Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica



Optimization models and algorithms for service planning

por

Gabriela Sánchez Yepez

como requisito para obtener el grado de

Doctorado en Ingeniería de Sistemas

Mayo 2026

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Secretaría de Estudios de Posgrado



Optimization models and algorithms for service planning

por

Gabriela Sánchez Yepez

como requisito para obtener el grado de

Doctorado en Ingeniería de Sistemas

Mayo 2026

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
Facultad de Ingeniería Mecánica y Eléctrica
Posgrado

Los miembros del Comité de Evaluación de Tesis recomendamos que la tesis "Optimization models and algorithms for service planning", realizada por la estudiante Gabriela Sánchez Yepez, con número de matrícula 1935064, sea aceptada para su defensa como requisito parcial para obtener el grado de Doctorado en Ingeniería de Sistemas.

El Comité de Evaluación de Tesis

Dra. María Angélica Salazar Aguilar
Directora

Dr. Igor Semionovich Litvinchev
Revisor

Dr. Roger Zirahuén Ríos Mercado
Revisor

Dr. Vincent André Lionel Boyer
Revisor

Dra. Pamela Jocelyn Palomo Martínez
Revisor

Dr. Omar Jorge Ibarra Rojas
Revisor

V.° B.°

Dr. Azael Martínez de la Cruz
Secretario de Estudios de Posgrado

SECRETARÍA DE ESTUDIOS DE POSGRADO

Institución 190001

Programa 557620

Acta núm. 386

Ciudad Universitaria, 25 de septiembre de 2026

En memoria de Cristina Ruiz Aguilar.

A mi familia.

Contents

Acknowledgements **xii**

Abstract **xiii**

Resumen **xv**

1 Introduction **1**

1.1 Motivation 2

1.2 Background 2

1.3 Objectives 3

1.4 Hypotheses 3

1.5 Thesis Structure 4

2 Literature Review **5**

2.1 Team Orienteering Problems and Variants 5

2.1.1 Solution Approaches 6

2.2 Workload Equity, Heterogeneity, and Compatibility in Routing Problems 7

2.3 Research Gap and Contributions 9

3 Team Orienteering Problem with Nonidentical Agents and Balanced Score **11**

3.1 Mathematical Modeling 11
3.2 Valid Inequalities for the TOPNB 14
3.2.1 Experimental Analysis 16
3.2.2 General Results 22
3.3 TOPNB Reformulation 26
3.3.1 Valid Inequalities 28

4 Branch and Cut for the TOPNB 31

4.1 Framework 31
4.2 Separation Algorithms 33
4.2.1 Cuts Acceptance Criteria 33
4.2.2 Logical and Infeasible Cover Cuts 33
4.2.3 Cuts Derived from a Max-Flow Problem 34
4.2.4 Matching Cuts 35
4.2.5 Lazy Cuts 35
4.3 Primal Heuristic 36
4.3.1 Constructing a Solution 36
4.3.2 Repairing a Solution 38
4.3.3 Improving a Solution 39
4.4 Computational Experiments 39
4.4.1 Performance of the B&C Algorithm 39
4.4.2 Cuts Analysis 41

5 Constraint Programming for the TOPNB 46

5.1 CP Model 46
5.2 Computational Experiments 48

5.2.1 CP Over Time 49

5.2.2 Linear versus Nonlinear Objective Function 51

6 Experimentation and General Results 54

6.1 Solution Quality 55

6.2 Relative Performance Analysis 56

6.3 Computational Efficiency 57

6.4 Case Study 60

6.4.1 Solution Quality 61

6.4.2 Characteristics of the Solutions 63

7 Conclusions 68

7.1 Summary of Main Findings 68

7.2 Scientific Contributions 70

7.3 Future Research 71

List of Figures

3.1 Percentage of instances with an objective value better, equal, or worse than the objective value obtained with the baseline model. 21

3.2 Gap reported by the solver. 23

3.3 Percentage deviation for common feasible solutions. 26

4.1 Branch-and-Cut framework. 32

4.2 Average of generated cuts by family and instance class. 41

4.3 Distribution of root node optimality Gaps across instance classes and model configurations. 43

5.1 Distribution of percentage improvements by class and time interval. . 51

6.1 Percentage deviation from the B&C objective value for each method and class. 56

6.2 Average CPU time by class for each method. 58

6.3 Percentage of fastest methods by class on instances where all methods reach the same solution. 59

6.4 Case study example. 60

6.5 Distribution of route durations obtained by CP under different time limits. 63

6.6 Distribution of route durations obtained in the ten AMH replicas. . . 64

6.7 Distribution of route scores obtained by CP under different time limits. 64

6.8 Distribution of route scores obtained in the ten AMH replicas. 65

List of Tables

3.1 MILP notation. 12

3.2 Tested models and valid inequalities included. 17

3.3 Distribution of instances by class and number of vehicles. 18

3.4 Tested instances for the valid inequalities analysis. 18

3.5 Deviations and relative CPU times. 20

3.6 Known optimal, new best known, and new optimal solutions. 22

3.7 Relative CPU times and optimal solutions. 24

3.8 Relative CPU times of common optimal solutions. 25

3.9 Reformulation notation. 27

4.1 Performance of B&C per class: optimal solutions and CPU times in seconds. 40

4.2 Root node performance summary by instance class. 42

4.3 Kruskal–Wallis test results for the comparison of cut configurations. . 44

5.1 CP notation. 47

5.2 CP percentage improvements by class over time. 50

5.3 Comparison of the nonlinear and linear CP formulations in terms of solution quality and number of optimal solutions. 52

5.4 Wilcoxon signed-rank test comparing the linear and nonlinear CP formulations globally and by instance class. 53

6.1 Number of optimal and best known solutions reached per method. . . 55

6.2 Case study data. 61

6.3 Comparison of AMH and CP for the case study. 62

6.4 CP performance under different time limits. 62

6.5 Operational indicators of the CP solutions. 65

6.6 Operational indicators of the AMH solutions. 66

Acknowledgements

I thank the Universidad Autónoma de Nuevo León and the Graduate Program in Systems Engineering for the opportunity to continue my academic training, and I extend my gratitude to CONAHCyT for the scholarship granted to me. My research would not have been possible without this doctoral fellowship.

I express my gratitude to the Sofía Kovalevskaya Foundation and the Mexican Mathematical Society (Sociedad Matemática Mexicana) for the award, which enabled me to complete my doctoral research and earn my degree.

I am grateful to my supervisor, María Angélica, for her guidance, infinite patience, and motivation during the development of this work.

Thanks to the members of my dissertation committee for taking the time to read this thesis and for their suggestions, which helped enhance its quality.

I also want to thank my family for their support, even at a distance.

My appreciation goes to all the people who met me on this journey. You have made my experience a lot more fun.

Finally, I also want to thank myself for never quitting.

Abstract

Gabriela Sánchez Yepez.

Ph.D. Candidate in Systems Engineering.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Title: Optimization models and algorithms for service planning.

Number of pages: 78.

This thesis studies a telecommunications service planning problem known as the Team Orienteering Problem with Nonidentical Agents and Balanced Score - (TOPNB). The problem consists of selecting a subset of service orders and determining their assignment and sequencing across available crews, with the objective of balancing the total collected score among them while satisfying operational and routing constraints.

The main objective of this research is to develop mathematical models and effective solution methods for this problem, contributing to the state of the art in vehicle routing, logistics, and transportation.

To this end, the following contributions are made:

1. An existing three-index mathematical formulation of the TOPNB is analyzed and strengthened through the incorporation of valid inequalities, improving the representation of the convex hull of integer solutions.
2. A reformulation of the existing model is introduced, leading to a more tractable model.
3. A branch-and-cut algorithm is developed to efficiently solve the proposed reformulation, achieving high-quality solutions and proving optimality for the majority of the benchmark instances.

4. A constraint programming model is proposed as an alternative formulation for the TOPNB. Computational results over an extensive set of artificial instances, as well as a real-world case study, show that this approach constitutes a competitive and reliable alternative.

A comprehensive computational analysis compares all proposed approaches, highlighting their relative strengths in terms of solution quality and efficiency. The results demonstrate that the developed methods constitute effective decision-support tools for real-world telecommunications applications, enabling improved resource allocation, balanced workload distribution among crews, and more efficient service planning.

Advisor: _____
María Angélica Salazar Aguilar, Ph.D.

Resumen

Gabriela Sánchez Yepez.

Candidata para obtener el grado de Doctorado en Ingeniería de Sistemas.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título: Optimization models and algorithms for service planning.

Número de páginas: 78.

Este trabajo estudia un problema de planificación de servicios de telecomunicaciones conocido como el problema de orientación por equipos con agentes heterogéneos y puntuación balanceada (en inglés, Team Orienteering Problem with Nonidentical Agents and Balanced Score (TOPNB)). El problema consiste en seleccionar un subconjunto de órdenes de servicio y determinar su asignación y la secuencia en que deben realizarse entre las cuadrillas disponibles, con el objetivo de equilibrar el puntaje total recolectado entre ellas, satisfaciendo al mismo tiempo las restricciones operativas y de ruteo.

El objetivo principal de esta investigación es desarrollar modelos matemáticos y métodos de solución eficaces para este problema, contribuyendo al estado del arte en ruteo de vehículos, logística y transporte.

Para ello, se hacen las siguientes contribuciones:

1. Se analiza y fortalece una formulación matemática previa del TOPNB mediante la incorporación de desigualdades válidas, mejorando el envolvente convexo de las soluciones enteras.
2. Se introduce una reformulación del modelo existente, lo que permite tener un modelo más tratable computacionalmente.
3. Se desarrolla un algoritmo de ramificación y corte para resolver eficientemente

la reformulación propuesta, logrando soluciones de buena calidad y probando optimalidad en la mayoría de las instancias del conjunto de prueba.

4. Se propone un modelo de programación por restricciones como una formulación alternativa para el TOPNB. Los resultados computacionales sobre un conjunto extenso de instancias artificiales, así como en un caso de estudio real, muestran que este enfoque constituye una alternativa competitiva y confiable.

Un análisis computacional integral compara todos los enfoques propuestos, destacando sus fortalezas relativas en términos de calidad de solución y eficiencia. Los resultados demuestran que los métodos desarrollados constituyen herramientas efectivas de apoyo a la toma de decisiones en aplicaciones reales de telecomunicaciones, permitiendo mejorar la asignación de recursos, equilibrar la carga de trabajo entre cuadrillas y optimizar la planificación de servicios.

Directora: _____
Dra. María Angélica Salazar Aguilar

Chapter 1

Introduction

Companies dedicated to provide services to customers at different geographical locations face the challenge of creating effective schedules by simultaneously combining service orders with workers' routes, shifts, and skills while complying with company policies and service requirements. In addition, these decisions must be made in a way that optimizes available resources, such as minimizing costs and travel distances, maximizing profits, or improving other performance metrics.

At the same time, company efficiency can be affected by nonmonetary factors such as service quality, customer satisfaction, and workload equity. For example, workload equity can increase acceptance of operational plans among drivers because they perceive that each crew is receiving a fair workload distribution, and, as a result, the quality of customer service improves. Consequently, workload equity has attracted attention for the tangible benefits it offers.

Although workload equity has been extensively studied in vehicle routing problems, its integration into profit-oriented routing problems remains comparatively limited. In particular, most Team Orienteering Problem formulations focus on maximizing the total collected profit, whereas equity considerations have received considerably less attention.

In this study, crew wages are directly related to workload, measured by the service orders each crew performs. Specifically, the total wage earned by a crew depends on the number and types of service orders assigned to it. Consequently, the main focus is on balancing wages among a set of crews in a team orienteering schedule. Since crews differ in their skills, each crew can perform only a subset of service orders. Accordingly, the problem incorporates compatibility constraints between customers and crews.

The results of this research are expected to contribute to the vehicle rout-

ing, logistics, and transportation literature by providing new models and solution methods.

1.1 Motivation

The problem presented in this thesis is motivated by a real-world situation faced by a local telecommunications company. The company offers different types of services to customers. Teams of technicians with different abilities and work shifts meet daily demand, meaning each crew can provide only a subset of services during its shift. Thus, a compatibility relationship exists between the service orders and the crews.

Each service has an associated score and service time that depend on its difficulty; specialized services yield higher scores and require longer service times than simple ones. It is important to note that the wage received for performing a service is proportional to its score; therefore, the total salary per crew depends on the number and types of service orders they perform.

Currently, the company assigns service orders to a supervisor, who builds the schedule based on experience, while technicians decide the order in which they perform the assigned services. However, this process does not always account for service and travel times properly, which can reduce productivity and increase costs. Moreover, assignments can be unfair, as total scores across crews are sometimes unbalanced.

Thereby, the following problem arises: how to select the subset of service orders to be performed by the available crews and the sequence in which they must be carried out, such that the collected score is balanced and the corresponding constraints are met.

1.2 Background

The core challenge we address in this work relates to different approaches in the literature to the vehicle routing problem family, where the Team Orienteering Problem (TOP) is one of the closest in features and among the most studied.

The relationship between the TOP and the studied problem allows it to be formulated, as shown in a previous study (Sánchez-Yepez, 2019), as a variant of the TOP with a different objective function and additional constraints to account for compatibility between vehicles and customers. This problem is referred to as the

TOP with nonidentical agents and balanced score (TOPNB). Two mathematical models and a solution algorithm based on a reactive GRASP (Greedy Randomized Adaptive Procedure) metaheuristic were proposed.

This thesis extends the previous work on the TOPNB by investigating strengthened and alternative mathematical formulations and exploring new solution methods for the problem.

1.3 Objectives

The main objective of the research is to study the telecommunications service planning problem, develop mathematical models for variants of the problem, and propose effective solution methods.

In order to achieve this goal, the following specific objectives are fulfilled:

- Strengthen existing mixed-integer linear programming formulations for the TOP with nonidentical agents and balanced score through the incorporation of valid inequalities.
- Develop and analyze alternative mathematical formulations that better capture the structural features of the problem.
- Design and implement different solution approaches, including exact and constraint-based methods.
- Perform a systematic comparison between the proposed approaches and existing methods from the literature in terms of solution quality and computational efficiency.
- Assess the applicability of the proposed methods in a real-world telecommunications context.

1.4 Hypotheses

- Strengthened mixed-integer linear programming formulations, combined with advanced exact solution methods, can significantly improve solution quality and computational performance.

- Alternative solution paradigms, such as constraint programming, provide competitive and robust performance for this class of problems, particularly for large-scale instances.

1.5 Thesis Structure

This thesis is organized into seven chapters, each addressing a specific component of the research.

Chapter 1 introduces the problem context, motivation, objectives, and research hypotheses, and outlines the overall structure of the thesis.

Chapter 2 presents a review of the relevant literature, focusing on the Team Orienteering Problems and related Vehicle Routing Problems.

Chapter 3 introduces the Team Orienteering Problem with Nonidentical Agents and Balanced Score (TOPNB). It presents the problem definition, the mathematical formulation, and a set of valid inequalities. An experimental analysis is conducted to evaluate different modeling configurations, leading to the selection of a robust formulation and the proposal of a reformulation.

Chapter 4 develops a Branch-and-Cut (B&C) algorithm to solve the TOPNB. This chapter describes the separation procedures, the primal heuristic, and the computational experiments used to evaluate the performance of the proposed approach.

Chapter 5 presents a Constraint Programming (CP) model derived from the TOPNB formulation. The chapter includes a computational analysis, with a focus on convergence behavior and the impact of linearizing the objective function.

Chapter 6 integrates all previous developments through a comprehensive computational study. It compares the proposed approaches in terms of solution quality and efficiency, and evaluates their performance against existing methods, including a mixed-integer linear programming (MILP) formulation and an Adaptive Multi-start Heuristic (AMH). Additionally, it includes a real-world case study, thereby connecting the methodological contributions with practical decision-making.

Finally, Chapter 7 summarizes the main findings and contributions, outlines directions for future research, and presents the scientific outcomes of this work, including peer-reviewed publications and presentations at national and international scientific meetings.

Chapter 2

Literature Review

The problem studied in this thesis lies at the intersection of team orienteering problems, workforce routing with heterogeneous skills, and workload balancing. Unlike classical TOP variants, it incorporates compatibility relations between crews and services, as well as the objective of balancing the collected score.

This combination of features is particularly relevant in real-world service systems, where decisions on which tasks to perform, how to assign them, and how to distribute the workload directly impact operational efficiency, service quality, and workforce satisfaction.

This chapter reviews the literature relevant to the problem studied in this thesis. The focus is on orienteering-type models, vehicle routing problems, and approaches that incorporate workload balancing. The chapter concludes by identifying the main research gaps and positioning the contributions of this work.

2.1 Team Orienteering Problems and Variants

The Orienteering Problem (OP) and its multi-vehicle extension, the Team Orienteering Problem, are well-established problems within the class of Vehicle Routing Problems with profits (Speranza et al., 2014). These problems consider a set of geographically distributed locations, each associated with a score, and aim to determine one or multiple routes, subject to a time budget, that maximize the collected score. Each location can be visited at most once, and all routes start and end at a depot.

A key distinguishing feature of OP and TOP compared to classical vehicle routing problems is the presence of selection decisions. In addition to answering the questions – *With which vehicle? In which order?* – they must also answer

the question – *Which customers to serve?* – because the profit associated with each customer makes it more or less attractive. Here, the vehicle decision refers to assigning customers to routes under the route duration budget. This selective nature makes OP and TOP particularly suitable for applications where not all customers can be served due to limited resources.

One of the most widely studied variants is the Team Orienteering Problem with Time Windows (TOPTW), in which service at each customer must occur within a predefined time interval (time window). This variant has been widely studied due to its relevance in applications such as tourist planning and service operations.

Over time, TOP variants have evolved to capture increasingly complex and realistic settings. These include capacity constraints (Hammami, 2024), clustered customers and time windows (Yu et al., 2024), multiple visits (Jung et al., 2024), time-dependent or interval-based profits (Li et al., 2024), synchronization and qualification requirements (Bendazzoli et al., 2024), mandatory visits (Kirac et al., 2023), stochastic travel times (Liu et al., 2025), and incompatibility constraints between pairs of nodes (Guastalla et al., 2025).

Despite these modeling advances, most TOP formulations continue to assume homogeneous agents and focus on maximizing the total collected profit. A notable exception is the heterogeneous TOP proposed by Chakraborty and Agarwal (2021), which allows nonidentical agents. Nevertheless, the objective remains the maximization of the total collected profit, without explicitly considering equity in the distribution of profits among agents.

Comprehensive surveys such as those by Vansteenwegen et al. (2011), Gunawan et al. (2016), and Vansteenwegen and Gunawan (2019) provide detailed overviews of OP and TOP variants, benchmark instances, and solution approaches.

2.1.1 Solution Approaches

A wide range of solution methods have been developed for the OP and the TOP. Exact methods include branch-and-cut (B&C), branch-and-price (B&P), and branch-and-cut-and-price (B&C&P), which are among the most effective approaches for solving medium-sized instances. For example, Bianchessi et al. (2018) propose a compact formulation combined with a B&C algorithm, while Kobeaga et al. (2024) improve the effectiveness of B&C through enhanced separation procedures. Similarly, Archetti et al. (2024) develop strengthened formulations with valid inequalities and dedicated separation algorithms, highlighting the importance of formulation

quality in exact methods. Branch-and-price algorithms have also demonstrated excellent performance for specialized TOP variants involving geometric constraints and uncertain travel times (Sundar et al., 2022; Liu et al., 2025).

For large-scale instances, heuristic and metaheuristic methods are widely used. These include Adaptive Large Neighborhood Search (Hammami, 2024), GRASP-based methods (Yahiaoui et al., 2023), Tabu Search (Tang and Miller-Hooks, 2005), evolutionary algorithms (Hu et al., 2018), and hybrid methods that combine heuristic exploration with exact optimization components (Hammami, 2024).

More recently, Constraint Programming (CP) has emerged as a flexible and effective paradigm for routing and scheduling problems. CP is particularly well-suited to modeling complex combinatorial structures and temporal constraints. Modern CP solvers such as IBM ILOG CP Optimizer provide specialized global constraints and interval variables that facilitate the modeling of routing and scheduling problems with complex temporal relationships (Laborie et al., 2018). In the context of TOP variants, CP-based approaches have shown strong performance. For instance, Gedik et al. (2017) and Kirac et al. (2023) demonstrate that CP can match or improve best known solutions on benchmark instances. Hybrid approaches integrating CP with metaheuristics or decomposition strategies (Hu et al., 2018; Pham et al., 2024) further enhance solution quality and computational performance.

Beyond TOP, CP has proven effective in complex routing problems involving synchronization and coordination (Montemanni and Dell'Amico, 2023; Rabbouch et al., 2019). In workforce routing, CP-based decomposition approaches (Bourreau et al., 2022) have demonstrated superior scalability compared to MILP formulations, particularly in large instances with multiple temporal dependencies.

2.2 Workload Equity, Heterogeneity, and Compatibility in Routing Problems

Workload equity has become an increasingly important consideration in vehicle routing, motivated by the need to distribute operational workloads more evenly among drivers while maintaining efficient routing plans. Surveys such as those by Matl et al. (2018) and Matl et al. (2019) analyze different modeling approaches and equity metrics, showing that workload equity can be incorporated as an objective, a constraint, or within multi-objective frameworks. Their analyses further demonstrate that the mathematical properties of the selected equity measure directly affect the quality of the resulting routing plans, with non-monotonic metrics potentially producing workload-inconsistent or artificially balanced solutions. They also high-

light that both the choice of the equity function and, more importantly, the workload resource being balanced (route duration, travel time, or load) significantly influence the resulting solutions.

From a broader Operations Research perspective, Karsu and Morton (2015) classify equity-oriented optimization approaches into Rawlsian criteria, lexicographic formulations, and inequality indices such as the range, variance, mean deviation, and Gini coefficient, providing a theoretical foundation for modeling workload equity. Building on these findings, Lehuédé et al. (2020) show that lexicographic minimax objectives overcome several shortcomings of traditional min–max and range-based criteria by avoiding artificially balanced routing solutions. More recently, Nekooghadirli et al. (2026) demonstrate that workload equity can be achieved with virtually no increase in routing cost when routes are reassigned over sufficiently long planning horizons. Despite these advances, the workload resource considered in equitable routing problems is almost exclusively operational, such as route distance or duration, rather than the distribution of rewards or profits among agents.

Heterogeneity and compatibility constraints have been more extensively studied in Workforce Scheduling and Routing Problems (WSRP). Surveys such as those by Castillo-Salazar et al. (2016) and Paraskevopoulos et al. (2017) identify key features of these problems, including multi-skilled workers, task requirements, time windows, and routing decisions. Solution approaches include mathematical programming (Garaix et al., 2018), hybrid metaheuristics (Gu et al., 2022), and integrated models that consider team formation and multi-skilled workers (Anoshkina and Meisel, 2019; Wang et al., 2021; Pekel, 2021). Although most WSRP models emphasize operational efficiency, workload equity has also been explicitly considered in personnel planning. In particular, Prot et al. (2015) formulate the Shift Design and Personnel Task Scheduling Problem with equity as the primary optimization objective, while Anoshkina and Meisel (2019) jointly consider team composition, routing, service quality, cost, and workload equity when assigning multi-skilled workers. Nevertheless, these problems generally seek to maximize task coverage while minimizing operational costs, rather than addressing selective service and profit maximization.

Similarly, Home Health Care (HHC) routing problems incorporate compatibility constraints, time windows, and uncertainty. Exact approaches such as branch-and-price (Cortés et al., 2014) have been successfully applied to solve these problems. Some variants allow unserved tasks and include reward structures (Rasmussen et al., 2012; Gobbi et al., 2019), while others explicitly incorporate workload equity through multi-objective formulations (Errarhout et al., 2016; Decerle et al., 2017). Nevertheless, the integration of selective routing decisions with equity-oriented objectives remains limited.

Within the TOP literature, only a few studies explicitly address workload equity, and existing approaches typically treat balancing as a secondary objective (Mirzaei et al., 2017; Estrada-Moreno et al., 2020), while the primary objective remains the maximization of collected profit. Moreover, although recent TOP variants have introduced heterogeneous agents (Chakraborty and Agarwal, 2021) and compatibility constraints motivated by practical applications (Yahiaoui et al., 2023; Guastalla et al., 2025), these studies continue to focus on routing efficiency and profit collection. In contrast to the equitable VRP literature, where equity is generally defined over operational workload measures, the distribution of collected profits among agents has received little attention as an equity criterion.

In summary, workload equity is now a well-established objective in routing and scheduling problems, and substantial progress has been made in understanding the implications of different equity measures and workload resources. However, the existing literature largely focuses on balancing operational resources in homogeneous routing settings. The joint consideration of workload equity, heterogeneous agents, compatibility constraints, and selective customer service remains largely unexplored, particularly when equity is defined over the distribution of collected profits in Team Orienteering Problems.

2.3 Research Gap and Contributions

The literature review reveals that, while the TOP and its variants have been extensively studied, several important aspects remain underexplored.

First, most existing formulations assume homogeneous agents, neglecting the heterogeneity that naturally arises in real-world applications.

Second, although some studies consider extensions such as time windows and resource constraints, the explicit incorporation of workload equity has received limited attention. In particular, balancing the collected score among agents as a primary objective has not been widely addressed in the literature.

Third, compatibility constraints between agents and customers, commonly arising in workforce applications, are rarely incorporated into TOP formulations.

Motivated by these gaps, this thesis makes the following contributions:

- It studies the Team Orienteering Problem with Nonidentical agents and Balanced score, a variant of the TOP that simultaneously incorporates heterogeneous agents, compatibility constraints between agents and customers, and a

workload equity objective based on balancing the collected score among agents.

- It strengthens previously developed mixed-integer linear programming formulations for the TOPNB, including valid inequalities that improve model performance.
- It develops a tailored branch-and-cut algorithm to efficiently solve the proposed formulations.
- It introduces a constraint programming model as an alternative solution approach and evaluates its performance.
- It provides a comprehensive computational comparison of all proposed methods and validates their applicability through a real-world case study.

Chapter 3

Team Orienteering Problem with Nonidentical Agents and Balanced Score

This chapter formally defines the problem addressed in this study, namely, the Team Orienteering Problem with Nonidentical Agents and Balanced Score (TOPNB). The problem consists of determining which service orders should be executed by the available crews and establishing the sequence in which these orders must be performed. The objective is to select a subset of orders and their corresponding routes to balance the total collected score among crews while satisfying all operational and structural constraints. This formal description specifies the sets, parameters, and decision variables involved, along with the mathematical structure that defines the problem.

The mathematical formulations presented in this chapter are based on those originally proposed in Sánchez-Yepez (2019). To maintain consistency with the notation adopted throughout this dissertation, the formulations are presented using a slightly revised notation while preserving their mathematical equivalence.

3.1 Mathematical Modeling

Let $G = (N, E)$ be a complete graph, where $N = \{o, 1, \ldots, n, d\}$ denotes the set of nodes, with o and d representing the initial and final depot, respectively, and $\{1, \ldots, n\}$ denoting the set of customers (also referred to as service orders). The set of available crews (vehicles) is denoted by K, where $|K| = m$. Each node has an associated nonnegative score (prize) p_i and a nonnegative service time h_i. All routes

start at node o and finish at node d, with both nodes having scores and service times of zero. The nonnegative travel time t_{ij} required to move from node i directly to node j is known, and the maximum total route duration allowed is $T_{\max}$. The compatibility between a service order and a crew is defined by the binary parameter c_{ik}. The goal is to determine a route for each crew such that the collected scores are balanced among them. A detailed description of sets and parameters is shown in Table 3.1.

Table 3.1: MILP notation.

Sets	
$\{o, d\}$	initial and final depots
$N \setminus \{o, d\}$	set of customers (service orders)
K	set of available crews (vehicles), where $\|K\| = m$
Parameters	
p_i	score (prize) associated with order $i \in N$
h_i	service time of order $i \in N$
t_{ij}	travel time from node i directly to node j, $i, j \in N$
$T_{\max}$	maximum route duration
c_{ik}	$\begin{cases} 1, \text{ if node } i \text{ is compatible with crew } k, \\ 0, \text{ otherwise, } i \in N, k \in K \end{cases}$
Decision variables	
x_{ijk}	$\begin{cases} 1, \text{ if crew } k \text{ travels directly from node } i \text{ to node } j \\ 0, \text{ otherwise, } i, j \in N, k \in K \end{cases}$
y_{ik}	$\begin{cases} 1, \text{ if node } i \text{ is visited by crew } k, \\ 0, \text{ otherwise, } i \in N, k \in K \end{cases}$

For the proposed formulation, the following assumptions are made:

- All problem data are known in advance and remain fixed during the daily planning horizon.
- Each crew performs exactly one route; therefore, the solution consists of exactly m routes.
- Every route originates at depot o and terminates at depot d.
- Each customer can be visited at most once and assigned to only one compatible crew.
- Some customers may remain unvisited.
- Once a customer is visited, its entire associated score p_i is collected and its complete service time h_i is incurred. Partial service is not allowed.

- Travel and service operations are assumed to be uninterrupted; therefore, waiting times, breaks, and other sources of delay are not explicitly considered.

There is no universally accepted metric to measure balance, and different equity measures capture different notions of balance Matl et al. (2018). In the motivating telecommunications application, work crews are compensated based on the scores collected from the service orders they complete. Therefore, balancing the collected scores provides a meaningful criterion for distributing service orders among the available crews. Accordingly, two complementary objective functions are considered.

The first objective maximizes the minimum collected score among the crews,

$$\text{Maximize} \quad \min_{k \in K} \left\{ \sum_{i \in N} p_i y_{ik} \right\}, \tag{3.1}$$

so that the lowest collected score is as large as possible.

The second objective minimizes the maximum absolute deviation from the average score,

$$\text{Minimize} \quad \max_{k \in K} \left\{ \quad \left| \mu - \sum_{i \in N} p_i y_{ik} \right| \quad \right\}, \tag{3.2}$$

where $\mu = \frac{1}{m} \sum_{i \in N} p_i$. This objective limits the largest deviation from the average score by keeping the collected scores as close as possible to the ideal equal allocation. Together, these objectives evaluate balance from two different perspectives while remaining easy to interpret and linearize.

Using the notation introduced in Table 3.1 and the assumptions stated above, the mixed-integer linear programming formulation associated with the first objective is presented below:

$$\text{Maximize} \quad S_{\min} \tag{3.3}$$

subject to:

$$S_{\min} \leq \sum_{i \in N} p_i y_{ik}, \qquad k \in K \tag{3.4}$$

$$\sum_{k \in K} \sum_{j \in N \setminus \{o\}} x_{ojk} = \sum_{k \in K} \sum_{i \in N \setminus \{d\}} x_{idk} = m \tag{3.5}$$

$$\sum_{i \in N \setminus \{d\}} x_{ilk} = \sum_{j \in N \setminus \{o\}} x_{ljk} = y_{lk}, \qquad k \in K,\ l \in N \setminus \{o, d\} \tag{3.6}$$

$$\sum_{k \in K} y_{ik} \leq 1, \qquad i \in N \setminus \{o, d\} \tag{3.7}$$

$$y_{ik} \leq c_{ik}, \qquad k \in K,\ i \in N \setminus \{o,d\} \quad (3.8)$$

$$\sum_{i \in N \setminus \{o,d\}} h_i y_{ik} + \sum_{i \in N \setminus \{d\}} \sum_{j \in N \setminus \{o\}} t_{ij} x_{ijk} \leq T_{\max}, \qquad k \in K \quad (3.9)$$

$$2 \leq u_{ik} \leq 2 + \sum_{j \in N \setminus \{o,d\}} y_{jk}, \qquad k \in K,\ i \in N \setminus \{o\} \quad (3.10)$$

$$u_{ik} - u_{jk} + 1 \leq (n+2)(1 - x_{ijk}), \qquad k \in K,\ i \in N \setminus \{o\},\ j \in N \setminus \{o,d\} \quad (3.11)$$

$$x_{ijk} \in \{0,1\}, \qquad i,j \in N,\ k \in K \quad (3.12)$$

$$y_{ik} \in \{0,1\}, \qquad i \in N,\ k \in K \quad (3.13)$$

$$u_{ik} \in \{2,\ldots,n+2\}, \qquad i \in N \setminus \{o\},\ k \in K. \quad (3.14)$$

The objective function (3.3), along with Constraints (3.4), balances the score among the crews. Constraint (3.5) ensures that m routes start and end at the respective depots. Flow conservation is imposed by Constraints (3.6), and Constraints (3.7) ensure that each customer is visited at most by one crew. The compatibility relationship is guaranteed by Constraints (3.8). Constraints (3.9) verify that the duration of each route does not exceed the established limit. The subtour elimination constraints are given by Constraints (3.10) and (3.11). Finally, Constraints (3.12) to (3.14) define the domains of the decision variables.

The formulation for the second objective function is obtained by replacing the objective function and introducing the corresponding linearization constraints, yielding the following formulation:

$$\text{Minimize} \quad D_{\max}, \quad (3.15)$$

subject to:

$$\mu - \sum_{i \in N} p_i y_{ik} \leq D_{\max}, \qquad k \in K \quad (3.16)$$

$$\sum_{i \in N} p_i y_{ik} - \mu \leq D_{\max}, \qquad k \in K \quad (3.17)$$

$$D_{\max} \geq 0, \quad (3.18)$$

$$(3.5) - (3.14).$$

3.2 Valid Inequalities for the TOPNB

The first approach of this study consists of strengthening previously proposed formulations by modifying or incorporating additional sets of constraints. This section

explains the proposed valid inequalities, which lead to different possible formulations, along with their validation and computational performance analysis.

One of the principal sets of constraints that make the problem complex is the subtour elimination constraints. Constraints (3.10) and (3.11) are adaptations of the Miller–Tucker–Zemlin (MTZ) subtour elimination constraints for the TOP (Vansteenwegen et al., 2011). The auxiliary variable u_{ik} defines the position of node i in the route of crew k.

Excluding the origin node, Constraint (3.10) defines the position range of the nodes in each crew route. Constraint (3.11) imposes the required ordering between consecutive nodes: if crew k travels from node i to node j, then u_{jk} must be at least one unit greater than u_{ik}, preventing the formation of subtours. Following the classical MTZ formulations, the big-M coefficient is defined as the total number of nodes in the graph, in this case $(n + 2)$. However, this general bound does not consider the compatibility restrictions between crews and customers. Therefore, it may provide a loose bound for crews with a limited set of compatible customers.

The compatibility constraints provide additional information that can be used to obtain a tighter big-M coefficient. Let

$$C_k = \{\, i \in N \setminus \{o, d\} : c_{ik} = 1 \,\}, \quad k \in K,$$

denote the set of customers compatible with crew k. By Constraint (3.8), only customers in C_k can be assigned to crew k, and therefore $\sum_{i \in N \setminus \{o,d\}} y_{ik} \leq |C_k|$, $k \in K$. Hence, Constraint (3.11) can be replaced by the tighter formulation

$$u_{ik} - u_{jk} + 1 \leq (|C_k| + 1)(1 - x_{ijk}).$$

From Constraint (3.10), it follows that

$$2 \leq u_{ik} \leq |C_k| + 2.$$

Therefore, for any $i \in N \setminus \{o\}, j \in N \setminus \{o, d\}$,

$$u_{ik} - u_{jk} + 1 \leq (|C_k| + 2) - 2 + 1 = |C_k| + 1.$$

Thus, when $x_{ijk} = 0$, the modified constraint is always satisfied, and no feasible solution is removed. Since $|C_k| \leq n$ for every $k \in K$, the proposed coefficient satisfies

$$|C_k| + 1 \leq n + 1 < n + 2.$$

Consequently, the proposed big-M coefficient is tighter than the original one and may improve the linear relaxation of the model.

Finally, an additional bound can be derived for the decision variables x_{ijk}. Recall that this variable determines the visit sequence and, implicitly, the number of arcs in each route. On the one hand, the maximum number of customers that a crew can visit equals the number of customers compatible with it. Hence, the maximum number of arcs in a route is $|C_k| + 1$, where the additional unit accounts for the return arc to the depot.

On the other hand, route duration constraints also impose a bound. In the most favorable case, when both service times and travel times attain their minimum values, the maximum possible number of arcs in a route is bounded by

$$\left\lfloor \frac{T_{\max}}{\min_{i \in C_k}\{h_i\} + \min_{i,j \in C_k}\{t_{ij}\}} \right\rfloor + 1. \tag{3.19}$$

Consequently, the proposed inequalities are valid for the problem:

1. Set1: **Modify** the set of constraints (3.11) by replacing it with (3.20):

$$u_{ik} - u_{jk} + 1 \leq (|C_k| + 1)(1 - x_{ijk}), \quad k \in K,\ i, j \in N \setminus \{o\}. \tag{3.20}$$

2. Set2: **Include** the set of constraints (3.21)

$$\sum_{i \in N \setminus \{o,d\}} y_{ik} \leq |C_k|, \quad k \in K. \tag{3.21}$$

3. Set3: **Include** the set of constraints (3.22)

$$\sum_{i \in N \setminus \{d\}} \sum_{j \in N \setminus \{o\}} x_{ijk} \leq \min \left\{ |C_k|, \left\lfloor \frac{T_{\max}}{\min_{i \in C_k}\{h_i\} + \min_{i,j \in C_k}\{t_{ij}\}} \right\rfloor \right\} + 1, \quad k \in K. \tag{3.22}$$

Different combinations of these inequalities yield alternative formulations of the problem. A computational study was conducted to identify the formulation achieving the best performance in terms of solution quality and execution time.

3.2.1 EXPERIMENTAL ANALYSIS

A total of seven models are analyzed. Their differences lie in the inclusion of distinct sets of valid inequalities, Table 3.2 indicates which sets are incorporated in each

formulation. Note that Set1 corresponds to a modified version of Constraints (3.11), therefore, whenever Set1 is included, Constraints (3.11) are replaced rather than added to the formulation.

Table 3.2: Tested models and valid inequalities included.

Model	**Set1**	**Set2**	**Set3**
A	√		
B		√	
C			√
D	√	√	
E	√		√
F		√	√
G	√	√	√

The experiments are based on the dataset introduced by Sánchez-Yepez et al. (2023), which adapts the classical benchmark instances of Chao et al. (1996) for the TOP. The instances are organized into seven classes according to the number of nodes (from 21 to 102, including depots), with variations in the number of vehicles (between two and four) and the maximum route duration.

Instances for which no customer can be visited within the allowed time limit are excluded from the analysis. These cases arise because the benchmark includes several route duration budgets, some of which are too restrictive to allow any customer to be visited while still ensuring a feasible return to the end depot. Consequently, the optimal objective value is zero, as the only feasible solution is to travel directly from the start depot to the end depot. After filtering out these trivial instances, the benchmark comprises 353 test instances. Table 3.3 reports the number of tested instances for each class and vehicle configuration. Note that contrary to the naming convention adopted in the literature, the instance classes in this work are labeled in ascending order of the number of nodes, from the smallest class to the largest. This convention is used consistently throughout the remainder of the document.

To evaluate model performance, only a subset of these instances was selected. The selection of a reduced subset for this preliminary computational study was primarily motivated by the combinatorial growth of the experimental design. Since seven candidate formulations were evaluated under two objective functions, testing the complete benchmark of 353 instances would have required 4,942 individual runs, each with a time limit of 7,200 seconds. Given the computational resources available, this was not feasible for an exploratory stage whose purpose is to identify which valid inequalities are worth incorporating into the final model.

To maintain a representative sample while controlling for problem size, three

Table 3.3: Distribution of instances by class and number of vehicles.

Class	Nodes	2 veh.	3 veh.	4 veh.	Total
I	21	11	11	11	33
II	32	17	16	15	48
III	33	20	20	20	60
IV	64	11	8	5	24
V	66	25	24	25	74
VI	100	20	19	17	56
VII	102	20	19	19	58
Total					**353**

instances were randomly selected from each class, one for each available fleet size. This sampling strategy ensures that all seven graph sizes and every fleet configuration represented in the benchmark are included in the study, yielding a total of 21 test instances (see Table 3.4). Since all instances within a class share the same underlying graph, differ only in the maximum travel time and the number of available vehicles, and have identical compatibility assignments for each fleet size, this subset allows a preliminary assessment of the relative impact of the proposed valid inequalities.

The formulation selected from this preliminary analysis is subsequently evaluated on the complete benchmark of 353 instances, where its performance is compared against the baseline formulation. For comparability with the baseline formulation, all experiments were conducted under identical computational conditions: a 2.10 GHz Intel Xeon(R) CPU E5-2620 v2 under the Ubuntu 20.04 operating system, with 64 GB of RAM, using CPLEX 12.9 as the branch-and-bound solver. The maximum computation time for solving each instance was set to 7200 seconds using a single thread and the default parameter setting.

Table 3.4: Tested instances for the valid inequalities analysis.

Instance	Class	n	v	Instance	Class	n	v	Instance	Class	n	v
p2.2.k	I	21	2	p3.3.m	III	33	3	p5.4.o	V	66	4
p2.3.h	I	21	3	p3.4.n	III	33	4	p4.2.r	VI	100	2
p2.4.g	I	21	4	p6.2.e	IV	64	2	p4.3.r	VI	100	3
p1.2.d	II	32	2	p6.3.g	IV	64	3	p4.4.q	VI	100	4
p1.3.e	II	32	3	p6.4.l	IV	64	4	p7.2.j	VII	102	2
p1.4.j	II	32	4	p5.2.s	V	66	2	p7.3.f	VII	102	3
p3.2.f	III	33	2	p5.3.j	V	66	3	p7.4.g	VII	102	4

Model performance is evaluated by comparing both objective values and CPU times across the test set. Objective values are compared against those obtained by

the baseline formulation, and the percentage deviation is computed according to Equation (3.23):

$$\text{Dev} = 100 \times \frac{Z^* - Z_x}{Z^*}, \tag{3.23}$$

where Z^* denotes the objective value obtained by the baseline model and Z_x corresponds to that produced by the tested formulation. To ensure consistent interpretation of Dev across objectives, when analyzing the Min–Max objective, the value is multiplied by -1. Under this convention, a negative deviation indicates that the tested model improves upon the baseline solution.

Additionally, the relative solution time with respect to the baseline model is evaluated. For each class of instances, the total CPU time required to solve all instances is first aggregated for the baseline model and then for each tested formulation. Using these aggregated values, a proportional measure is computed to quantify the relative computational effort of the tested formulation with respect to the baseline.

Formally, let T_c^{base} and T_c^{new} denote the total solution times for class c under the baseline and the proposed model, respectively. The relative time is computed according to Equation (3.24):

$$\text{RelTime}_c = \frac{T_c^{\text{new}}}{T_c^{\text{base}}} \times 100 \tag{3.24}$$

This metric expresses the computational time of the tested formulation as a percentage of the baseline, allowing for a direct comparison of efficiency across classes. Table 3.5 summarizes the results, reporting the minimum, average, and maximum deviations for each formulation together with the relative computational time for both objective functions.

Overall, model B provides the best performance for the Max–Min objective: its deviation from the baseline is minimal, and solutions are obtained in comparable time. The remaining models exhibit larger maximum deviations, though they yield similar results on average. For the Min-Max objective, model F has the best average deviation, one of the smallest maximum deviations, and is one of the fastest models. Model C achieves the second-best performance for this objective.

To complement the information reported in Table 3.5, we compare the objective values obtained by each formulation and compute the percentage of instances for which the objective value is worse, equal, or better than that of the baseline model. These percentages are illustrated in Figure 3.1.

Analyzing the Max–Min objective, we can verify that model G did not achieve a better objective function value in any instance compared to the baseline model

Table 3.5: Deviations and relative CPU times.

Model	Max-Min				Min-Max			
	Dev (%)			Time (%)	Dev (%)			Time (%)
	min	avg	max		min	avg	max	
A	-13.73	1.01	20.93	103.51	-17.39	0.15	8.57	95.98
B	-0.54	0.33	5.39	99.98	-0.61	0.19	3.45	99.95
C	-3.92	2.44	20.93	107.33	-7.83	-0.82	0.30	93.13
D	-3.18	2.13	20.93	103.52	-17.39	1.58	31.25	95.99
E	-3.18	4.22	20.93	112.05	-17.39	-0.35	29.31	95.87
F	-2.48	2.39	15.32	107.37	-7.83	-0.96	3.45	93.15
G	0.00	4.73	21.62	112.05	-17.39	0.72	31.25	95.87

solutions. Thus, the minimal deviation is equal to zero. We can also see that, excluding model B, the remaining models have a similar percentage of instances with equal objective values; thus, their average deviations are similar. With the Min–Max objective, we can see that the models with the best average results have higher percentages of instances with better and equal objective values than the original formulation.

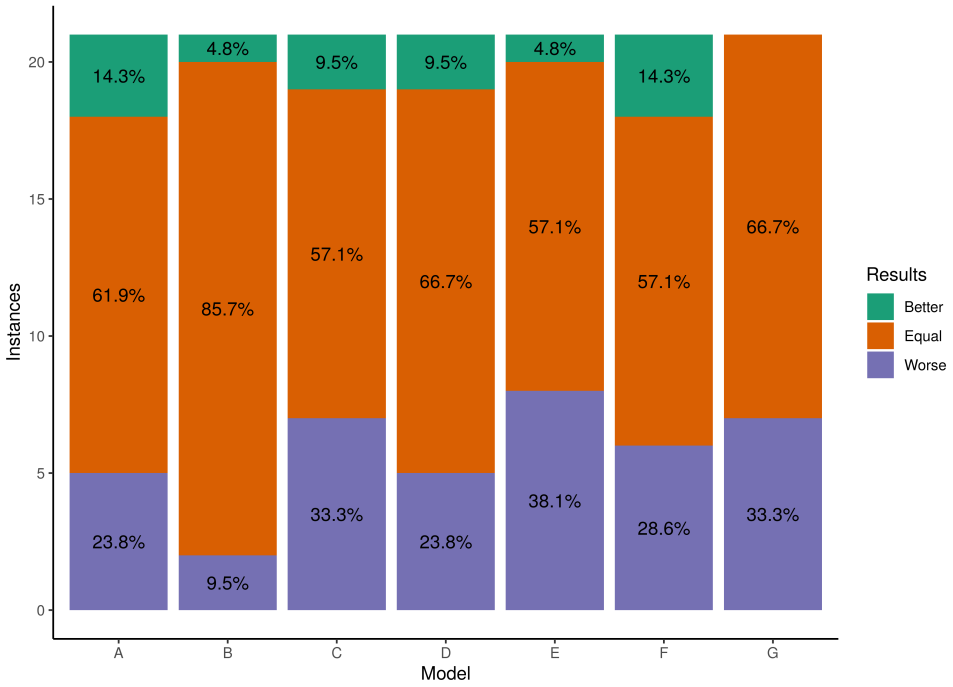
We can also conclude that model B shows consistent results for both objective functions, being the one whose results are most closely aligned with the baseline model. Additionally, Model F exhibits the highest percentage of instances that improve upon the baseline model's objective values.

Finally, we analyze the quality of the solutions obtained by each model by comparing them against the reference values available for each instance. In particular, three categories are considered: (i) known optimal solutions (obtained by the baseline model), (ii) new optimal solutions, and (iii) new best known solutions.

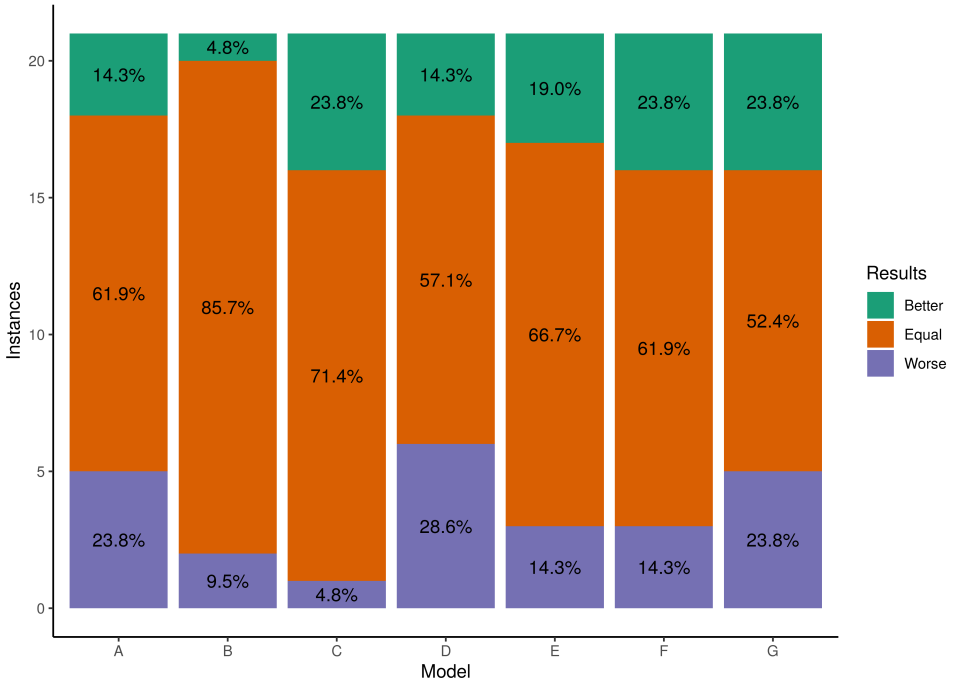
For each instance, the maximum objective value across all tested models is identified. A solution is classified as a *new best known* only if this value strictly exceeds the reference value obtained from the baseline model. Among all models, only those attaining this maximum value are considered as achieving the new best known result.

Table 3.6 summarizes the number of known optimal solutions (KOpt), the new best known solutions (BK), and new optimal (NOpt) solutions that each model achieves. It can be seen that model F achieved good performance on both objective functions.

Although model F is not the fastest, it has shown one of the best average



(a) Max–Min objective function



(b) Min-Max objective function

Figure 3.1: Percentage of instances with an objective value better, equal, or worse than the objective value obtained with the baseline model.

Table 3.6: Known optimal, new best known, and new optimal solutions.

Model	Max–Min			Min-Max		
	KOpt	**BK**	**NOpt**	**KOpt**	**BK**	**NOpt**
A	11/12	3	0	10/10	1	0
B	**12/12**	0	0	10/10	0	0
C	11/12	1	0	10/10	1	**1**
D	11/12	2	0	10/10	2	0
E	10/12	1	0	10/10	3	0
F	11/12	2	0	10/10	2	**1**
G	10/12	0	0	10/10	4	0

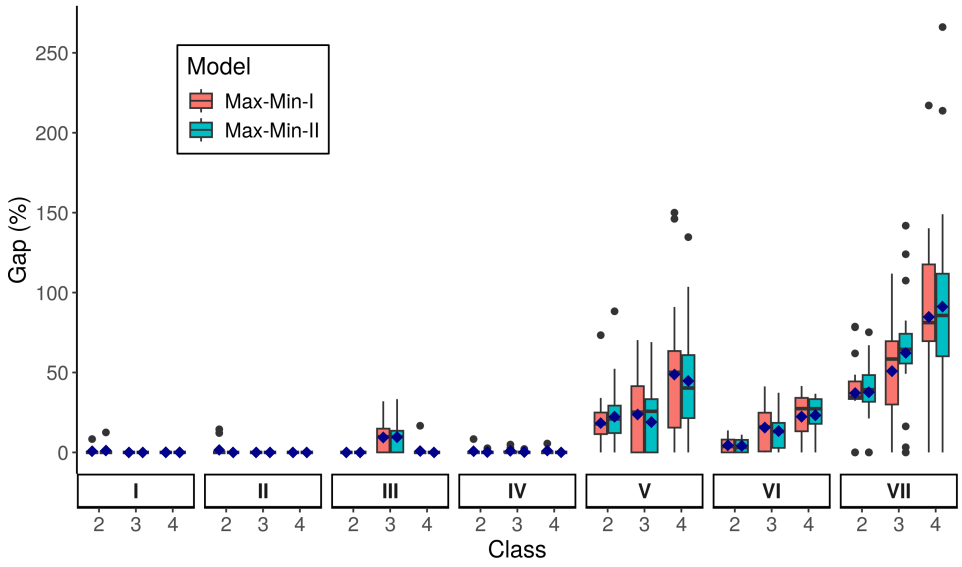
deviations. It maintains the highest proportion of solutions in which the objective value exceeds the values of the baseline model. Additionally, it demonstrated that for both objective functions, it finds nearly all known optimal solutions and even attains a new optimum for the Min–Max model. These results indicate that model F provides the most robust overall performance across objective functions. Therefore, it is selected as the preferred formulation.

3.2.2 General Results

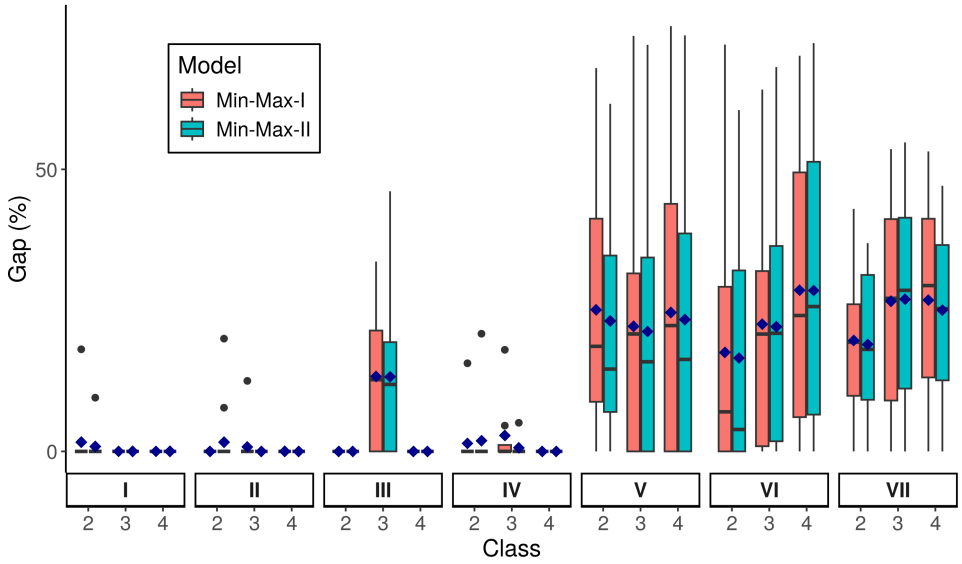
After selecting the preferred formulation, we tested it on the complete benchmark set (353 instances) and conducted a new analysis. The aim was to compare its performance with that of the baseline formulation (without valid inequalities) to determine which provides the best overall performance. Again, using a single thread, we set the maximum computation time to 7200 seconds. Henceforth, the baseline models are called Max–Min-I and Min–Max-I, respectively. Similarly, the formulations with valid inequalities (derived from model F) will be called Max–Min-II or Min–Max-II, as the case may be.

Figure 3.2 displays the distribution of the Gap percentages reported by CPLEX for each class and number of vehicles. We highlight the average Gap with a blue diamond. Note that, regardless of the formulation, the solver reported feasible solutions for all the tested instances.

Table 3.7 shows the percentage of CPU time required compared to the baseline model and the number of optimal solutions obtained for each objective function. We highlight in bold text the cases for which the solver reports optimality in the whole instance set. These relative times are computed analogously to Equation (3.24).



(a) Max–Min objective function



(b) Min-Max objective function

Figure 3.2: Gap reported by the solver.

Table 3.7: Relative CPU times and optimal solutions.

Class	v	**Max–Min-II**		**Min-Max-II**	
		Time (%)	**Opt**	**Time (%)**	**Opt**
I	2	89.33	10/11	111.93	10/11
	3	14.39	**11/11**	23.29	**11/11**
	4	86.10	**11/11**	114.71	**11/11**
II	2	54.90	**17/17**	223.48	15/17
	3	87.21	**16/16**	72.77	**16/16**
	4	192.31	**15/15**	116.46	**15/15**
III	2	91.30	**20/20**	188.73	**20/20**
	3	93.71	8/20	99.73	6/20
	4	53.38	**20/20**	76.04	**20/20**
IV	2	81.27	10/11	96.85	10/11
	3	61.84	7/8	71.92	7/8
	4	20.03	**5/5**	156.60	**5/5**
V	2	100.64	5/25	102.84	5/25
	3	110.95	12/25	98.26	9/25
	4	104.59	5/24	110.05	7/24
VI	2	97.12	8/20	97.49	10/20
	3	99.51	5/19	101.27	5/19
	4	100.57	2/17	100.30	2/17
VII	2	100.14	3/20	99.76	3/20
	3	100.00	2/19	99.93	2/19
	4	99.96	3/19	100.02	3/19
	Total		195/353		192/353

As expected, the optimality Gap and CPU time increase as the number of nodes in the instances increases (see instances from classes V–VII), and the best proportion of optimal solutions (more than 76%) is reached for instances from classes I to IV. We can also see that, in general, the model with valid inequalities is faster than the formulations without them.

To determine whether including valid inequalities accelerates solver convergence, we analyzed CPU times only for instances for which both models reach the optimal solution. Table 3.8 shows the percentage of time required for the formulations with valid inequalities compared to the baseline model. It can be seen that the Max–Min-II formulation improves the CPU times in 4 out of 7 classes. In the general case, the percentage deviations between the Max–Min-I and Max–Min-II models are 9.62% and 1.43% for the Min–Max-I and Min–Max-II models (in favor of the baseline models).

Table 3.8: Relative CPU times of common optimal solutions.

Class	**CPU Time (%)**	
	Max–Min-II	**Min–Max-II**
I	40.12	73.23
II	137.35	113.92
III	59.58	117.46
IV	83.31	106.05
V	261.09	213.84
VI	88.15	108.32
VII	109.33	76.62

Similarly, we consider the instances for which the models with valid inequalities and without them reached a feasible solution. We calculate the percentage deviation between the objective values, referencing the values of the original model (Max–Min-I or Min–Max-I, as the case may be). Thus, a negative result indicates that the model with valid inequalities has a better objective value than the model without them. The results are shown in Figure 3.3, where the average deviations per class are highlighted with a blue diamond and are also explicitly reported as numerical values.

We observe that, on average, the Min–Max-II formulation improves results over the Min–Max-I formulation in classes V, VI, and VII. Meanwhile, the Max–Min-I formulation improves results only for class VI. Nevertheless, the average deviations do not exceed 4%. Overall, the averages are 0.69% and -0.38% for the Max–Min and Min-Max objective functions, respectively. This shows us that the results between the formulations are quite similar.

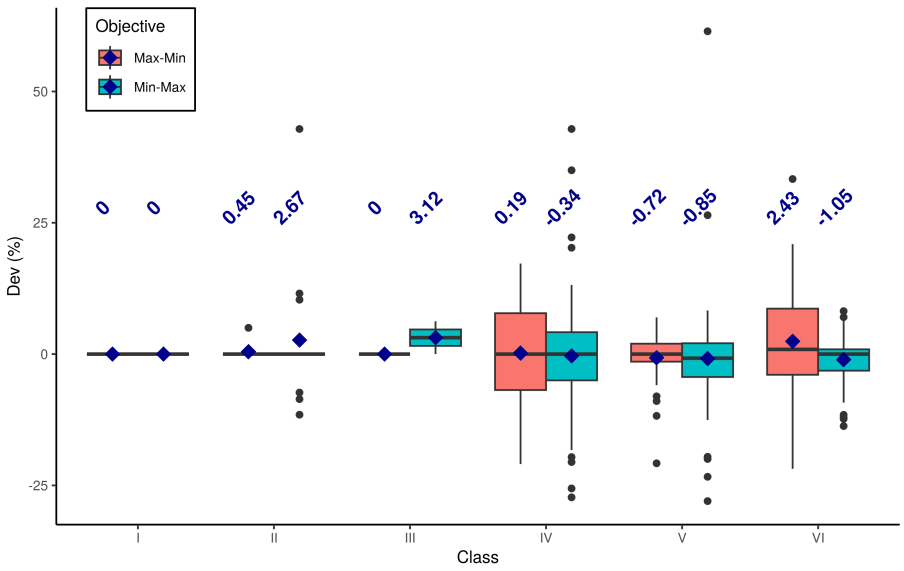


Figure 3.3: Percentage deviation for common feasible solutions.

While the average deviations indicate that the Max–Min-I formulation is better than the Max–Min-II formulation, the difference is minimal. Moreover, the Max–Min-II formulation increases the number of optimal solutions found by almost 2%.

Despite the inclusion of valid inequalities, the solver finds feasible solutions but often fails to prove optimality within the imposed time limit. The presence of a three-index decision variable, together with the structure of the subtour elimination constraints, makes the formulation computationally demanding. Consequently, a reformulation based on two-index decision variables is adopted. For the remainder of this thesis, only the Max-Min objective function is considered.

3.3 TOPNB Reformulation

The reformulation is built upon an undirected graph representation. Several parameters from the three-index formulation are preserved, while additional parameters are newly introduced. In particular, let

$$I_k = \{\, i \in N \setminus \{o, d\} : c_{ik} = 0 \,\}, \quad k \in K,$$

denote the set of customers that cannot be served by crew k. The notation used throughout the reformulation is summarized in Table 3.9. The goal of this reformulation is to reduce model size while preserving solution quality and model validity.

Table 3.9: Reformulation notation.

Sets	
$\{o, d\}$	initial and final depots
$N \setminus \{o, d\}$	set of customers (service orders)
K	set of available crews
I_k	set of customers incompatible with crew $k \in K$
Parameters	
p_i	score (prize) associated with order $i \in N$
h_i	service time of order $i \in N$
t_e	travel time associated with edge $e \in E$
$T_{\max}$	maximum route duration
Decision variables	
x_{ek}	$\begin{cases} 1, \text{ if crew } k \text{ traverses edge } e \\ 0, \text{ otherwise, } k \in K,\ e \in E \end{cases}$
y_{ik}	$\begin{cases} 1, \text{ if customer } i \text{ is visited by crew } k \\ 0, \text{ otherwise, } i \in N,\ k \in K \end{cases}$
Auxiliary notation	
$E(S)$	set of edges with both endpoints in S, $S \subseteq N$
$\delta(S)$	set of edges with exactly one endpoint in S, $S \subseteq N$
$N(T)$	set of nodes incident to edges in $T \subseteq E$
$x_k(T)$	$\sum_{e \in T} x_{ek}$, for $T \subseteq E,\ k \in K$
$y_k(S)$	$\sum_{i \in S} y_{ik}$, for $S \subseteq N,\ k \in K$

Note: For simplicity, we write $N(e)$ for $e \in E$ instead of $N(\{e\})$.

Using the notation summarized in Table 3.9, the reformulated problem can be expressed as the following model defined on the undirected graph $G = (N, E)$.

$$\text{Maximize} \quad z = \min_{k \in K} S_k \tag{3.25}$$

subject to:

$$S_k = \sum_{i \in N} p_i y_{ik}, \qquad k \in K \tag{3.26}$$

$$x_k(\delta(o)) = x_k(\delta(d)) = 1, \qquad k \in K \tag{3.27}$$

$$x_k(\delta(i)) = 2y_{ik}, \qquad i \in N \setminus \{o, d\},\ k \in K \tag{3.28}$$

$$\sum_{k \in K} y_{ik} \leq 1, \qquad i \in N \setminus \{o, d\} \tag{3.29}$$

$$y_{ik} = 0, \qquad k \in K,\ i \in I_k \tag{3.30}$$

$$\sum_{e \in E} t_e x_{ek} + \sum_{i \in N} h_i y_{ik} \leq T_{\max}, \quad k \in K \tag{3.31}$$

$$x_k(E(S)) \leq |S| - 1, \qquad S \subset N,\ |S| \geq 2,\ k \in K \tag{3.32}$$

$$x_{ek} \in \{0, 1\}, \qquad e \in E,\ k \in K \tag{3.33}$$

$$y_{ik} \in \{0, 1\}, \qquad i \in N,\ k \in K. \tag{3.34}$$

The objective function (3.25) aims to balance the collected score among crews by maximizing the minimum score. This nonlinear objective is handled using a linearization analogous to that introduced for the previous formulation (objective (3.1)), with the corresponding adaptations to the present model. The score of each crew $k \in K$ is given by Equation (3.26), while connectivity is enforced by (3.28). Depot constraints (3.27) ensure each route starts at the origin depot and ends at the final depot. Constraints (3.29) enforce that each node $i \in N \setminus \{o, d\}$ can be visited at most once, and Constraints (3.30) prevent assignments that are considered incompatible. The maximum route duration for each crew is imposed by Constraints (3.31). Subtours are eliminated through the Generalized Subtour Elimination Constraints (GSEC) in (3.32). Finally, Constraints (3.33) and (3.34) specify the decision variable domains.

In addition, the following set of constraints is incorporated into the model:

$$x_k(E) = y_k(N) - 1, \quad k \in K. \tag{3.35}$$

These constraints are valid when fixing $y_{ok} = y_{dk} = 1$, meaning that both depots are always included in every route. Under this condition, the constraints establish a relationship between the number of selected edges and the number of visited nodes for each crew, which must correspond to the structure of a simple path. Consequently, it strengthens the formulation by reinforcing the consistency between routing and visit decisions.

3.3.1 Valid Inequalities

To strengthen the formulation and improve the quality of the linear relaxation, we incorporate several families of valid inequalities for the TOPNB. These inequalities do not eliminate any feasible integer solution, but help tighten the relaxation and guide the branch-and-cut algorithm toward high-quality solutions. In total, we consider five classes of inequalities, which are described below.

First, we consider *logical cuts* (Fischetti et al., 1998). Observe that if an edge $e \in E$ is used by crew k, then both of its incident nodes must necessarily be visited

by that crew. This simple logical implication yields the following inequalities:

$$x_{ek} \leq y_{ik}, \qquad e \in \delta(i),\ i \in N \setminus \{o, d\},\ k \in K. \tag{3.36}$$

The second family corresponds to *infeasible cover cuts* (Gendreau et al., 1998). Let $B \subset N$ be a set of nodes such that no feasible route can visit all nodes in B. This situation may occur, for example, when the total travel time required to visit all nodes in B exceeds the available time budget. In such cases, the following inequality is valid:

$$y_k(B) \leq |B| - 1, \qquad k \in K. \tag{3.37}$$

The validity of this inequality follows from the fact that if a feasible route for crew k cannot visit all nodes in B due to time budget limitations, then at least one node in S must remain unvisited. Therefore, the sum of the visit variables over B cannot exceed $|B| - 1$.

The third class corresponds to *infeasible path cover cuts* (Fischetti et al., 1998). Let $P = \{o, i_1, i_2, \ldots, i_m, d\}$ be a path such that

$$\sum_{i \in P} p_i \leq \bar{z},$$

where $\bar{z}$ is a valid lower bound on the objective value z. In this situation, traversing all edges of P would produce a route whose total score does not exceed the current lower bound and therefore cannot correspond to an optimal solution. This observation leads to the following inequality:

$$x_k(E(P)) \leq y_k(P) - 2, \qquad k \in K. \tag{3.38}$$

To establish validity, observe that every feasible route must include both the origin o and the destination d, implying $y_{ok} + y_{dk} = 2$. The inequality, therefore, prevents crew k from traversing the complete set of edges in $E(P)$ as a contiguous path when the associated score is insufficient.

Next, we consider *connectivity cuts*, originally proposed by Gendreau et al. (1998). These inequalities ensure that the set of visited nodes forms a connected route and prevent the formation of subtours disconnected from the main path. For $S \subset N$, the following inequalities are valid:

$$x_k(\delta(S)) \geq 2y_{ik}, \qquad k \in K,\ S \subset N,\ 2 \leq |S| \leq n + 1,\ i \in S,\ \{o, d\} \not\subset S \tag{3.39}$$

and

$$x_k(\delta(S)) \geq 2y_{ik}, \qquad k \in K,\ S \subset N,\ 2 \leq |S| \leq n + 1,\ i \notin S,\ \{o, d\} \subset S. \tag{3.40}$$

These inequalities are also known as subtour elimination constraints (SEC), as they ensure that any subset of visited nodes remains properly connected to the rest of the route.

Finally, we consider the *cycle cover cuts* introduced by Fischetti et al. (1998). For any subset $D \subset N$ such that $\{o, d\} \subset D$, the following inequality is valid:

$$x_k(E(D)) \leq y_k(D) - 1, \qquad k \in K. \tag{3.41}$$

These inequalities prevent the formation of cycles entirely contained within D. Indeed, since $\{o, d\} \subset D$, we have $y_k(D) - 1 \geq 1$, ensuring that the number of selected edges inside D cannot form a cycle without violating the constraint.

Additional details regarding the separation and implementation of these inequalities within the branch-and-cut framework are presented in Chapter 4.

Chapter 4

Branch and Cut for the TOPNB

In this chapter, we describe the main components of the proposed Branch-and-Cut (B&C) algorithm used to solve the TOPNB. The algorithm builds upon the undirected formulation presented in the previous chapter and incorporates several families of valid inequalities, which were introduced in Section 3.3.1.

4.1 Framework

The solution approach consists of three main components: a preprocessing phase, separation procedures for the valid inequalities, and a primal heuristic designed to obtain feasible solutions during the search process. The overall workflow of the proposed B&C algorithm is illustrated in Figure 4.1.

In the preprocessing phase, infeasible edges are removed, and nodes that cannot be reached within the available time budget are discarded from the graph. This step reduces the instance size before the optimization process begins.

After preprocessing, the linear programming (LP) relaxation of the problem is solved. The primal heuristic is then applied to the LP solution to construct feasible solutions that may improve the incumbent solution. Next, the separation algorithms are applied to identify violated inequalities, which are dynamically added to the formulation in order to strengthen the LP relaxation. If cuts are added, the strengthened LP relaxation is solved again, and the process is repeated until no additional cuts are generated. Once no additional cuts are generated, the algorithm proceeds with the branch-and-bound search (B&B). Throughout this phase, lazy constraints are used to check the feasibility of integer solutions before they are accepted as incumbents.

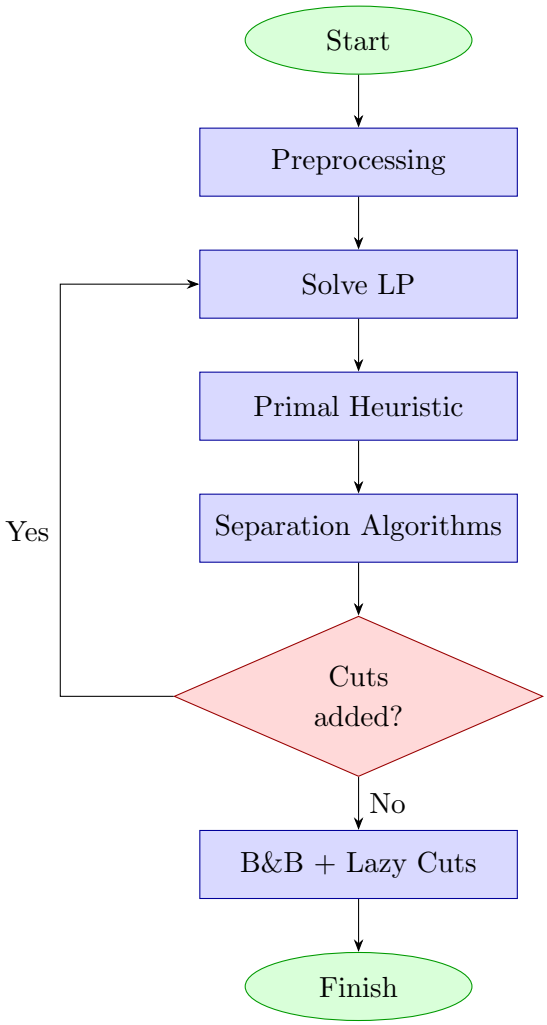


Figure 4.1: Branch-and-Cut framework.

4.2 Separation Algorithms

This section describes the separation procedures used to identify violated inequalities during the cutting phase of the B&C algorithm. The separation algorithms are applied only at the root node of the search tree and are not executed during the branching phase. The goal of these procedures is to strengthen the linear relaxation of the model by dynamically generating violated cuts.

Let x^* denote the fractional solution obtained from the linear relaxation of the problem. We define $G^* = (N^*, E^*)$ as the support graph associated with x^*, where $E^* = \{e \in E \mid x^*_e > 0\}$ and $N^* = N(E^*)$. For each crew $k \in K$, we consider the subgraph $G^*_k = (N^*_k, E^*_k)$, where $E^*_k \subseteq E^*$ corresponds to the edges associated with crew k, and $N^*_k = N(E^*_k)$.

4.2.1 Cuts Acceptance Criteria

When a violated inequality is identified, it is necessary to determine whether it should be incorporated into the model. To evaluate the effectiveness of a candidate cut, we measure the distance between the current fractional solution x^* and the hyperplane defined by the inequality.

Consider a cut of the form $\alpha^\top x \leq \beta$. The distance from x^* to the hyperplane $\alpha^\top x - \beta = 0$ is given by

$$d(x^*) = \frac{|\alpha^\top x^* - \beta|}{\|\alpha^\top\|}.$$

Since the inequality is violated when $\alpha^\top x - \beta > 0$, we define the signed distance

$$d'(x^*) = \frac{\alpha^\top x^* - \beta}{\|\alpha^\top\|}.$$

If $d'(x^*) > 0$, the inequality is violated by the current solution. In practice, a tolerance parameter ϵ is introduced to avoid adding weak cuts. Therefore, a candidate cut is added to the model only if $d'(x^*) > \epsilon = 0.01$.

4.2.2 Logical and Infeasible Cover Cuts

Logical constraints (LC) are systematically checked during the separation phase and added whenever a violation is detected. In addition, we search for violated Infeasible

Cover Cuts (ICC).

For each crew $k \in K$, a candidate set $S \subseteq N$ is constructed by iteratively selecting nodes in decreasing order of their fractional values y^*_{ik}, with $i \in N$. Nodes are added to S while the condition $y_k(S) > |S| - 1$ remains satisfied.

Once the candidate set S has been constructed, we verify whether it is feasible to visit all nodes in S within the available time budget. Determining the exact minimum travel time is equivalent to solving a Traveling Salesman Problem (TSP). In this work, rather than solving a TSP for each candidate set generated during the B&C separation process, we adopt a simple heuristic separation procedure based on a lower bound obtained from a minimum spanning tree (MST), computed using Kruskal's algorithm. The MST calculation considers only the travel distances between nodes and omits their individual service times.

Since the MST provides a valid lower bound on the travel cost of any Hamiltonian tour, if the MST cost alone exceeds the available time budget, visiting all nodes in S is infeasible, and the corresponding ICC is added to the model. However, because the procedure relies on a relaxed lower bound, both by replacing the TSP with an MST-based bound and by omitting service times, it does not guarantee the detection of all violated ICCs.

The use of efficient exact TSP solvers, such as Concorde, provides an alternative to the MST-based feasibility check considered here. Evaluating their integration into the separation procedure and comparing their performance with the MST-based approach adopted in this work was beyond the scope of this study and remains a relevant direction for future research.

4.2.3 Cuts Derived from a Max-Flow Problem

For each crew $k \in K$, the nodes in N^*_k are ordered in decreasing order of their fractional values y^*_{ik}. On the support graph G^*_k, two max-flow problems are solved for each node $i \in N^*_k$: one from the source pair (o, d) to node i, and another from node i to (o, d). These max-flow computations are used to identify violated connectivity cuts, cycle cover cuts, and infeasible path cover cuts. Throughout this process, let $\bar{z}$ denote the value of the incumbent solution, which is dynamically updated as the algorithm progresses.

First, for each node $i \in N^*_k$, we compute the max-flow from the source-sink pair (o, d) to node i. This induces a partition (S_1, S_2) such that $\{o, d\} \subset S_1$ and $i \in S_2$. If $\sum_{j \in S_1} p_j < \bar{z}$, an Infeasible Path Cover Cut (IPCC) is generated for S_1.

Subsequently, we attempt to add a Cycle Cover Cut (CCC) on S_1. If the CCC is rejected according to the acceptance criteria, a Connectivity Cut (CC) associated with node i is added instead.

Next, for each node $i \in N_k^*$, we compute the max-flow from node i to the pair (o, d), resulting in a partition (S_1, S_2) such that $i \in S_1$ and $\{o, d\} \subset S_2$. In this case, we first attempt to add a Generalized Subtour Elimination Constraint (GSEC) on S_1. If the GSEC is rejected, a Connectivity Cut (CC) associated with node i is added. Furthermore, if $\sum_{j \in S_1} p_j < \bar{z}$, an IPCC is generated for the augmented set $S_1 \cup \{o, d\}$.

To avoid generating redundant cuts, once a valid cut is accepted for a given shore S, all nodes $j \in S$ are excluded from further iterations.

4.2.4 Matching Cuts

Matching cuts (MC) are separated using the modified algorithm proposed by Kobeaga et al. (2024). The procedure is applied independently for each crew $k \in K$.

Let $G_{k\lambda}^* = (N_{k\lambda}^*, E_{k\lambda}^*)$ denote the support graph associated with crew k at level λ, where

$$E_{k\lambda}^* = \{e \in E_k^* \mid 0 < x_{ek}^* < \lambda\}$$

and

$$N_{k\lambda}^* = N(E_{k\lambda}^*).$$

The set of candidate handles H corresponds to the connected components in $G_{k\lambda}^*$. For each candidate handle H, the corresponding set of teeth is identified using the approach described by Fischetti et al. (1998).

The level λ is selected from the set of distinct fractional values $\{y_{ik}^* \mid i \in N\}$. For each value of λ, the corresponding support graph is constructed and potential handle–teeth structures are evaluated to detect violated matching inequalities. Whenever a violated inequality satisfying the acceptance criteria is found, the corresponding matching cut is added to the model.

4.2.5 Lazy Cuts

During the B&C process, the solver may encounter integer solutions that violate some connectivity constraints. Therefore, whenever a new incumbent solution is

found, we verify whether the corresponding routes satisfy all required constraints.

For each crew $k \in K$, we extract the route R_k that departs from the origin depot. By connectivity, this route must eventually reach the final depot. Let $R = \{R_k \mid k \in K\}$ denote the set of routes in the incumbent solution.

If the objective value of R coincides with that of the current incumbent, the solution is accepted. Otherwise, the max-flow separation procedure described above is applied in order to detect violated connectivity constraints.

Additionally, if the collected score along route R_k satisfies $\sum_{i \in R_k} p_i \leq \bar{z}$, an IPCC is generated for route R_k.

4.3 Primal Heuristic

To generate an initial feasible solution, we employ a primal heuristic (see Algorithm 1) that is executed at every iteration of the cutting phase at the root node and whenever an integer solution is found, regardless of feasibility. The heuristic is composed of three stages: constructing a candidate solution, repairing potential infeasibilities, and subsequently improving the solution.

A solution $\mathcal{S}$ is defined as a collection of $|K|$ routes, one for each crew. Each route is represented by an ordered sequence of nodes $R = \{o, i_1, i_2, \ldots, i_{|R|-2}, d\}$, where o and d denote the origin and final depots, respectively, and the intermediate nodes correspond to the customers visited by the crew.

Because this heuristic runs frequently during the exact search process, it is designed to be fast. The overall computational complexity of the algorithm is $\mathcal{O}(|K| \cdot |N|^2)$. The time is mostly spent during the construction and repair phases, where the algorithm iteratively checks the customers to add or remove from the route of each crew without performing complex or exhaustive searches.

4.3.1 Constructing a Solution

For each crew $k \in K$, the construction phase starts by randomly selecting an edge $e \in E_k^*$ according to probabilities proportional to the fractional values x_{ek}^*, using a roulette wheel selection mechanism. This edge serves as the initial fragment of the route.

Algorithm 1 Primal Heuristic

Construction phase
$\mathcal{S} \leftarrow \emptyset$
for each crew $k \in K$ **do**
    Select edge $e \in E_k^*$ with probability proportional to its LP relaxation value x_{ek}^* (roulette wheel)
    Initialize route R_k with edge e
    while R_k can be extended **do**
        Identify candidate edges $e \in E_k^* \cap \delta(i)$ such that $N(e) \cap R_k = \{i\}$
        Select an edge e from this set with probability proportional to x_{ek}^*
        Add the adjacent node to R_k
    end while
    if $o \notin R_k$ **and** $d \notin R_k$ **then**
        Reverse R_k if it minimizes the connection distance to o and d
    end if
    if $o \notin R_k$ **then**
        Add o to the start of R_k
    end if
    if $d \notin R_k$ **then**
        Add d to the end of R_k
    end if
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{R_k\}$
end for

Repair phase
while duplicate visits exist in $\mathcal{S}$ **do**
    Remove duplicated node from the route with the largest collected score
end while
for each route $R_k \in \mathcal{S}$ **do**
    while time budget of R_k is violated **do**
        Remove node $i \in R_k$ maximizing $\frac{\delta t_i}{p_i}$
    end while
end for

Improvement phase
Sort unvisited nodes $i \in N \setminus \{o, d\}$ in descending order of score p_i
for each unvisited node i **do**
    Let $R_k \in \mathcal{S}$ be the route with the minimum collected score
    Best-insert i into R_k if the marginal duration maintains time budget $T_{\max}$
end for
return $\mathcal{S}$

The route is then progressively expanded by extending its endpoints. To extend the route from i, we examine edges $e \in E_k^* \cap \delta(i)$ such that $N(e) \cap R = \{i\}$, meaning that the edge connects i to a node that does not yet belong to the route. One of these candidate edges is selected through roulette-wheel sampling using x_{ek}^* as weights, and the corresponding adjacent node is added to the route.

This extension procedure is repeated until it is no longer possible to add new nodes without creating a cycle. The process stops when a depot is reached or when no admissible extension is available. The nodes encountered during this construction phase are recorded in the order in which they are visited.

If the resulting route is missing one of the depots, the missing depot is simply added to the correct end of the sequence. When both depots are missing, we evaluate both possible directions for the route: connecting the origin to the first node and the destination to the last, or vice versa. We choose the direction that results in the shortest connection distance, and then attach the depots to the ends of the route.

Applying this procedure independently to every crew produces a complete solution. However, the resulting solution may not be feasible. For example, the same node may appear in multiple routes, or a route may exceed the available time budget.

4.3.2 Repairing a Solution

After constructing the initial solution $\mathcal{S}$, a repair phase is applied to eliminate infeasibilities. This phase first removes duplicate visits and then enforces the time budget on every route.

Let $\mathcal{S} = \{R_k \subset N_k^* \mid k \in K\}$. If a node i appears in more than one route, that is, if $i \in R_k \cap R_{k'}$ for some $(k, k') \in K^2$, the node is removed from the route with the largest collected score. This procedure is repeated until every node is visited by at most one crew.

Subsequently, if a route $R \in \mathcal{S}$ violates the time limit, nodes are removed iteratively. At each iteration, the node $i \in R$ maximizing the ratio

$$\frac{\delta t_i}{p_i}$$

is deleted from the route, where δt_i represents the reduction in route duration obtained by removing node i. The process continues until the time constraint is satisfied.

This repair process is fully guaranteed to restore feasibility. In the worst-

case scenario, the iterative removal of nodes will reduce a route to an empty trip containing only the initial and final depots, which trivially satisfies the maximum route duration $T_{\max}$.

4.3.3 Improving a Solution

In the final phase, the current solution is refined using an insertion improvement strategy. The procedure considers each node i that is not included in any route of $\mathcal{S}$ and attempts to insert it into one of the existing routes.

Nodes are first considered for insertion into the route $R_k \in \mathcal{S}$ corresponding to the crew with the lowest collected score, provided that the compatibility condition $i \notin I_k$ is satisfied. If the insertion is not feasible for that crew, other crews are examined following the same criterion.

When a feasible insertion is possible, the best insertion rule is applied, meaning that node i is placed in the position that results in the smallest increase in the route duration.

The procedure continues by examining the remaining unvisited nodes and stops when no further node can be feasibly inserted into any route of $\mathcal{S}$.

4.4 Computational Experiments

This section presents the computational results obtained with the proposed B&C algorithm. The objective of these experiments is to evaluate the effectiveness of the approach.

To ensure direct comparability with the results previously reported, all experiments were conducted under the same computational conditions and using the same benchmark dataset described in Chapter 3. In particular, the B&C algorithm was implemented in C++ and solved using CPLEX 22.1.0.

4.4.1 Performance of the B&C Algorithm

The proposed B&C algorithm was tested on 353 benchmark instances with the CPLEX relative MIP Gap tolerance set to 0.0, requiring CPLEX to prove opti-

mality rather than terminate at a prescribed positive relative optimality Gap. The algorithm successfully identified feasible solutions for all instances and demonstrated high robustness across all problem classes. Table 4.1 summarizes its performance by reporting the number of optimal solutions (Opt), together with the minimum, average, and maximum CPU times for each instance class. The notation < 1.00 denotes a CPU time of less than one second.

Table 4.1: Performance of B&C per class: optimal solutions and CPU times in seconds.

Class	**Opt**	**CPU time**		
		Min	**Avg**	**Max**
I	33	< 1.00	< 1.00	< 1.00
II	48	< 1.00	0.04	1.00
III	60	< 1.00	0.13	3.00
IV	24	1.00	19.75	107.00
V	74	< 1.00	10.08	150.00
VI	51	< 1.00	988.52	7200.00
VII	58	< 1.00	228.14	3803.00
Total	348			

In terms of solution quality, the B&C algorithm proved optimality in 348 out of 353 instances, corresponding to 98.6% of the test set. Moreover, it identified the best-known solutions for all instances considered. These results highlight the effectiveness of the approach in producing high-quality solutions consistently across different instance sizes and configurations.

The performance of the algorithm varies across instance classes. For smaller instances (classes I–III), all instances were solved to optimality with minimal computational effort, as reflected by sub-second average solution times. As the problem size increases (classes IV–VII), the difficulty of the instances grows, yet the algorithm maintains a high optimality rate and continues to deliver high-quality solutions.

From a computational perspective, the results in Table 4.1 indicate that the B&C algorithm exhibits strong scalability. For small and medium-sized instances, solution times are typically below one second or only a few seconds. For larger instances, the computational effort increases significantly, with some cases reaching the imposed time limit of 7200 seconds. Despite this, the algorithm remains efficient and is capable of solving most instances within reasonable time limits.

Overall, the results demonstrate that the proposed B&C algorithm is both effective and robust. It consistently produces optimal or near-optimal solutions while maintaining acceptable computational times across a wide range of instance

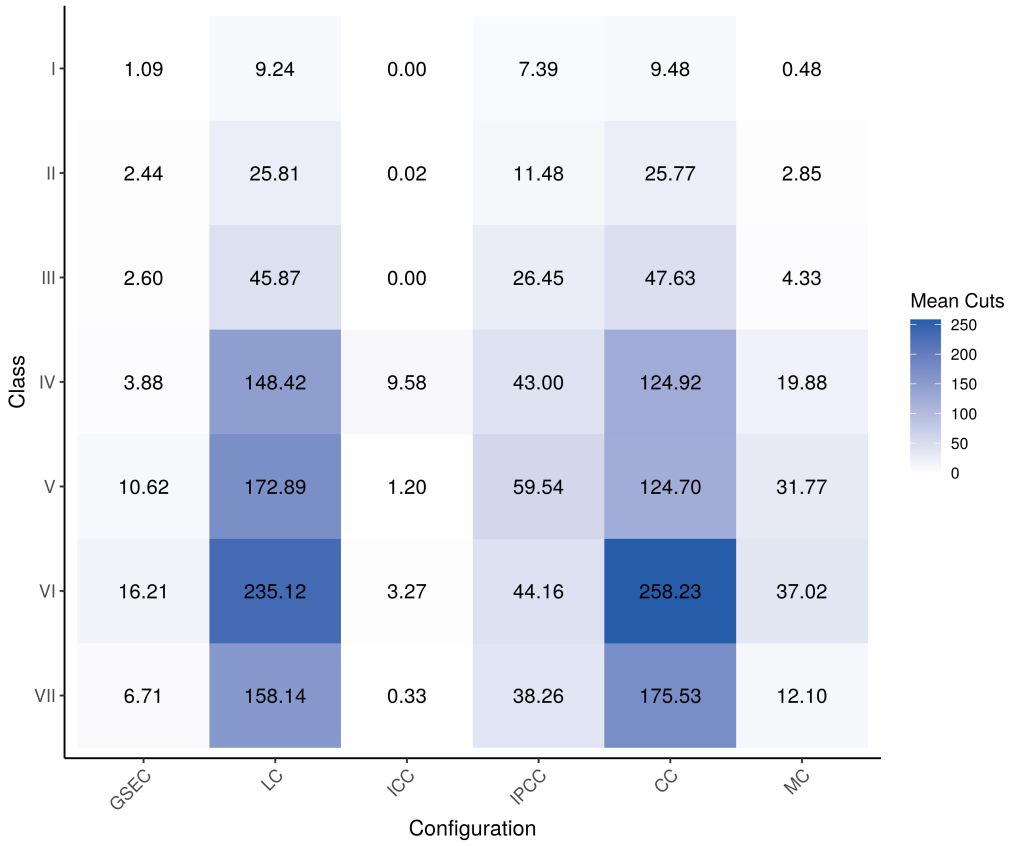


Figure 4.2: Average of generated cuts by family and instance class.

sizes.

4.4.2 Cuts Analysis

This section analyzes the role and impact of cutting planes within the proposed B&C algorithm, considering both their contribution to the separation process and their effect on solution quality.

Figure 4.2 reports the average number of generated cuts per family and instance class. As expected, the number of cuts generally increases with problem size, except for class VII. When contrasted with the results in Table 4.1, this suggests that instances in class VII are comparatively less challenging than those in class VI, which may explain the lower number of separated cuts.

Among the different families, CC, LC, and IPCC cuts are the most prominent. Connectivity cuts (CC) and logical cuts (LC) play a central role in strengthening

Table 4.2: Root node performance summary by instance class.

Class	Opt	Opt (%)	Gap (%)	CPU time (s)
I	25	75.76	2.67	< 1.00
II	26	54.17	11.60	< 1.00
III	18	30.00	10.48	< 1.00
IV	0	0.00	14.13	2.25
V	27	36.49	6.85	1.77
VI	4	7.14	17.25	8.52
VII	11	18.97	13.02	5.64
Total	111	31.44		

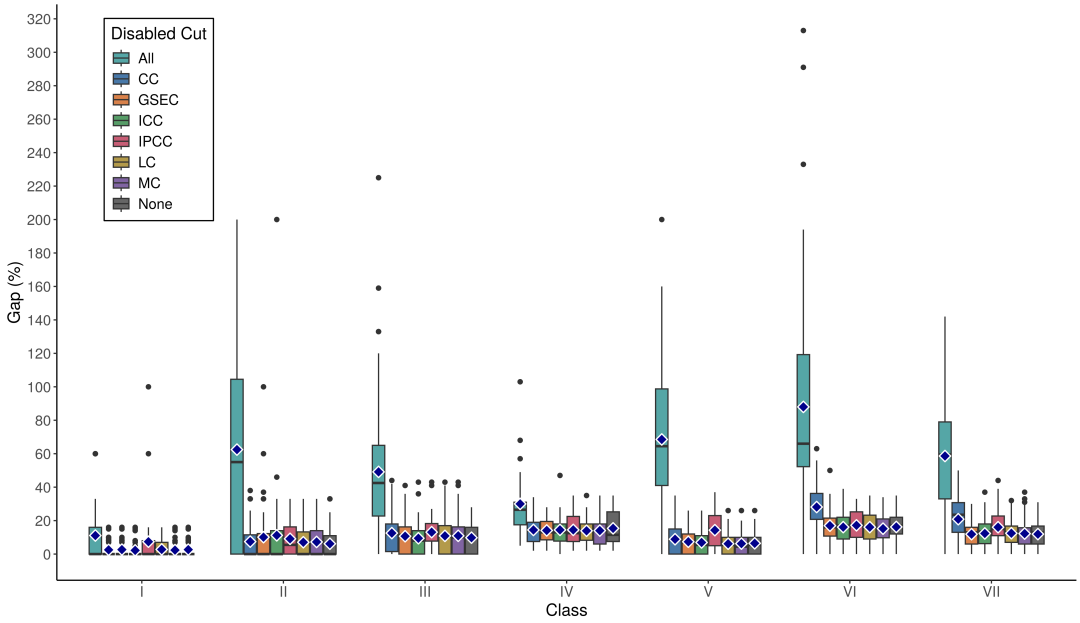
the formulation, while IPCC cuts significantly eliminate infeasible profit allocations. Other families, such as GSEC and ICC, also contribute to tightening the formulation, although their impact is more instance-dependent.

To further assess their impact, we analyze the quality of solutions at the root node. Table 4.2 summarizes, for each class, the number and percentage of optimal solutions, the average Gap, and the average CPU time. Overall, approximately 31% of all optimal solutions are already identified at the root node. The average optimality Gap at this stage is 10.8%, while the computational effort required represents only 1.41% of the total runtime, highlighting the effectiveness of the separation phase in providing strong bounds early in the search process.

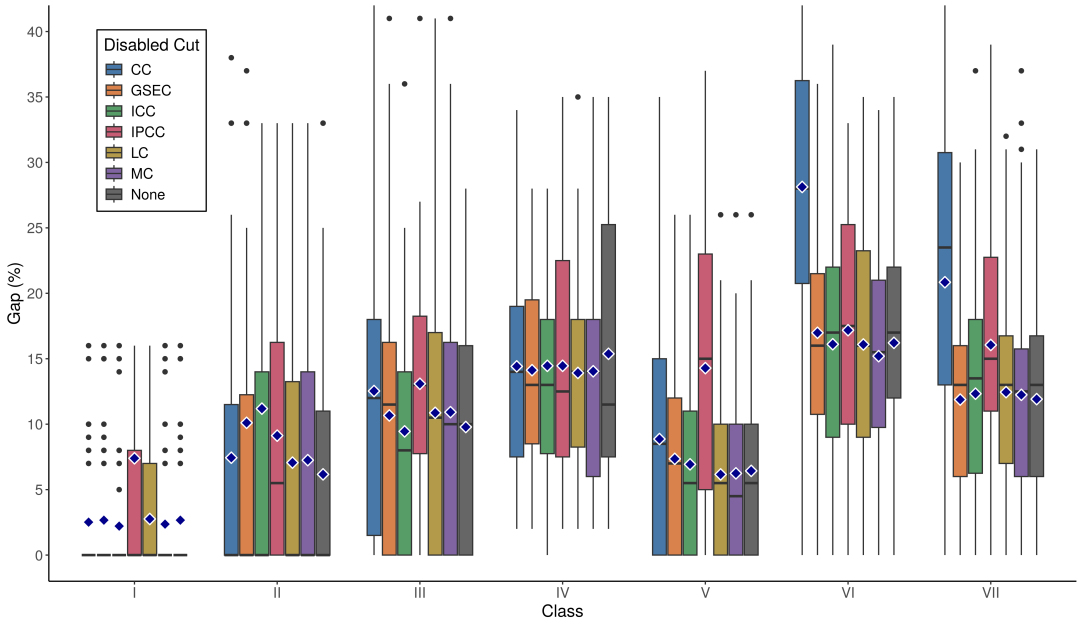
To isolate the contribution of each cut family, we evaluate the root node relaxation under different configurations in which individual cut families are selectively disabled. Figure 4.3 presents the distribution of optimality Gaps (%) across instance classes and model configurations. The *None* configuration corresponds to the complete model, in which no cut family is disabled, whereas *All* represents the baseline configuration in which all cut families are disabled. Because the substantially larger Gaps obtained with *All* make the distributions of the remaining configurations difficult to distinguish, this configuration is excluded from Figure 4.3b. The vertical range of this figure is also restricted to provide a clearer view of the boxes; consequently, some outliers fall outside the displayed range. The blue diamonds are used to indicate mean values.

Additionally, to determine whether the Gap distributions differ among cut configurations, a Kruskal–Wallis test was performed separately for each instance class. The results are reported in Table 4.3. Statistical significance is assessed at the 5% level.

For class I, the Kruskal–Wallis test does not reject the null hypothesis that the



(a) All model configurations.



(b) Detailed view excluding the no-cuts configuration (*All*). Some extreme observations fall outside the displayed range.

Figure 4.3: Distribution of root node optimality Gaps across instance classes and model configurations.

Table 4.3: Kruskal–Wallis test results for the comparison of cut configurations.

Class	**I**	**II**	**III**	**IV**	**V**	**VI**	**VII**
p**-value**	0.3924	0.0002	< 0.0001	0.0061	< 0.0001	< 0.0001	< 0.0001

configurations have the same Gap distribution. Although the mean Gap increases when all cuts are disabled, most observations remain concentrated near zero. The higher mean appears to be driven by a relatively small number of large Gaps rather than by a consistent separation among configurations.

In contrast, the test identifies statistically significant differences among configurations for classes II–VII. This indicates that at least one configuration has a different Gap distribution within each of these classes. However, the Kruskal–Wallis test is an overall test and does not identify which specific pairs of configurations differ. Moreover, because the *All* configuration is clearly separated from the other configurations in several classes, the global significance may be driven largely by the deterioration observed when all cuts are disabled.

The distributions in Figure 4.3 support this interpretation. Disabling all cuts considerably weakens the root node relaxation in classes II–VII. For these classes, the mean Gap under the *All* configuration is approximately 30% or higher. Moreover, in five of the six classes, it is around 50% or higher, reaching nearly 90% in class VI. This configuration also exhibits substantially greater variability and several extreme Gaps, particularly in classes II, III, V, and VI. Class I behaves differently: under the *All* configuration, its mean Gap remains close to 10%, and most observations are concentrated near zero despite the presence of some large values. Overall, these results show that the combined use of the cut families strengthens the root node bound and produces more consistent performance across instances.

The detailed view in Figure 4.3b further shows that disabling a single cut family has a much smaller effect than disabling all cuts simultaneously. Mean Gaps remain below approximately 20% in nearly all family configurations. The main exceptions occur when CC cuts are disabled in classes VI and VII, where the mean Gaps increase to approximately 28% and 21%, respectively. Disabling IPCC cuts also produces a noticeable deterioration in class V, where the mean Gap increases from approximately 6% for the complete model to about 14%. A smaller increase is also observed in class VII.

These class-dependent patterns are consistent with the structural roles of the cut families in the TOPNB. The effect associated with IPCC cuts may reflect their role in eliminating infeasible profit allocations arising from the objective of balancing scores across agents. Similarly, the greater sensitivity to CC cuts in the larger classes

is consistent with the increasing difficulty of controlling disconnected or fractional routing structures. In contrast, disabling LC or MC cuts produces relatively small changes without a consistent pattern across instance classes, suggesting that their individual contribution may depend on their interaction with the remaining cut families. ICC cuts show a more noticeable effect in class II, but their removal does not produce a consistent pattern across the remaining classes. Overall, the results suggest that the strength of the complete formulation arises from the complementary action of the cut families rather than from the uniform dominance of a single family.

Chapter 5

Constraint Programming for the TOPNB

Within the literature review, CP was identified as a promising approach for solving variants of the TOPTW. Motivated by these results, we evaluate the performance of a CP-based formulation on a relaxed version of the problem, in which time window constraints are omitted. The following chapter presents a CP formulation for the TOPNB as an alternative to the MILP formulations introduced previously.

5.1 CP Model

The proposed formulation relies on interval and sequence variables together with global constraints. Interval variables represent activities executed over continuous time intervals, characterized by a start time, an end time, and a duration. These variables can be either mandatory or optional, allowing the model to decide whether a given activity is included in the solution. When an optional interval variable is set to `Absent`, the corresponding activity is excluded and ignored by all constraints and expressions involving interval variables. Using the notation defined in Table 5.1, the formulation is implemented in IBM ILOG CPLEX CP Optimizer.

The following interval and sequence variables are introduced to model customer visits and the sequence of visits performed by each crew:

- $\texttt{visits}_i$: interval variable associated with customer i. This variable is optional for customers and mandatory for depots ($\{o, d\}$).
- $\texttt{visitalts}_{ik}$: optional interval variable representing the execution of customer i by crew k. Compatibility constraints are enforced by fixing $\texttt{visitalts}_{ik}$ to

Table 5.1: CP notation.

Sets	
$\{o, d\}$	initial and final depots
N	set of customers and depots
$N' = N \backslash \{o, d\}$	set of customers (service orders)
K	set of available crews
Parameters	
p_i	score associated with order $i \in N'$
h_i	service time of order $i \in N'$
t_{ij}	travel time from i to j, $i, j \in N$
$T_{\max}$	maximum route duration
c_{ik}	compatibility between customer $i \in N'$ and crew $k \in K$

`Absent` whenever $c_{ik} = 0$.

- R_k: interval sequence variable representing the ordered route of crew $k \in K$.

In this model, the duration of each interval variable explicitly incorporates the service time h_i. To enforce the duration limits of the route, the span of each sequence R_k is constrained by $T_{\max}$ using the `LengthMax` global constraint. A transition matrix $M = \texttt{TransitionDistance}(i, j, t_{ij})$ is defined to capture travel times between customer locations.

Based on this structure, the CP formulation of the TOPNB is given by:

$$\text{Maximize} \quad S_{\min} \tag{5.1}$$

subject to:

$$S_{\min} \leq \sum_{i \in N'} p_i \cdot \texttt{PresenceOf}(\texttt{visitalts}_{ik}), \qquad k \in K, \tag{5.2}$$

$$\texttt{First}(R_k) = \texttt{visits}_o, \qquad k \in K, \tag{5.3}$$

$$\texttt{Last}(R_k) = \texttt{visits}_d, \qquad k \in K, \tag{5.4}$$

$$\texttt{Alternative}(\texttt{visits}_i, \{\texttt{visitalts}_{ik} : k \in K\}), \qquad i \in N', \tag{5.5}$$

$$\texttt{NoOverlap}(R_k, M), \qquad k \in K. \tag{5.6}$$

The objective function in (5.1), together with constraints (5.2), maximizes the minimum score collected by any crew, thus capturing the balancing objective of the TOPNB. Constraints (5.3) and (5.4) ensure that each route starts and ends at its corresponding depot. Constraints (5.5) ensure that each customer is assigned to at most one crew. Finally, the `NoOverlap` constraints (5.6) enforce route feasibility by

preventing temporal overlaps and ensuring that travel and service times between consecutive visits are respected.

Although the formulation adopts a linearized objective for consistency with the MILP model, CP allows a direct representation of the original max–min objective:

$$\text{Maximize} \quad \texttt{IloMin}\left(\sum_{i \in N'} p_i \cdot \texttt{PresenceOf}(\texttt{visitalts}_{ik}),\ k \in K\right). \tag{5.7}$$

This alternative formulation exploits the native ability of CP solvers to handle nonlinear expressions, which may enhance propagation and improve computational performance.

5.2 Computational Experiments

In this section, we analyze several aspects of the proposed solution method, including model convergence and the impact of linearizing the objective function.

A limitation of the CP approach is that, although it is able to find feasible solutions for all instances, it cannot prove optimality unless the entire search space is exhaustively explored. In contrast, the B&C method can prove optimality. For this reason, the optimality of CP solutions is assessed using the B&C results as a benchmark. More precisely, CP solutions are processed prior to analysis: if the objective value obtained by CP matches that of B&C for an instance where B&C proves optimality, then the CP solution is classified as optimal.

The CP model was implemented in C++ using the CPLEX 22.1.0 API. All computational experiments were conducted on the same hardware and software environment as the B&C and the baseline methods, and on the same set of instances, ensuring a consistent comparison. The maximum runtime per instance varies depending on the specific analysis; however, all experiments were conducted with a single thread and default parameter settings.

Since the CP model is based on integer decision variables, all input parameters must be defined as integers. In particular, the values in the M matrix are required to be non-negative integers. Therefore, the distances between each pair of nodes are scaled and rounded by multiplying them by 100 and applying the `ceil` function. To ensure consistent scaling across parameters, the service times h_i are processed in the same way, while the maximum time limit $T_{\max}$ multiplied by 100 yields an integer value without further rounding.

This scaling and rounding process introduces a maximum absolute error of strictly less than 0.01 per pair of nodes and per service time in the original scale. Consequently, for a route k, the maximum accumulated error in its duration is strictly bounded by $0.02 \times (|C_k| + 1)$. Ultimately, because the `ceil` function always rounds distances and service times upward, the CP model provides a conservative approximation of the original problem. Therefore, any route feasible for the CP model is also feasible with respect to the original Euclidean distance matrix M and the original service times. Finally, the feasibility of all obtained routes is always verified using the original Euclidean distances and service times.

5.2.1 CP Over Time

A comprehensive analysis was conducted to examine the temporal dynamics of the model. The objective is to characterize the behavior of the CP model over time, highlighting both its strengths and potential limitations.

The model was evaluated on the adapted instance set under three execution time limits: 600, 1200, and 1800 seconds. These limits were selected to assess whether the CP approach could match or improve the results of the AMH, which has been shown in the literature (Sánchez-Yepez et al., 2023) to produce high-quality solutions within relatively short execution times. Restricting the maximum CPU time to 30 minutes also ensured a practical computational budget. This decision was supported by previous analyses indicating that the branch-and-cut solved most instances within relatively short execution times, while the MILP formulation had already been evaluated with a two-hour time limit. Thus, the objective was to determine whether the CP approach could provide competitive results without relying on another computationally intensive model.

Based on these runs, the percentage improvement achieved across time intervals was computed. Based on these runs, the percentage improvement achieved across time intervals was computed. Specifically, using the results obtained at 10 minutes as the baseline, the percentage improvement was calculated when the runtime was extended to 20 minutes (Mark 1). Similarly, the improvement from 20 to 30 minutes was computed to obtain the corresponding values for Mark 2.

Table 5.2 reports, for each class, the number of instances with improvement (Total), as well as the minimum, average, and maximum percentage improvement, and the number of new optimal solutions (Opt) found with respect to the previous time mark.

Table 5.2: CP percentage improvements by class over time.

Class	Mark 1					Mark 2				
	Total	**Min**	**Avg**	**Max**	**Opt**	**Total**	**Min**	**Avg**	**Max**	**Opt**
I	0	-	-	-	0	0	-	-	-	0
II	0	-	-	-	0	1	5.56	5.56	5.56	1
III	1	7.69	7.69	7.69	1	0	-	-	-	0
IV	2	2.78	3.17	3.57	0	0	-	-	-	0
V	5	0.72	2.59	5.26	2	5	0.71	1.8	4.55	2
VI	13	0.28	1.82	8.14	2	4	0.28	1.07	1.82	0
VII	8	0.4	2.17	6.32	1	1	1.30	1.30	1.30	0

The results show that class I exhibits no improvement at any time point, whereas class II shows a single improvement when the runtime is extended from 20 to 30 minutes, corresponding to a 5.56% increase. Overall, 29 improvements are observed when moving from 10 to 20 minutes, with an average improvement of 2.34%, whereas only 11 improvements are obtained from 20 to 30 minutes, with an average of 1.83%. As expected, given the problem complexity, most improvements occur in larger instances.

Figure 5.1 shows the distribution of the percentage deviations for the two time marks and for each class. The averages are highlighted with a blue diamond mark. We can see that none of them is larger than 9%, and the averages (as shown in Table 5.2) for the larger classes (V to VII) are between 1% and 3%.

These results provide a clear justification for limiting the runtime in subsequent experiments. Extending the execution time from 10 to 20 minutes yields a substantial number of improvements, whereas increasing it further to 30 minutes produces comparatively smaller gains, both in frequency and magnitude. This behavior is also reflected in Figure 5.1, where the distributions for the second interval are more concentrated and exhibit lower variability, indicating that most of the significant progress is achieved within the first 20 minutes. Consequently, a time limit of 1200 seconds is adopted in the following experiments as a suitable balance between solution quality and computational efficiency.

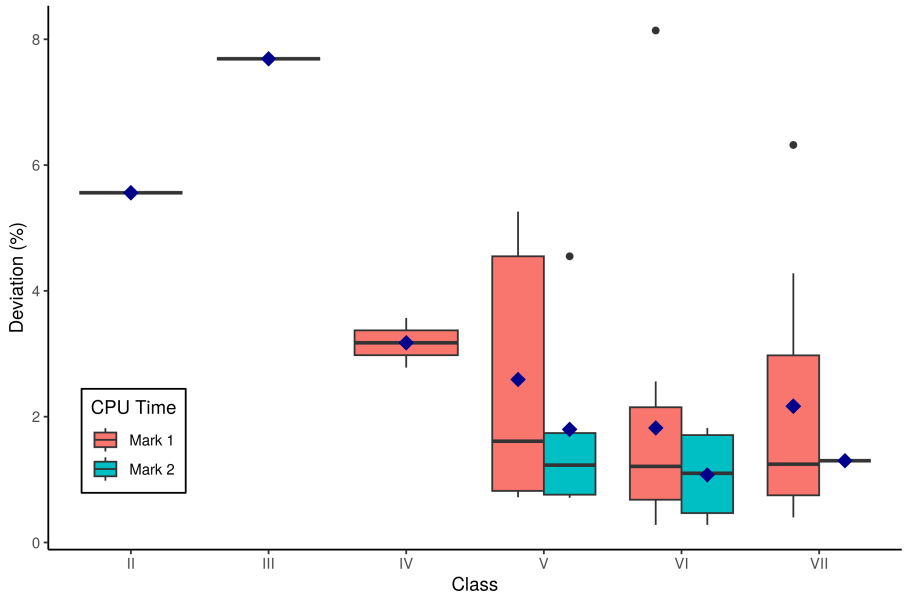


Figure 5.1: Distribution of percentage improvements by class and time interval.

5.2.2 Linear versus Nonlinear Objective Function

In a preliminary study (Sánchez-Yepez, 2019), the problem was solved as an MILP using CPLEX, which required linearizing the objective function. In contrast, constraint programming allows us to model the objective function in its original nonlinear form. Motivated by this, we perform a comparative analysis between two versions of our CP model: one with the original nonlinear objective function and another with a linearized version.

Both versions were tested over the complete set of instances using a time limit of 1200 seconds, as determined by the convergence analysis. Since both models consistently produced feasible solutions, the comparison focuses on solution quality and the number of optimal solutions found, where optimality is assessed with respect to the results obtained by the B&C approach, as described in the previous section.

Table 5.3 summarizes the empirical comparison between the two formulations. For each instance class, it reports the number of instances for which both formulations obtain the same objective value (Equal), the nonlinear formulation performs better, or the linear formulation performs better, together with the number of optimal solutions obtained by each formulation.

The results in Table 5.3 show that the two formulations obtain identical objective values in most instances. Overall, 252 of the 353 instances (71.4%) yield

Table 5.3: Comparison of the nonlinear and linear CP formulations in terms of solution quality and number of optimal solutions.

Class	Solution quality			Optimal solutions	
	Equal	**Nonlinear**	**Linear**	**Nonlinear**	**Linear**
I	25	8	0	32	24
II	44	4	0	45	41
III	58	2	0	57	55
IV	18	1	5	16	19
V	55	9	10	52	49
VI	17	17	22	11	14
VII	35	10	13	33	29
Total	252	51	50	246	231

the same solution quality, while the nonlinear formulation performs better in 51 instances (14.4%) and the linear formulation in 50 (14.2%).

An analysis by instance class reveals a more differentiated pattern. For classes I–III, the formulations obtain identical objective values in 75.8%, 91.7%, and 96.7% of the instances, respectively. Whenever the solutions differ in these classes, the nonlinear formulation performs better: it improves upon the linear formulation in 8, 4, and 2 instances, respectively, whereas the linear formulation does not outperform it in any instance. This pattern is also reflected in the number of optimal solutions, with the nonlinear formulation obtaining 32 versus 24 in class I, 45 versus 41 in class II, and 57 versus 55 in class III.

From class IV onward, the comparison becomes more balanced. In class IV, the linear formulation performs better in 5 instances, compared with only 1 for the nonlinear formulation, and obtains more optimal solutions (19 versus 16). In classes V–VII, the formulations differ more frequently, particularly in class VI, where only 17 of the 56 instances yield identical objective values. The linear formulation yields a better objective value more frequently across these three classes. However, this does not always translate into a larger number of optimal solutions: the nonlinear formulation obtains more optimal solutions in classes V and VII, whereas the linear formulation does so in class VI.

Across the complete instance set, the nonlinear formulation obtains 246 optimal solutions, corresponding to 69.69% of the 353 instances, compared with 231 optimal solutions (65.44%) for the linear formulation. Overall, the descriptive results indicate very similar solution quality between the two formulations, with a modest advantage for the nonlinear formulation in terms of the total number of optimal solutions.

To determine whether the observed differences in solution quality are statistically significant, a Wilcoxon signed-rank test was performed globally and independently for each instance class, using a significance level of $\alpha = 0.05$. Since multiple hypothesis tests (one for each class) were conducted, their p-values were adjusted using the Holm procedure to control the family-wise error rate. Table 5.4 reports the Wilcoxon test statistic (W), the corresponding p-value, and the Holm-adjusted p-value for each instance class, together with the results of the global comparison.

Table 5.4: Wilcoxon signed-rank test comparing the linear and nonlinear CP formulations globally and by instance class.

Class	W	p**-value**	**Adj.** p**-value**
Global	2539.0	0.5499	-
I	36.0	0.0051	0.0360
II	10.0	0.0445	0.2669
III	3.0	0.1855	0.9277
IV	2.5	0.9670	1.0000
V	91.5	0.5647	1.0000
VI	341.0	0.7553	1.0000
VII	109.0	0.8159	1.0000

The statistical results show that, globally, there is no statistically significant difference between the two formulations ($p = 0.5499$). Analyzing the results by instance class reveals that only class I exhibits a statistically significant difference after applying the Holm correction (adjusted $p = 0.0360$), providing statistical evidence in favor of the nonlinear formulation, consistent with the descriptive results in Table 5.3.

Overall, the results indicate that the two formulations have comparable performance across the complete instance set. Most instances yield identical objective values, the number of instances in which each formulation performs better is nearly the same, and the global statistical test does not identify a systematic difference. Nevertheless, the nonlinear formulation obtains more optimal solutions overall and shows a clear advantage for the smallest instances, with statistically significant evidence of superiority in class I. Considering these results together with the simpler modeling structure obtained by preserving the objective function in its original form, the nonlinear CP formulation is selected for the remainder of this work.

In the next chapter, the selected nonlinear CP model is compared with other solution approaches to provide a broader evaluation of its performance.

Chapter 6

Experimentation and General Results

This chapter presents a comparative analysis of the solution approaches considered for the problem. Specifically, the performance of the branch-and-cut algorithm and the constraint programming model with a nonlinear objective function proposed in this work is evaluated against existing methods from the literature (Sánchez-Yepez et al., 2023), namely an adaptive multi-start heuristic (AMH) and a mixed-integer linear programming (MILP) formulation.

To ensure a fair and reproducible comparison, all methods are executed on the same computational environment and evaluated over the same set of benchmark instances. Furthermore, the comparison is based on a common set of performance metrics that capture both solution quality (the number of optimal and best known solutions and deviations relative to the B&C results) and computational efficiency (execution times).

The objective of this comparison is to assess the performance of the four methods in terms of solution quality, computational efficiency, and their ability to prove optimality. In this context, the B&C algorithm is used as a reference method to establish optimal solutions whenever possible, thereby enabling a unified evaluation framework.

The results of the analysis highlight the strengths and limitations of each approach and provide a clear positioning of the proposed methods with respect to the state of the art.

Table 6.1: Number of optimal and best known solutions reached per method.

Class	Optimal				Best known			
	B&C	**MILP**	**AMH**	**CP**	**B&C**	**MILP**	**AMH**	**CP**
I	**33**	**33**	**33**	32	**33**	**33**	**33**	32
II	**48**	**48**	47	45	**48**	**48**	47	45
III	**60**	54	54	57	**60**	54	54	57
IV	**24**	23	17	16	**24**	23	17	16
V	**74**	25	44	52	**74**	25	44	52
VI	**51**	19	14	11	**56**	19	15	12
VII	**58**	12	31	33	**58**	12	31	33
Total	**348**	214	240	246	**353**	214	241	247

6.1 Solution Quality

Table 6.1 reports the number of optimal and best known solutions obtained by each approach across the different instance classes. Overall, the B&C algorithm achieves the best solution quality, identifying the best-known solution for all instances in the test set and proving optimality in 98.58% of the cases.

In contrast, the MILP, AMH, and CP approaches achieve optimality in 60.62%, 67.99%, and 69.69% of the instances, respectively. While these methods are competitive in smaller or less complex instance classes (classes I and II), their performance deteriorates as the problem size or difficulty increases.

A more detailed analysis by class reveals that the difference between B&C and the other approaches becomes particularly pronounced in classes V, VI, and VII. In these cases, the number of optimal solutions found by MILP and AMH drops significantly, whereas CP shows a more robust behavior, consistently outperforming both MILP and AMH. Regarding best known solutions, B&C maintains its dominance across all classes, even in class VI where other methods show minor variations.

These results confirm the effectiveness of the B&C algorithm as a reference method and highlight the relative strengths of CP compared to MILP and AMH, particularly in more challenging instances.

While the previous analysis focuses on the ability to identify high-quality solutions, it does not provide a detailed view of the overall solution quality across all instances.

6.2 Relative Performance Analysis

To complement this evaluation, the relative deviation of each method with respect to the B&C solutions is analyzed over the entire set of instances. The percentage deviation is computed using Equation (6.1):

$$\text{Dev} = \frac{Z_{B\&C} - Z_x}{Z_{B\&C}} \times 100, \tag{6.1}$$

where $Z_{B\&C}$ denotes the objective value obtained by the B&C algorithm and Z_x represents the objective value obtained by method x.

It is worth noting that B&C solutions serve as a reference and may not always correspond to the proven optimal values, though they represent the best solutions available within the time limit.

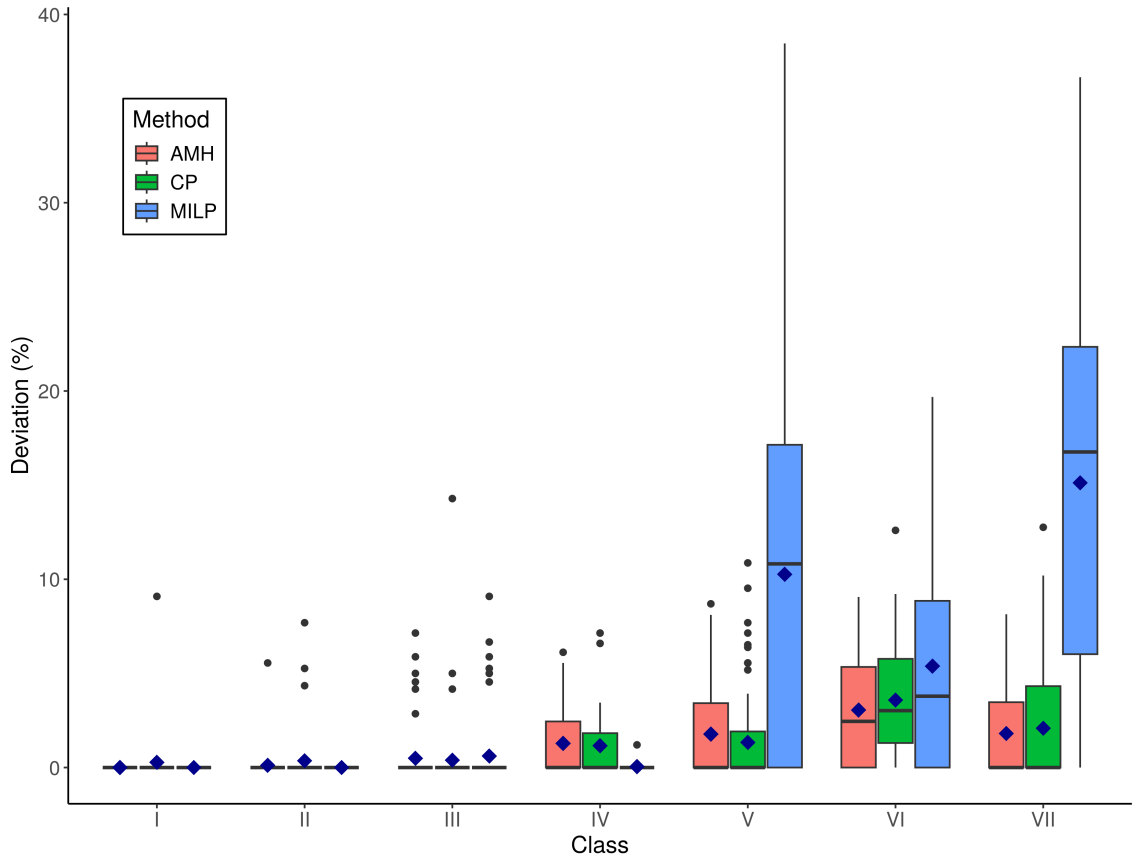


Figure 6.1: Percentage deviation from the B&C objective value for each method and class.

Figure 6.1 presents the distribution of these deviations across instance classes. Overall, the results reveal significant differences in both the magnitude and the

variability of deviations across the methods. For the smaller classes (I–III), all approaches exhibit insignificant deviations, indicating that high-quality solutions are consistently obtained.

However, as instance difficulty increases (classes IV–VII), clear performance differences emerge. In particular, MILP shows substantially larger deviations and higher variability, especially in classes V and VII, indicating a lack of robustness in more challenging instances. In contrast, CP and AMH exhibit relatively low median deviations, though AMH shows greater dispersion and a higher number of outliers.

These results indicate that CP provides a more stable and reliable solution quality across the full set of instances, whereas MILP tends to degrade significantly in harder cases.

6.3 Computational Efficiency

A direct comparison of computational times across all methods is not straightforward as time limits differ between approaches; such comparisons may lead to biased conclusions. Additionally, the CP implementation does not provide the time at which the best known solution is obtained, but only the total CPU time upon termination (1200 seconds). Therefore, the analysis is conducted in two complementary steps.

The first analysis focuses on methods with comparable timing information, namely B&C, MILP, and AMH. Figure 6.2 presents the average computational times across instance classes. To improve readability and account for the large variability in running times, the vertical axis is displayed on a logarithmic scale.

As shown in Figure 6.2, the computational effort increases with the instance class for all methods, although the growth rate differs significantly among them. The MILP approach exhibits the highest computational times across all classes, with a clear increase from medium-sized instances (classes III–V) onwards, reaching several thousand seconds for the largest classes. This behavior highlights its limited scalability for larger instances.

In contrast, the AMH method shows a more stable, moderate increase in computational time, remaining consistently lower than the MILP approach across all classes. This indicates that AMH provides a good balance between solution quality and computational efficiency.

The B&C algorithm demonstrates very low computational times for small instances (classes I–III), but its performance becomes more variable for larger classes.

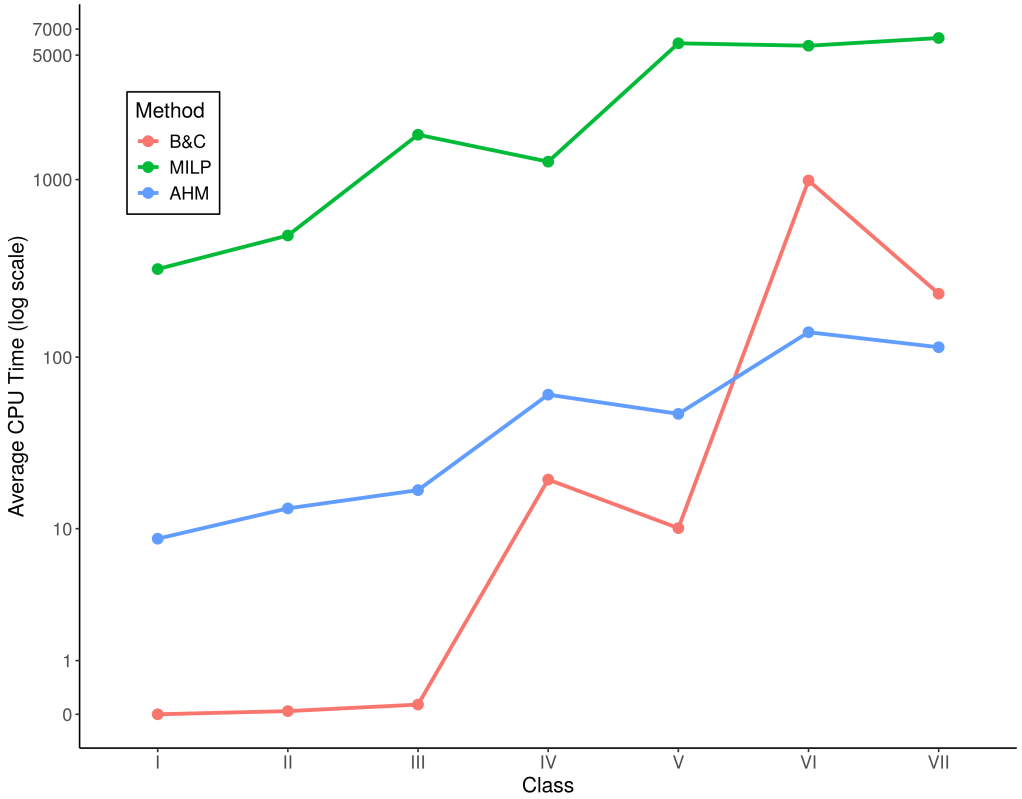


Figure 6.2: Average CPU time by class for each method.

In particular, a significant increase is observed in class VI, followed by a decrease in class VII (behavior already explained in Chapter 4). This variability suggests that, while B&C is highly efficient for simpler instances, its performance may depend more strongly on instance characteristics as problem size increases.

The second analysis incorporates CP under conditions that minimize the impact of the methodological differences. The analysis is restricted to the subset of instances where all methods reach the same best known solution. This ensures that differences in running times are not influenced by the quality of the solution. For these instances, the fastest method is identified for each case and the results are aggregated by class. Figure 6.3 presents the percentage of instances in which each method achieves the shortest computational time.

The results show that the B&C algorithm dominates in most classes, achieving the highest percentage of fastest executions. In several classes, it solves all instances faster than the other methods. This result highlights its strong computational performance when the optimal or best known solution is attainable within the time limit.

The MILP and AMH methods contribute marginally in comparison, with only

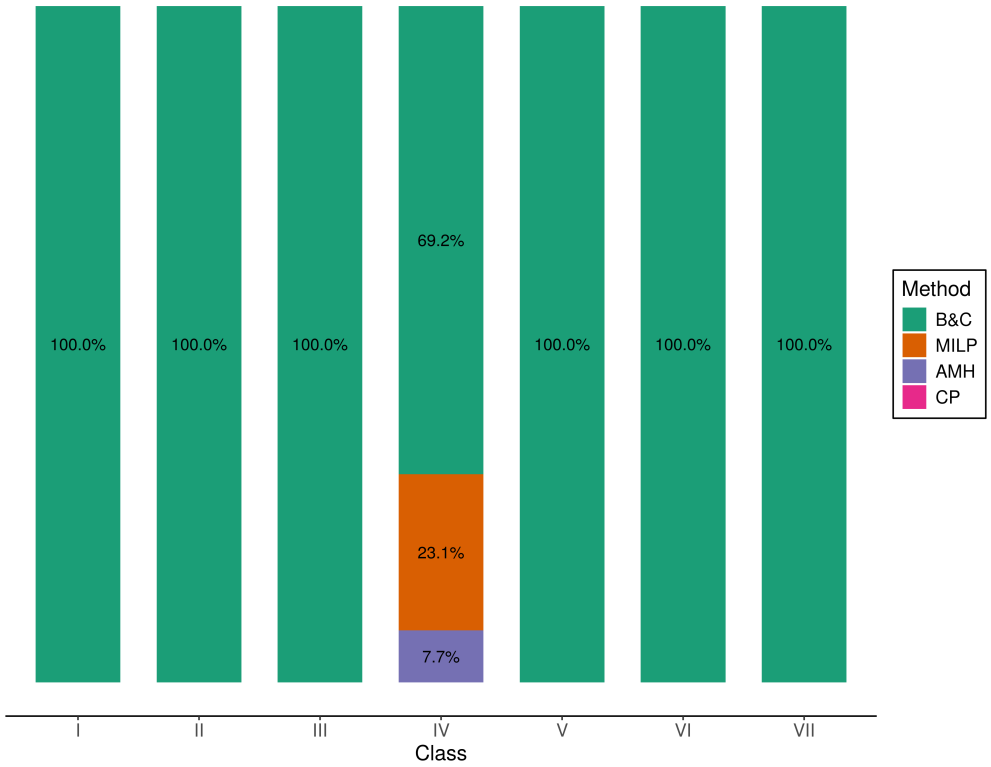


Figure 6.3: Percentage of fastest methods by class on instances where all methods reach the same solution.

a small proportion of fastest instances and only in class IV. This suggests that, although these approaches can reach competitive solutions, they are generally slower than B&C when all methods achieve the same solution quality. Regarding the CP approach, it is not identified as the fastest method in any instance, although this result is influenced by the available timing information.

Taken together, these results indicate clear differences among the evaluated methods. The B&C approach stands out as the most effective overall, achieving the highest number of optimal solutions and reaching all best known solutions.

The CP approach delivers competitive solution quality, ranking second-highest in the number of optimal and best known solutions. Nevertheless, its computational performance should be interpreted with caution due to the limitations of the available timing information. By contrast, the AMH method shows intermediate behavior, while MILP becomes less competitive, particularly for larger instances.

From a computational perspective, B&C also performs best under comparable conditions, as it is the fastest method in the majority of instances in which all approaches reach the same best known solution.

Overall, these results indicate that B&C is the most robust method, combining high solution quality with strong computational efficiency, while the remaining methods offer alternative trade-offs.

6.4 Case Study

The case study considered in this section is based on a real-world problem faced by a Mexican telecommunications company and corresponds to the instance introduced in the literature (Sánchez-Yepez et al., 2023). The problem involves daily workforce scheduling, in which field service crews must be assigned to customer service requests.

The original operational data cannot be disclosed. Consequently, a synthetic instance was developed to preserve the main operational characteristics and decision-making conditions of the real problem while ensuring data confidentiality. The instance assumes that any household requiring telecommunication services may become a potential customer. Based on this assumption, customer locations are generated randomly within the service area of a selected branch office using the zip codes associated with that branch. This procedure produces a realistic spatial distribution of service requests while preserving the statistical properties of the original instance. An example of the resulting customer distribution is shown in Figure 6.4.

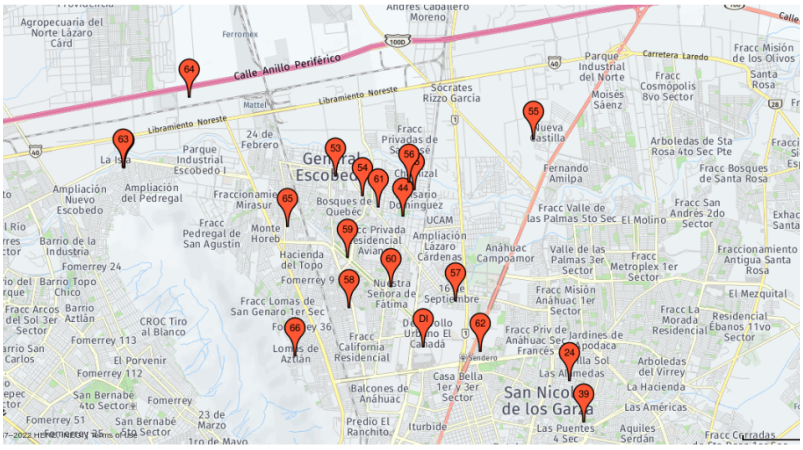


Figure 6.4: Case study example.

The operational setting reflects the field service operations of the company. Seven different service types are considered, together with two crew categories. A total of 52 crews are available, including 8 crews of type T1 and 44 crews of type T2, each working an 8-hour shift. A preliminary analysis of historical daily operations was conducted to define the parameters of the instance. This analysis provided the number of service orders for each service type, the corresponding service times,

Table 6.2: Case study data.

Service Type		Type 1	Type 2	Type 3	Type 4	Type 5	Type 6	Type 7
No. Orders		2	65	75	63	18	8	20
Service Time (min)		18.44	29.3	37.03	51.9	21.48	74.09	60.52
Score		1	2	4	5	1	6	7
Compatibility	**T1**						√	√
	T2	√	√	√	√	√		

and the scores. The compatibility between service types and crew categories was established according to the technical qualifications and operational capabilities of each crew type.

Table 6.2 summarizes the parameters of the case study, including the service types, the compatibility between services and crew categories, the number of service orders, the required service times (in minutes), and the associated scores. A check mark (√) indicates that the corresponding crew is qualified to perform the service, whereas an empty cell indicates that the service cannot be assigned to that crew.

In Sánchez-Yepez et al. (2023), results are reported exclusively for the AMH approach, so building upon this setting, we evaluate the CP formulation on the same instance and compare its performance against the reported AMH results. It is important to note that, although the B&C algorithm could potentially be extended to the case study, the original formulation assumes an undirected graph, whereas the available data represent network distances on a directed graph. Adapting the formulation and its separation procedures to account for this asymmetry falled outside the scope of the present work. Therefore, no results obtained using the B&C method are reported.

6.4.1 Solution Quality

Table 6.3 reports a summary of the results obtained by the AMH approach in Sánchez-Yepez et al. (2023) over multiple independent executions. The method consistently achieves an objective value of 16, with an average runtime of 121.83 seconds per run and a total computational time of 1218.30 seconds.

The CP approach was evaluated on the same instance using a time limit of 10 minutes. The solution obtained matches the objective value reported for AMH, reaching a value of 16.

From a solution quality perspective, both methods achieve identical results. In terms of computational effort, AMH requires multiple executions to assess solution stability, whereas CP obtains a solution of comparable quality within a single run. Although the computational effort cannot be directly compared due to the limited timing information available for CP, these results provide evidence that CP is capable of matching the performance of the heuristic approach while requiring a single execution.

Table 6.3: Comparison of AMH and CP for the case study.

Method	**Objective value**	**CPU Time (s)**		**Runs**
		Average	**Total**	
AMH	16	121.83	1218.30	10
CP	16	–	600.00*	1

* Time limit.

To better understand the behavior of the CP approach, Table 6.4 shows the results obtained using different time limits. We can see the evolution of the objective value over time. With a time limit of 300 seconds, CP returns a Max–Min value of 15. Increasing the time limit to 600 seconds yields a value of 16. No better solution is obtained with the longer time limits considered. CP attains the best known objective value with a time limit of 600 seconds.

Table 6.4: CP performance under different time limits.

CPU Time (s)	**Objective value**
300	15
600	16
1200	16
1800	16
2700	16
3600	16

These results highlight a fundamental difference between both approaches. While AMH relies on multiple independent executions to ensure solution quality and possible stability, CP is able to achieve a solution of identical quality within a single run.

This result is particularly relevant and supports using CP as a practical alternative for solving real instances of the problem, especially in settings where other methods cannot be applied due to the structure of the data.

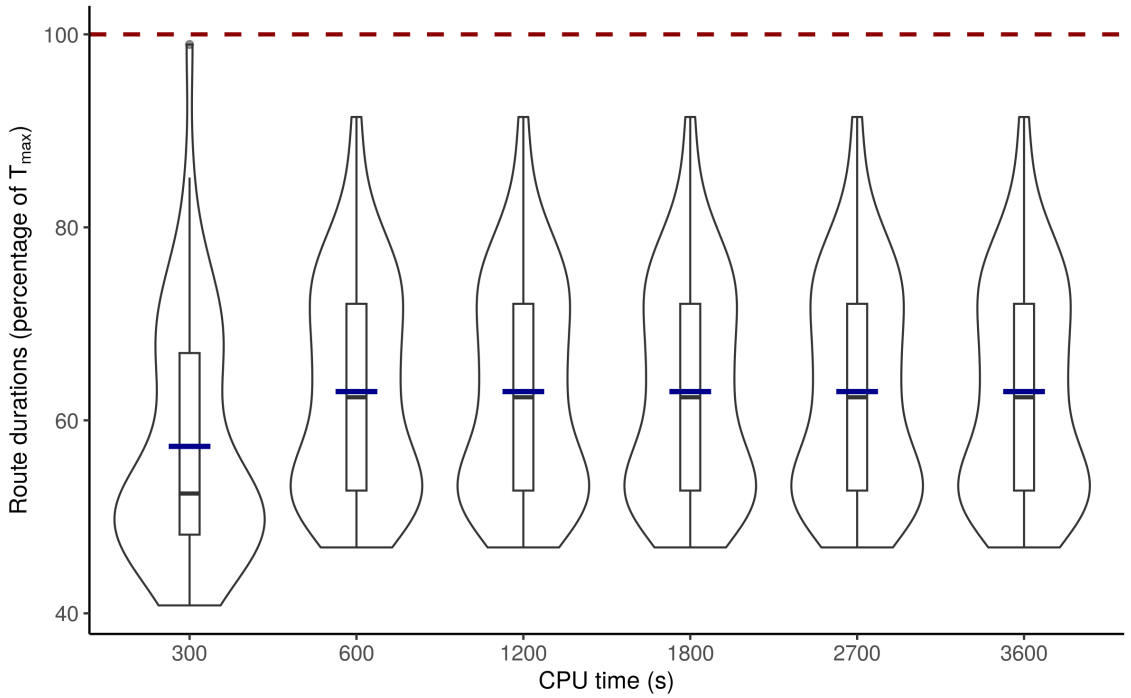


Figure 6.5: Distribution of route durations obtained by CP under different time limits.

6.4.2 Characteristics of the Solutions

Since the objective value alone does not fully describe the operational characteristics of a solution, we also analyzed the route scores, route durations, and route structures produced by the CP and the AMH. This analysis provides further information about route balance and the possible implications of using these solutions in practice.

Figures 6.5 and 6.6 show the distributions of route durations obtained by CP and AMH, respectively. Figures 6.7 and 6.8 present the corresponding route score distributions. In all figures, the blue mark represents the mean of each distribution. Route durations are expressed as a percentage of the maximum route duration, $T_{\max}$.

The CP results fall into two groups. The first group contains the solution obtained with a time limit of 300 seconds, while the second contains the identical solutions obtained with time limits of 600 seconds or more. Table 6.5 summarizes the main indicators of these two solutions.

Increasing the CP time limit from 300 to 600 seconds improves the Max–Min value from 15 to 16. At the same time, customer coverage decreases from 249 customers (99.2%) to 248 customers (98.8%). The longest route decreases from 98.95% to 91.44% of $T_{\max}$, while the route-duration standard deviation decreases

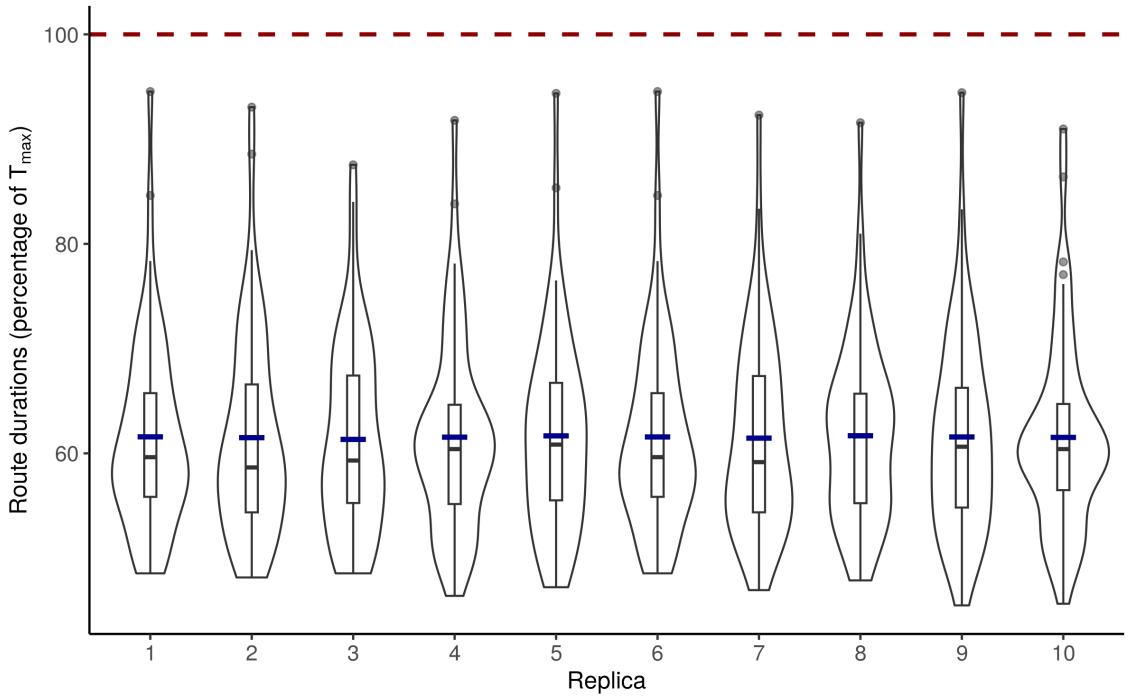


Figure 6.6: Distribution of route durations obtained in the ten AMH replicas.

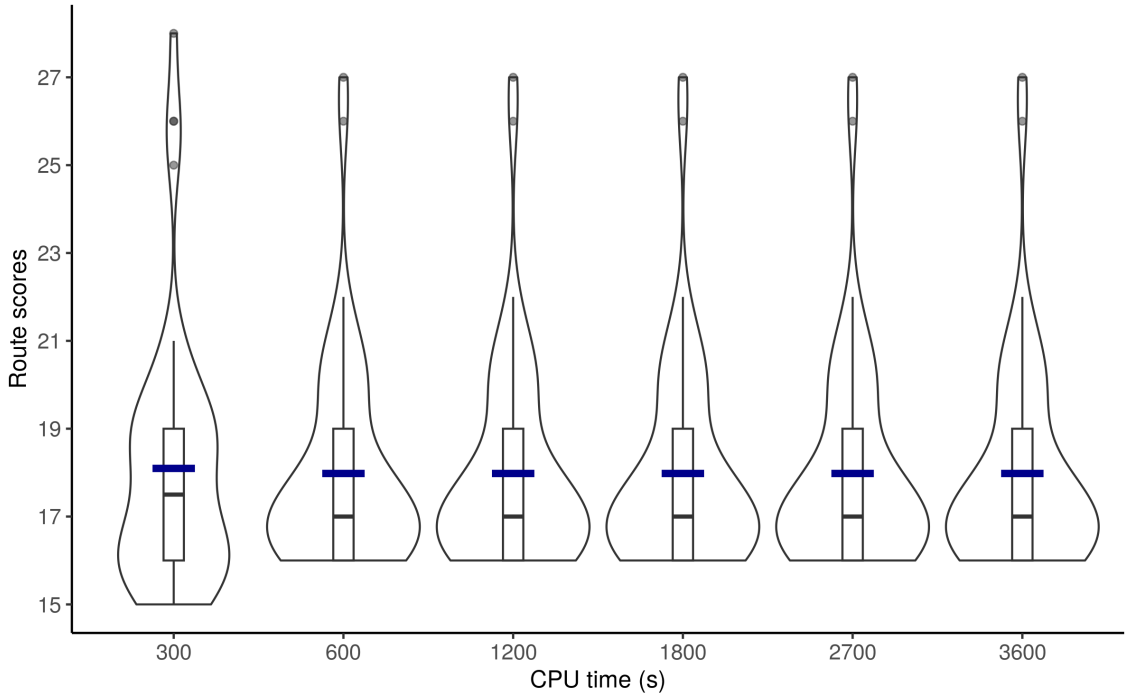


Figure 6.7: Distribution of route scores obtained by CP under different time limits.

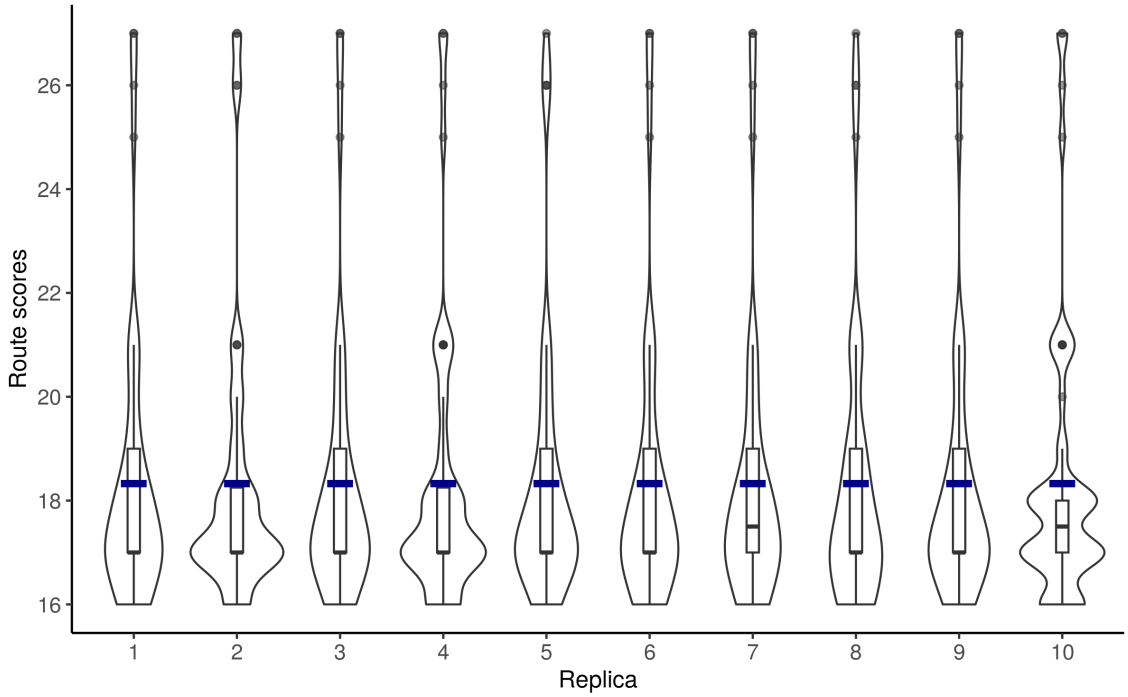


Figure 6.8: Distribution of route scores obtained in the ten AMH replicas.

Table 6.5: Operational indicators of the CP solutions.

Measure	**CPU time limit (s)**	
	300	**600+**
Objective value	15	16
Average score	18.10	17.98
Score range	15–28	16–27
Score standard deviation	2.90	2.35
Customers served	249	248
Mean route duration (% of $T_{\max}$)	57.30	62.98
Route duration standard deviation	12.26	11.05

from 12.26 to 11.05 percentage points of $T_{\max}$, a reduction of approximately 10%. The score standard deviation also decreases from 2.90 to 2.35, and the score range narrows from 15–28 to 16–27. Thus, the CP solution obtained from 600 seconds onward improves the minimum route score and reduces the variation in both scores and durations, but it leaves three customers unserved.

Table 6.6 provides the corresponding indicators for the ten AMH replicas. For each replica, it reports the standard deviation of the route scores (Score SD), the mean route duration as a percentage of $T_{\max}$, and the standard deviation of the route durations (Route duration SD) in the same units. All replicas reach a Max–Min value of 16, a mean score of 18.33, and serve all 251 customers.

Table 6.6: Operational indicators of the AMH solutions.

Replica	**Score SD**	**Mean route duration (% of $T_{\max}$)**	**Route duration SD (% of $T_{\max}$)**
1	2.67	61.58	9.07
2	2.68	61.50	9.68
3	2.64	61.33	9.10
4	2.69	61.54	9.17
5	2.63	61.67	9.40
6	2.67	61.57	9.08
7	2.65	61.45	9.16
8	2.66	61.68	8.52
9	2.67	61.56	9.45
10	2.68	61.52	9.13

Despite their different route structures, the AMH solutions have very similar operational indicators. Their average route durations range only from 61.33% to 61.68% of $T_{\max}$, while their score standard deviations range from 2.63 to 2.69. The main differences appear in the route duration standard deviation, which ranges from 8.52 to 9.68 percentage points of $T_{\max}$.

At an objective value of 16, the two methods show different forms of balance. The route duration standard deviation ranges from 8.52 to 9.68 for AMH, compared with 11.05 for CP, indicating lower variation in route durations for the AMH solutions. In contrast, CP has a lower route score standard deviation of 2.35, compared with values from 2.63 to 2.69 for AMH. Both methods have the same score range of 16–27. All AMH replicas serve 251 customers, whereas the CP solution serves 248. Since the main objective is to balance route scores, the CP result is consistent with this goal, while customer coverage and route duration balance provide additional criteria for selecting a solution.

The methods also differ in the type of plans they provide. CP returns the same route plan for every time limit from 600 seconds onward, showing stability with respect to the time limits considered and supporting its use as a practical alternative. AMH produces ten different plans with the same best known objective value and full customer coverage, giving the decision maker several alternatives.

From a practical perspective, the methods provide the company supervisor with a starting point for assigning and sequencing the service orders. Instead of building the complete schedule manually based only on experience, a process that may take several minutes even hours, the supervisor can obtain a feasible plan within minutes and then adjust it when necessary. The generated plans consider crew–service compatibility, work shifts, service times, travel times, and route score balance at the same time. This can reduce planning effort and the risk of assigning more work than a crew can complete during its shift.

The balance of the score also has a direct practical effect because the wages of the technicians are proportional to the scores of the services they performed. A more balanced distribution therefore supports a more transparent and equitable distribution of aerning among crews. At the same time, including service and travek times in the planning process can improve the use of working hours and reduce the risk of overtime, unfinished service order, unnecesary travel and the associated operating costs.

The process could also be automated while keeping the supervisor responsible for the final decision. A stable solution provides a consistent daily plan, whereas multiple solutions with similar performance give the supervisor alternatives when operating conditions change. For example, if crew availability or the set of service orders changes, an alternative plan may be reviewed instead of rebuilding the schedule from the beginning. This can shorten response times, support faster replanning, and reduce the financial and service risks caused by delays or interrupted routes.

Chapter 7

Conclusions

This chapter summarizes the main findings of this research, discusses its primary contributions and practical implications, evaluates the research hypotheses, and outlines directions for future research.

7.1 Summary of Main Findings

This research provides a comprehensive analysis of different optimization approaches for the Team Orienteering Problem with Nonidentical Agents and Balanced Score, which is motivated by a real-world problem from a Mexican telecommunications company.

The main findings are summarized as follows:

1. The mathematical formulation and valid inequalities presented in Chapter 3 define a robust modeling framework for the TOPNB. The experimental analysis shows that the three-index formulation with valid inequalities results in a more tractable formulation since the solution method achieves consistent performance across different instance classes and objective functions, with low average deviations and a high proportion of improvements over the baseline (the formulation without valid inequalities). Moreover, under this formulation, the solution method achieves all known optimal solutions by the baseline model and is able to identify new optima, highlighting its practical effectiveness despite marginal differences in average performance.

2. The reformulation of the TOPNB used within the branch-and-cut algorithm described in Chapter 4 provides the best overall performance. The proposed approach identifies the best-known solution for all instances in the test set and

proves optimality in 98.58% of the cases. In terms of computational efficiency, it dominates most instance classes, achieving the highest proportion of fastest executions.

3. The constraint programming model introduced in Chapter 5 provides a competitive alternative solution approach. The results show that CP is the second-best method in terms of solution quality, while offering more stable and consistent performance across the full set of instances compared to MILP and AMH approaches. Additionally, the CP model successfully solves the case study, matching the objective value reported for AMH in the literature in a single run.

Based on the experimental findings summarized above, the first research hypothesis, which states that *strengthened mixed-integer linear programming formulations combined with advanced exact solution methods can significantly improve solution quality and computational performance*, is supported by Findings 1 and 2. The incorporation of valid inequalities together with the proposed branch-and-cut algorithm improves the tractability of the mathematical model, identifies additional optimal solutions beyond those obtained with the baseline formulation, proves optimality in 98.58% of the benchmark instances, and achieves superior computational performance.

The second research hypothesis, which states that *alternative solution paradigms such as constraint programming provide competitive and robust performance for this class of problems*, is supported by Finding 3. Although the CP model does not outperform the proposed B&C in terms of overall solution quality, it consistently produces high-quality solutions, exhibits stable performance across the complete benchmark set, and successfully solves the real-world case study in a single execution. These findings demonstrate that CP constitutes a competitive and robust alternative for solving the TOPNB.

These results are particularly relevant when considering the current operational context. Service orders are typically assigned by a supervisor based primarily on experience, while technicians independently determine the sequence in which tasks are executed. This decentralized and experience-driven process does not explicitly account for service and travel times, potentially leading to reduced productivity, higher operational costs, and an unbalanced distribution of workload among crews.

In this context, the proposed optimization models provide a structured and data-driven decision-support framework that improves current planning practices.

First, the models support the selection of the most valuable subset of service orders under limited operational capacity. By explicitly incorporating assignment

and routing decisions, they enable more efficient crew allocation, reduce idle time, better account for travel times, and improve overall resource utilization.

Second, the incorporation of a balancing objective ensures a more equitable distribution of workload and collected score among crews. This directly addresses one of the main limitations of the current process, contributing to fairer task allocation, improved workforce satisfaction, and more consistent operational performance.

Third, the availability of multiple solution approaches provides flexibility for implementation. In practice, decision-makers can select the most appropriate method depending on time constraints and required solution quality, enabling both high-quality planning and faster, near-real-time decision-making.

Finally, the results obtained in the case study confirm that the proposed approaches can be effectively applied in realistic operational settings. This demonstrates their potential to support both tactical planning and day-to-day scheduling decisions in telecommunications service operations.

7.2 Scientific Contributions

The scientific contributions derived from this research include three peer-reviewed journal articles (two published and one currently under review), six presentations at national and international scientific meetings, and active participation in the Sociedad Mexicana de Investigación de Operaciones (SMIO). The corresponding publications and presentations are listed below.

Peer-reviewed journal articles

- G. Sánchez-Yepez, M. A. Salazar-Aguilar, and P. J. Palomo-Martínez. Team orienteering problem with nonidentical agents and balanced score. *International Journal of Production Research*, 61(23):7957–7971, 2023. DOI: 10.1080/00207543.2022.2162146
- V. Boyer, G. Sánchez-Yepez, and M. Angélica Salazar-Aguilar. Branch and cut for the team orienteering problem with nonidentical agents and balanced score. *Expert Systems with Applications*, 307:131007, 2026. DOI: 10.1016/j.eswa.2025.131007
- "A constraint programming approach for telecommunications service planning" (under review).

Conference Presentations

- **2024** — "Modeling service planning with constraint programming," *Optimization Days*. Oral presentation.
- **2024** — "Exploring constraint programming for telecom service planning," *XII Congreso de la Sociedad Mexicana de Investigación de Operaciones – XII Congreso Latino Iberoamericano de Investigación Operativa*. Oral presentation.
- **2023** — "Optimizando la planificación de servicios de telecomunicaciones," *Seminario de la Maestría en Logística y Cadena de Suministro*. Invited talk.
- **2023** — "Service planning optimization with balanced wages," *XI Congreso de la Sociedad Mexicana de Investigación de Operaciones*. Oral presentation.
- **2022** — "A constraint programming approach for service planning optimization," *X Congreso de la Sociedad Mexicana de Investigación de Operaciones*. Oral presentation.
- **2022** — "Planificación de servicios: un caso de estudio," *I Taller en Ingeniería de la Decisión y sus Aplicaciones a la Ingeniería de Sistemas, UANL*. Poster presentation.

7.3 Future Research

The results presented in this thesis open several promising directions for future research, particularly in enhancing solution methodologies and extending the problem to more realistic settings. Some of the improvements and research directions derived from this work are as follows:

- **Enhancement of the CP model**: although the proposed approach demonstrates competitive performance, its efficiency could be further improved by adopting advanced search strategies. In particular, incorporating warm-start mechanisms that use high-quality solutions from heuristic methods, such as AMH, to initialize the CP solver is a promising direction.

- **Enhancement of the B&C separation procedure**: the current separation procedure uses an MST-based feasibility check to identify violated ICCs. Future research could explore exact TSP-based separation using efficient solvers such as Concorde and compare it with the proposed MST-based procedure. Such an analysis would allow the trade-off between the computational effort required by the separation procedure and its ability to identify violated ICCs to be evaluated.

- **Adaptation of the B&C approach**: the current B&C formulation is defined on an undirected graph and associates a single travel time with each edge. Ex-

tending the formulation to a directed graph and adapt the associated cuts used within the framework would allow asymmetric travel times to be represented. This extension would enable the B&C to be evaluated using network-based travel information, including instances that more closely reflect the characteristics of the real application and would facilitate comparisons with the other solution approaches under such settings.

- **TOPNB extension with time windows**: a natural extension of the TOPNB that consists of incorporating additional operational constraints, such as time windows. From a modeling perspective, this extension can be naturally integrated into the CP framework. Preliminary experiments suggest that the approach remains viable under this extension; however, challenges related to proving optimality persist and require further investigation.

- **Multi-objective extension of the TOPNB**: while the TOPNB focuses on balancing the collected score among crews, the classical Team Orienteering Problem emphasizes maximizing the total collected score. A natural extension is to consider both criteria simultaneously, leading to a multi-objective or bi-criteria formulation that captures the trade-off between efficiency (maximizing the total score) and equity (balanced workload distribution). This extension would provide a more flexible and realistic decision-making framework in practice.

- **Stochastic extensions of the TOPNB**: the models studied in this thesis consider deterministic travel and service times. In practice, some times these parameters could be stochastic due to traffic variability, weather conditions, and unforeseen field complications. Future research could therefore investigate extensions of the TOPNB in which travel times, service times, or both are uncertain or dynamically updated. This would allow the impact of such variability on assignment and routing decisions to be studied and would extend the problem toward more realistic operational settings.

Overall, these research directions aim to enhance the computational performance, practical applicability, and realism of the proposed approaches, while extending the TOPNB toward more complex operational settings that better reflect the conditions encountered in real-world service planning problem.

Bibliography

Y. Anoshkina and F. Meisel. Technician teaming and routing with service-, cost-and fairness-objectives. *Computers & Industrial Engineering*, 135:868–880, 2019. DOI: 10.1016/j.cie.2019.05.016.

C. Archetti, F. Carrabs, R. Cerulli, and F. Laureana. A new formulation and a branch-and-cut algorithm for the set orienteering problem. *European Journal of Operational Research*, 314(2):446–465, 2024. DOI: 10.1016/j.ejor.2023.09.038.

A. Bendazzoli, M. Urbani, M. Brunelli, and F. Pilati. A cooperative team orienteering optimisation model and a customised resolution metaheuristic. *Computers & Operations Research*, 163:106488, 2024. DOI: 10.1016/j.cor.2023.106488.

N. Bianchessi, R. Mansini, and M. G. Speranza. A branch-and-cut algorithm for the team orienteering problem. *International Transactions in Operational Research*, 25(2):627–635, 2018. DOI: 10.1111/itor.12422.

E. Bourreau, T. Garaix, M. Gondran, P. Lacomme, and N. Tchernev. A constraint-programming based decomposition method for the generalised workforce scheduling and routing problem (GWSRP). *International Journal of Production Research*, 60(4):1265–1283, 2022. DOI: 10.1080/00207543.2020.1856436.

V. Boyer, G. Sánchez-Yepez, and M. Angélica Salazar-Aguilar. Branch and cut for the team orienteering problem with nonidentical agents and balanced score. *Expert Systems with Applications*, 307:131007, 2026. DOI: 10.1016/j.eswa.2025.131007.

J. A. Castillo-Salazar, D. Landa-Silva, and R. Qu. Workforce scheduling and routing problems: literature survey and computational study. *Annals of Operations Research*, 239(1):39–67, 2016. DOI: 10.1007/s10479-014-1687-2.

S. Chakraborty and Y. K. Agarwal. Solving the team orienteering problem with nonidentical agents: A Lagrangian approach. *Networks*, 78(4):443–460, 2021. DOI: 10.1002/net.22023.

I.-M. Chao, B. L. Golden, and E. A. Wasil. The team orienteering problem. *European Journal of Operational Research*, 88:464–474, 1996. DOI: 10.1016/0377-2217(94)00289-4.

C. E. Cortés, M. Gendreau, L. M. Rousseau, S. Souyris, and A. Weintraub. Branch-and-price and constraint programming for solving a real-life technician dispatching problem. *European Journal of Operational Research*, 238(1):300–312, 2014. DOI: 10.1016/j.ejor.2014.03.006.

J. Decerle, O. Grunder, A. Hajjam El Hassani, and O. Barakat. A general model for the home health care routing and scheduling problem with route balancing. *IFAC-PapersOnLine*, 50(1):14662–14667, 2017. DOI: 10.1016/j.ifacol.2017.08.1907.

A. Errarhout, S. Kharraja, and C. Corbier. Two-stage stochastic assignment problem in the home health care. *IFAC-PapersOnLine*, 49(12):1152–1157, 2016. DOI: 10.1016/j.ifacol.2016.07.659.

A. Estrada-Moreno, A. Ferrer, A. A. Juan, J. Panadero, and A. Bagirov. The non-smooth and bi-objective team orienteering problem with soft constraints. *Mathematics*, 8(9):1461, 2020. DOI: 10.3390/math8091461.

M. Fischetti, J. J. S. González, and P. Toth. Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10(2):133–148, 1998. DOI: 10.1287/ijoc.10.2.133.

T. Garaix, M. Gondran, P. Lacomme, E. Mura, and N. Tchernev. Workforce scheduling linear programming formulation. *IFAC-PapersOnLine*, 51(11):264–269, 2018. DOI: 10.1016/j.ifacol.2018.08.289.

R. Gedik, E. Kirac, A. Bennet Milburn, and C. Rainwater. A constraint programming approach for the team orienteering problem with time windows. *Computers & Industrial Engineering*, 107:178–195, 2017. DOI: 10.1016/j.cie.2017.03.017.

M. Gendreau, G. Laporte, and F. Semet. A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32(4):263–273, 1998. DOI: 10.1002/(SICI)1097-0037(199812)32:4¡263::AID-NET3¿3.0.CO;2-Q.

A. Gobbi, D. Manerba, R. Mansini, and R. Zanotti. A kernel search for a patient satisfaction-oriented nurse routing problem with time-windows. *IFAC-PapersOnLine*, 52(13):1669–1674, 2019. DOI: 10.1016/j.ifacol.2019.11.440.

H. Gu, Y. Zhang, and Y. Zinder. An efficient optimisation procedure for the workforce scheduling and routing problem: Lagrangian relaxation and iterated local search. *Computers & Operations Research*, 144:105829, 2022. DOI: 10.1016/j.cor.2022.105829.

A. Guastalla, R. Aringhieri, and P. Hosteins. The team orienteering problem with service times and mandatory & incompatible nodes. *Computers & Operations Research*, 183:107170, 2025. DOI: 10.1016/j.cor.2025.107170.

A. Gunawan, H. C. Lau, and P. Vansteenwegen. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255(2):315–332, 2016. DOI: 10.1016/j.ejor.2016.04.059.

F. Hammami. An efficient hybrid adaptive large neighborhood search method for the capacitated team orienteering problem. *Expert Systems with Applications*, 249: 123561, 2024. DOI: 10.1016/j.eswa.2024.123561.

W. Hu, M. Fathi, and P. M. Pardalos. A multi-objective evolutionary algorithm based on decomposition and constraint programming for the multi-objective team orienteering problem with time windows. *Applied Soft Computing*, 73:383–393, 2018. DOI: 10.1016/j.asoc.2018.08.026.

H.-B. Jung, H.-I. Kim, and D.-H. Lee. Team orienteering with possible multiple visits: Mathematical model and solution algorithms. *Computers & Industrial Engineering*, 190:110097, 2024. DOI: 10.1016/j.cie.2024.110097.

Ö. Karsu and A. Morton. Inequity averse optimization in operational research. *European Journal of Operational Research*, 245(2):343–359, 2015. DOI: 10.1016/j.ejor.2015.02.035.

E. Kirac, R. Gedik, and F. Oztanriseven. Solving the team orienteering problem with time windows and mandatory visits using a constraint programming approach. *International Journal of Operational Research*, 46(1):20–42, 2023. DOI: 10.1504/IJOR.2023.128542.

G. Kobeaga, J. Rojas-Delgado, M. Merino, and J. A. Lozano. A revisited branch-and-cut algorithm for large-scale orienteering problems. *European Journal of Operational Research*, 313(1):44–68, 2024. DOI: 10.1016/j.ejor.2023.07.034.

P. Laborie, J. Rogerie, P. Shaw, and P. Vilím. IBM ILOG CP optimizer for scheduling: 20+ years of scheduling with constraints at IBM/ILOG. *Constraints*, 23: 210–250, 2018. DOI: 10.1007/s10601-018-9281-x.

F. Lehuédé, O. Péton, and F. Tricoire. A lexicographic minimax approach to the vehicle routing problem with route balancing. *European Journal of Operational Research*, 282(1):129–147, 2020. DOI: 10.1016/j.ejor.2019.09.010.

J. Li, J. Zhu, G. Peng, J. Wang, L. Zhen, and E. Demeulemeester. Branch-price-and-cut algorithms for the team orienteering problem with interval-varying profits.

European Journal of Operational Research, 319(3):793–807, 2024. DOI: 10.1016/j.ejor.2024.07.015.

X. Liu, X. Jiang, X. Luo, Z. Zhang, and P. Ji. Exact and heuristic algorithms for team orienteering problem with fuzzy travel times. *Expert Systems with Applications*, 278:127369, 2025. DOI: 10.1016/j.eswa.2025.127369.

P. Matl, R. F. Hartl, and T. Vidal. Workload equity in vehicle routing problems: A survey and analysis. *Transportation Science*, 52(2):239–260, 2018. DOI: 10.1287/trsc.2017.0744.

P. Matl, R. F. Hartl, and T. Vidal. Workload equity in vehicle routing: The impact of alternative workload resources. *Computers & Operations Research*, 110:116–129, 2019. DOI: 10.1016/j.cor.2019.05.016.

M. H. Mirzaei, K. Ziarati, and M.-T. Naghibi. Bi-objective version of team orienteering problem (BTOP). In *2017 7th International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 1–7, Mashhad, Iran, 2017. IEEE. DOI: 10.1007/s40092-019-0315-9.

R. Montemanni and M. Dell'Amico. Constraint programming models for the parallel drone scheduling vehicle routing problem. *EURO Journal on Computational Optimization*, 11:100078, 2023. DOI: 10.1016/j.ejco.2023.100078.

N. Nekooghadirli, M. Gendreau, J.-Y. Potvin, and T. Vidal. Workload equity in multiperiod vehicle routing problems. *International Transactions in Operational Research*, 33(1):245–267, 2026. DOI: 10.1111/itor.70012.

D. C. Paraskevopoulos, G. Laporte, P. P. Repoussis, and C. D. Tarantilis. Resource constrained routing and scheduling: Review and research prospects. *European Journal of Operational Research*, 263(3):737–754, 2017. DOI: 10.1016/j.ejor.2017.05.035.

E. Pekel. Examination of daily task constraint in technician routing and scheduling problem. *Avrupa Bilim ve Teknoloji Dergisi*, (28):207–212, 2021. DOI: 10.31590/ejosat.995415.

H. G. Pham, T. Mai, and M. H. Hà. Constraint programming-based cutting plane algorithm for a combination of orienteering and maximum capture problem. In *Information and Communication Technology: 13th International Symposium, SOICT 2024*, volume 2350 of *Communications in Computer and Information Science*, pages 354–368, Singapore, 2024. Springer. DOI: 10.1007/978-981-96-4282-3_29.

D. Prot, T. Lapègue, and O. Bellenguez-Morineau. A two-phase method for the shift design and personnel task scheduling problem with equity objective. *International Journal of Production Research*, 53(24):7286–7298, 2015. DOI: 10.1080/00207543.2015.1037023.

B. Rabbouch, F. Saâdaoui, and R. Mraihi. Constraint programming based algorithm for solving large-scale vehicle routing problems. In H. Pérez García, L. Sánchez González, M. Castejón Limas, H. Quintián Pardo, and E. Corchado Rodríguez, editors, *Hybrid Artificial Intelligent Systems*, pages 526–539, Cham, Switzerland, 2019. Springer. ISBN 978-3-030-29859-3. DOI: 10.1007/978-3-030-29859-3_45.

M. S. Rasmussen, T. Justesen, A. Dohn, and J. Larsen. The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies. *European Journal of Operational Research*, 219(3):598–610, 2012. DOI: 10.1016/j.ejor.2011.10.048.

G. Sánchez-Yepez. *Optimización de la Planificación de Servicios*. Masters thesis, Universidad Autónoma de Nuevo León, San Nicolas de los Garza, Mexico, September 2019. Available at: `http://eprints.uanl.mx/17848/1/1080288722.pdf`.

G. Sánchez-Yepez, M. A. Salazar-Aguilar, and P. J. Palomo-Martínez. Team orienteering problem with nonidentical agents and balanced score. *International Journal of Production Research*, 61(23):7957–7971, 2023. DOI: 10.1080/00207543.2022.2162146.

M. G. Speranza, C. Archetti, and D. Vigo. Vehicle routing problems with profits. In *Vehicle Routing: Problems, Methods, and Applications*, chapter 10, pages 273–297. SIAM, Philadelphia, USA, 2nd edition, November 2014. DOI: 10.1137/1.9781611973594.ch10.

K. Sundar, S. Sanjeevi, and C. Montez. A branch-and-price algorithm for a team orienteering problem with fixed-wing drones. *EURO Journal on Transportation and Logistics*, 11:100070, 2022. DOI: 10.1016/j.ejtl.2021.100070.

H. Tang and E. Miller-Hooks. A tabu search heuristic for the team orienteering problem. *Computers & Operations Research*, 32(6):1379–1407, 2005. DOI: 10.1016/j.cor.2003.11.008.

P. Vansteenwegen and A. Gunawan. *Orienteering Problems: Models and Algorithms for Vehicle Routing Problems with Profits*. EURO Advanced Tutorials on Operational Research. Springer, Cham, Switzerland, 2019. ISBN 978-3-030-29745-9. DOI: 10.1007/978-3-030-29746-6.

P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10, 2011. DOI: 10.1016/j.ejor.2010.03.045.

H. Wang, B. Alidaee, J. Ortiz, and W. Wang. The multi-skilled multi-period workforce assignment problem. *International Journal of Production Research*, 59(18): 5477–5494, 2021. DOI: 10.1080/00207543.2020.1783009.

A.-E. Yahiaoui, A. Moukrim, and M. Serairi. Grasp-ils and set cover hybrid heuristic for the synchronized team orienteering problem with time windows. *International Transactions in Operational Research*, 30(2):946–969, 2023. DOI: 10.1111/itor.13111.

V. F. Yu, N. Y. Salsabila, S.-W. Lin, and A. Gunawan. Simulated annealing with reinforcement learning for the set team orienteering problem with time windows. *Expert Systems with Applications*, 238:121996, 2024. DOI: 10.1016/j.eswa.2023.121996.

AUTOBIOGRAPHY

Gabriela Sánchez Yepez

Ph.D. Candidate in Systems Engineering
Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica

Thesis:
Optimization models and algorithms for service planning

I was born on March 1994.

In 2017, I received a B.S. degree in Physical and Mathematical Sciences from the Universidad Michoacana de San Nicolás de Hidalgo. Two years later, I got my Master's degree in Systems Engineering from Universidad Autónoma de Nuevo León. Complementing my masters education, I did a research stay at the Applied Economy Department in the University of Malaga, Spain.

In October of 2021, I received an award for the second best master's thesis granted by the Mexican Society of Operations Research (Sociedad Mexicana de Investigación de Operaciones).

In October of 2023, I earned the Sofía Kovalevskaya award, granted by the Sofía Kovalevskaya Foundation and the Mexican Mathematical Society (Sociedad Matemática Mexicana).