codes to drivers of responsibility. There are a lot of possible allocations and the decision-maker must choose the one that best fits, his or the company's, best interests. Note that the preferences of different decision-makers can be different depending on each interests, as such, the best-fit solution can vary.

In this work, the partition of the geographic space by allocating a set of postal codes to driver's is studied. The problem is focused on the delivery of parcels to clients. For each possible allocation there are different criteria that can influence the decision-maker decision, as such, it is important to find the set of solutions that optimize the criteria. Moreover, it is important to allow the decision-maker to make a thoughtful decision on the solution to adopt, with resource to a tool.

3.4 Problem data

Real world data provide real insight over the real problems companies face. However, real data is often incomplete, prone to outliers and errors, and complicated. Real data has a lot of complex interactions and behaviours that can guide and delude the search process through a wrong path. Often, an extensive and thorough data analysis must be conducted to process and filter the relevant data attributes.

Generated data can increase the data quality and confidence by providing complete and balanced data, it provides an adequate representation of the data for the problem at hand and allows to scale the data to different dimensions to assess different scenarios.

Many times, it is an iterative process, such that the solution developed to solve the problem can provide useful insight over the data generation and solution model, allowing for improvements on the models.

The data for this problem is consisted of a list of active clients for each working day, with the information of their location and the number of parcels for each client.

In this work, there was a data generation process to provide input data to this problem, which is explained in detail in chapter 4.

3.5 Problem statement

The problem developed in this work takes as input a list containing the active clients on each of D working days, $C = \{C_0, C_1, ..., C_D\}, C_k = \{c_0, c_1, ..., c_{|C_k|-1}\}$ and a graph representing the postal codes layout, $G = (V, \emptyset)$. Each active client is defined by their location and number of packages to be delivered, c_j. The active customers on each day, their location and demand are given in advance, they are deterministic. The graph G is only consisted by postal codes, labelled as V and no neighbourhood relationship between nodes of the graph. The depot for the problem is labelled as w.

In this problem, the BU $b \in V$ corresponds to a postal code, represented by a centroid in a 2D plane. Each BU has the set of clients active on each day.

A driver's area of responsibility, P, also known as sector, consists of a set of BUs, such that $P \subseteq V$. It represents a collection of BUs that a single driver is responsible for and for its clients.

The allocation of V into sectors is given by X and the number of sectors is given in advance and constant on every day, $|X| = L$. Each driver has a vehicle and the vehicles are homogeneous, which mean, they share the same characteristics. A feasible allocation can be represented by X, such that, $X = \{P_0, P_1, ..., P_{L-1}\}$ and P_k represents a partition of V, thus, $\bigcup_{k=0}^{L-1} P_k = V$ and $\bigcap_{k=0}^{L-1} P_k = \emptyset$. There is no known information for a previous allocation to take into account, so it is not a redistricting problem.

In order to validate the allocation defined in X, daily partitions are also defined as $X^d = \{P_0^d, P_1^d, ..., P_{L-1}^d\}, d \in \{0, 1, ..., D-1\}$. These daily partitions need to be disjoint but not necessarily cover the entirety of set V, $\bigcap_{k=0}^{L-1} P_k^d = \emptyset, d \in \{0, 1, ..., D-1\}$ and $C_d \subset \bigcup_{k=0}^{L-1} P_k^d, d \in \{0, 1, ..., D-1\}$.

An instance I, is consisted of a global allocation X and the set of daily allocation $\{X^0, X^1, ..., X^{D-1}\}$. The solution to the problem is the non-dominated front consisted of several instances, $S = \{I_0, I_1, ..., I_{|S|-1}\}$, such that $\forall_{x \in S} \nexists_{y \in S} y \preceq x$, from which the decision-maker can choose from.

3.6 Expected results

The goal of this work is to develop a system that aids a decision-maker in the last-mile delivery business to find the allocation of postal codes to driver's areas of responsibility that best fits his interests. There are three main processes for this, the first of them is to build possible allocation; the second to find the set of instances that minimize a series of constraints; and the last, to develop a tool on which the decision-maker can decide on the solution that best matches with his interests.

3.7 Summary

This chapter describes the proposed solution to address the logistics management and planning with resource to sectorization techniques.

Section 3.1 describes a real world example in detail, exposing the business logic, how parcels are managed and organized, business characteristics, etc. The company apply sectorization at two levels, one for defining the geographic areas and another for the drivers' zones. However, no algorithm approach is employed and it is done manually by empirical observation and knowledge.

In section 3.2, the background for the proposed solution is layout, abstracting from the business logic and formulating the problem, allowing to state where the solution will come into play. And section 3.3 describes the problem under study in this dissertation.

Lastly, sections 3.5 and 3.6 formalize the problem and expose the expected results to this work, respectively.

Chapter 4

Data Generation

Generated data ideally should be indistinguishable from real-world data so that the procedures built on this generated data can be applied and transposed to real-world scenarios. However, it is often impossible to achieve such high standards as it is very challenging to model all the possible interactions and dependencies the real world has. As such, a good model must lay its foundations on real-world context-specific data.

This chapter presents the process behind the data generation employed in this work. The methodology and possible approaches are discussed, and the implementation is explained.

For the data generation process, it was considered three distinct phases: (i) the definition of postal codes, (ii) the definition of potential clients, and (iii) the definition of delivery demand. Each of these phases is explored in its corresponding subsection. The first phase explores possible methodologies for getting the Postal Codes representation. In the second phase, potential clients are generated in each postal code. The third phase presents how the package demand can be simulated.

A medium-sized European metropolitan area was used as a case study, the Porto Metropolitan Area, also known as Área Metropolitana do Porto (AMP).

4.1 Methodology

In a real-world case, clients are not uniformly distributed across the geographical area. Their distribution can be dependent on several aspects, from demographics to geographic ones. In this work, the distribution of clients on geographical space is based on the population density. Highly populated areas have more people than lowly populated areas and thus, are more likely to generate a higher number of packages. Therefore, the idea behind the client generation process is to use a density function for the AMP to define potential clients and, consequently, the locations of packages.

A continuous probability density function for the AMP is not very handful for generating clients, and it is hard to define. However, it is possible to resort to a discrete function by using the definition of CP7. This representative unit is practical because it is small enough to represent the population accurately and is also used in the parcel business logic.

The location and density of potential clients can be generated through an approximation of the population probability density function.

4.1.1 Postal Codes representation

To the best of my knowledge, there are no known accurate and correct geometrical representation of the CP7 for the AMP. Some approximations were found, however they were incomplete. As such, there was the need to produce a geometrical representation for the AMP CP7s. For this purpose, three main ideas have emerged: (i) mapping of CP7 to coordinates, (ii) mapping of coordinates to CP7, and (iii) partitioning of the space.

The first idea involves mapping an extensive collection of publicly available addresses to their respective CP7s and GPS coordinates. Then, proceed to aggregate addresses with the same CP7 and obtain an approximation of the geometrical shape of the CP7. There are several approaches to this. The first that came to mind was to apply the Convex Hull algorithm, which produces the smallest convex shape that encloses all points of that CP7. The second approach was similar but applying alpha-shapes, producing bounding shapes that enclose all points. These shapes are not necessarily convex and depend on a parameter α, so that a disk of radius $\frac{1}{\alpha}$ can be constructed between every pair of points [21]. Thus, it is essential to specify an adequate value for α. One problem with these two approaches was that the available data contained few points for an accurate representation of the CP7. Some locations were poorly represented, and the geometrical shapes obtained did not represent a correct partition of the geographic space, with intersections and empty spaces. Moreover, the several addresses and CP7 were mapped to the same GPS coordinate.

The second idea was inspired by the Monte-Carlo method. A large sample of random coordinates inside the geographical area was generated and fed into a publicly available maps API, such as OpenStreetMap. The API would then return the address of those coordinates from which the CP7 attribute is part of. After a large number of requests, a mapping from coordinates to CP7s would be achieved. The next step would be to construct the geometrical representation of the CP7s as in the previous idea. However, the same problems would arise, intersections and empty spaces. If the collected data were big enough, the problem would be on a smaller scale. The main problem of this method is that for a large area, this method can be resource expensive and needs many coordinates to achieve good representation on smaller and more intricate CP7.

The third and employed method is to approximate the CP7 regions by a plane partitioning algorithm. The geometric representation of the centroid of each CP7 is publicly available. The application of the Voronoi Diagram produces a partition of the geographical space into several Voronoi cells corresponding to each CP7. Voronoi cells are convex polygons such that any point inside that Voronoi cell is closer to its centroid than to any other centroid. This method enforces an approximation of the CP7 region.

4.1.2 Potential clients

As CP7 tends to be small regions, one can discretize the population density function accordingly to the CP7 representation through the Voronoi Diagram.

The following step is to associate the population density with the partition of the space. The population density is not available for each CP7; however, in Portugal, the Instituto Nacional de Estatística, INE, performs Census every 10 years to analyze demographic data on the country's population. The last one was performed in 2021. Census contain information about the demography of Portugal, and one of such information is the number of residents in each "subsection". "Subsections" are divisions of the smaller administrative unit, also known as parish or freguesias [14]. By crossing the data from the Census, population per subsection, and intersecting with the Voronoi Diagram, an approximation of the population in each CP7 can be derived.

After each CP7s population has been associated, there is a generation of potential clients inside each, according to its population count. These points represent potential clients and are distributed based on the population density across the geographic area.

Potential clients are generated uniformly inside each CP7 representation. Another approach would be to generate clients on top of a representation of the streets to describe the clients' location more accurately.

4.1.3 Delivery demand

In this subsection, the process behind generating demands for each working day is explained, which will output the generated data that will serve as input for the problem.

Clients order packages, and at some point, the package arrives at the parcel delivery company to be delivered to their address. The package delivery demand fluctuates and changes gradually over time, with seasonal and demographic variations, to name a few.

At first, an agent-based system was considered, with positive and negative feedback loops to keep the system evolving without exploding into chaos or dying into stagnation. Each client had a probability of having a package to be delivered. If a package were delivered, the probability would increase. Over time, this probability would decay. This approach gave interesting results; however, controlling the system's behavior was difficult, and empirical observations defined the parameters. For example, defining an average value for the number of packages per day or enforcing small changes over a specific period.

The employed method to generate the demand for each day assumed a Poisson distribution. Let $Pois(\lambda)$ be a Poisson distribution with λ as the expected number of packages per client per day. Let *target* be the number of active clients, in other words, the number of clients with at least one package to be delivered, and N, the total number of potential clients. This can be formulated considering the probability of a client having at least a package, $P(X > 0)$, as shown below:

$$P(X > 0) = \frac{target}{N} \tag{4.1}$$

$$\Leftrightarrow \frac{\lambda^0}{0!} e^{-\lambda} = 1 - \frac{target}{N} \tag{4.2}$$

$$\Leftrightarrow \lambda = -\ln(1 - \frac{target}{N}) \tag{4.3}$$

This produces an expected number of active clients equal to *target*, but the number of packages to be delivered will be greater or equal to *target*, as each active client has at least one package to be delivered.

The Poisson approach works best for defining an average expected value; however, it fails in modeling the system's dynamic behavior, as each iteration is not dependent on any previous iteration. Also, the Poisson distribution is lower bounded to 0 and not upper-bounded, allowing for an infinitely large number of packages. For a large enough value, the probabilities are very small and approach 0, thus they can considered 0. In this work, such probabilities are equalized to 0.

By running this method for several iterations, a list of active clients and their corresponding packages to be delivered for each day is generated.

4.2 Implementation

In this section the details about the implementations of the processes detailed above are explained.

4.2.1 Postal Codes representation

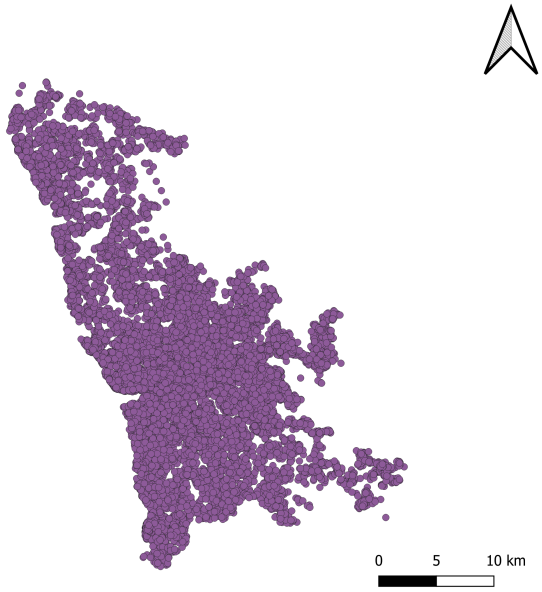
For this phase, the program QGIS on version 3.24.1-Tisler was used to perform the operations needed to conduct this work. This software allows the treatment, analysis, and process of geo-referenced data.

A shapefile containing all postcodes of Europe was fed into the system. An intersection of the postal codes with the AMP polygon filtered all the CP7 that is part of AMP, as figure 4.1a shows. The Voronoi Diagram was then built from the AMP CP7s. As the resulting structure extends infinitely to the borders, it is cropped by the AMP polygon. Figure 4.1b shows a zoomed-in region of the resulting structure, more specifically, in the city of Porto.

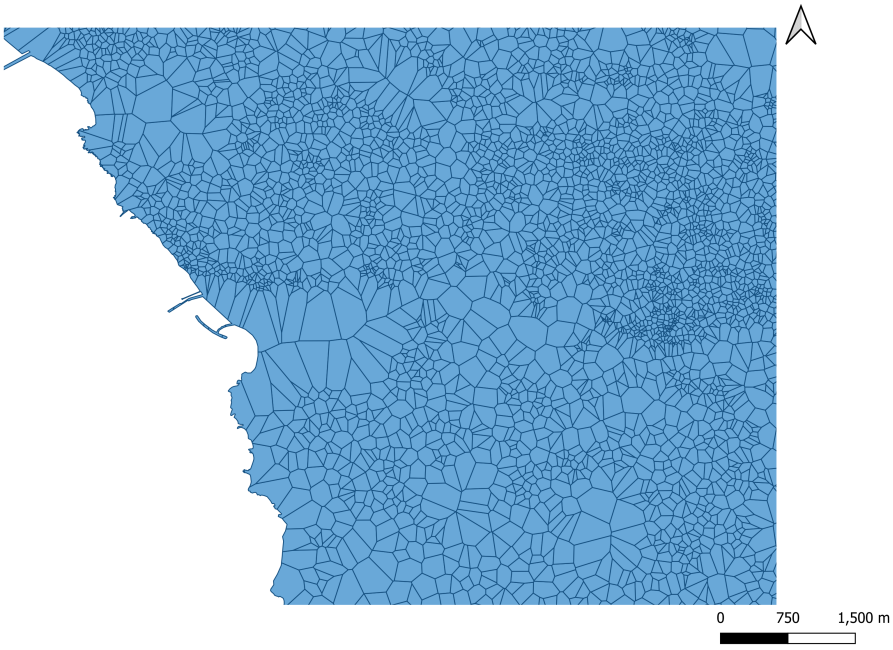
The data collected consisted of 20,004 CP7s, however, a portion of them was mapped to the same coordinates. As such, filtering the CP7s, resulted in a Voronoi Diagram with a total of 17,318 Voronoi cells. This set of CP7 was the representation used for the AMP.

4.2.2 Potential clients

The Census data from the INE contains different demographic attributes sorted by subsections and a shapefile representing those subsections. The data is hierarchically organized to different



(a) CP7s of AMP.



(b) Zoomed-in region of Voronoi Diagram.

Figure 4.1: AMP geospatial information.

resolutions, for example, entire Portugal, North of Portugal, or even by counties. For the purpose of this work, it is focused on the AMP region. At the time of this work, the 2021 data was not publicly available; thus, the 2011 data was used. The representation of the AMP and its subsections are presented in figure 4.2

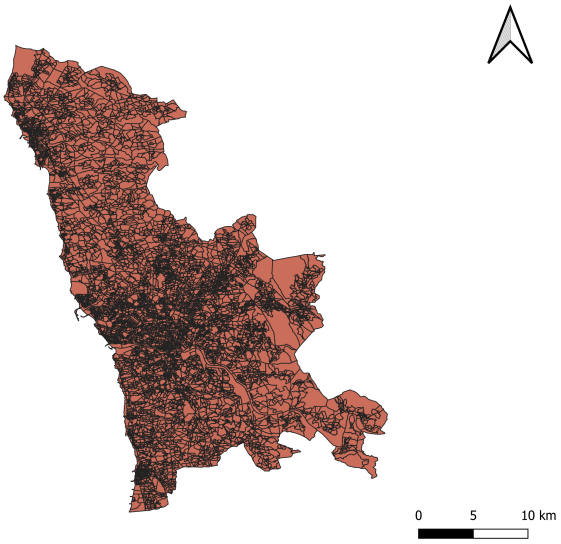


Figure 4.2: AMP subregions.

At this stage, the Voronoi cells have no associated population; thus, the next step is to associate the population of each of the subregions from the 2011 Census with the Voronoi cells. One possible way to achieve the desired result is to intersect the subregions from the Census with the Voronoi cells; this assumes a uniform population distribution within the subregions. By intersecting the subregions with the Voronoi cells, the results consist of tiny sections that can be joined to either make up the Voronoi cells or the subregions. By calculating the ratio of areas between a section and the subregion that originated it, the expected population in each tiny section,$pop_{section}$, is calculated, following the formula in 4.4. Figure 4.3 shows a zoomed-in region of the resulting sections by intersecting the AMP with the Voronoi cells.

$$pop_{section} = pop_{subregion} * \frac{area_{section}}{area_{subregion}} \tag{4.4}$$

Having calculated the population by section, it proceeds to group sections that are part of the same Voronoi cell while summing the population from each constituent section, resulting in the population by Voronoi cell, pop_{cell}. In the end, a representation of the Voronoi cells with the demographic information is achieved. In figure 4.4a and 4.4b it is observed, respectively, the

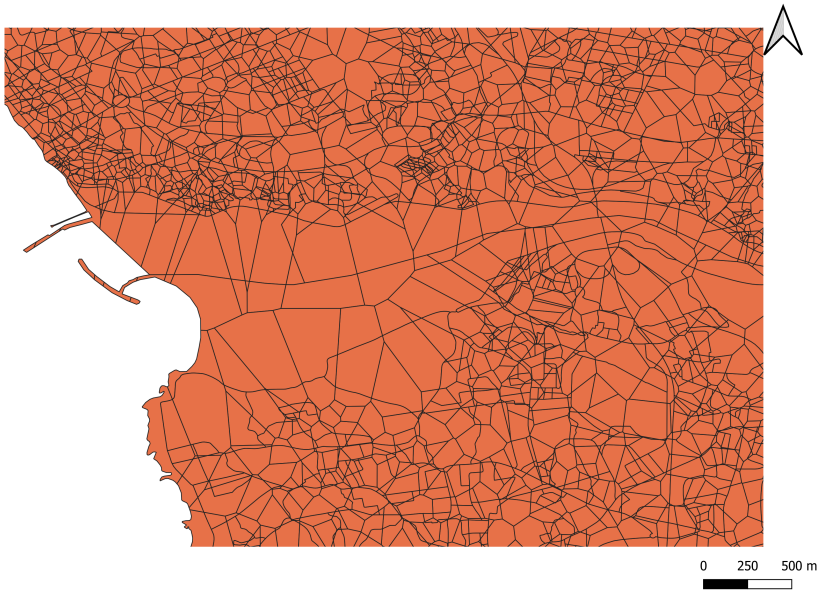


Figure 4.3: Zoomed in section resulting from the intersection.

population by Voronoi cell for the AMP and the population density by Voronoi cell zoomed-in in the Porto region, which was calculated by dividing the population count by the area.

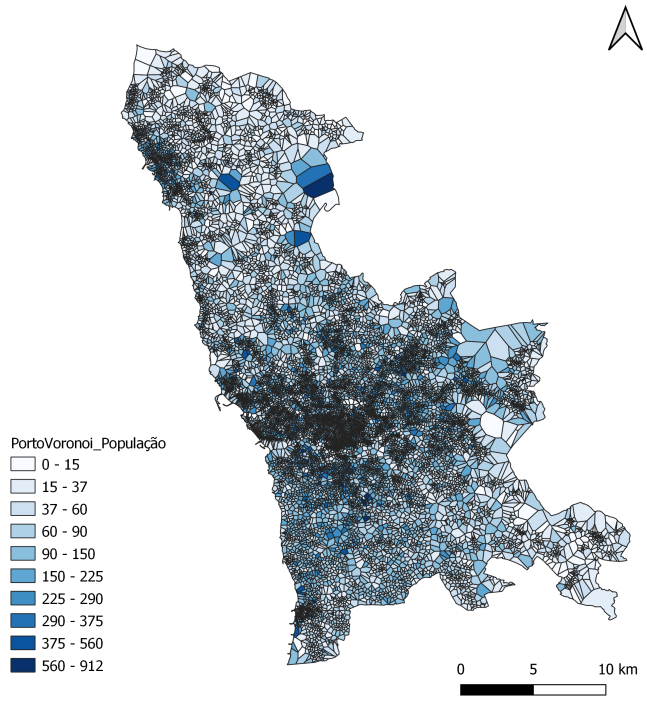
As the final step of this phase, random points representing potential clients are generated inside each Voronoi cell, accordingly to the population count. As points are distributed uniformly inside the Voronoi cell's area, they will be distributed accordingly to a discrete population density function. A zoomed-in example can be seen in figure 4.5, where a more significant amount of points occurs in high-density regions and a smaller number of points in low-density regions.

The number of points on each cell was not necessarily equal to the population count. In order to have greater control over the number of points, num_{points}, and avoid situations where the population count was less than 1, thus yielding 0 clients, the formula in 4.5 was employed.

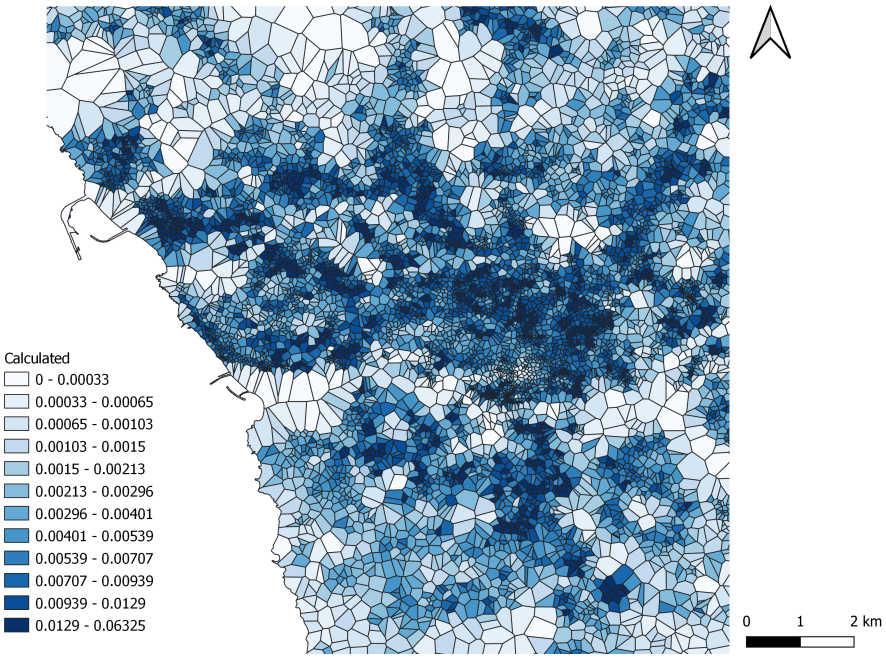
$$num_{points} = \lceil \beta * pop_{cell} \rceil \tag{4.5}$$

where β is a scaling factor, and, pop_{cell} is the population of each Voronoi cell. In this work, a $\beta = \frac{1}{5}$ was used. This value is justified by the large population count, 1,258,991 inhabitants, resulting in 266,126 total potential clients. The ceiling function guarantees that a cell always has at least 1 potential client, unless that cell is unpopulated.

This work considered the population of each subregion to be the resident population of each subregion. Other demographic attributes are available, from the number of present individuals and even discrimination by sex and age gap, which could also be used to map different consumer profiles and behaviors.



(a) Population count per Voronoi cell in AMP.



(b) Population density per Voronoi cell in Porto.

Figure 4.4: AMP demographic information.

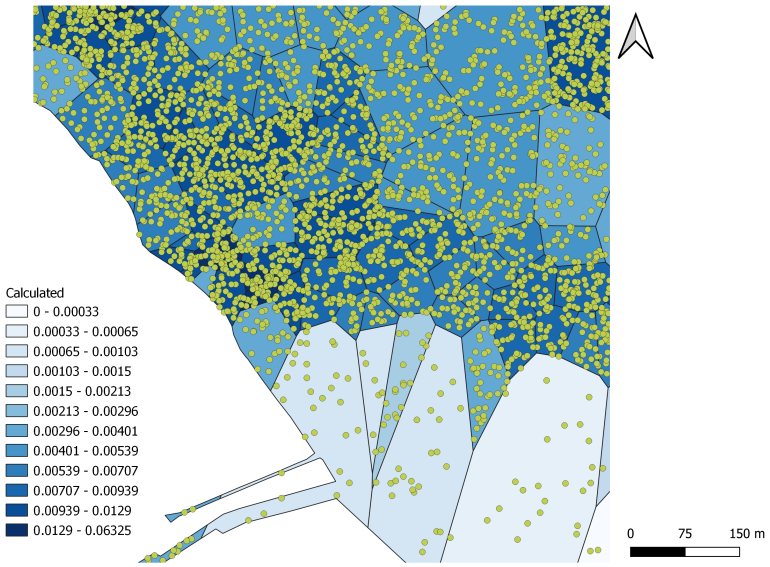


Figure 4.5: Random clients in CP7 Voronoi cells.

4.2.3 Delivery demand

In this phase, the aim is to produce the delivery of each day. As explained before, a Poisson distribution-based approach was employed.

A program was developed in Python, which handled the data created in the previous phase and generated delivery demands for several working days. The data was imported from CSV file, comprising the coordinates of each client created and the respective CP7.

The pseudocode in Algorithm 1 shows how the data is generated for each day. The algorithm takes as input the potential clients created in the previous phase, C, the parameter λ defined by the formula in 4.1 and the upper bound for the number of packages per client, as *max_packages*, and return a list with the active clients with a package to be delivered on that day.

Algorithm 1: Generate daily delivery

Data: $C, \lambda, max\_packages$
Result: $active\_points$

```
1  active_points ← ∅;
2  for c ∈ C do
3      c.packages ← Pois(λ);
4      if c.packages > max_packages then
5          c.packages ← max_packages;
6      end
7      if c.packages > 0 then
8          active_points ← active_points + c;
9      end
10 end
```

Line 1 initiates the list of the active clients as an empty set. The loop from 2 to 9 iterates over the clients and determines whether they have packages to be delivered or not. Line 3 generates the number of packages to be delivered accordingly to a Poisson distribution with an expected value of λ. Line 4 and 5 apply the upper bound as *max_packages*. Line 7 and 8 add the client to the list of active clients if any packages are to be delivered.

For this work, the desired number of active clients per day, *target* in 4.1 was set to 3000, as it goes along the lines of the insights provided by the company.

Chapter 5

Handling the Problem

The chapter describes the approach followed in handling the last-mile delivery problem. Includes a discussion of all solution methods, techniques, and requirements needed to solve the problem. The diagram in Figure 5.1 is intended to illustrate the conceived approach, with the different procedures and techniques and the way they are articulated.

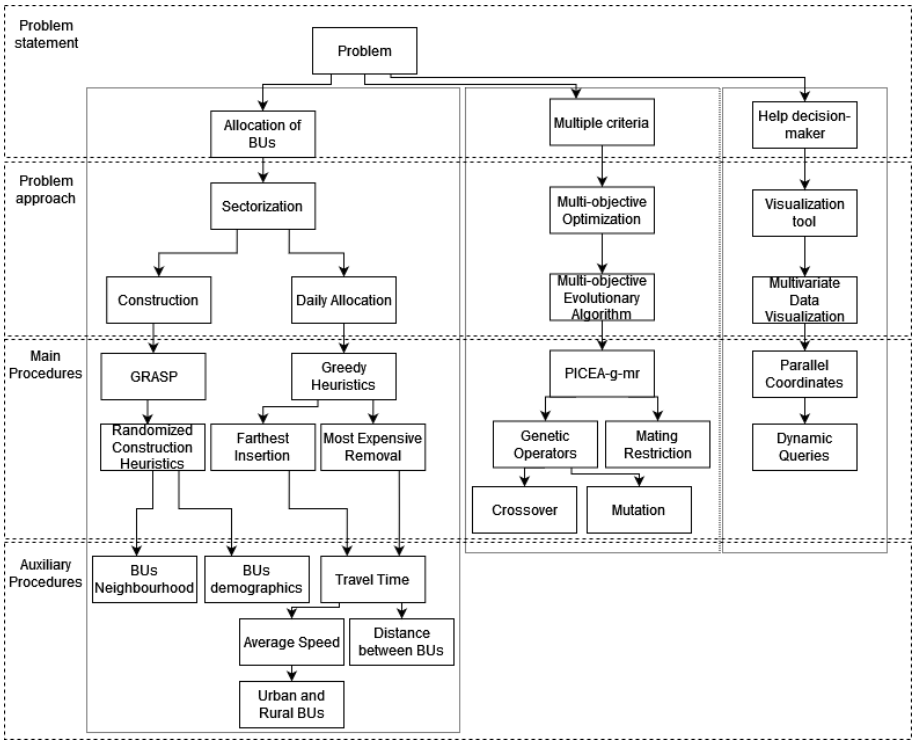


Figure 5.1: Diagram of the approach followed.

5.1 Approach

The problem implicates allocating postal codes to drivers' areas of responsibility to fit the decision-maker's best interests. Postal codes represent a small, meaningful, working partition of the geographic space, which is useful for the last-mile logistics inner-workings. Another logistics company can find that another representation better suits their business logic. From now on, postal codes, CP7s, are considered the individual elements of the problem and can be transposed to some other representation that better suits the problem's requirements. This fundamental element is called the basic unit (BU).

The allocation of the BUs to drivers' areas of responsibility is a complex task, as there are several context-specific requirements and criteria; thus, resorting to Sectorization was used to deal with that. Sectorization allows for the partition of the geographical space into sectors, such that a problem is more easily solved, and it has a lot of aspects similar to the task of allocating BUs. So, the use of Sectorization is adequate for this context.

Several criteria are regarded in each BU allocation; however, they often conflict with each other; a decrease in the value associated with one may lead to an increase in another. A solution to the problem will be a set of allocations that meets such conflicts - hence the use of multi-objective optimization and satisfies the decision-maker's interests.

In order to achieve the set of allocations that represent the solution to the problem, a multi-objective optimization method is applied, specifically a Multi-Objective Evolutionary Algorithm. These kinds of algorithms are used to solve multi-optimization problems and have a family of instances representing the solution to the problem, thus adequate for this problem.

In this problem, the allocation of BUs is the primary process under study; however, the driver's behavior is also modeled to validate the allocation. By this, a daily allocation and respective driver's routes are computed to provide comparability between solutions.

In order to minimize the disparity between sectors on different working days and the computational effort in computing the allocation for each day, a concept of rigid centers and flexible borders was employed. A rigid center would be a set of BUs in the center of the sector that will always belong to that sector on any given day; thus, a driver will always be responsible for those BUs. On the other hand, flexible borders are BUs located on the border of a sector and, on a given day, could belong to another sector to accommodate the delivery demand and balance the workload between sectors. This concept aims to reduce the problem complexity while not jeopardizing the solution quality.

Thus, allocating the border BUs to sectors for each working day is necessary. For this, it was used a greedy heuristic and a deterministic procedure. A critical aspect of this problem is the need to define delivery routes to validate the quality of the sectors. As such, two different approaches were possible.

One of them was to reallocate the border BUs for each day as a redistricting problem, making use of an heuristic and then build the routes of each sector, taking into account the active clients on each day.

Another approach is to reverse the logic and build the routes for each sector first and only then, based on the routes, reallocate the border BUs to sectors. However, this approach removes the need to assign BUs daily as the routes are already defined. This approach was the opted one.

5.2 Auxiliary Procedures

In order to support the solution approach to this problem, some supplemental procedures were developed. These procedures are not part of the approach itself; however, they play an essential role in modeling the approach and enhancing the solution.

5.2.1 Basic Units demographics

The basic units of this problem are the CP7s. The driver's areas of responsibility, also known as sectors, consist of CP7s. Active clients belong to a CP7, and on its delivery route, a driver only leaves the BU after he has served all its clients.

There is relevant information that can be obtained to help solve the problem and guide the search process to a better solution quicker. One assumption that was made due to empirical knowledge is that the number of packages is greater in areas where there are more inhabitants, as more people have the potential to make an order and be active clients. As such, one could extract the population per CP7 and try to build sectors that cover an equal amount of population, assuming that an equal amount of population will produce an equal amount of work for each working day.

Motivated by this assumption, a process similar to the one used in the data generation, chapter 4, is applied, as in sections 4.1.1 and 4.1.2. In the end, a representation of the BUs is obtained, consisting of the centroid coordinates, the estimated population count, and the estimated population density for each CP7.

5.2.2 Neighborhood

At this stage, the basic units are just points in space with no neighborhood relation; thus, it is impossible to define any connectivity relation. Within a sector, it must be possible to travel between two BUs without leaving the sector. So, it is essential to define a neighborhood relation between BUs. For this, some ideas were studied.

The goal of the neighborhood is to produce edges between CP7 that share a connection and are accessible from one another in the real world, such as by roads, bridges, and so on. As the actual shape of each CP7 is not accessible, some approximation must be conducted. Some relevant observations are that edges must connect CP7 close to each other, often sharing a border, and edge-crossing is not desirable.

One possibility is to consider the k closest CP7 as neighbors of some other CP7, also known as the k-Nearest Neighbour Graph, k-NNG. This produced satisfactory results for a small $2 < k < 6$. However, crossing edges is an evident factor; the relationship is not symmetric, and edges are biased towards more dense areas than sparse ones.

The method chosen for this work was the Delaunay Triangulation, DT, closely related to the Voronoi Diagram. The DT is obtained by connecting every two points whose Voronoi cells share an edge. It produces a crossing-free and angle-optimal triangulation.

Other subsets of DT were also considered, such as the Gabriel Graph and the Relative Neighbourhood Graph. Gabriel Graph states that two points, p and q, that share an edge must have an empty circumference between them of diameter $d(p,q)$, which is the euclidean distance between p and q [43]. On the other hand, the Relative Neighbourhood Graph states that two points, p and q, share an edge, if there is no point r that is both closer to p and q than p and q are to each other [35]. They both do not produce a triangulation of the CP7s but produce a neighboring relationship that could be worth exploring.

Another alternative that, as far as known, has not been considered before is the use of a geometric spanner, also known as $t-spanner$, to form the neighborhood graph. The t-spanner achieves the property that the shortest path length between two points, p and q, is at most $t \times d(p,q)$, where t is known as the stretch factor [1], which could be interesting to explore.

Edges might fall outside the AMP polygons, especially on the DT, as it forms a convex shape, and AMP is not a convex polygon. So a filter of the edges that are fully contained within the polygon could be considered. However, even though an edge falls outside the polygon, it is not impossible for two BUs to be connected, as there could be traffic ways between those two BUs that go outside the AMP polygon.

5.2.3 Average speeds

The calculation of travel time between two different points or BUs is generally calculated by dividing the distance by the average velocity:

$$t(p,q) = \frac{d(p,q)}{v(p,q)} \tag{5.1}$$

where $t(p,q)$ is the travel time between points p and q, $d(p,q)$ is the distance between points p and q, and, $v(p,q)$ is the average velocity between points p and q.

However, this is far from reality; the average speed between BUs at a similar distance is not always the same. In urban areas, there are traffic lights, traffic jams, and low-speed limits, whereas in rural areas, the traffic is less, and speed limits are higher; thus, the average velocity in rural areas tends to be higher than in urban areas.

The first idea was to consider three possible travel types: urban to urban, urban to rural (and vice-versa), and rural to rural. This representation would express the three possible transitions from one region to another. Then, for each of these region types there would be a respective average velocity.

However, one might argue that when traveling a greater distance, the driver tends to choose faster travel ways and will stay in them for longer. Take, for example, an urban to urban travel; inside a city, if a driver has to travel a short distance, he will tend to choose the shortest path

through the available roads. In contrast, if he has to travel a longer distance, he might take an arterial road which would take him to his destination faster, at a greater average speed. Following this line of thought, the average distance should also consider the distance between points. For this, consider the following equation for the average velocity between two points:

$$v(x, v_{min}, v_{max}, l, d) = \frac{1}{1 + e^{-\frac{l}{d} \times x + l}} \times (v_{max} - v_{min}) + v_{min} \tag{5.2}$$

The following equation is based on the sigmoid function and its parameters are x, which represents the distance between two points, v_{min} which is the minimal velocity, v_{max} which is the max velocity, l is a coefficient which determines the slope of the function and d is the value for x at which the function return the mean between v_{min} and v_{max}.

It achieves some interesting properties: $v(0, v_{min}, v_{max}, l, d) = \frac{1}{1+e^{l}} \times (v_{max} - v_{min}) + v_{min}$, which for a big enough l approximates $v(0, v_{min}, v_{max}, l, d) \approx v_{min}$; $v(d, v_{min}, v_{max}, l, d) = \frac{v_{min}+v_{max}}{2}$; $v(\infty, v_{min}, v_{max}, l, d) = v_{max}$. It provides an increasing and continuous function for the average velocity, that aims to simulate transitions from lower speed traffic ways to higher speed traffic ways, start and end of a trip and approximation to a maximum average velocity when driving for long periods.

For each of the three travel types, define a set of parameters that represent the average velocity in function of the velocity. Consider the distance in meters and the velocity in km/h.

Urban to urban = $v_{u,u}(x) = v(x, 15, 30, 5, 2000)$

Urban to rural = $v_{u,r}(x) = v(x, 25, 40, 5, 5000)$

Rural to urban = $v_{r,u}(x) = v(x, 25, 40, 5, 5000)$

Rural to rural = $v_{r,r}(x) = v(x, 30, 60, 5, 10000)$

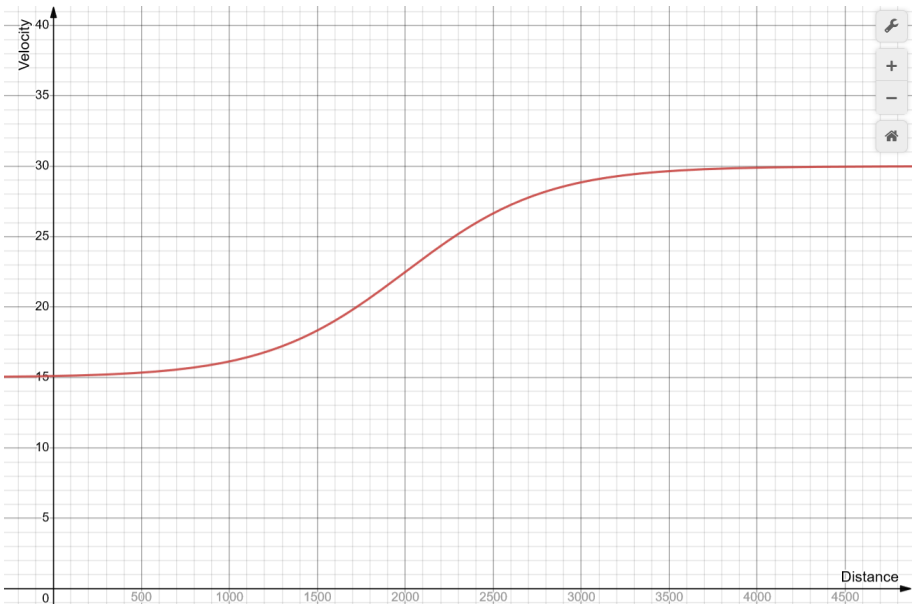


Figure 5.2: Urban to urban average velocity in function of the distance.

5.2.4 Distance between entities

The BUs and client data have their GPS coordinates in longitude and latitude. The distance between two entities is needed to calculate the average velocities and travel times. Calculating the Euclidean distance between the two coordinates would not be very accurate and would not be meaningful when dealing with real world values. For small enough distances the approximation might be acceptable, however, for larger distances it would not account for the curvature of the Earth's surface. Moreover, when dealing with velocities it is wise to translate the distance to meters, for a more accurate handling.

In order to calculate the distance between two entities, the Haversine formula was used. It returns the shortest distance on a sphere between two sets of coordinates [46]. It is a particular case of the more general law of cosines.

The Haversine formula states that the distance d is equal to:

$$d(\varphi_1, \varphi_2, \phi_1, \phi_2) = 2R_{earth} \times \arcsin\sqrt{\sin^2(\frac{\varphi_2 - \varphi_1}{2}) + \cos\varphi_1 \times \cos\varphi_2 \times \sin^2(\frac{\phi_2 - \phi_1}{2})} \quad (5.3)$$

where φ_i represents the latitude and ϕ_i the longitude of point Q_i, and, R_{earth} the radius of the Earth, in this case, 6,371,000 meters. On possible modification would be to use a straight path in rural areas and the Manhattan distance in urban ones, to accommodate city/street planning.

5.2.5 Urban and Rural areas

In order to define the travel type transitions involving rural and urban regions, the need arises to define which BUs belongs to urban and rural regions. The first thought is to think about the population density.

The authors in [19] propose a new definition for rural and urban areas. They distinguish between 3 regions: high-density clusters as contiguous grid cell of 1 km^2 that have at least 1,500 inhabitants per km^2 and a minimum population of 50,000; urban clusters with a density of at least 300 inhabitants per km^2 and a minimum population of 5,000 and the rest as rural areas.

This work considers only two regions, urban and rural, and some insight can be extracted from the definitions presented before. The urban areas will consist of BUs with a minimum density and population. Everything else will be set as rural area.

As such, the procedure will consist of setting BUs as urban or rural and then counting the population on each cluster. However, there is one problem with such implementation: outliers, like low-density areas inside cities (parks, hospitals, public services) or isolated high-density areas classified as urban, with no other around, would exist.

In order to mitigate this problem, it is proposed a procedure that for each BUs checks the ratio of neighboring CP7s that are urban. If it is above some threshold, it is upgraded to urban otherwise, if it is below some other threshold, it is downgraded to rural. The process is repeated until there are no more updates on the BUs.

An algorithm similar to Flood Fill is used for counting the total population. A Breath-First-Search on the connecting urban BUs is conducted to find all the urban BUs that are part of the same urban cluster, while counting the population. Every cluster that has less than the minimum population is set to rural.

5.2.5.1 Data

The data that serves as input was obtained by the process in chapter 4. The data consists of a set of working days, each with a list of the active clients on that day. Each active client has a number of packages to be delivered.

There is no dependence on the working days with one another, which means that their order is not significant, they could be randomly ordered, and their meaning would be the same.

5.3 Main Procedures

5.3.1 Solution representation

The representation of a solution goes through partitioning the set of BUs into L sets. For this, one can represent an instance S as a 1D integer array of $|V|$ elements numbered between 0 and $L-1$, representing the sector each BU belongs to, with $|V|$ being the number of BUs and L the number of sectors; $S : \{r_0, r_1, ..., r_{|V|-1}\}, r_i \in \{0, 1, ..., L-1\}$.

This representation is enough to get the necessary definition and helper structures described in the previous sections, such as borders and routes for each day, as they rely on deterministic definitions, heuristics, and procedures.

5.3.2 Flexible borders

This work employs the concept of rigid centers and flexible borders. In this subsection, the definition of borders is detailed.

A border-radius is defined, stating how large the border is. The smallest border would consist of BUs that have another BU from another sector as a neighbor, known as border level-zero. Having this first definition of a border level-zero, one needs to define the concept of border-radius. Two possible approaches were thought. The first one considers a distance metric, such that every BU that lays at r distance of a same-sector border level-zero BU would also be considered a border. Another approach considers several layers of borders, such that a BU is a border level-r if it has a same-sector border level-$(r-1)$ neighbor. The second method was chosen; below it is presented the formulation:

$$x \in b_{P_i}^0 \implies x \in P_i \wedge \exists n \in N_x | n.s \neq P_i \tag{5.4}$$

$$\forall k \in \{1, 2, ..., r\}, x \in b_{P_i}^k \implies \exists n \in N_x | n \in b_{P_i}^{k-1} \wedge n.s = x.s = P_i \tag{5.5}$$

where x represents a BU of sector P_i, $b^k_{P_i}$ represents the border level-k of sector P_i, N_x represents the neighbours of basic unit x and $x.s$ represents the sector of basic unit x. Statement 5.4 states that elements of a sector that have a neighbour of another sector are considered border level-zero. And statement 5.5, states that elements are considered border level-k if they have a neighbour of the same sector of border level-$(k-1)$.

5.3.3 Optimization criteria

This subsection explains the optimization criteria considered for this problem. In total, there are four criteria to optimize: Compactness, Balance, the Efficiency of the allocation, and Dissimilarity. All of this are meant to be minimized.

5.3.3.1 Compactness

The idea behind Compactness has been described in 2.2.3. In this work, the definition of Compactness for each design consists of the sum of maximum distance of a sector's BU to a dummy center for each day. The dummy center for a sector P_i, $c(P_i)$, is an abstract point that results from the average between all points from the allocation of that sector.

$$comp(I) = \sum_{d=0}^{D-1} \sum_{P_i^d \in X^d} max_{b \in P_i^d} d_{c(P_i^d),b} \tag{5.6}$$

$$c(P_i^d) = \left(\frac{\sum_{b \in P_i^d} b.x}{|P_i^d|}, \frac{\sum_{b \in P_i^d} b.y}{|P_i^d|} \right) \tag{5.7}$$

where $d_{c(p),b}$ is the distance of the BU b to the center of its sector. The distance used in this formulation was the Euclidean distance between the two coordinates. A more accurate representation could use the Haversine distance. The center of a sector P_i, $c(P_i)$, is calculated by the average of the points location, where $b.x$ and $b.y$ are the x-coordinate and y-coordinate of the BU b.

5.3.3.2 Balance

Balance aims to level the amount of work between each sector and driver. Ideally, each driver should have the same amount of work. In many cases, the average amount of work per driver can be calculated beforehand. For example, one could consider that the number of deliveries/packages should be the same for every driver. As the number of packages is deterministic for each day, it is possible to calculate the number of packages per driver.

For this work, it was considered the amount of work as the number of minutes each driver is expected to need to complete his assigned delivery, in other words, the time cost of his route. In this formulation, it is impossible to know the average time a driver takes to complete a route as it depends on the allocation. However, it is possible to define the optimal number of minutes a route

should have, t_{opt}, which can be seen as a working day for the driver. In other words, sectors' routes should be balanced at t_{opt} minutes. Thus, it is defined the following formulation for balance:

$$bal(I) = \sum_{d=0}^{D-1} \sum_{P_i^d \in X^d} bal_{sector}(P_i^d) \tag{5.8}$$

$$bal_{sector}(P_i^d) = (t_{P_i^d} - t_{opt})^2 \tag{5.9}$$

where $t_{P_i^d}$ is the expected time cost of the route in sector P_i on day d. This formulation has two objectives, the first one aims to encourage the transfer of work from over-worked sectors to under-worked sectors resulting in a global decrease; and the second one is to encourage the transfer from highly under-worked/over-worked to low under-worked/over-worked, also resulting in a global decrease. This formulation penalizes highly imbalanced allocations.

5.3.3.3 Efficiency

One of the most critical metrics for a last-mile parcel delivery company is the percentage of packages delivered. As such, it was considered as optimization criteria for this problem, which is called efficiency.

$$efficiency(I) = 1 - \frac{\sum_{d=0}^{D-1} \frac{\sum_{p \in X^d} pack_p}{|pack^d|}}{D} \tag{5.10}$$

where $pack_p$ is the packages delivered on the route of sector p on day d and $pack^d$ the set of packages to deliver on day d. The expression calculates the average percentage of packages delivered per day.

5.3.3.4 Dissimilarity

Dissimilarity aims to calculate the difference between different daily allocations. As the data has no dependence between days, a formulation considering allocations on consecutive days would not work. As previously mentioned, the centers' BUs would always belong to the same sector. Only those on the border can belong to another sector. The first idea was to consider the number of basic units that have a different sector assigned between the daily and global allocation and sum this count for each of the days as the following formulation:

$$dissimilarity(I) = \sum_{d=0}^{D-1} \sum_{r \in L} \sum_{u \in X \setminus P_r} |P_r^d \cap P_u| \tag{5.11}$$

The following formulation counts, for every day, the number of basic units of sector P_r for each day d from another sector P_u global allocation. However, one would like to give more weight

to basic units farther away from the sector's border. If two border BUs belong to a route that is not in their original sector, then the one farther away from that route's sector has a more significant contribution than another BU that is closer.

$$dissimilarity(I) = \sum_{d=0}^{D-1} \sum_{r \in L} \sum_{u \in X \setminus P_r} \left(\sum_{b \in |P_r^d \cap P_u|} \frac{d_{c(P_u),b}}{max_{i \in P_u} d_{c(P_u),i}} \right) \tag{5.12}$$

where $|P_r^d \cap P_u|$ is the set of BUs of P_u that are daily allocated to P_r^d; $d_{c(P_u),b}$ is the distance from the center of the sector and $max_{c,i \in s} d_{c(P_u),i}$ is a normalization factor that consists of the maximal distance of the center to a basic unit of that sectors global allocation.

Remember that this is only applied to the routes, as there is no need to assign the remaining BUs that are not active. However, another formulation was built where an assignment of the BUs was conducted from the routes. For this, the dissimilarity measure had its basis on the entropy formulation, where the frequency of each point belonging to a sector was taken into account:

$$dissimilarity(I) = -\sum_{P_i \in X} \sum_{b \in P_i} f(b, P_i) \times \log_D(f(b, P_i)) \tag{5.13}$$

where $f(b, P_i)$ is the frequency that basic unit b belonged to sector P_i during the D days. It was interpreted that this metric was more meaningful in a situation when full allocation of the BUs on each day is applied. On this formulation, if a BU always belonged to the original sector, then, that BU would have a null contribution to the measurement.

5.3.4 Construction of global allocation

The construction procedure aims to return a feasible and good-quality allocation. An approach similar to GRASP was considered. GRASP works by generating randomized solutions and then applying a local search; however, this work is only interested in the first, the Randomized Construction Heuristics.

A Randomized Construction Heuristics or Randomized Greedy Heuristics aims to produce somewhat random solutions with good quality. A totally randomized construction can produce every feasible solution either with good or bad quality but has no information or search process when building it; thus, most of the time, the solution quality is not satisfactory. A greedy construction uses heuristics to give information during build process when generating the solution by always choosing the most promising candidate and tends to produce a good quality solution; however, rarely the best. The idea behind the Randomized Greedy Heuristics is that the optimal solution might be in the neighborhood of the greedy solution. Thus, introducing some degree of randomization when choosing a candidate can generate different, good-quality and promising solutions.

The Randomized Greedy Heuristics iterates over an incomplete solution and keeps adding new components until reaching a feasible solution. These components come from a Candidate List, CL. The CL is comprised of all the possible candidates to an incomplete solution and their corresponding heuristic value. The CL is reduced by selecting the most promising candidates, the Reduced Candidate List, RCL. Two simple approaches to determine the RCL are: take the k cheapest candidates from the CL, which is known as fixed-size RCL or to add the candidates that differ by α from the cheapest solution, $h(c) <= h(c_{min}) + \alpha \times (h(c_{max}) - h(c_{min}))$, where $h(c_{min})$ and $h(c_{max})$ are the heuristic value of the minimum and maximum candidate, which is known as dynamic-size RCL. It was opted for the last one.

As an incomplete solution, a set of random basic units was defined, as many as there will be sectors, known as seeds. Seeds are the starting point of sectors, and from there, sectors will expand and grow during the Randomized Greedy Heuristics. The CL consists of BUs that are not assigned to a sector and are neighbors of a BU assigned to a sector. The heuristics calculates the costs of assigning that basic unit to that sector. A basic unit can be a candidate for several sectors simultaneously.

The heuristics used for calculating the candidate's value considers the Compactness and the balance of the sector. The Compactness was measured according to a definition in 5.3.3.1. However, the balance, for this case, was defined according to the amount of population on the assigned BUs, assuming that sectors with the same population have an equal amount of work expected. The two metrics described are combined to form the candidate's heuristic:

$$h(P_i, b) = max_{u \in (P_i \oplus b)} d_{c(P_i),u} \times \frac{\sum_{u \in (P_i \oplus b)} u.pop}{\overline{pop}} \tag{5.14}$$

where $h(P_i, b)$ is the heuristic function of assigning BU i to sector P_i, $d_{c(P_i),u}$ is the distance of the BU u to the center of sector P_i, $u.pop$ is the amount of population on BU u, and, $\overline{pop}$ is the average population by sector.

The continuity relation is kept always during the construction phase, as only the unassigned neighbour of each sector are taken into account as candidates.

5.3.5 Daily generation

Daily generation involves the generation of driver's routes for each sector. The route definition is vital as it provides a meaningful way of validating the allocation, providing the work hours of each route, the number of packages delivered, and the allocation for each day. As such, the definition of routes is an essential factor. Routes are made by the drivers, taking into account their experience and knowledge; thus, even though they might not be optimal, they tend to have very good quality. Routes are defined from the depot and must return to it.

In this work, multiple routes are constructed, one for each sector, which can be considered a Multiple Travel Salesperson Problem, which is, itself, an optimization problem. There are several approaches for solving this family of problems, and many studies have been done in this area,

from advanced metaheuristics to Evolutionary Algorithms, to name a few. However, due to time complexity constraints, it was decided to go for a simple deterministic approach; to use a greedy heuristic that would iteratively build the routes assigning each BU to an adequate route. A fully-connected graph was considered for this process.

There are two types of active clients: the ones that belong to the rigid centers and the ones that belong to the borders. The first kind will inherently be part of the sector's route; the second kind can be part of a route of the same or any other sector.

5.3.5.1 Route generation

In the first phase, a route was defined using the active BUs on the rigid center. The route was constructed with the resource to the Farthest Insertion heuristic. The idea behind the Farthest Insertion heuristic is to find the farthest candidate from the route and insert it where the insertion cost is lower. The initial route is given only by two stops, starting and ending at the depot, w. From there, the route is expanded given by the formulation:

$$c_{far} = max_{c \in C} min_{t \in T} d(c,t) \tag{5.15}$$

$$min_{(i,j) \in T}(cost(i,c_{far}) + cost(c_{far},j) - cost(i,j)) \tag{5.16}$$

where C is the list of candidate points to add to the tour, T is the current tour, $d(c,t)$ is the distance from c to t, (i,j) is an arc in T and $cost(i,j)$ is a cost function from i to j.

The definition in 5.15 finds the candidate point that lays the farthest from the route by comparing the minimal distance to a point in the route. The definition in 5.16 finds the arc where the insertion of a point is minimal by adding the costs of the newly created arcs and subtracting the existing one.

$$travel\_time(a,b) = \frac{dist(a,b)}{vel_{a.t,b.t}(dist(a,b))} \tag{5.17}$$

where $dist(a,b)$ is the distance according to 5.2.4, $a.t$ is the type of the BU a, either urban or rural, $vel_{T_1,T_2}(x)$ is the average velocity depending on the BUs type, according to 5.2.3, as such the average velocity depends on the BUs urban/rural types and the distance between them. The formulation in 5.17 is used as cost function in 5.16, $travel\_time(a,b) = cost(a,b)$.

This process will create an inner route on the active BUs belonging to the rigid center. The next step will be to assign the remaining active BUs to routes; however, these BUs do not have to be part of their original sector's route.

The second phase makes use of an heuristic for assigning and inserting BUs to routes. The heuristic will be based on the minimum insertion, described in 5.16 and the current capacity of the route. At this stage, routes depart and return to the depot. As such, long arcs from the depot to the first and last BU exist. In order to avoid situations where a BU lays almost over those arcs but

away from the rest of the deliveries and where far BUs can be assigned to a route, only k routes for which the BU is closest to its sector's center are considered. This approach also decreases the time complexity as it reduces the need to compute the insertion on every route.

$$min\_insertion(T,b) = min_{(i,j)\in T}(travel\_time(i,b) + travel\_time(b,j) - travel\_time(i,j)) \tag{5.18}$$

$$time(T,b) = T_{time} + min\_insertion(T,b) + (T_d + b_d) \times t_d \tag{5.19}$$

$$f(x) = \begin{cases} x, & \text{if } x \leq t_{opt} \\ \frac{x^2}{t_{opt}}, & \text{if } t_{opt} < x \leq t_{opt} + extra_t \\ \frac{(2x - extra_t)^2}{t_{opt}}, & \text{otherwise}(x > t_{opt} + extra_t) \end{cases} \tag{5.20}$$

$$h(T,b) = f(time(T,b)) \times dist(b, c(T.s)) \tag{5.21}$$

where $min\_insertion(T,b)$ is the cost of the minimum insertion of b in tour T, $time(T,b)$ is the total route time after inserting b in tour T, T_{time} is the tour's required time, T_d is the number of deliveries in route T, b_d the number of deliveries in BU b and t_d the average time per delivery. t_{opt} is the ideal work time, $extra_t$ is the extra work time allowed, $f(x)$ is a transforming function of x which represents a time measure, $c(T.s)$ is the center of the sector of route T and $h(T,b)$ is a function that assigns a heuristic value of inserting b in tour T.

In 5.18, the minimum insertion of a BU in a route is calculated as stated before in 5.16, is return the minimum time cost of inserting a BU and the correspondent arc. The total time of a route after inserting a BU is given in 5.19 and is calculated by the total traveling times plus the total delivery time, and it is fed into 5.20, which transforms that time into a value, such that is has a more significant impact is it surpasses the optimal working time and even greater if it surpasses the extra working time, $t_{opt} + extra_t$. Finally, the heuristic value is given in 5.21 combining the transforming function by a factor, such that points closer to the center of the route's sector have a higher tendency to be part of that route.

The transforming function in 5.20 has the goal, during the construction of the routes, to balance routes between themselves while penalizing more routes that surpass the optimal working time and even more to the ones that surpass the maximum extra working time. It produces a continuous, not differentiable function with a positive slope. An example for the 5.20 with an optimal working time of 8 hours, 480 minutes, and a maximum extra working time of an extra 1 hour, so 540 minutes, can be seen in Figure 5.3.

The remaining available border BUs that have not yet been assigned will have a heuristic value calculated for each of the k routes of the closest sectors. Then, a greedy procedure will choose the pair BU and route with the smallest heuristic value and insert the BU into the route accordingly.

After all the BUs have been assigned to a route, the delivery route is created, consisting of the client delivery order inside each of the BUs. All active clients must be served within the same BU before the driver proceeds to another BU. One can solve this as an open TSP. Once again, a

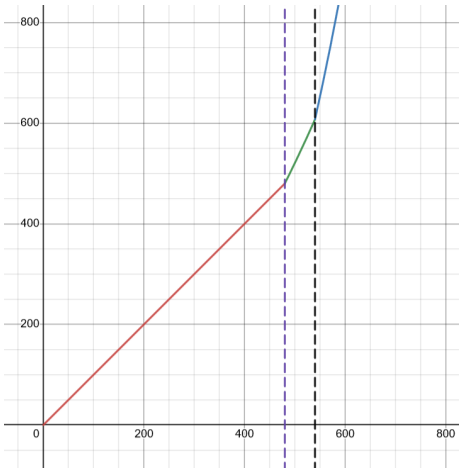


Figure 5.3: Example of transforming function.

greedy heuristic was used to get a route for the problem, as, in most cases, the number of clients is not considerable. The chosen heuristic was the Farthest Insertion method. For each delivery route inside a given BU, it is already given an incomplete route consisting of 2 points, the start and end points, as two BUs located right before and after the given BU in the route in which it is being constructed the delivery route.

5.3.5.2 Delivery route trimming

The previous definition is not enough, as all the BUs and their clients are assigned, and, in some cases, it will result in delivery routes with more than the extra working time allowed. In order to prevent such situations, a trimming of the routes is conducted, which aims to remove the most expensive clients, clients that significantly impact the route's time.

A heuristic is employed to achieve this, which formulates the reverse of the minimal insertion by calculating the decrease in time by removing the travel time to a client and the delivery time and adding the travel time of going directly to its neighbors in the route plan. However, in this heuristic, the delivery time is divided by the number of packages to deliver to a client, such that, if two clients take approximately the same travel time, the heuristic will remove the one with fewer packages to deliver, giving preference to more delivered packages.

$$min_{(i,j,k)\in T}(travel\_time(i,k) - (travel\_time(i,j) + travel\_time(j,k)) - \frac{t_d}{j_p} \tag{5.22}$$

where i, j, k represent clients in the delivery route, and j_p the number of packages to deliver to client j.

In 5.22, client j is removed from the delivery route, and the process is repeated until the total delivery route time is less than the maximum work time allowed. (i,j,k) are three clients that form two consecutive arcs in T.

5.3.6 Genetic operators

The genetic operators are responsible for the crossover and mutation of an instance. Crossover is the operation of generating a new instance from two given instance. The two instance given are known as parents, and the solution obtained as child or offspring that shares features from the two parents. The mutation is an operation that introduces a small degree of randomness in the offspring to generate an unseen instance. The crossover operator promotes exploitation and the mutation operator promotes exploration.

The crossover operator is reasonably straightforward. There are given two allocations X_1 and X_2, consisting of a set of sectors. The intersection of its sectors is computed:

$$\bigcup_{P_1 \in X_1, P_2 \in X_2} P_1 \cap P_2 \tag{5.23}$$

Each intersection is assigned a heuristic value that determines how much of a feature is shared between the two parents. The heuristic has a normalization factor, such that the size of bigger sectors does not neglect the smaller ones present in both solutions. The formulation is the following:

$$h(P_1, P_2) = \frac{|P_1 \cap P_2|}{max(|P_1|, |P_2|)} \tag{5.24}$$

Then, the L greatest intersections are chosen as part of the future sectors; each intersection is an incomplete sector. As the next step, the construction procedure described in 5.3.4 is applied. Note that the crossover operator does not guarantee the contiguity relation, as the intersection can be non-continuous. However, the construction procedure encourages split fractions to be connected, through the Compactness portion in the heuristic for each candidate.

This procedure already produces an offspring with characteristics shared by both parents and some degree of randomization. However, the closer the parents are to each other, the minor the variation there will be; if they are equal, there is no difference between the parents and the child.

As a result, a mutation operator was developed to introduce some degree of randomization. Giving a complete instance, choose a sector at random and remove the BUs that are part of its border level-zero. The basic units of other sectors that are neighbors to the removed BUs are also removed from their respective sectors. This results in an incomplete instance, so the construction procedure described in 5.3.4 is applied again. This operator actuates at a small degree on the border of the sectors, possibly generating a slightly different allocation.

In order to activate the mutation, a random number is generated and compared to a mutation threshold, which determines how often the mutation happens. If a mutation is executed, the random number is generated again to possibly trigger a new mutation until a maximum number of mutations is reached.

5.3.7 PICEA-g-mr

As this work aims to aid the decision-maker with allocating BUs to sectors, a set of non-dominated instances must be provided to proceed with the decision. Due to the multitude of objectives, a multi-optimization method was considered, specifically an MOEA approach. As the number of objectives fits in the many-objective category, a total of four, a PICEA-g framework was used, together with a mating restriction to guide the search process.

5.3.7.1 PICEA-g

PICEA-g has been shown to perform well and even outperform other state-of-the-art multi-optimization algorithms in MaOPs. PICEA-g is an MOEA, based on an evolutionary structure and elitist framework. Two populations co-exist, one of instances/individuals and another of goals, which are co-evolved together. Goals aim to represent the decision-maker's preferences and are used to guide the solution towards the Pareto-Front. There is an offline archive, whose function is to keep track of the non-dominated front, containing the approximation of the Pareto-front.

PICEA-g starts by generating a set of random feasible instances and goals. From there, for several generations, instance parents are subject to genetic operators to produce offsprings. At the same time, goals are randomly generated according to the non-dominated front bounds. The offspring and parents of each population are pooled together. The non-dominated instances are selected accordingly to the non-domination relationship. In contrast, the goals do not go through any selection. Both populations are pooled according to a fitness assignment that considers the non-domination relationship of instances and goals. The best individuals/goals are kept for the next generation.

The PICEA-g allows an approximation of the Pareto-front, which will contain the instances that will be presented to the decision-maker so that he can choose the one that best fits his interests.

5.3.7.2 Mating restriction

The mating restriction is intended to mate/recombine similar instances. Recombining instances that are very distant from each other can lead to instances that lose the progress made towards the Pareto-front and do not take advantage of promising features of both parents. In contrast, recombining similar instances can lead to children that share promising features with their parents. Thus, mating restriction intends to make instances more probable to recombine with similar ones.

The definition of similarity is problem-dependent, so there is a need to develop a function that maps the distance between two instances. There are two possible alternatives: to consider the distance in the objective-space, such that instances with similar criteria values are closer, or to consider distance in the instance-space, such that instances with similar allocations are closer. For this problem, it was opted the last one, as instances that are similar to each other on the objective space could not share similar features; thus, the recombination would not be very promising and could appear as a random offspring.

In order to calculate the distance between different individuals, a similarity measurement was developed, such that, the closer two individuals are, the greater the similarity measure and the greater the probability of they being recombined with each other.

An instance consists of sectors, which in their turn, consist of BUs. This can be seen as sectors being a set of clusters or partitions of the available set. As such, a similarity measurement used in cluster analysis, known as Mutual Information, was employed, which gives the dependence of two distributions or how much two partitions have in common [56].

$$similarity(X_1, X_2) = \sum_{P_1 \in X_1} \sum_{P_2 \in X_2} P_{X_1 X_2}(P_1, p_2) \times log \frac{P_{X_1 X_2}(p_1, p_2)}{P_{X_1}(P_1) \times P_{X_2}(P_2)} \tag{5.25}$$

$$P_A(a) = \frac{|a|}{N} \tag{5.26}$$

$$P_{AB}(a,b) = \frac{|a \cap b|}{N} \tag{5.27}$$

, where $X1$ and $X2$ are allocations consisted of sectors and N is the number of BUs.

This metric is always non-negative and computable for every pair of solutions and it provides a comparison framework, such that, if $similarity(X_1, X_2) > similarity(X_1, X_3)$, then X_1 is closer to X_2 than X_1 is to X_3. Instead of considering the most similar solution to mate with, as it would not promote exploration and diversity in the population, some selection mechanisms are applied from evolutionary algorithms. In this case, a Roulette Wheel selection method is employed.

Roulette Wheel is a selection method, in which the probability of an individual being selected is proportional to its fitness, as such, fitter individuals are more likely to be chosen.

Linear scaling is applied to control the selection pressure. Selection pressure can be informally defined as the amount of preference given to fitter individuals than to the average one, sometimes expressed as $S_p = \frac{f(max)}{\overline{f}}$. Linear scaling is followed by a normalization of the values to get a probability distribution. The linear scaling is as follows:

$$f(i) = a \times g(i) - b \tag{5.28}$$

$$a = \frac{S_p \overline{f} - \overline{f}}{g(max) - \overline{f}} \tag{5.29}$$

$$b = a \times \overline{f} - \overline{f} \tag{5.30}$$

, where f is the scaled fitness function, g the raw fitness function, $\overline{f}$ is the average of the target function, and S_p is the selection pressure value. The scaling is done such that the average of the values before and after scaling is preserved but the selection pressure is equal to S_p.

5.3.8 Choosing a solution

After the optimal front has been generated, it is up to the decision-maker to select the individual that best fits his interests. During the search process, a set of non-dominated instances have been found. However, not all of them will fit the decision maker's best interests. As such, a toll must be developed to help reduce the number of solutions and select one.

During the search process, a set of criteria were considered on which the solutions were tested on the Pareto dominance. This set of criteria guided the search towards the Pareto-front. When comparing several solutions that are similar to each other, it can be challenging for the decision-maker to be able to say with confidence which of the solutions fits his preferences the best. Sometimes, metrics are abstract measurements, and it can be hard to judge slight differences between solutions. In order to help the decision, at this stage, it is possible to add other metrics that can influence the selection process. For example, one can consider other Compactness metrics, the expected financial cost of an assignment, or the greenhouse emission of a given allocation. Using other metrics can provide a meaningful comparison between solutions that, otherwise, would be very hard to decide on.

In order to provide a meaningful representation of the M criteria that describe an allocation, a multidimensional visualization was considered. Scatter plots work best for two criteria, where a 2D mapping can be conducted. Three criteria can also be represented in scatter plots, either by 3D representation or by using an extra variable, such as size or color. As the number of criteria increases, the representation can be complicated and complex to comprehend. One powerful technique to represent multidimensional data is using multiple axes, such as in Parallel Coordinates plots.

In Parallel Coordinates, one vertical axis exists for each variable, and the axes are evenly spaced. Each data item, in this case, an allocation's criteria values, are represented by a polygonal line that intersects the axis in the attribute's value. This representation is effective but can become very crowded for many data items. It is possible to interact with Parallel Coordinates through dynamic queries using range sliders. One might try to analyze data using this representation; however, it has some disadvantages, such that it can be hard to recognize patterns on axes that are not side by side, as the order of the axis can be relevant. Complex relations are not easily identified; however, this tool's purpose is not to understand the relationship between multiple criteria but to select the best-fit solution. The authors in [39] make use of this technique, where they define aspiration criteria that allow for filtering data and only consider those instances that have metrics below that criteria, as Figure 5.4 shows.

Another possible alternative, which is based on the same principle, are Stardinates, which are Parallel Coordinates in a circular shape. However, they also possess some disadvantages, as the axis arrangement does not work well for a small number of dimensions, and with a high number of dimensions, the perception of the data decreases. Moreover, data lines closer to the center tend to cover each other and are harder to perceive.

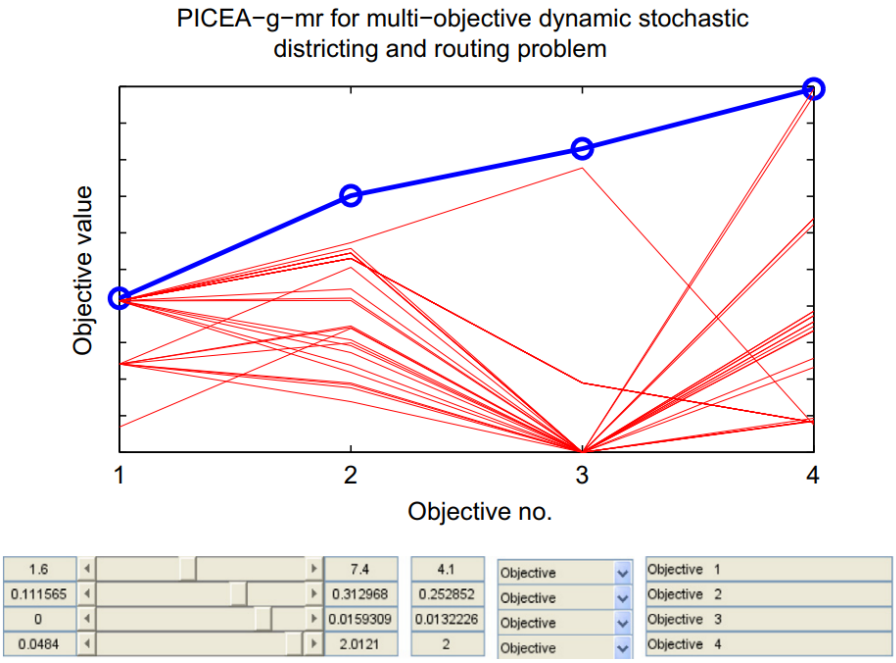


Figure 5.4: Selecting a preferred design: phase 3 [39].

Chapter 6

Implementation

In this section, the implementation of the approach is presented and explained, where the ideas are moved into the code and practice, to achieve the desired result. The program was implemented in Python 3.10.4 and some procedures with the resource to QGIS.

6.1 Neighbourhood

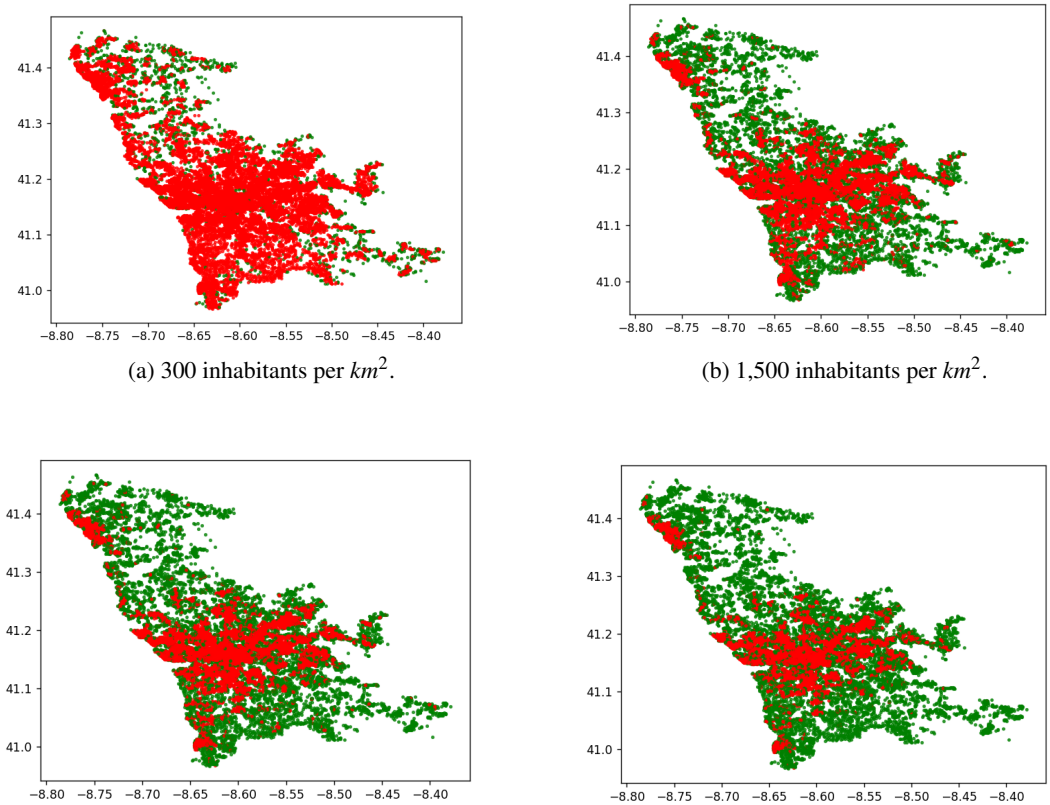
The Delaunay Triangulation was first approached with the resource of the QGIS geometric tools. The output was a geometric polygon consisting of several triangles. A line extraction tool was applied to extract the edges that establish the neighborhood relationship between BUs. It gave a series of lines that made up the DT. However, the result was not the expected, and a strange behavior arose when selecting or filtering lines: others would also be selected. For example, when selecting a single edge, another edge would often be selected, and only a few times, only the correct one was selected. There was no solution found nor any explanation for this problem.

The DT was achieved with the resource of a Python package; however, due to a different coordinate system, it was impossible to intersect and filter edges that felt out of the AMP. As such, edges that were too long were excluded from the neighborhood. This avoids situations where points far away are connected and allows edges between small connections that could exist in the real world. A value of 5000 meters was used; anything below that length would be excluded.

6.2 Urban and Rural areas

In order to identify the urban and rural areas, a Python script was developed accordingly to the definition in 5.2.5.

The BUs will be considered urban or rural according to the population density. In the definition given before, a high-density urban cluster has a population of 1500 inhabitants per km^2, whereas



(a) 300 inhabitants per km^2.

(b) 1,500 inhabitants per km^2.

(c) 2,000 inhabitants per km^2.

(d) 3,000 inhabitants per km^2.

Figure 6.1: Population density (green as rural, red as urban).

an urban cluster has a population density of 300 inhabitants per km^2. The BUs data available has the CP7s population per m^2; the conversion yields 0.0003 population per m^2 for the urban clusters.

In Figure 6.1 we can observe values for different population densities. Each point is a centroid of the BU; the red color means an urban BU and the green color a rural one. One can observe that in 6.1a the majority of the basic units is an urban area. As the next step, the definition of high-density clusters was used in 6.1b in order to evaluate the results, which gave more satisfactory results. Other values were used in order to understand the data better. To the author's interpretation, the values that better represented the actual demographic types were 1,500 and 2,000 inhabitants per km^2, in Figures 6.1b and 6.1c. However, Figure 6.1d, showed more clear and well defined urban clusters. The value chosen was 2000 inhabitants.

The next step of the process was to clear the outliers, which are BUs with a greater ratio of neighbors of other type. The algorithm is described in Figure 6.2. It iterates over all the points and checks whether a rural point has a ratio of urban neighbors above or equal to the urban threshold; if so, it is upgraded to urban. On the other hand, if an urban point has a ratio of urban neighbors below or equal to the rural threshold, it is downgraded to rural. The values used in this work for the threshold are 60% for the urban threshold and 30% for the rural threshold. Which means, that if a rural point has at least 60% of its neighbors as urban, it is considered urban, whereas if an

urban BU has no more than 30% of urban neighbors, it is considered rural. The algorithm iterates over all points as long as there is any modification and stops when it achieves convergence. The results can be seen in 6.3.

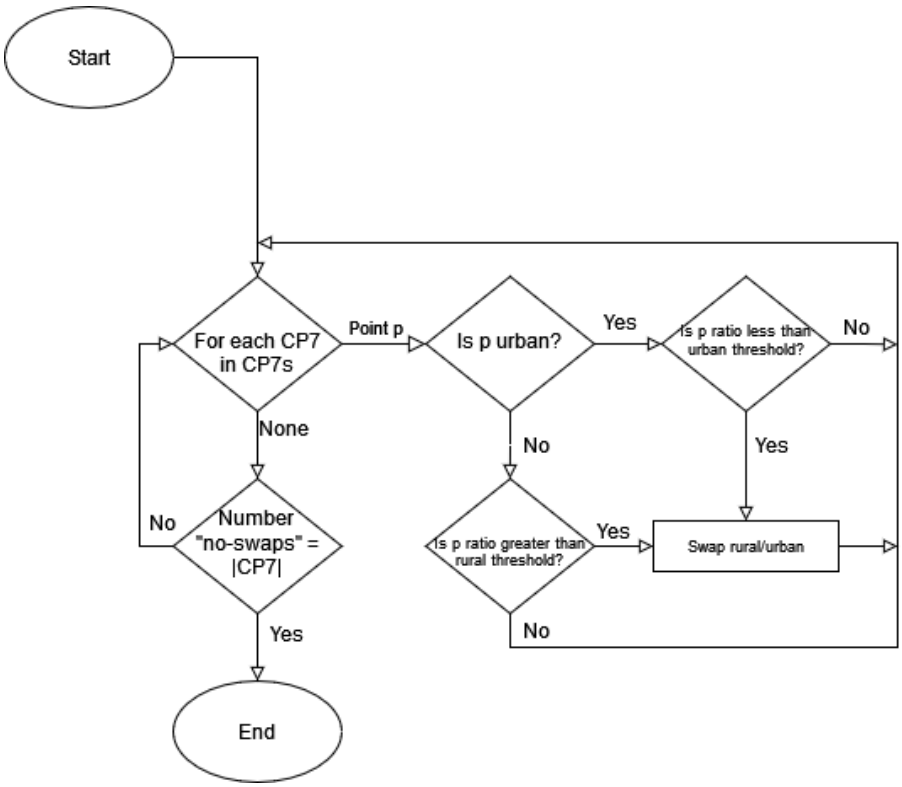


Figure 6.2: Removal of urban and rural outliers.

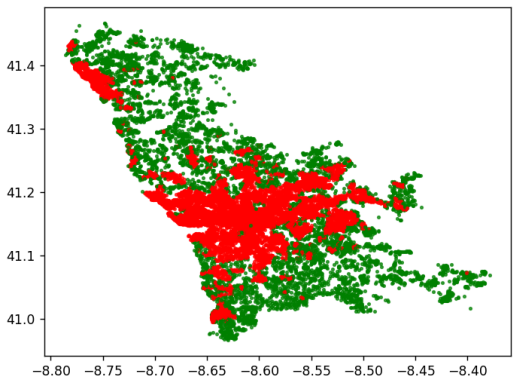


Figure 6.3: Urban and rural outliers removed, with 60% and 30 % for the urban and rural thresholds, respectively.

As the next step, we enforce a minimum population per cluster, as the definition in 5.2.5 states. The definition for high-density clusters has a minimum population of 50,000 and for urban clusters, a minimum of 5,000 inhabitants in the contiguous region. The algorithm is described in

Figure 6.4. It has a Flood Fill like approach to identify all urban connecting BUs that are part of the same cluster of BUs and count the total population of each cluster. If a cluster has less than the minimum population, it is all set to rural. This aims to remove small high-density clusters that can occur in the center of villages, for instance.

Several values for the minimum population have been tested in 6.5. A minimum population of 2,000 inhabitants is enough to clear many small clusters, but values of 15,000 and 30,000 are too restrictive and remove clusters of considerable size. The value that worked the best was the one used in the previous definition of urban clusters, a minimum of 5,000, as seen in 6.5b.

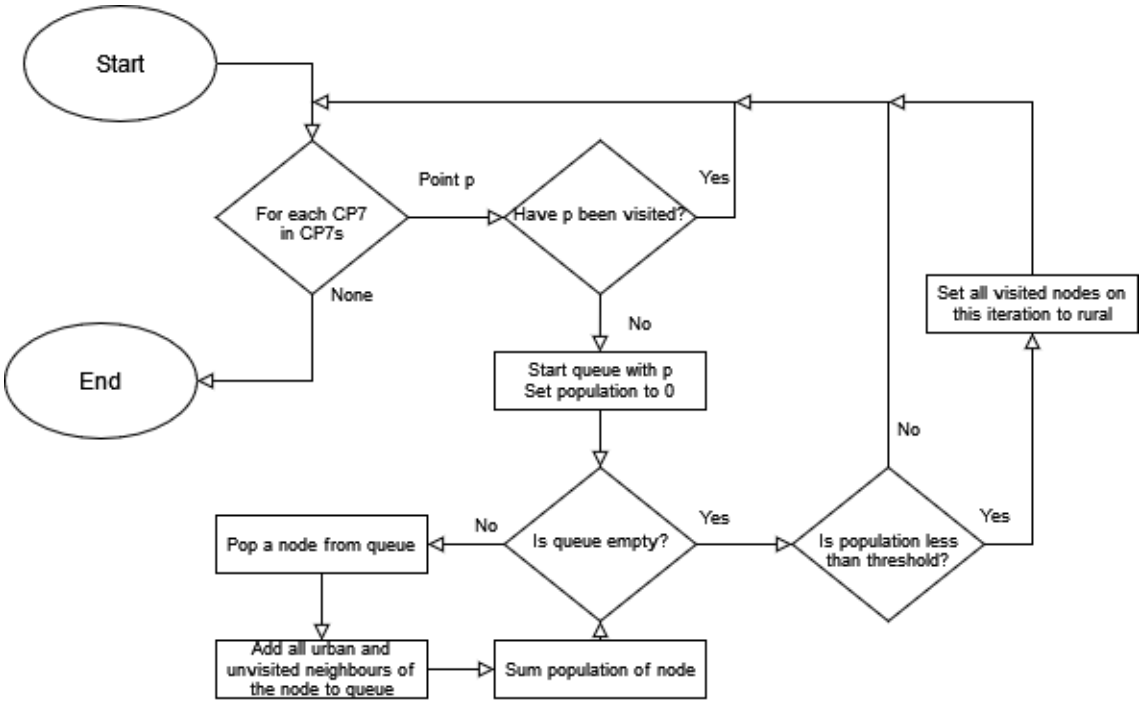


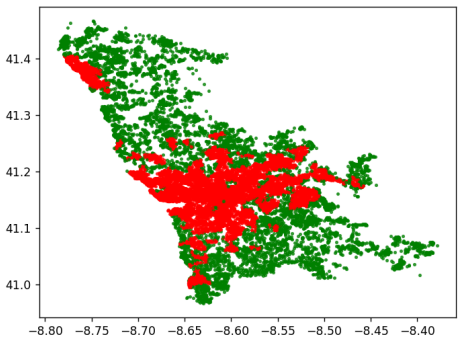
Figure 6.4: Removal of low population clusters.

6.3 Allocation construction

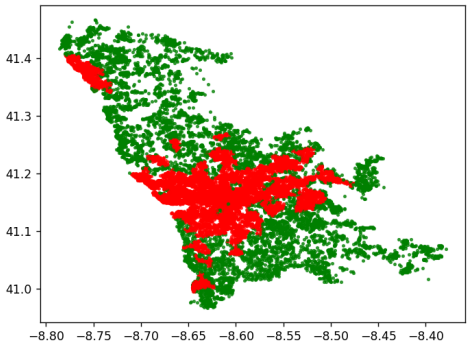
The construction is based on Randomized Construction Heuristics. It introduces randomness in a greedy construction procedure to generate a variety of solutions with good quality and a good starting point for the algorithm.

The algorithm is described in Figure 6.6. It starts by identifying all the possible candidate BUs for that solution that are not yet assigned to a sector. It associates a heuristic value to each candidate BU and sector pair. Note that a BU can be a candidate for more than one sector. Then, while there are candidates to be assigned, the main loop will iterate.

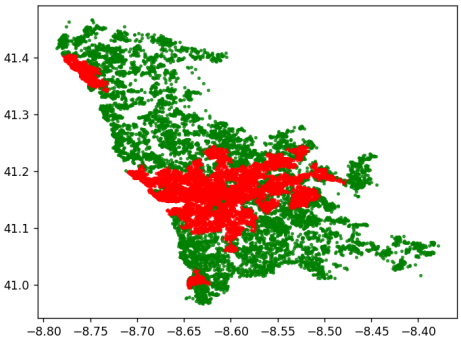
A candidate consists of an unassigned BU, a sector to which it can be assigned, and the heuristic value of inserting that BU in that sector. Candidates are obtained by iterating over the sector and listing the unassigned BUs that are adjacent to the level 0 border of that sector. There are two



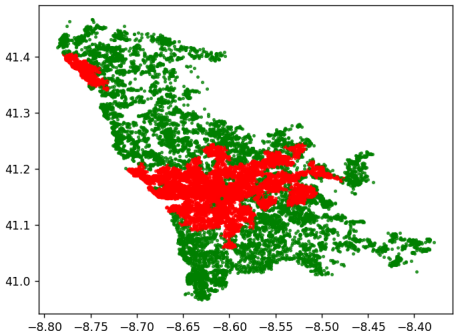
(a) Minimum of 2,000 inhabitants.



(b) Minimum of 5,000 inhabitants.



(c) Minimum of 15,000 inhabitants.



(d) Minimum of 30,000 inhabitants.

Figure 6.5: Minimum cluster population.

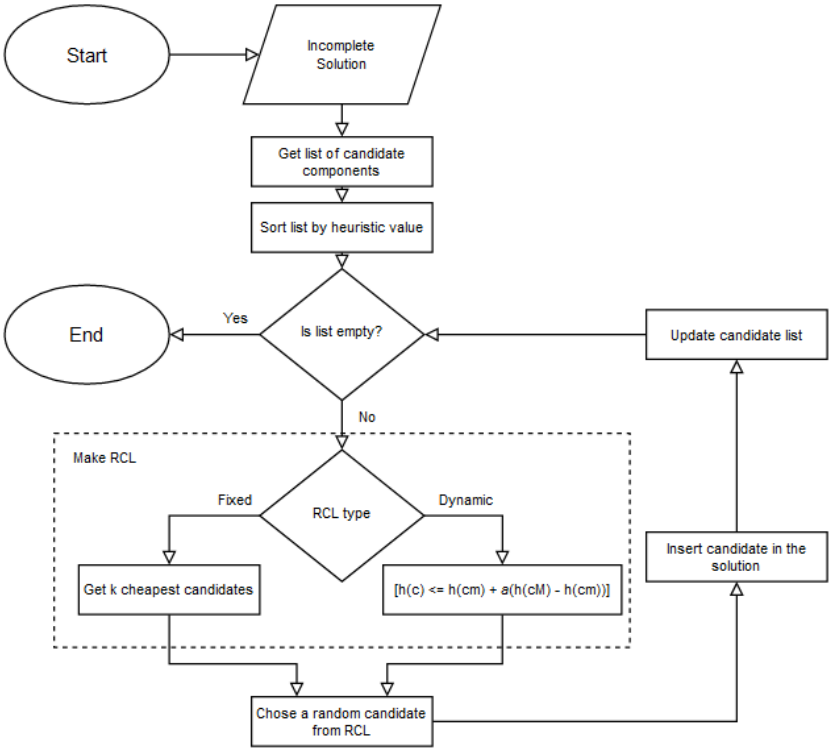


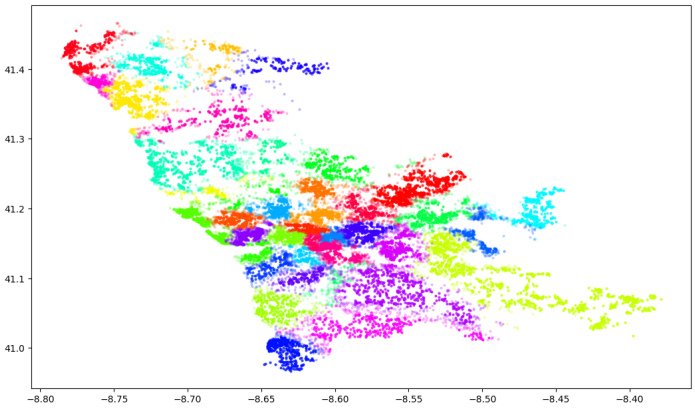
Figure 6.6: Random Construction Heuristics.

ways of building the RCL: a fixed-RCL or a dynamic-RCL. From there, a random candidate is chosen to integrate the solution.

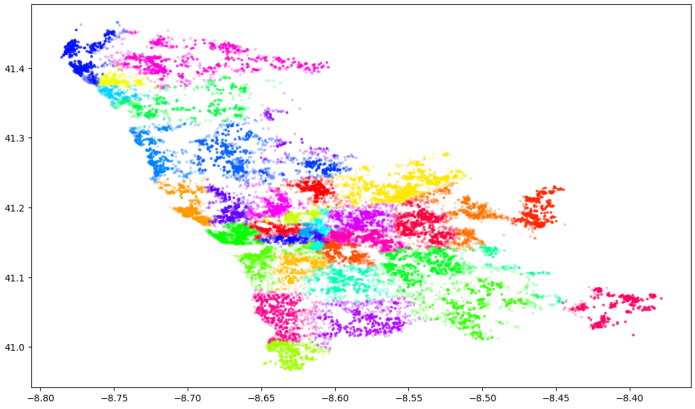
As the heuristic considers metrics about the sector into which the BU will be inserted, every candidate that refers to an addition to the same sector needs to be recomputed. Moreover, new candidates can be added to the candidate list, as a result of adding the BU to the sector. This can be implemented by removing all sector candidates from the CL and recomputing the CL for that sector.

Delta evaluation is employed here to speed up the heuristics calculation. Delta evaluation is a technique that consists of only recomputing parts of the heuristic that depend on the changes made to the solution. In this case, the computation of the compactness is only made once for each newly added BU, and its value can be used to recompute the new compactness if adding a new BU when calculating the heuristics value. Also, the list of candidate solutions is not all recomputed again, only the sector's candidates that have changed.

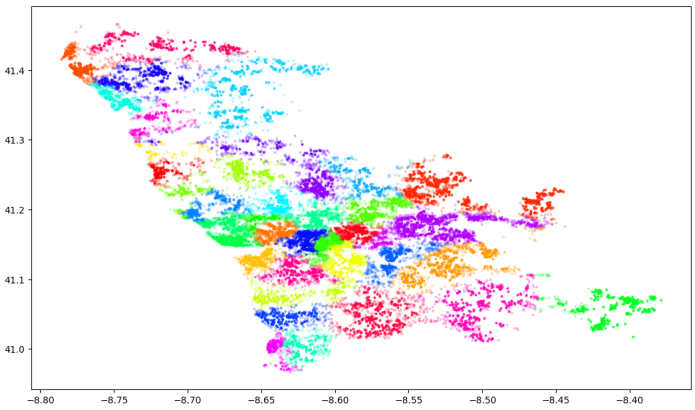
In Figure 6.7, possible solutions generated by the Construction procedure can be seen.



(a) Possible Construction result, example 1.



(b) Possible Construction result, example 2.



(c) Possible Construction result, example 3.

Figure 6.7: Possible solutions from the Random Greedy Construction (translucent points represent sectors' borders).

6.4 Daily generation

Daily generation is the process of generating an allocation and routes for each day to validate the allocation. As explained in 5.3.5, the process has three main procedures: (i) allocating BUs to routes, (ii) generating the delivery schedule, and (iii) trimming the route to make them feasible.

Figure 6.8 shows the main algorithm for assigning and inserting BUs to routes for each day. It starts by identifying the active BUs for the day. Then, a route is built on the active BUs that belong to the rigid center of the sector. It is done using the Farthest Insertion heuristics.

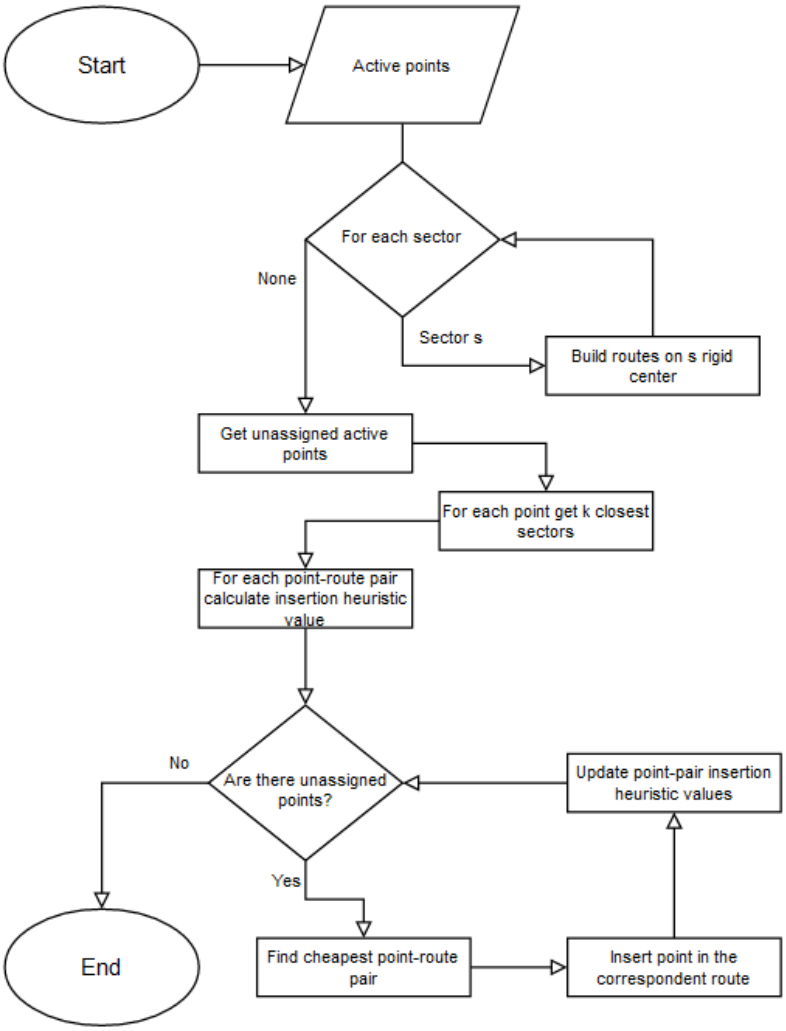


Figure 6.8: Generation of basic unit routes.

The initial route is given only by the start and end points, which are both the depot. Then, the minimal distance to a point in the route is computed for each point to be added. After this procedure, the main loop is entered. It finds the BU that lies the farthest from the route and calculates the minimal insertion on the current route. Then, it inserts the BU in that position. The remaining BUs to be inserted have their distances updated concerning the newly inserted BU.

The procedure repeats until there are no more BUs to be inserted. It has a $\mathcal{O}(n^2)$ time and space complexity.

After this, the remaining BUs that belong to the borders need to be assigned to routes and inserted into them. Firstly it is identified the active BUs that belong to the border; thus, they have not been assigned yet. Then, for each BU, we find the k closest sector. This is done by calculating the distance of every pair of BUs to the sectors' centroids. For this work, a $k = 3$ was considered.

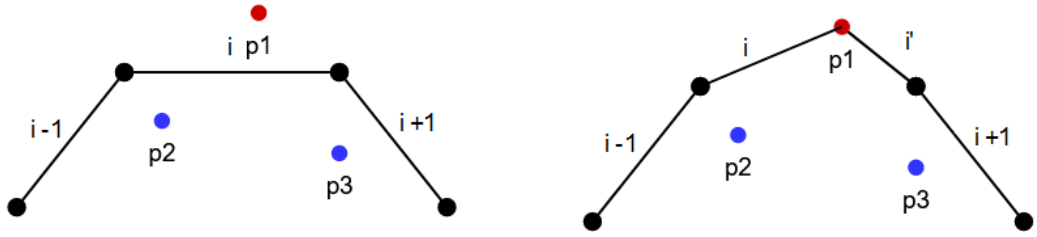
Secondly, for each pair of BUs and its k closest sectors, a heuristic value was calculated accordingly to the formulation in 5.21. It involves iterating over all arcs in the route and finding the insertion that minimizes the heuristics value. The index of the best insertion and the corresponding value are saved for each pair.

At this stage, the main loop is entered. The cheapest BU and sector pair is found, and the BU is inserted in the corresponding arc. Then, due to the heuristics formulation, the heuristic values for the pairs that have the sector on which a BU was inserted need to be updated to accommodate the new changes, such as the increase in travel and delivery time and the transforming function.

Initially, the update of the heuristic values for a BU route pair was made by recomputing the minimal intersection on the sector's route by iterating over every arc in the route. However, this is expensive, and there is no need to iterate over all the arcs in the tour. Consider the situation in Figure 6.9, where the red point, point $p1$ is the point to be inserted in arc i, point $p2$ has the minimum insertion in arc i as well and point $p3$ has minimum insertion in arc $i+1$. Figure 6.9a represents the route before the insertion, and 6.9b represents the situation after the insertion. There are two possible update situations: the minimum insertion of a point was where the insertion was made, and the minimum insertion of a point was not where the insertion was made.

Consider point $p2$, such that the minimum insertion of a point was where $p1$ was inserted, arc i. In this case, consider that the best insertion of $p2$ would be in arc $i-1$. However, in a different disposition of the points, the minimum insertion could be in any arc of the route. To better describe this, the new minimum insertion would be the minimum of the second-best insertion for that point before the insertion and the two new added arcs, i and i'. This would require keeping an updated and ordered list for all possible insertions, which would result in recomputing the values all over again. So, for this case, it is needed to iterate over all arcs and recompute the minimum insertions.

Now, consider point $p3$, such that minimum insertion was not where $p1$ was inserted. In order to update the minimum insertion of this point, it is necessary to recompute the heuristics value for that insertion to accommodate the changes of adding point $p1$. Given that the increase in time of the heuristics function is the same for every arc and it goes through a transforming function that always has a positive slope, it is guaranteed that the arc $i+1$ will be the best insertion for point $p3$ in all route, excluding the newly formed arcs i and i'. As such, the new minimum insertion will result from comparing the arc $i+1$ with the newly formed arcs i and i' as the previous computations for point $p3$ did not consider the new arcs. Hence, for this case, there is only the need to recompute the heuristic values for three arcs. This reduced the computational effort of the algorithm by a major factor.



(a) Before insertion of $p1$.

(b) After insertion of $p1$.

Figure 6.9: Route insertion update (red point - point to be inserted, blue points - candidate points).

At this stage, routes consist of an order list of BUs that have been generated. The next step consists of generating the delivery route, which means defining the order in which clients are visited. The algorithm approach for each route is presented in Figure 6.10.

The route has the depot at its start and end, which does not have any client to deliver in; as such, they are ignored. For every basic unit, the active clients are found, and a route is generated for those clients. An initial route is already given, from the centroid of the previous basic unit to the centroid of the following basic unit, and the active clients must be inserted in between that route. For this, the Farthest Insertion heuristic was employed. Then, the order of the active clients is extracted from the generated route for each BU. Ultimately, it yields a delivery route consisting of an ordered list of active clients representing the delivery order.

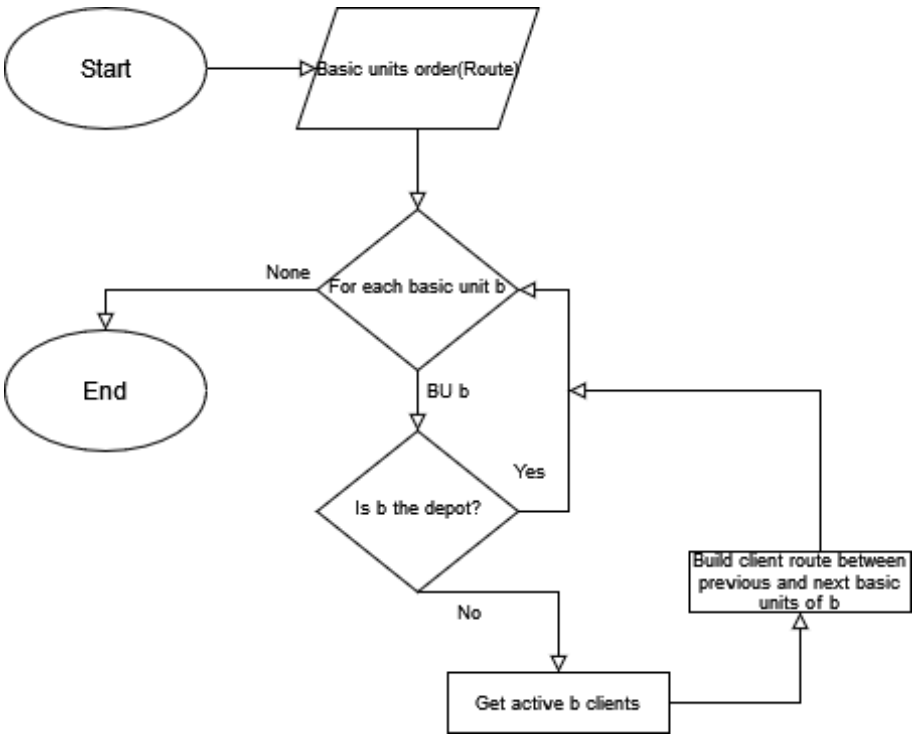


Figure 6.10: Generation of delivery route.

When building the BU route, an heuristics was employed that tried to minimize the time imbalance between routes, penalizing routes with an expected required working time greater than the maximum and extra time allowed. However, when generating the BU routes, there is no guarantee that the extra time limit is not surpassed as all BUs are allocated. Moreover, the delivery route has an equal or greater expected time than the corresponding BU route. To avoid this situation, a trimming of the delivery route is employed and is described in Figure 6.11.

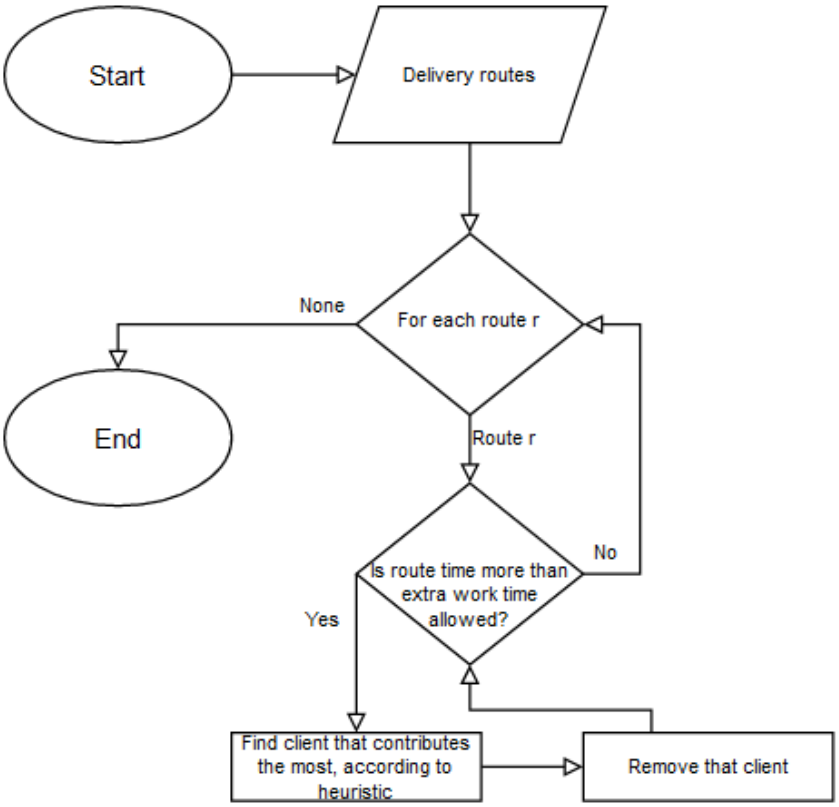


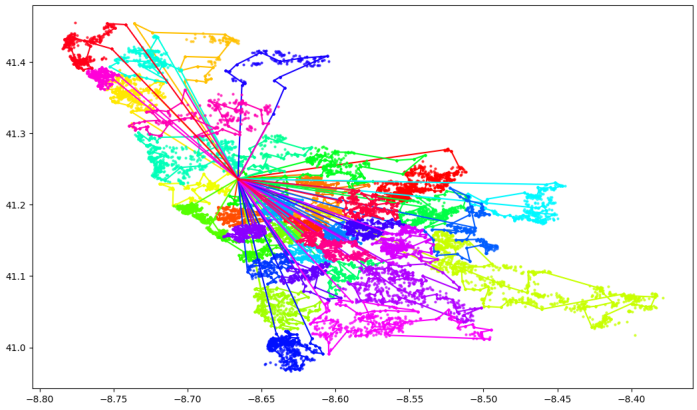
Figure 6.11: Trimming of route.

The procedure for trimming a route is simple: find the most expensive and least valuable client, according to a heuristic, and remove that client while the route time exceeds the maximum extra working time.

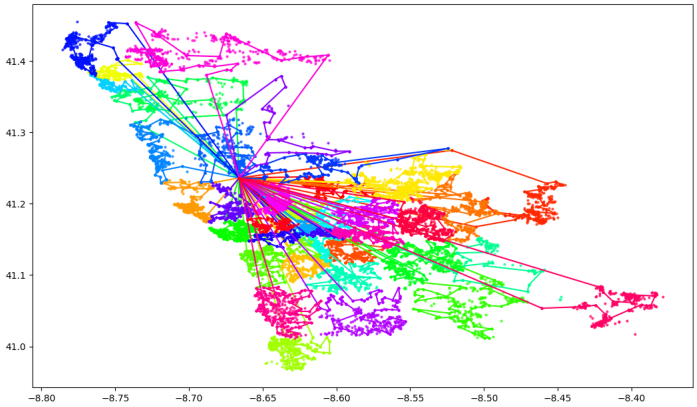
For this work, a working time of 8 hours and an extra working time of 1 hour were considered. In Figure 6.12, routes for a sample day of the global solutions in 6.7 are presented.

6.5 Genetic operators

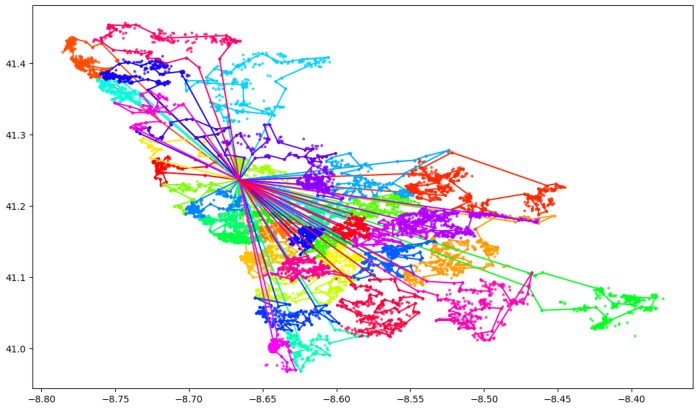
The genetic operator is responsible for guiding the search process. It consists of the crossover of two solutions generating a new solution, known as offspring, that shares feature of both parents.



(a) Possible construction 1.



(b) Possible construction 2.



(c) Possible construction 3.

Figure 6.12: Possible daily routes for the sectors defined in Figure 6.7.

On the other hand, the mutation operator is responsible for introducing randomness and variation in the offsprings and population.

The crossover operator is described in Figure 6.13. Given two individuals I_1 and I_2, all the intersections are computed, and each intersection is designated an intersection ratio. Then, the L best intersections are chosen to be part of an incomplete solution. Finally, the construction heuristics that takes as input an incomplete allocation and assigns the remaining BUs.

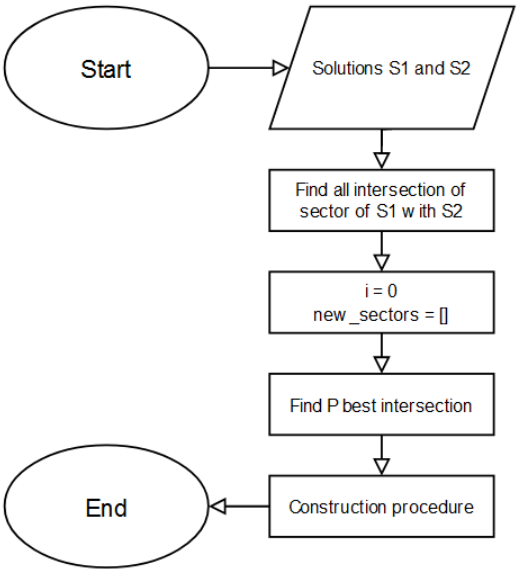


Figure 6.13: Crossover operator.

Even though the crossover operator introduces some randomness, depending on the parent solutions, there can be no randomness. So a mutation operator was introduced to achieve that result. The procedure is described in Figure 6.14.

Given a solution S, a uniform random variable can trigger a mutation depending on the mutation rate. If so, a random sector is chosen, and its zero-level border is deleted, as well, as the neighbor of the deleted border belonging to other sectors. This will create a set of contiguous BUs that are not assigned and will be reassigned by the construction procedure. The random number is generated again, which can then trigger a new mutation on the same solution. If it does not trigger a mutation, then the procedure stops.

Here, an exemplification of the process is explained in detail. Consider an 8x8 grid, where each grid cell represents a basic unit; each color represents a sector/partition, five sectors per instance, and a Von Neumann neighborhood (4-neighborhood). Example solutions S1 and S2 are presented in Figure 6.15a and 6.15b, respectively. The intersection of the two solutions is presented in 6.15c.

After getting the intersections, with the resource to the heuristics in 5.24, the five best sectors were chosen as part of the incomplete solution. In Figure 6.16a, the size of each sector and intersection is calculated, followed by the calculation of the heuristics values, in 6.16b, where

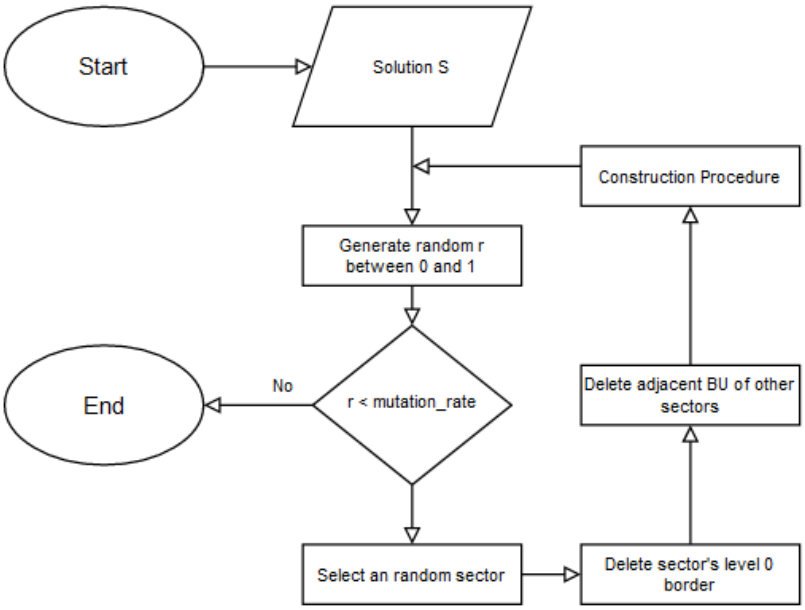
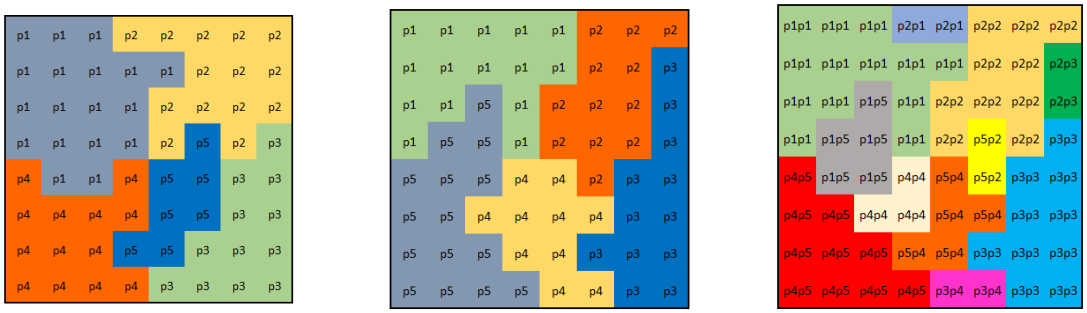


Figure 6.14: Mutation operator.

the best five sectors are highlighted, forming an incomplete solution. In the end, the construction procedure is executed over the incomplete solution, as shown in Figure 6.17.

In order to introduce some randomness, a sector is chosen at random; in this case, partition 1 was chosen, and its zero-level border was removed. Then, the neighboring BUs of the removed border that belong to other sectors are also removed, as it is exemplified in Figure 6.18.



(a) Possible solution 1. (b) Possible solution 2. (c) Intersection of S1 and S2.

Figure 6.15: Crossover: intersection of two solutions.

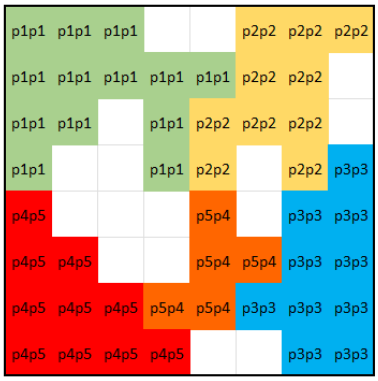
S1 \ S2		p1	p2	p3	p4	p5
		15	12	12	10	15
p1	18	13	0	0	0	5
p2	14	2	10	2	0	0
p3	12	0	0	10	2	0
p4	13	0	0	0	3	10
p5	7	0	2	0	5	0

(a) Size of intersections.

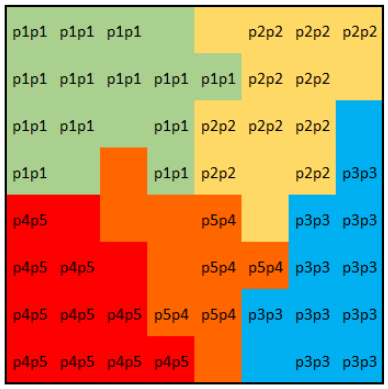
S1 \ S2		p1	p2	p3	p4	p5
		16	13	11	9	15
p1	19	0.68	0	0	0	0.26
p2	13	0.13	0.77	0.15	0	0
p3	14	0	0	0.71	0.14	0
p4	13	0	0	0	0.23	0.67
p5	5	0	0.15	0	0.56	0

(b) Heuristic value of intersection.

Figure 6.16: Crossover: sectors selection.

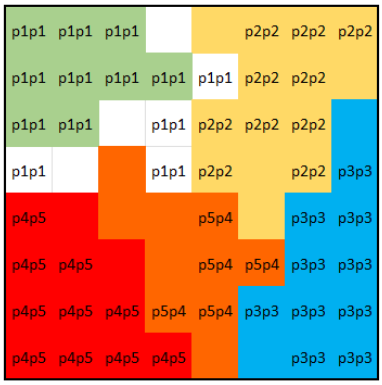


(a) Chosen sectors.

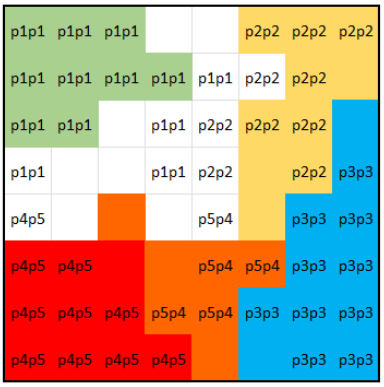


(b) Possible solution after crossover.

Figure 6.17: Crossover: building the complete offspring.



(a) Removal of level-zero border.



(b) Removal of neighbouring elements.

Figure 6.18: Mutation operator.

6.6 PICEA-g-mr

The underlying workings of the PICEA-g-mr are described in Figure 6.19. In the first stage, a population of solutions is randomly initialized, and an offline archive is updated with the generated population of instances. In the next step, the goal bounds are updated according to the member of the offline archive, and the goals can be randomly generated according to the updated goal bounds. After the initiation phase, the main loop can be executed.

The first step is to generate the population of offsprings from the current population of solutions. In the PICEA-g-mr, a mating restriction is conducted to apply the crossover operator to instances closer to each other. Children are mutated in order to keep diversity in the population. The parent population and its offsprings are combined into a joined population of solutions. Children's goals are generated according to the current goal bounds and joined with the current population.

Next, a fitness assignment is conducted on the joint instances and joint goal populations. This fitness is used to select the next generation populations. The selection method works by counting the number of non-dominated instances. If that number is less than the population size, N, the non-dominated solutions are assigned the maximum fitness, and then the N best solutions are selected for the next population. If the number of non-dominated individuals is not less than N, then the best N fitness non-dominated solutions are selected to be part of the next generation. Regarding the goals, the best N_g goals, according to their fitness, are selected to be part of the next generation.

The new populations update the archive with the new generation of instances. The goals bounds are updated according to the criteria values of the instances in the archive. Then, the loop repeats to generate a new population closer to the Pareto-front until reaching the maximum iterations.

Non-dominated individuals

In the PICEA-g it is important to find the non-dominated solutions, solutions that are not Pareto-dominated by any other solution. Taking the example in Figure 6.20a, and recalling the definition on 2.3.1, in a minimization problem, we can identify points $p1$, $p2$ and $p3$ as non-dominated points, because no solutions Pareto-dominates those solutions. Point $p4$ is dominated by point $p2$ and point $p5$ is dominated by every other point. A possible implementation for finding the non-dominated front would to check every pair of individuals and comparing each of the objective values to check if one individual Pareto-dominates the other or not. This would have a time complexity of $\mathcal{O}(Mn^2)$, where M is the number of criteria.

Considering $p2$; by checking all points that lay on the blue rectangle of Figure 6.20b, one can identify all the solutions that are Pareto-dominated by point $p2$. This can be achieved by performing a range query on the points.

There are some data structures that can perform range queries efficiently. such as KD-Trees that have a search time complexity of $\mathcal{O}(n^{1-\frac{1}{M}}+k)$ and Range Trees that with a technique called fractional cascading can perform search in $\mathcal{O}(log^{M-1}n+k)$, where k is the number of returned

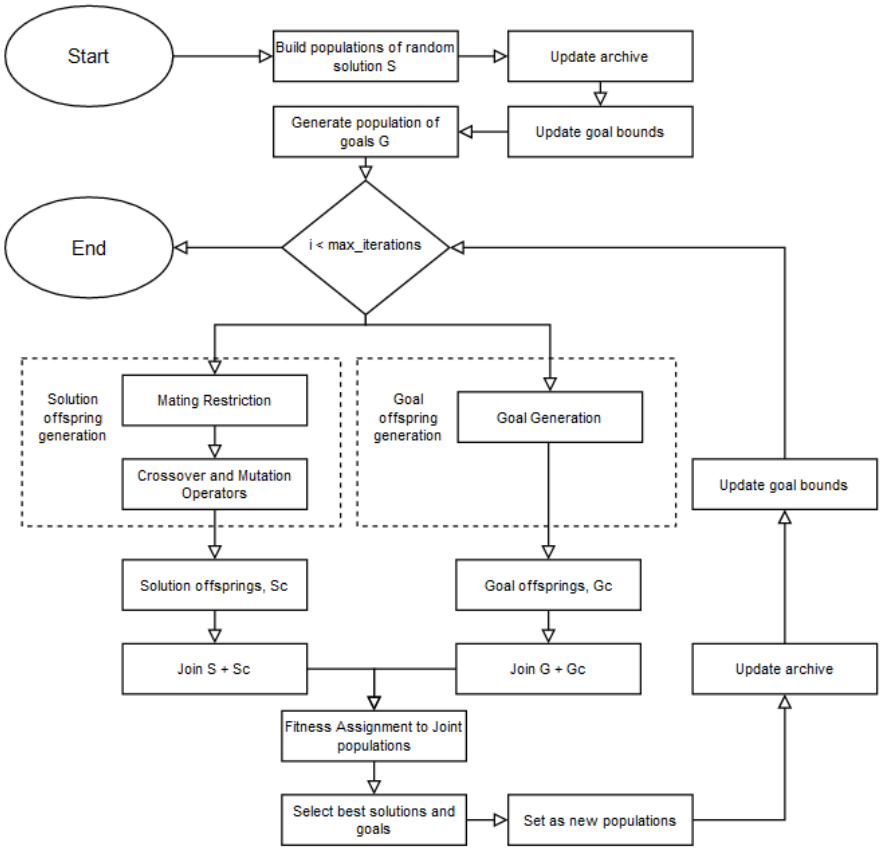
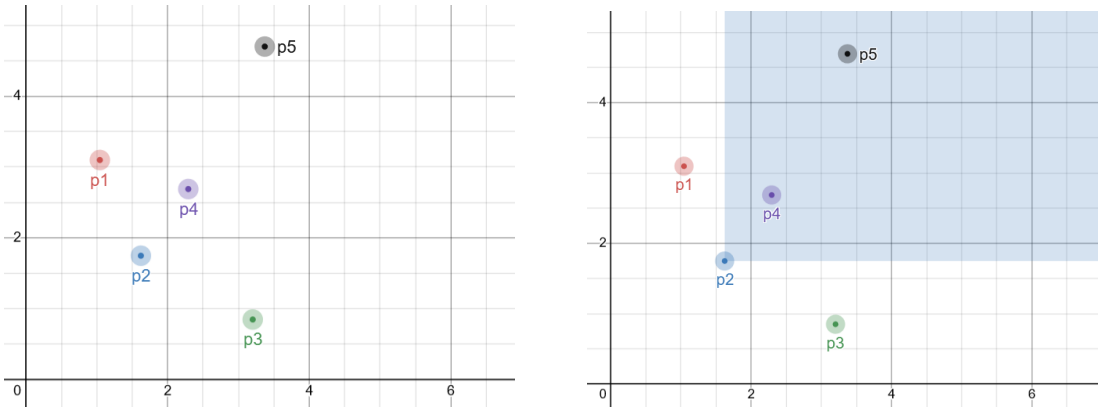


Figure 6.19: PICEA-g-mr.

points, as such, they are output-sensitive [8]. Note that there is no need to query points returned on a previous query, as the query would return a subset of the previous. Taking Figure 6.20b as example, any points returned from the interval range query for point $p4$ would already have been returned in the query for point $p2$ if $p2$ has been query first.

For the purpose of this work, a KD-Tree approach was implemented in Python to get the non-dominated solutions. KD-Trees work by building a binary Tree over the M-dimensional points, partitioning the set of points at the median of alternating dimensions. It organizes the data in a M-dimensional space. It has a time complexity of $\mathcal{O}(M \times nlog(n))$ for building the KD-Tree and a worst time complexity of $\mathcal{O}(n \times n^{1-\frac{1}{M}} + n^2)$ for querying all points, for finding all the non-dominated individuals. The algorithms 2 and 3 present the pseudocode for the construction and query of a KD-Tree, respectively.

The proof for the worst time complexity is given below in 6.1. It depends on the number of visited points, v, which, in its turn, depends on the order on which points are visited.



(a) Non-dominated individuals criteria.

(b) Range query for point $p2$.

Figure 6.20: Example of non-dominated individuals.

$$\sum_{v} \mathcal{O}(n^{1-\frac{1}{M}} + k) \tag{6.1}$$

$$= \sum_{v} \mathcal{O}(n^{1-\frac{1}{M}}) + \sum_{v} \mathcal{O}(k) \tag{6.2}$$

$$= \mathcal{O}(n \times n^{1-\frac{1}{M}} + n^2) \tag{6.3}$$

For this, consider a layout of instances all on different non-dominated fronts. If the first iteration visits the least dominated front, then the algorithm only has one iteration, and the number of returned points from the query is $n-1$. On the other hand, if the order of solutions visited is from the most dominated to the least, then n iterations are needed, and the total number of returned points is $(0+1+...+n-1) = \mathcal{O}(n^2)$. Thus the order in which points are visited matters. However, as the Pareto-front is approximated, most instances will be on the least-dominated front; thus, many iterations are needed, but the number of returned points by the queries is low.

Algorithm 2: *Build_KD_Tree*

Data: $P, depth$

Result: *node*

if $|P| = 1$ **then**
  return *leaf with point in P*
end
$depth \leftarrow depth \bmod M$;
divide P into $P1$ and $P2$ at the median in criteria at index $depth$ at l;
$n_{left} \leftarrow Build\_KD\_Tree(P1, depth+1)$;
$n_{right} \leftarrow Build\_KD\_Tree(P2, depth+1)$;
return *node with* n_{left} *and* n_{right} *as child and* l *as median*

Algorithm 3: *Query_KD_Tree*

Data: $n, query$
Result: $points$
if *n is leaf* **then return** *point in n is point in query* ;
if $region(n.left) \in query$ **then return** *points in subtree n.left* ;
else if $region(n.left) \cap query \neq \emptyset$ **then**
    $Query\_KD\_Tree(n.left, query)$;
end
if $region(n.right) \in query$ **then return** *points in subtree n.right* ;
else if $region(n.right) \cap query \neq \emptyset$ **then**
    $Query\_KD\_Tree(n.right, query)$;
end

By performing a range query on every point, one can identify all the solutions that are not returned in any range queries, thus, are non-dominated. The algorithm is presented in 4. The algorithm as worst time complexity of $\mathcal{O}(M \times n\log(n) + n \times n^{1-\frac{1}{M}} + n^2)$, combining the KD-Tree build and algorithm for finding the non-dominated solutions.

Algorithm 4: Get non-dominated solutions.

Data: S
Result: $non\_dominated$
$kd \leftarrow KD\_Tree(S)$;
$non\_dominated \leftarrow S$;
for $s \in non\_dominated$ **do**
    $dominated \leftarrow kd.query(s)$;
    for $sd \in dominated$ **do**
        $non\_dominated.remove(sd)$;
    end
end
return $non\_dominated$

6.7 Choosing a solution

In order to aid the decision-maker in finding a solution, a Parallel Coordinates visualization tool was developed. It maps each solution's criteria to the visualization and uses range sliders to define the aspiration criteria. Each data item consists of criteria values mapped to a vertical axis, evenly horizontally spaced. Each value is scaled individually according to the minimum and maximum values for that criteria. Each data item corresponds to a polygonal line between consecutive criteria values for that item. For this work, instead of polygonal lines, Bézier Curves are used to achieve the desired effect.

A color palette was employed according to the first criteria to give more comparison capability with axes farther away and a more pleasing look. A distinctive color identifies the aspiration criteria.

In Figure 6.21, a possible use-case is demonstrated with data obtained in one algorithm's execution. It illustrates a possible decision-maker in the process of choosing an instance that best fits the interests.

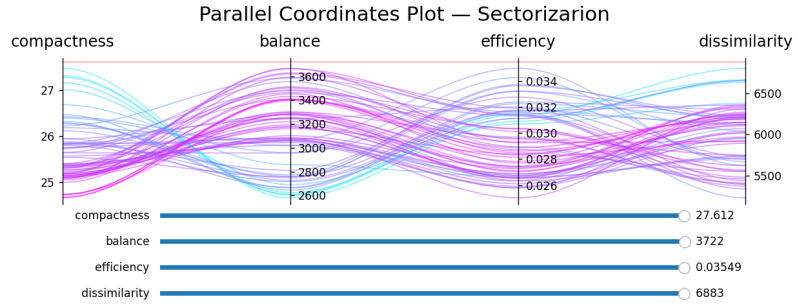
- In the beginning, the decision-maker is presented with all the instances, each represented by a line crossing at the correspondent criteria values, as shown in Figure 6.21a.
- The decision-maker has access to a set of range sliders, on which it is possible to define the aspiration criteria, represented by a line in red. The decision-maker aims for more balanced and efficient instances; as such, he lowers the range sliders to a value of his liking. The instances that violate the aspiration criteria are discarded. The decision-maker keeps decreasing the range sliders, resulting in Figure 6.21b.
- At this stage, there are still too many instances for the decision-maker to choose from. This is an iterative process, in which the aspiration criteria can be reduced and the instances filtered. In Figure 6.21c, the decision-maker has reached a state where most instances have been filtered out; a small set of similar ones is also presented.
- In Figure 6.21d, the decision-maker proceeds to select a single and preferred instance out of the set of non-dominated instances.

Building Bézier curves

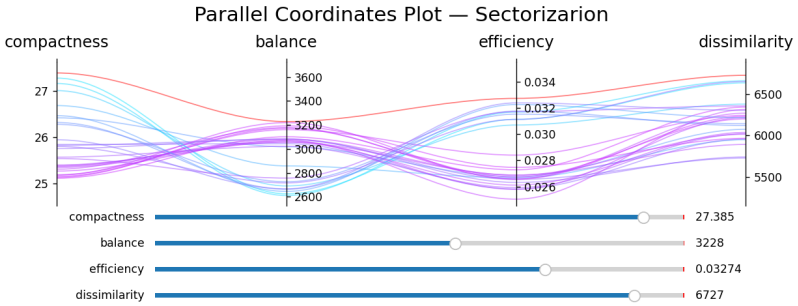
Bézier curves are parametric curves that define a continuous and smooth curve between two endpoints, with respect to a set of control points [42]. For this work, a cubic Bezier curve was used, which means, there are two endpoints and two control points. Let $B_{i,0}$ and $B_{i,1}$ be the curve endpoints, such that $B_{i,0} = (x_i, y_i)$ and $B_{i,1} = (x_{i+1}, y_{i+1})$, where x_i represents the horizontal coordinate of axis corresponding to criteria listed as i and y_i the value for criteria listed as i for a data item. Then, the control points, $B^c_{i,0}$ and $B^c_{i,1}$, represent the two control points for the curve between criteria i and $i+1$ for a data item, such that $B^c_{i,0} = (x_i + \frac{(x_{i+1}-x_i)}{3}, y_i)$ and $B^c_{i,1} = (x_i + \frac{2\times(x_{i+1}-x_i)}{3}, y_{i+1})$.

For every pair of consecutive axes, a Bézier Curve is defined and drawn, such that curve endpoints that lay on axes, not on the extremities, will be part of different two Bézier curves, $B_{i,1} = B_{i+1,0} = B_m$. As the curves join on the axis, they form a longer and more complex structure, called a composite curve. In order to make a smooth transition between curves, the tangent of the two curves on the point where they meet, B_m, must be co-linear. The tangent of a curve at endpoint $B_{i,0}$ is determined by the vector $B^c_{i,0} - B_{i,0}$ and for endpoint $B_{i,1}$ is determined by $B_{i,1} - B^c_{i,1}$ [42].

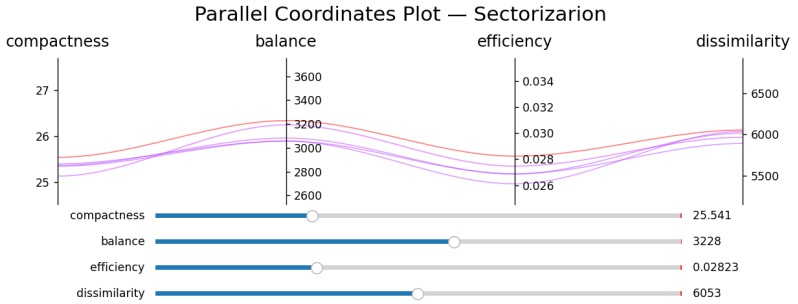
Putting all of this together, at the point B_m, the tangent at B_m of the first curve is $B^c_{i,0} - B_m$ and for the second curve $B_m - B^c_{i,1}$. So, the tangent is controlled by the control points. The formulation



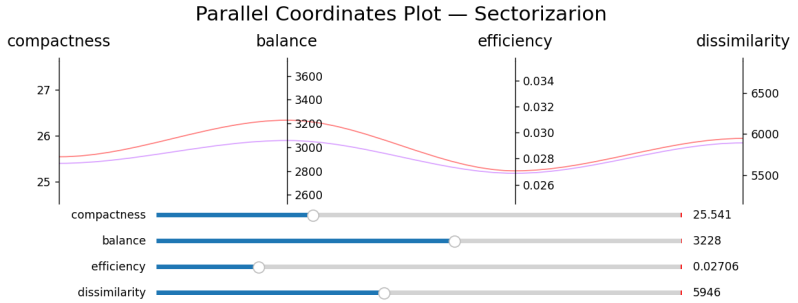
(a) Selecting a solution - start.



(b) Selecting a solution - iterative process 1.



(c) Selecting a solution - iterative process 2.



(d) Selecting a solution - reaching a decision.

Figure 6.21: Example of tool usage.

above, produces co-linear tangents on B_m as the control points lay on the same y coordinate, $B^c_{i,1} = (x_i + \frac{2\times(x_{i+1}-x_i)}{3}, y_{i+1})$ and $B^c_{i+1,0} = (x_{i+1} + \frac{(x_{i+2}-x_{i+1})}{3}, y_{i+1})$. Figure 6.22 shows an example of the Bézier curves following the description above, for a data values of $\{1, 1.8, 1.4\}$ and axis on x coordinates $\{0, 1, 2\}$.

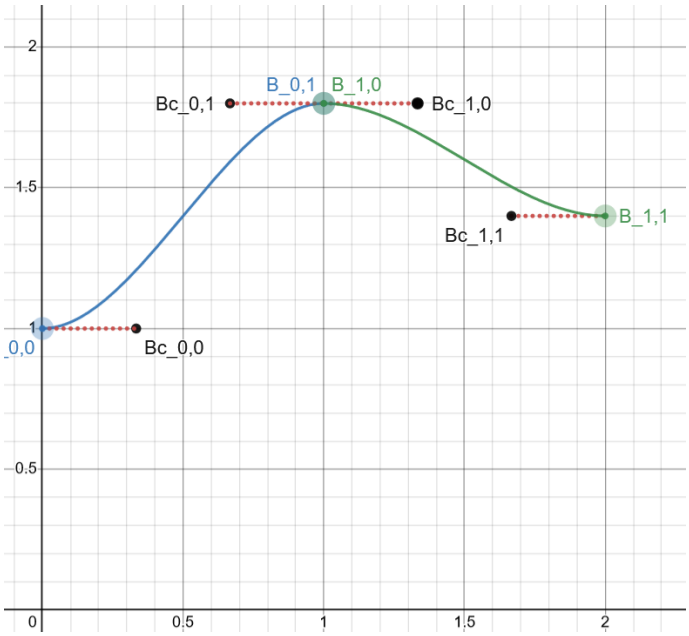


Figure 6.22: Bézier curves and correspondent control and end points.

Chapter 7

Results

The chapter describes the experiments carried out and presents the results achieved. An analysis of these results is also included.

7.1 Experiments

Two experiments were proposed in this work to evaluate the behavior of the algorithm and fit to the problem under study.

Experiment 1

The first experiment aims to evaluate how the algorithm behaves on different values for the radius of the sectors. Three values are considered for the radius r:

- 0 : flexible border level-0.
- 1 : flexible border level-0 and level-1.
- 2 : flexible border level-0, level-1 and level-2.

The radius of the border affects how the daily allocation and the correspondent routes are constructed.

Experiment 2

The second experiment aims to evaluate how the algorithm behaves on different values of α for the Randomized Greedy Heuristics. Three values are considered for α:

- 0.05 : α is set to 0.05.
- 0.15 : α is set to 0.15.

- 0.25 : α is set to 0.25.

The Randomized Greedy Heuristics is used in the Construction procedure, when generating the global allocation and in the Genetic operators. The crossover operator employs the Construction procedure for generating a complete allocation and the mutation operator employs the procedure to generate new borders between sectors.

Experiment 3

The third experiment aims to evaluate how the algorithm behaves with and without *mating restriction* in the PICEA-g framework. Two values are considered for *mating restriction*:

- False : *mating restriction* is set to False.
- True : *mating restriction* is set to True.

The *mating restriction* promotes closer instances to be mated with each other more often then far away instances, during the genetic operator in the PICEA-g.

7.2 Experiments methodology

The data was generated for 30 working days, with an expected average of 3000 active clients per day. The problem size consists of 17,318 Basic Units.

The same instance of 30 working days is used in all algorithm repetitions and experiments. The algorithm was implemented in Python 3.10.4 and the repetitions were conducted on two computers with a Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz processor and 16.0 GB of memory.

For all experiments, apart from the parameter under study, the values were fixed and are expressed in Table 7.1.

Parameter	Values
Instance population size	40
Goals population size	120
Archive size	50
Iterations	50
Radius	1
Number of sectors	40
Working days	30
Mating restriction	True
Selection Pressure	2
Mutation rate	0.2
Dynamic RCL	True
α	0.2
w (depot)	41°14’10.0"N 8°39’57.5"W

Table 7.1: Algorithm parameter settings.

Ideally, the size of the instance population should be higher, around 100, for example, to take more advantage of the evolutionary framework of the algorithm. Moreover, the number of iterations should also be higher, allowing the algorithm to converge to the Pareto-Front. These decisions were made due to time constraints. The algorithm's time complexity is linearly dependent on the population size and the number of iterations. The number of repetitions is also relevant as it gives more confidence in the results the greater the number of repetitions. In order to be able to have several repetitions per value, a trade-off between the quality of the repetitions and the number of repetitions had to be made.

The algorithm repetitions return a set of non-dominated instances representing the approximation to the Pareto-Front. Each repetition might return a new and different set of non-dominated instances. For each pair experiment and value, 4 repetitions were conducted. In order to aggregate the results from several repetitions, the non-dominated instances of the joint instances of all repetitions for a pair experiment and value were considered. Let this set of non-dominated instances be the representative result of all repetitions of a pair experiment and value.

In order to analyze the results of each experiment and compare the approximation of the Pareto-front between different values, three metrics were used: (i) the Set-Coverage, also known as C-metric, (ii) the Schott's Spacing of the instances, known as SS metric, and, (iii) the Hypervolume Indicator, also known as S. The values are rounded to 4 decimal places.

The Set-Coverage of A and B measures the percentage of instances in B that are dominated by at least one solution in A. Note that $C(A,B) \neq 1 - C(B,A)$, the relationship is not symmetric. If $C(A,B) = 1$, then all the instances in B are dominated by at least one instance in A. In contrast, if $C(A,B) = 0$, then no instance in B is dominated by an instance in A [62]. This metric is also used in the work of [39].

$$C(A,B) = \frac{|\{x \in B | \exists y \in A : y \preceq x\}|}{|B|} \tag{7.1}$$

Schott's Spacing measures the spread of the instances in the approximation of the Pareto-Front, how well the instances are distributed over the Pareto-Front. It measures the distance variance between neighboring instances. A value of 0 means that all the instances are equally spaced from one another. A set of instances that minimizes this metric tends to better represent the Pareto-Front, as solutions are not crowded in the same location [53].

$$SS(A) = \sqrt{\frac{1}{|A|-1} \times \sum_{x \in A} (\overline{d} - d_x)^2} \tag{7.2}$$

$$d_x = min_y(\sum_{i}^{M} |f_i(x) - f_i(y)|) \tag{7.3}$$

The Hypervolume Indicator measures the volume of the M-dimensional region between the non-dominated front A and a Reference point, R. For the same Reference point, the higher the

hypervolume, the better the front represents the Pareto-front. It can be seen as the size of the dominated space [62].

$$H(A) = \Lambda(\bigcup_{p \in A, p \leq R} [p, R]) \tag{7.4}$$

where $\Lambda(.)$ is the Lebesgue measure, and $[p,R]$ is the minimum bounding box containing p and R.

The Reference point needs to be chosen. Notice that for different values of R, the hypervolumes measures change and the ordering of the fronts can change.

Definition 7.2.1 (Nadir-point)**.** The Nadir Point is constructed by all the worst objective values on the Pareto-Front, such that $z^n = (max_{x \in PF}(f_1(x), ..., max_{x \in PF}(f_{M-1}(x))$.

In this work, it was considered to use the Nadir Point as a Reference point for the Hypervolume Indicator metric. However, there is no known Pareto-Front from which the Nadir Point can be extrapolated. As such, an approximation of the Nadir Point was calculated.

The maximum objective values of all the non-dominated individuals from each repetition was calculated. This point represented an approximation of the Nadir Point in relation to all repetitions.

Note that, a more accurate representation, according to the definition in 7.2.1, would calculate the non-dominated individuals from all the repetitions and only then calculate the maximum values. This approach was not used, as then, the calculated point could be below a point in a non-dominated front from some repetition and that was not desirable.

As the repetitions return an approximation of the Pareto-Front, the maximum values could not correspond to the actual Nadir Point; a multiplicative factor was added to calculate the Reference point. The formulation is given in 7.5, where $\kappa \geq 1$. For this work, a value of $\kappa = 1.1$ was used.

$$z_t^n = (max_{x \in \cup A}(f_1(x), ..., max_{x \in \cup A}(f_{M-1}(x)) \tag{7.5}$$

$$R = \kappa \times z_t^n \tag{7.6}$$

where A is the union of all the results from all the repetitions of all experiments.

7.3 Results

Experiment 1

This experiment aims to evaluate the algorithm on different values for the border radius; a total of 4 repetitions for each value was conducted. Table 7.2 presents information on the repetitions for this experiment, namely the number of repetitions for each value, the average time per repetition and the standard deviation (STD) of the repetitions times.

Radius	Number of Repetitions	Average Time (d:h:m:s)	STD Time (h:m:s)
0	4	0:21:13:52	1:52:15
1	4	2:04:44:43	1:22:06
2	4	3:05:44:22	1:12:41

Table 7.2: repetition times for different radius values.

It is possible to observe that as the radius parameter increases, so does the average repetition time of the algorithm.

Table 7.3 compares different radius values among themselves using the C-metric. Table 7.4 compares different radius values using the SS and HyperVolume Indicator.

A \ B	0	1	2
0	0.0	0.0563	0.0145
1	0.0	0.0	0.1087
2	0.0	0.0563	0.0

Table 7.3: Comparative results using the Set Coverage for different values of radius.

Radius	N	SS	Hypervolume
0	141	9.0102	2302066.3651
1	142	54.9890	1761681.5498
2	138	24.1263	1338661.3418

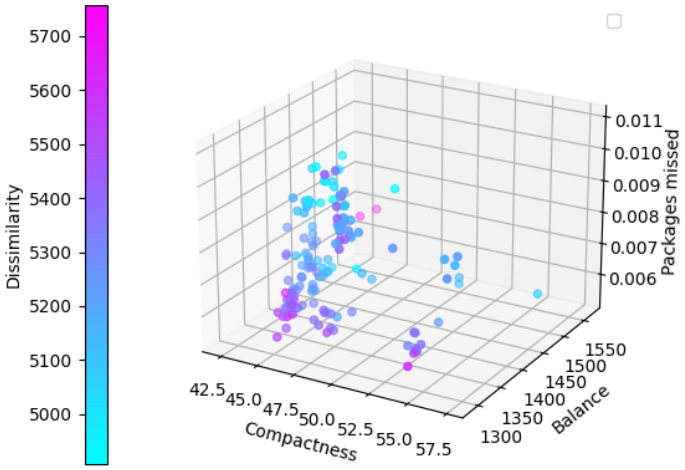
Table 7.4: Comparative results, stating the number of individuals, using the Spacing, Hypervolume Indicator for different values of radius.

Figures 7.1, 7.2, and, 7.3 show several visualizations of the representative non-dominated front for each of the values, radius of 0, of 1, and, of 2, respectively.

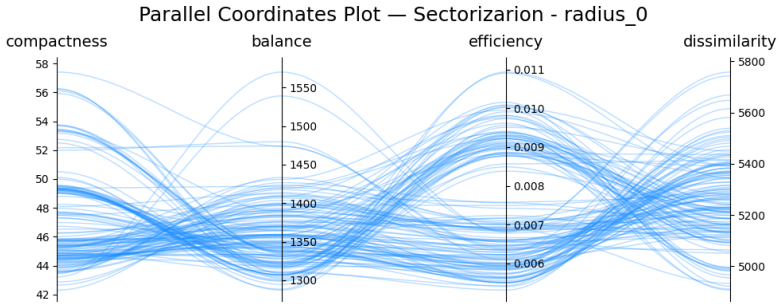
About 10% of the instances of radius 2 are dominated by at least one of radius 1. The radius of 1 has a high SS metric; as such, the solutions are not uniformly distributed over the non-dominance front. On the other hand, the 5% of the instances of radius 1 are dominated by the ones of radius 0 and 2. The low values might indicate that they represent different regions of the Pareto-front.

A radius of 0 has a Hypervolume Indicator higher than the rest of the values and a lower SS metric. This indicates that the instances of radius of 0 are more uniformly distributed and represent better the Pareto-Front. The ones of radius 1, are not very uniformly distributed, however, they represent the Pareto-Front better than the instances of radius 2, which might indicate a high density region, as it is possible to observe in Figure 7.2a.

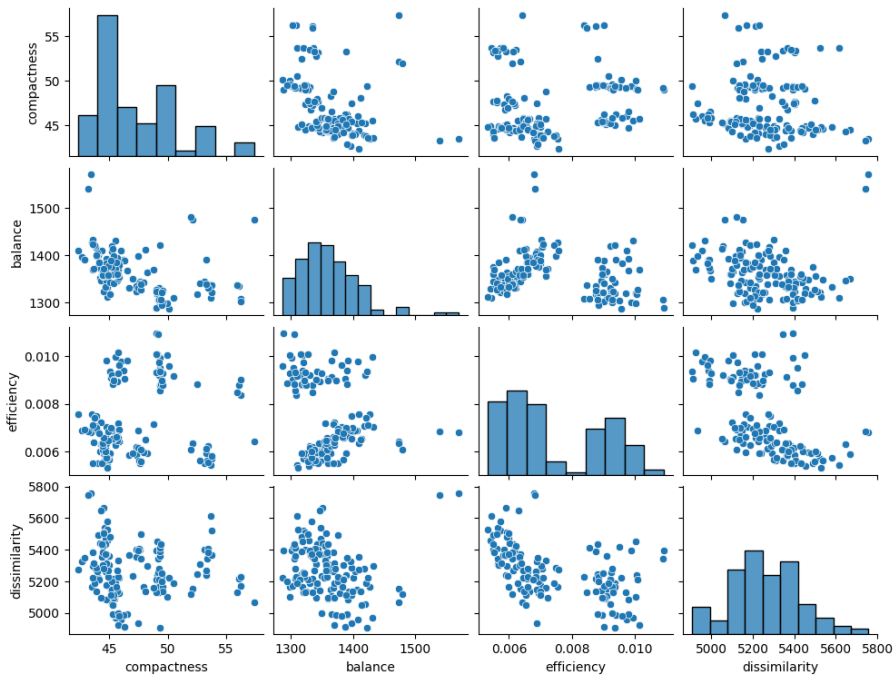
Comparing the instances criteria of the 3 values for radius, in Figure 7.4, it is possible to observe that the Dissimilarity values increase with the radius, as it is expected. Moreover, the Efficiency also increases with the radius, so, one can hypothesize that the greedy heuristics used



(a) 3D Scatter plot.

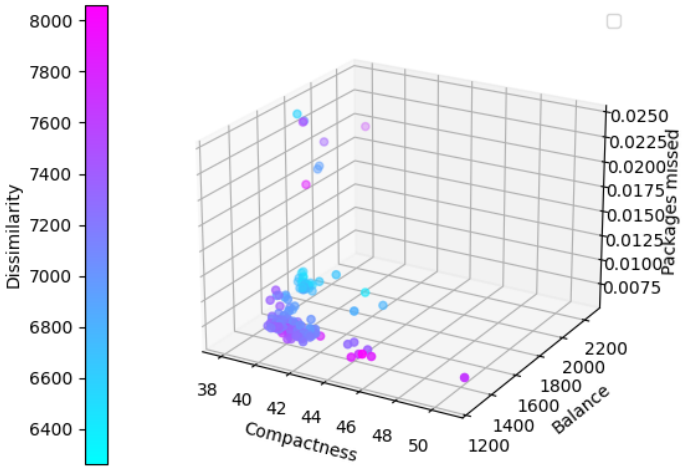


(b) Parallel Coordinates Visualization.

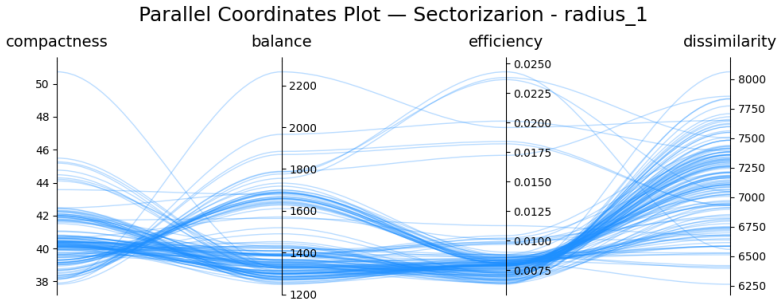


(c) Pair-wise scatter plots.

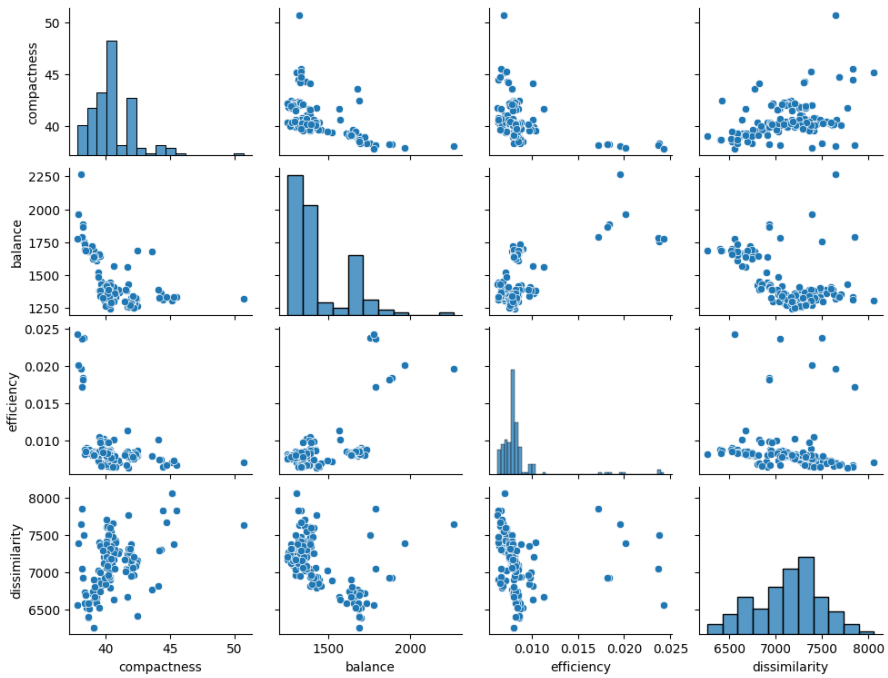
Figure 7.1: Non-dominated front visualizations for radius of 0.



(a) 3D Scatter plot.

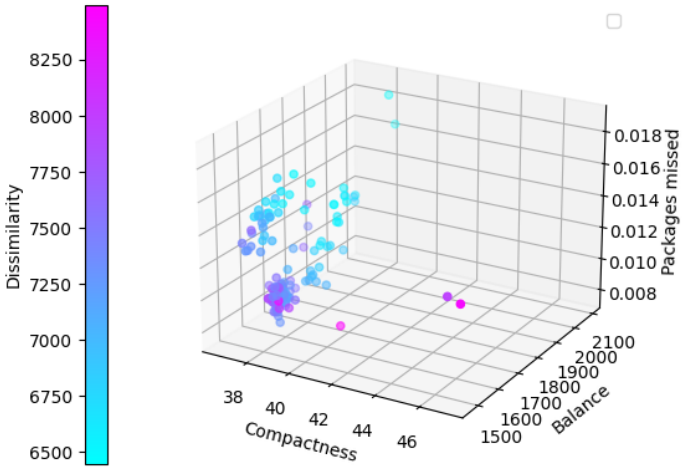


(b) Parallel Coordinates Visualization.

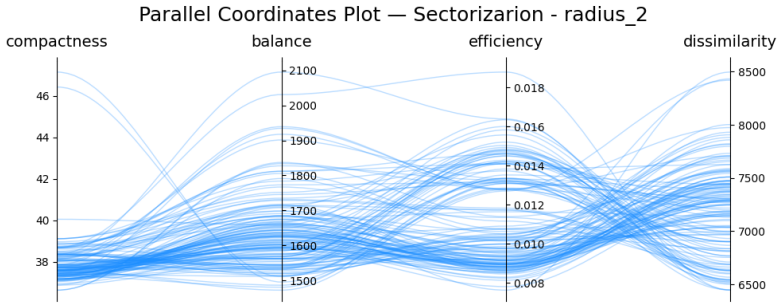


(c) Pair-wise scatter plots.

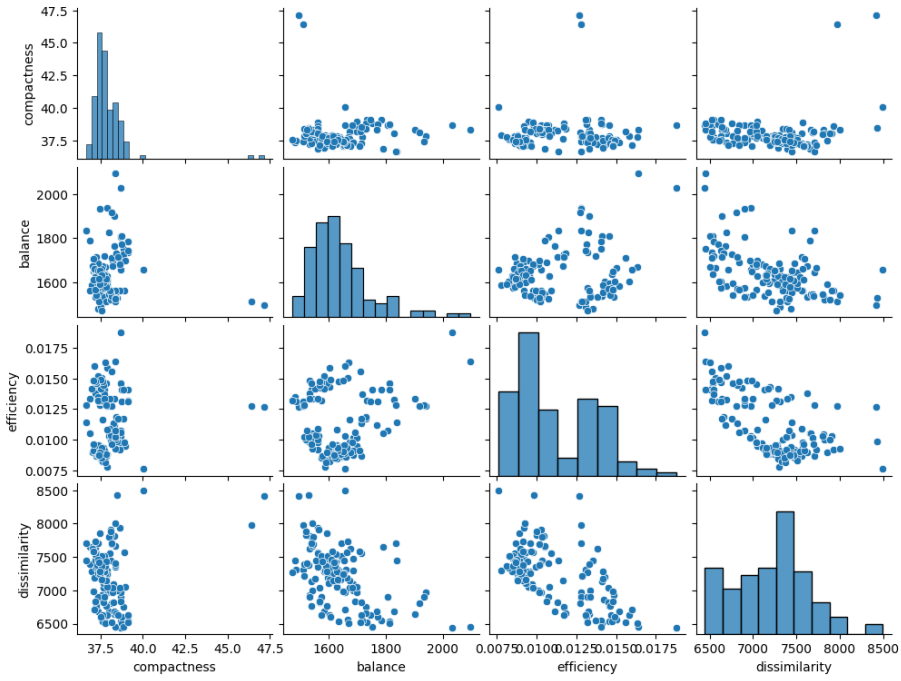
Figure 7.2: Non-dominated front visualizations for radius of 1.



(a) 3D Scatter plot.



(b) Parallel Coordinates Visualization.



(c) Pair-wise scatter plots.

Figure 7.3: Non-dominated front visualizations for radius of 2.

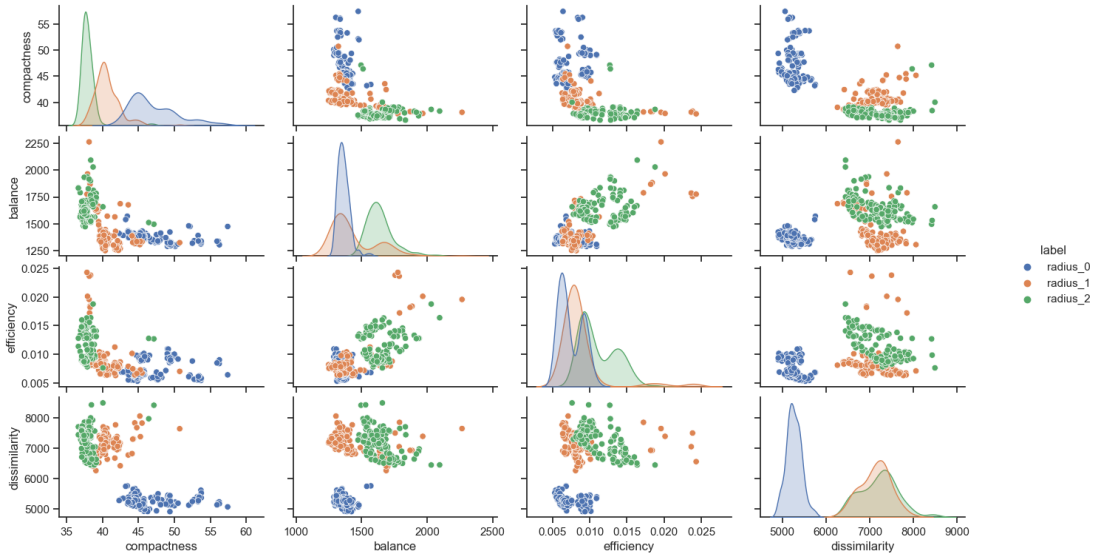


Figure 7.4: Pair-wise scatter plot for non-dominated fronts for different values of radius.

in the Daily generation of the routes do not perform well as the radius, and, the consequent amount of BUs to allocate, increases.

Experiment 2

This experiment aims to evaluate the algorithm on different values for the parameter α, a total of 4 repetitions for each value was conducted. Table 7.5 presents information on the repetitions for this experiment, namely the number of repetitions for each value, the average time per repetition and the standard deviation (STD) of the repetitions times.

α	Number of Repetitions	Average Time (d:h:m:s)	STD Time (h:m:s)
0.05	4	2:06:27:42	1:23:26
0.15	4	2:05:44:57	3:20:31
0.25	4	2:05:16:45	1:47:25

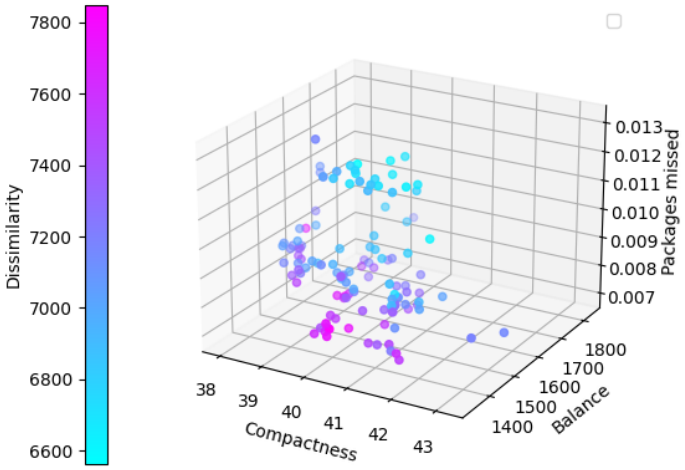
Table 7.5: repetition times for different α values.

It is possible to observe that the repetition time stays constant for different values of α.

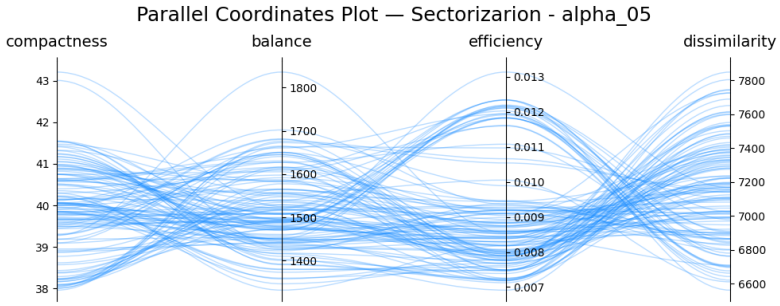
Table 7.6 compares different α values among themselves using the C-metric. Table 7.7 compares different α values using the SS and HyperVolume Indicator.

Figures 7.5, 7.6, and, 7.7 show several visualizations of the representative non-dominated front for each of the values, α equal to 0.05, to 0.15, and, to 0.25, respectively.

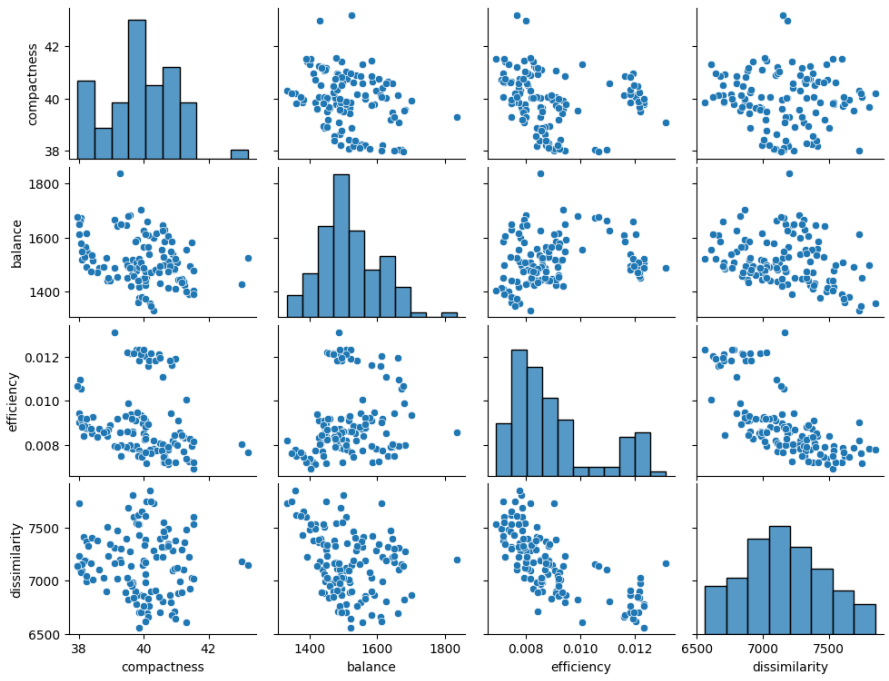
It is possible to see that an $\alpha = 0.25$ achieved better results. About 82% of the instances in 0.05 and 57% of 0.15 are dominated by 0.25. The instances of $\alpha = 0.15$ are more than half dominated by the other values. More repetitions of the experiment would give more confidence on the low performance of $\alpha = 0.15$.



(a) 3D Scatter plot.

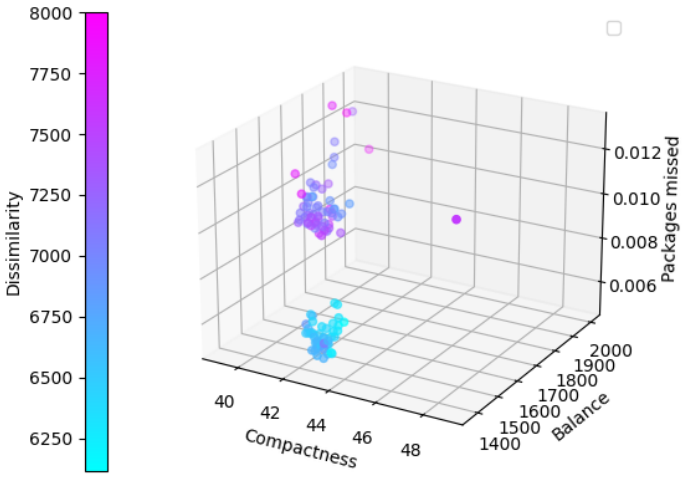


(b) Parallel Coordinates Visualization.

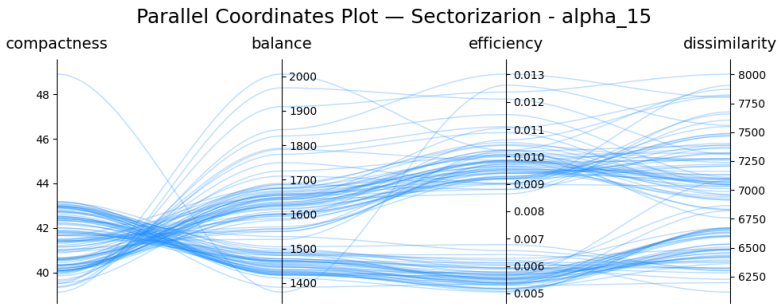


(c) Pair-wise scatter plots.

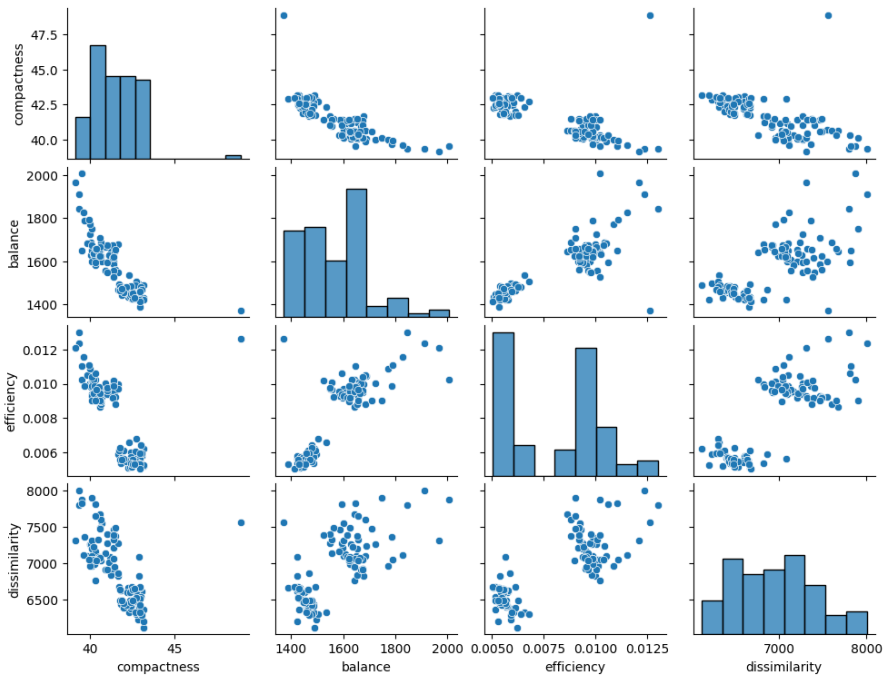
Figure 7.5: Non-dominated front visualizations for $\alpha = 0.05$.



(a) 3D Scatter plot.

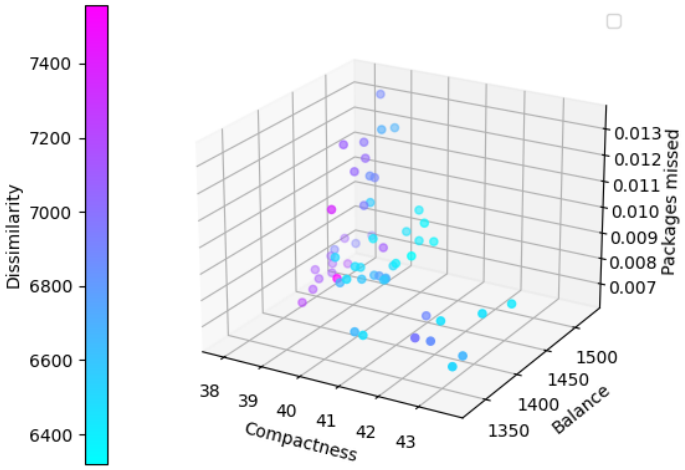


(b) Parallel Coordinates Visualization.

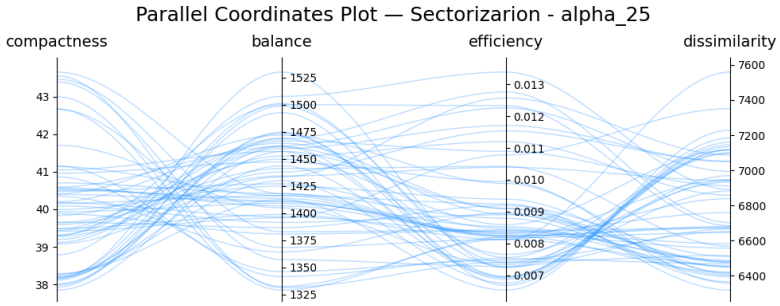


(c) Pair-wise scatter plots.

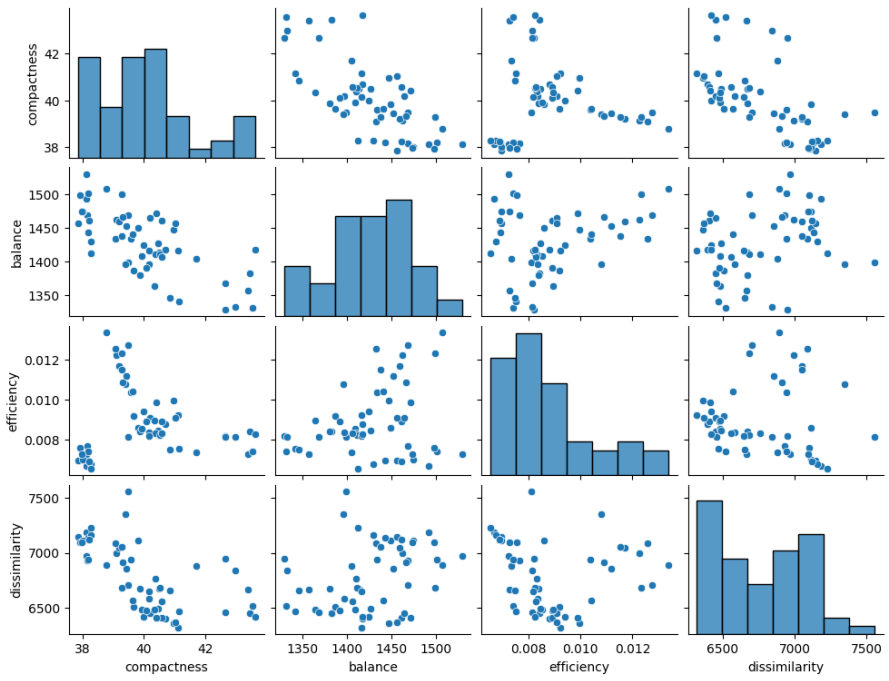
Figure 7.6: Non-dominated front visualizations for $\alpha = 0.15$.



(a) 3D Scatter plot.



(b) Parallel Coordinates Visualization.



(c) Pair-wise scatter plots.

Figure 7.7: Non-dominated front visualizations for $\alpha = 0.25$.

A \ B	0.05	0.15	0.25
0.05	0	0.5514	0.0179
0.15	0.0163	0.0	0.0
0.25	0.8211	0.5701	0.0

Table 7.6: Comparative results using the Set Coverage for different value of α.

α	N	SS	Hypervolume
0.05	123	23.49783	1490914.5067
0.15	107	40.4349	1729637.2174
0.25	56	32.8756	1723250.5220

Table 7.7: Comparative results, stating the number of individuals, using the Spacing, Hypervolume Indicator for different values of α.

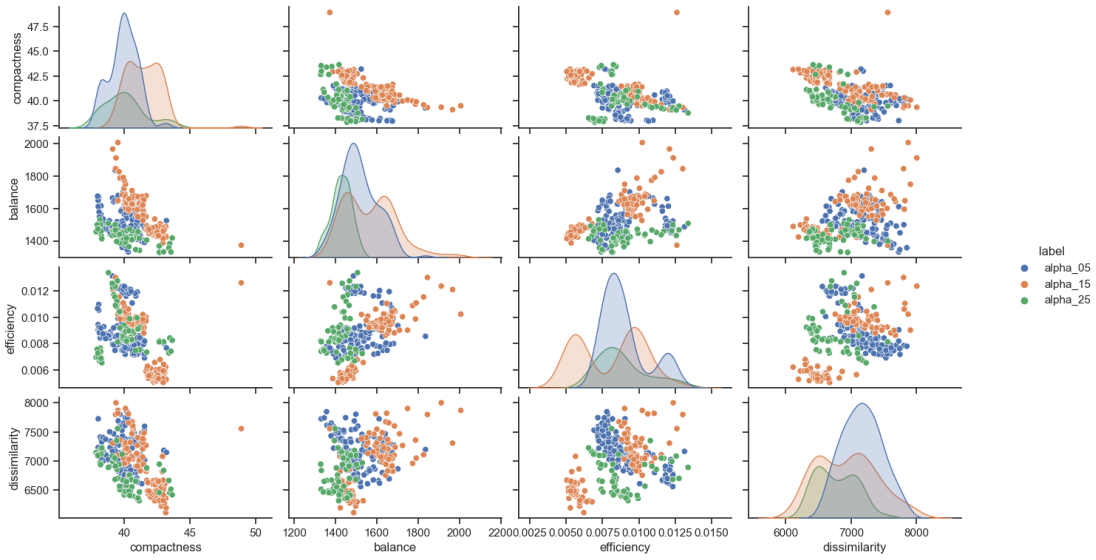


Figure 7.8: Pair-wise scatter plot for non-dominated fronts for different values of α.

The instances in $\alpha = 0.05$ are more uniformly distributed than the others as they have the lowest SS value; being the worst distributed the instances of $\alpha = 0.15$. The Hypervolume Indicator is close between values of 0.15 and 0.25 for α, but they are both higher than the one for $\alpha = 0.05$, which might indicate that values of 0.15 and 0.25 for α are closer to the Pareto-Front. Notice that $\alpha = 0.15$ has more instances than $\alpha = 0.25$, thus, for similar non-dominance fronts, more instances result in a higher HyperVolume value, as they represent better the space, with more "definition".

Even though, the instances of $\alpha = 0.15$ do not show good results using the Set Coverage metric, they represent well the Pareto-Front, according to the HyperVolume Indicator. The SS metric and Figure 7.6 show that the instances are concentrated in two clusters. This might indicate that the instances of α of 0.05 and 0.25 dominate one of the clusters, thus the half of instances

being dominated, according to the Set Coverage. Further analysis is needed.

On the other hand, $\alpha = 0.25$ shows good results on the C-metric and HyperVolume, and can be concluded that it presented better results in this experiments.

Concluding, this might indicate that the Construction heuristic can be improved to better fit the problem, as higher values of randomness work better; or that the optimal allocation is not in a very close neighborhood of the purely greedy allocation, hence, some degree of randomness is needed. This sustains the high complexity of the problem.

Experiment 3

This experiment aims to evaluate the algorithm with or without *mating restriction*; a total of 4 repetitions for each value was conducted. Table 7.8 presents information on the repetitions for this experiment, namely the number of repetitions for each value, the average time per repetition and the standard deviation (STD) of the repetitions times.

In this experiment, the repetitions from the Experiment 2 for $\alpha = 0.25$ were utilized for *mating restriction* set to True, in order to reduce the computation and time effort of the experiments. As such, an $\alpha = 0.25$ was also set for *mating restriction* set to False.

Mating-restriction	Number of Repetitions	Average Time (d:h:m:s)	STD Time (h:m:s)
False	4	2:03:23:41	1:52:19
True	4	2:05:16:45	1:47:25

Table 7.8: repetition times for different α values.

It is possible to observe that the repetition time stays constant for different values of α.

Table 7.9 compares the two values for *mating restriction* using the C-metric. Table 7.10 compares different *mating restriction* values using the SS and HyperVolume Indicator.

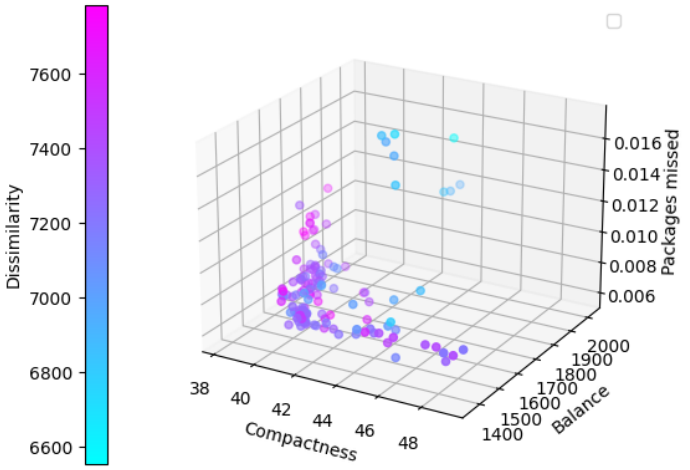
A \ B	False	True
False	0.0	0.0
True	0.9051	0.0

Table 7.9: Comparative results using the Set Coverage for different value of *mating restriction*.

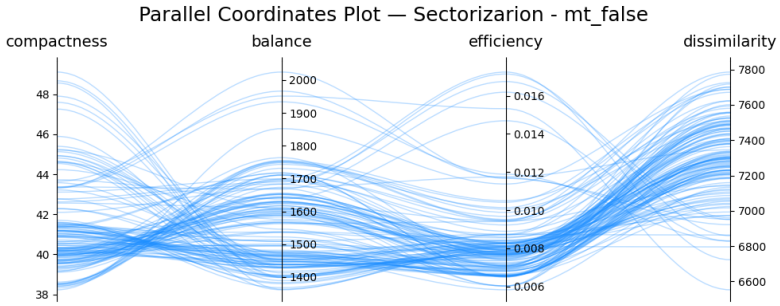
Mating-restriction	N	SS	Hypervolume
False	137	31.9160	1422863.2634
True	56	32.8756	1723250.5221

Table 7.10: Comparative results, stating the number of individuals, using the Spacing, Hypervolume Indicator for different values of *mating restriction*.

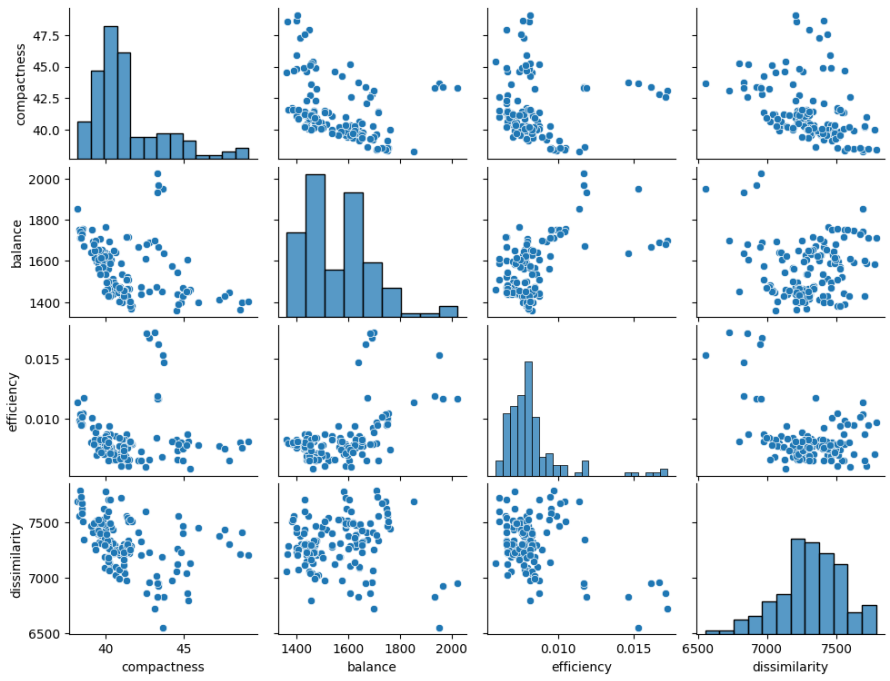
Figures 7.9 and 7.10 show several visualizations of the representative non-dominated front for each of the values, *mating restriction* equal to False and to True, respectively.



(a) 3D Scatter plot.

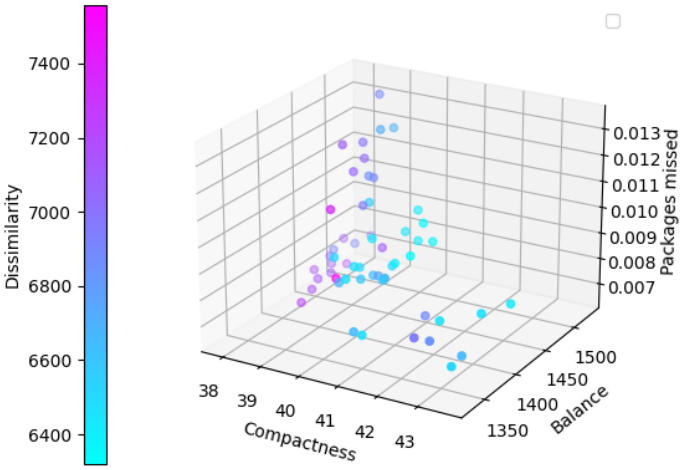


(b) Parallel Coordinates Visualization.

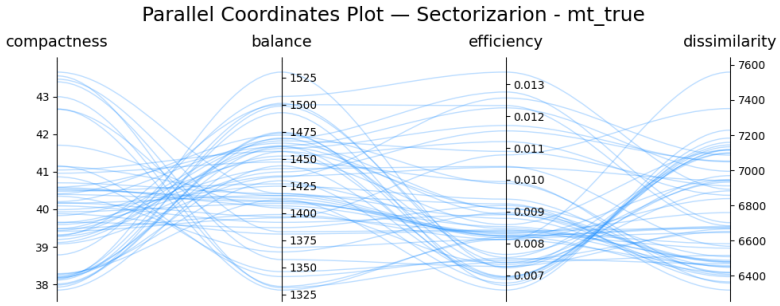


(c) Pair-wise scatter plots.

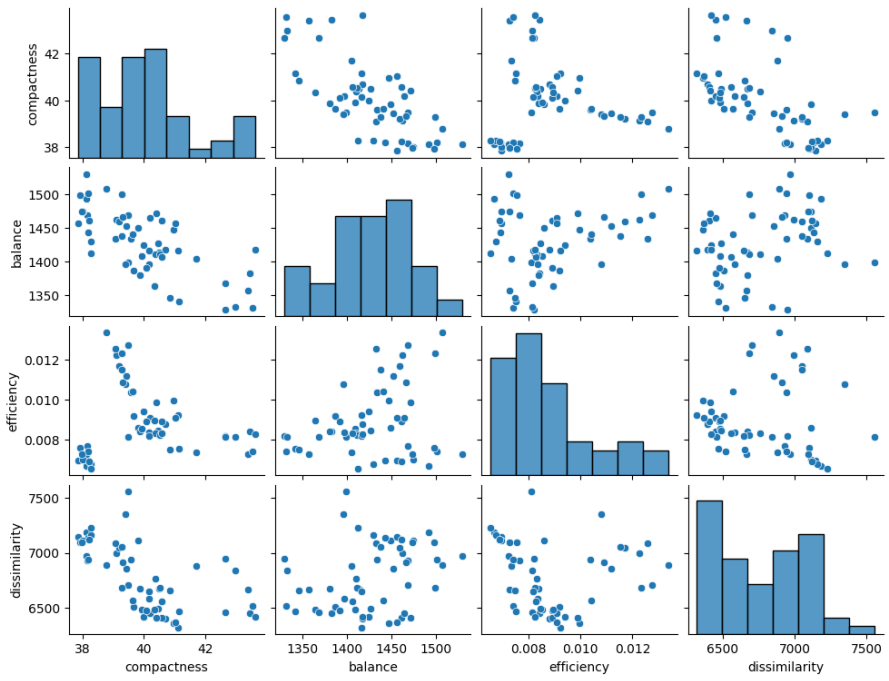
Figure 7.9: Non-dominated front visualizations for *mating restriction* set to False.



(a) 3D Scatter plot.



(b) Parallel Coordinates Visualization.



(c) Pair-wise scatter plots.

Figure 7.10: Non-dominated front visualizations for *mating restriction* set to True.

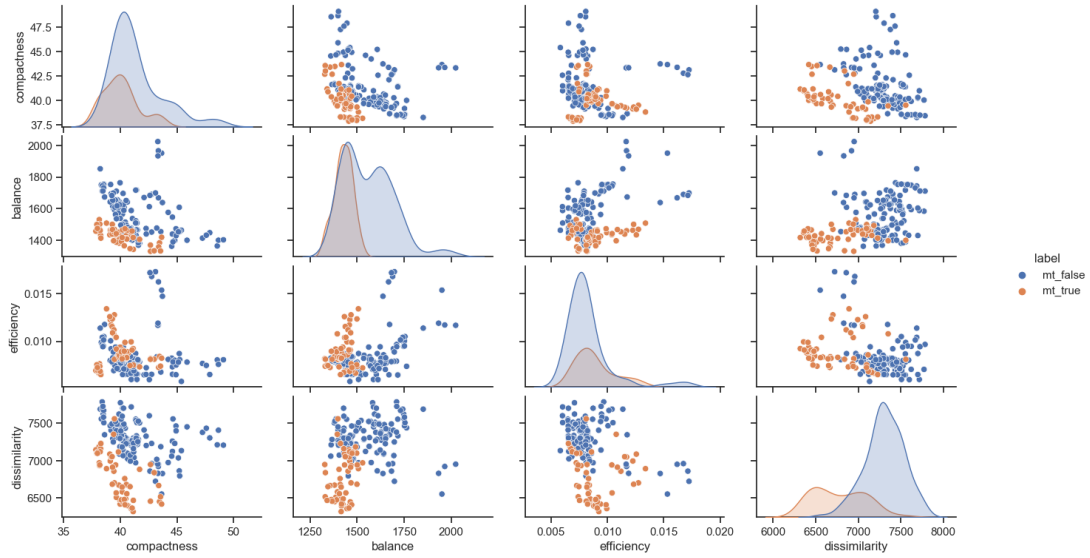


Figure 7.11: Pair-wise scatter plot for non-dominated fronts for different values for the *mating restriction*.

In this experience, it is possible to clearly observe that the *mating restriction* produced better results and a better and faster convergence toward the PF. About 90% of the individuals with *mating restriction* $= False$ are dominated by the individuals with *mating restriction* $= True$, which is supported by Figure 7.11. Moreover, the HyperVolume Indicator is greater with *mating restriction*, even though the number of instances is less.

Chapter 8

Conclusions

Last-mile delivery is a regular process in many business chains. It makes part of everyday life. It is a complex problem, and adequate planning and management of the last-mile logistics are crucial for last-mile delivery companies. This dissertation studies the definition of geographic areas, composed of smaller geographic regions, with routing. Geographic areas allow for the organization of parcels and each one is of responsibility of a driver, who has to deliver the parcels.

The problem studied is one of large dimension, with a total of 17,318 Basic Units, which represent the Porto Metropolitan Area, and an expected number of deliveries of 3,000, with 40 vehicles. A real-world example was taken into consideration to understand better the problem that last-mile logistics companies face and serves as a foundation for the problem modeling.

This dissertation modeled the last-mile logistics business in two ways. First, a data generation model for the problem was developed, aiming to reproduce real-world input data by simulating the parcel demand. Secondly, a Sectorization with Multi-Objective Optimization approach was implemented to solve the last-mile logistics problem.

The data generation simulates the delivery demand according to the population density. A space partitioning algorithm and the population demographics are combined to get a representation of the population density.

Last-mile delivery companies define drivers' areas of responsibility, in which drivers must deliver the parcels allocated to them. The nature of these areas of responsibility makes Sectorization adequate for this problem. The literature review also supports this, as several examples have used sectorization to design drivers' areas of responsibility with routing. The work developed in this dissertation successfully implemented a Sectorization approach capable of designing such areas, that are relevant to the business logic (answering **RQ1**). The approach is based on Sectorization for defining the drivers' areas and greedy heuristics for the daily allocation and routing.

The multiplicity of objectives in the last mile logistics business and the design of sectors require Multi-Objective Optimization. The corresponding literature applied to this problem is not

vast; however, it was possible to select a method that became suitable. A Preference Inspired Co-Evolutionary Algorithm with Goal vectors using Mating Restriction was implemented to optimize the multiple criteria that constitute the problem. It was shown that the implementation could minimize the criteria and generate a set of possible designs that converge towards the Pareto-Front (answering **RQ2**). Due to time constraints, the algorithm did not run for many iterations, which might not have allowed it to converge to the actual Pareto-Front or local optima. Moreover, the *mating restriction* has been shown to help reach a faster convergence.

The implementation takes a considerable time to reach a solution, in the order of days. However, the large dimension of the problem justifies this, and, as the definition of drivers' areas of responsibility does not go through change very often, the execution time is acceptable.

A graphical tool has been developed to aid the decision-maker in choosing the best-fit allocation to the company's interests. It is an interactive tool that allows to compare the criteria of different solutions and filter the ones that do not match the company's goals (answering **RQ3**). A Parallel-Coordinates visualization has been adopted due to the high number of criteria. The usage of Parallel-Coordinates is supported by examples in the literature, providing a complete overview of the instances' criteria and allowing for an online selection of the instances.

8.1 Work Contributions

The work developed in this dissertation dealt with a last-mile delivery case study and solved associated multi-objective optimization problems.

More specifically, the following contributions are recapped:

- **Literature Review on Sectorization with Routing:** A review on the Sectorization with routing was performed, delivering the state-of-the-art Compactness metrics and methods employed to solve the problem.

- **Generation of Problem-Specific Data:** A visit to a last-mile logistics company was conducted to understand the data and the business logic, allowing for the modeling and generation of problem-specific data.

- **Reference Implementation:** The PICEA-g using Mating Restriction framework was implemented in Python and made available in a public repository [1].

- **Publishable Contribution:** The contents of this dissertation were summarized and synthesized into a paper, to be submitted to a domain-related conference or journal in the near future.

[1] https://github.com/gmtorres/PICEA-g

8.2 Future Work

This problem is one of high complexity. Several improvements could be made in the different steps of this dissertation, in the data generation model, the problem approach and solution, and the visualization tool.

The data generation models the problem under study. There are many alternatives to the process employed; however, some could improve the current result. A better and more accurate representation of the Postal Codes could help model the parcel demand better. The client generation could be made on top of a representation of the streets for a more accurate client location. Moreover, one could implement different client behavior based on the demographic information, generating different parcel demands. Also, a model supporting seasonal and demographic changes in client demand could be interesting.

The Delaunay Triangulation defines the neighborhood between Basic Units; a more accurate representation or other neighborhood graphs, such as Gabriel Graph or Relative Neighborhood Graph, could lead to different results. There could be different Compactness metrics that would fit better the problem under study. An analysis of other Compactness metrics and how they affect the solution to the problem could also be interesting. The current implementation and criteria are very dependent on the definition of the delivery routes. Other, and better, heuristics could also affect the solutions' quality.

A summative evaluation of the visualization tool would be of great value as it would validate the proposed visualization tool and collect improvements from real decision-makers dealing with last-mile logistics daily. A feature that could allow to select and visualize a particular allocation so that, the decision-maker could better understand and compare different allocations, would benefit the ability to choose the adequate instance.

Lastly, regarding the experiments, the number of iterations and the population size could be greater, bringing more confidence and quality to the results obtained, followed by a more thorough analysis of the results.

References

[1] Reyan Ahmed, Greg Bodwin, Faryad Darabi Sahneh, Keaton Hamm, Mohammad Javad Latifi Jebelli, Stephen Kobourov, and Richard Spence. Graph spanners: A tutorial review. *Computer Science Review*, 37:100253, 2020.

[2] Eduardo Álvarez Miranda and Jordi Pereira. A districting application with a quality of service objective. *Mathematics*, 10, 1 2022.

[3] Florian Arnold and Kenneth Sörensen. What makes a vrp solution good? the generation of problem-specific knowledge for heuristics. *Computers and Operations Research*, 106:280–288, 6 2019.

[4] Johannes Bader and Eckart Zitzler. Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 19(1):45–76, 2011.

[5] Matthias Bender, Jörg Kalcsics, and Anne Meyer. Districting for parcel delivery services – a two-stage solution approach and a real-world case study. *Omega (United Kingdom)*, 96, 10 2020.

[6] Emna Benzarti, Erol Sahin, and Yves Dallery. Modelling approaches for the home heath care districting problem. 02 2013.

[7] Emna Benzarti, Evren Sahin, and Yves Dallery. Operations management applied to home care services: Analysis of the districting problem. *Decision Support Systems*, 55:587–598, 05 2013.

[8] Mark De Berg, Otfried Cheong, Marc Van Kreveld, and Mark Overmars. *Computational geometry: Algorithms and applications*. Springer Berlin Heidelberg, 2008.

[9] Burcin Bozkaya, Erhan Erkut, and Gilbert Laporte. A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research*, 144:12–26, 01 2003.

[10] Alexander Butsch. *Districting Problems - New Geometrically Motivated Approaches*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2016.

[11] M. Camacho-Collados, F. Liberatore, and J. M. Angulo. A multi-criteria police districting problem for the efficient and effective design of patrol sector. *European Journal of Operational Research*, 246:674–684, 10 2015.

[12] Stephanie Chevalier. Global retial e-commerce market size 2014-2023 | Statista retail e-commerce sales worldwide from 2014 to 2025, 2022. https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/, accessed 2022-02-09.

[13] Didem Cinar, Konstantinos Gakis, and Panos M. Pardalos. A 2-phase constructive algorithm for cumulative vehicle routing problems with limited duration. *Expert Systems with Applications*, 56:48–58, 9 2016.

[14] Instituto Nacional de Estatística. Instituto Nacional de Estatística - 2011 base cartográfica, 2021. https://censos.ine.pt/xportal/xmain?xpid=CENSOS&xpgid=censos_base_cartogr, accessed 2022-06-21.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[16] Statista Research Department. Logistics industry - market size 2018-2024 | Statista size of the global logistics industry from 2018 to 2024, 2021. https://www.statista.com/statistics/943517/logistics-industry-global-cagr/, accessed 2022-01-06.

[17] Statista Research Department. Logistics industry - road freight market size in Europe 2010-2025 | Statista size of the road freight market in europe from 2010 to 2025, 2021. https://www.statista.com/statistics/1068472/road-freight-market-size-europe/, accessed 2022-01-06.

[18] Antonio Diglio, Stefan Nickel, and Francisco Saldanha da Gama. Towards a stochastic programming modeling framework for districting. *Annals of Operations Research*, 292:249–285, 9 2020.

[19] Lewis Dijkstra and Hugo Poelman. A harmonised definition of cities and rural areas: the new degree of urbanisation, 2014.

[20] Djordje Dugošija, Aleksandar Savić, and Zoran Maksimović. A new integer linear programming formulation for the problem of political districting. *Annals of Operations Research*, 288:247–263, 5 2020.

[21] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551–559, 1983.

[22] Roland G. Fryer and Richard Holden. Measuring the compactness of political districting plans. *The Journal of Law and Economics*, 54(3):493–535, 2011.

[23] Peichao Gao, Haoyu Wang, Samuel A. Cushman, Changxiu Cheng, Changqing Song, and Sijing Ye. Sustainable land-use optimization using nsga-ii: theoretical and experimental comparisons of improved algorithms. *Landscape Ecology*, 36:1877–1892, 7 2021.

[24] GlobalTranz. What is last mile logistics & why are more shippers looking at this transportation function?, 2017. https://www.statista.com/statistics/943517/logistics-industry-global-cagr/, accessed 2022-01-06.

[25] Rosa Gonzalez Ramirez, Neale Smith, Ronald Askin, and Dr. Vyacheslav Kalashnikov. A heuristic approach for a logistics districting problem. *International Journal of Innovative Computing, Information and Control*, 6:3551–3562, 08 2010.

[26] Rosa G. González-Ramírez, Neale R. Smith, Ronald G. Askin, José Fernando Camacho-Vallejo, and Jose Luis González-Velarde. A grasp-tabu heuristic approach to territory design for pickup and delivery operations for large-scale instances. *Mathematical Problems in Engineering*, 2017, 2017.

[27] Wes Gurnee and David B. Shmoys. Fairmandering: A column generation heuristic for fairness-optimized political districting. In *ACDA*, 2021.

[28] Jialin Han, Yaoguang Hu, Mingsong Mao, and Shuping Wan. A multi-objective districting problem applied to agricultural machinery maintenance service network. *European Journal of Operational Research*, 287:1120–1130, 12 2020.

[29] Pierre Hansen, Nenad Mladenović, Raca Todosijević, and Saïd Hanafi. Variable neighborhood search: basics and variants. *EURO Journal on Computational Optimization*, 5:423–454, 9 2017.

[30] Maram Hasan and Rajdeep Niyogi. A meta-heuristic based multi-agent approach for last mile delivery problem. volume 1, pages 498–505. SciTePress, 2020.

[31] Robert E. Helbig, Patrick K. Orr, and Robert R. Roediger. Political redistricting by computer. *Commun. ACM*, 15(8):735–741, aug 1972.

[32] Fuyu Hu, Saini Yang, and Wei Xu. A non-dominated sorting genetic algorithm for the location and districting planning of earthquake shelters. *International Journal of Geographical Information Science*, 28:1482–1501, 2014.

[33] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. *2008 IEEE Congress on Evolutionary Computation*, pages 2424–2431, 2008.

[34] Jochen Janssens, Joos Van Den Bergh, Kenneth Sörensen, and Dirk Cattrysse. Multi-objective microzone-based vehicle routing for courier companies: From tactical to operational planning. *European Journal of Operational Research*, 242:222–231, 4 2015.

[35] Jerzy W. Jaromczyk and Godfried T. Toussaint. Relative neighborhood graphs and their relatives. *Proceedings of the IEEE*, 80:1502–1517, 1992.

[36] Ahmad I. Jarrah and Jonathan F. Bard. Large-scale pickup and delivery work area design. *Computers and Operations Research*, 39:3102–3118, 12 2012.

[37] Hongtao Lei, Gilbert Laporte, and Bo Guo. Districting for routing with stochastic customers. *EURO Journal on Transportation and Logistics*, 1:67–85, 6 2012.

[38] Hongtao Lei, Gilbert Laporte, Yajie Liu, and Tao Zhang. Dynamic design of sales territories. *Computers and Operations Research*, 56:84–92, 2015.

[39] Hongtao Lei, Rui Wang, and Gilbert Laporte. Solving a multi-objective dynamic stochastic districting and routing problem with a co-evolutionary algorithm. *Computers and Operations Research*, 67:12–24, 3 2016.

[40] Meiyan Lin, Kwai Sang Chin, Chao Fu, and Kwok Leung Tsui. An effective greedy method for the meals-on-wheels service districting problem. *Computers and Industrial Engineering*, 106:1–19, 4 2017.

[41] Flavien Lucas, Romain Billot, and Marc Sevaux. A comment on "what makes a vrp solution good? the generation of problem-specific knowledge for heuristics". *Computers and Operations Research*, 110:130–134, 10 2019.

[42] Michael E. Mortenson and Michael E. Mortenson. *Mathematics for computer graphics applications*. Industrial Press, 1999.

[43] Christoph Norrenbrock. Percolation threshold on planar euclidean gabriel graphs. *The European Physical Journal B - Condensed Matter and Complex Systems*, 89, 6 2014.

[44] Thomas Ormerod and E Chronicle. Global perceptual processing in problem solving: The case of the traveling salesperson. *Perception & psychophysics*, 61:1227–38, 09 1999.

[45] Yanfeng Ouyang. Design of vehicle routing zones for large-scale distribution systems. *Transportation Research Part B: Methodological*, 41:1079–1093, 2007.

[46] C. C. Robusto. The cosine-haversine formula. *The American Mathematical Monthly*, 64(1):38–40, 1957.

[47] Ana M. Rodrigues and J. Soeiro Ferreira. Sectors and routes in solid waste collection. In João Paulo Almeida, José Fernando Oliveira, and Alberto Adrego Pinto, editors, *Operational Research*, pages 353–375, Cham, 2015. Springer International Publishing.

[48] Diego Gabriel Rossit, Daniele Vigo, Fernando Tohmé, and Mariano Frutos. Visual attractiveness in routing problems: A review. *Computers & Operations Research*, 103:13–34, 2019.

[49] Roger Z. Ríos-Mercado and Hugo Jair Escalante. Grasp with path relinking for commercial districting. *Expert Systems with Applications*, 44:102–113, 2 2016.

[50] Roger Z. Ríos-Mercado and Elena Fernández. A reactive grasp for a commercial territory design problem with multiple balancing requirements. *Computers and Operations Research*, 36:755–776, 3 2009.

[51] Roger Z. Ríos-Mercado, Ada M. Álvarez Socarrás, Andrés Castrillón, and Mario C. López-Locés. A location-allocation-improvement heuristic for districting with multiple-activity balancing constraints and p-median-based dispersion minimization. *Computers and Operations Research*, 126, 2 2021.

[52] M. Gabriela Sandoval, Juan A. Díaz, and Roger Z. Ríos-Mercado. An improved exact algorithm for a territory design problem with p-center-based dispersion minimization. *Expert Systems with Applications*, 146, 5 2020.

[53] Jason R Schott. Csdl-t-1251 fault tolerant design using single and multicriteria genetic algorithm optimization, 1995.

[54] N. Srinivas and Kalyanmoy Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[55] Aydin Teymourifar, Ana Maria Rodrigues, and José Soeiro Ferreira. A new model for location-allocation problem based on sectorization. *Engineering World*, 3:92–96, 2021.

[56] Hanneke van der Hoef and Matthijs J. Warrens. Understanding information theoretic measures for comparing clusterings. *Behaviormetrika*, 46:353–370, 10 2019.

[57] Leonardo Vanneschi, Roberto Henriques, and Mauro Castelli. Multi-objective genetic algorithm with variable neighbourhood search for the electoral redistricting problem. *Swarm and Evolutionary Computation*, 36:37–51, 10 2017.

[58] Rui Wang, Robin C. Purshouse, and Peter J. Fleming. Preference-inspired coevolutionary algorithms for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 17:474–494, 2013.

[59] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[60] Thomas Wortmann and Jonathan Natanian. Multi-objective optimization for zero-energy urban design in china: A benchmark. 05 2020.

[61] Kui Zhang, Hexiang Yan, Han Zeng, Kunlun Xin, and Tao Tao. A practical multi-objective optimization sectorization method for water distribution network. *Science of the Total Environment*, 656:1401–1412, 3 2019.

[62] Eckart Zitzler. Evolutionary algorithms for multiobjective optimization: methods and applications. 1999.