FERNANDO AUGUSTO MARTIN FERRI

OPTIMIZATION TECHNIQUES APPLIED TO A CONSTRUCTION SECTOR PROBLEM:

"MULTI-TRIP PICKUP AND DELIVERY PROBLEM, WITH SPLIT LOADS, PROFITS AND MULTIPLE TIME WINDOWS"

São Paulo

2021

FERNANDO AUGUSTO MARTIN FERRI

OPTIMIZATION TECHNIQUES APPLIED TO A CONSTRUCTION SECTOR PROBLEM:

"MULTI-TRIP PICKUP AND DELIVERY PROBLEM, WITH SPLIT LOADS, PROFITS AND MULTIPLE TIME WINDOWS"

Trabalho de formatura apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Diploma de Engenheiro de Produção.

São Paulo

2021

FERNANDO AUGUSTO MARTIN FERRI

OPTIMIZATION TECHNIQUES APPLIED TO A CONSTRUCTION SECTOR PROBLEM:

"MULTI-TRIP PICKUP AND DELIVERY PROBLEM, WITH SPLIT LOADS, PROFITS AND MULTIPLE TIME WINDOWS"

Trabalho de formatura apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Diploma de Engenheiro de Produção.

Orientadores:

Débora Pretti Ronconi (Poli-USP)

Wahiba Ramdane Cherif-Khetaff (Mines Nancy)

Atef Jaballah (Mines Nancy)

São Paulo

2021

FICHA CATALOGRÁFICA

Ferri, Fernando Augusto Martin

Optimization techniques applied to a contruction sector problem: “Multi-trip Pickup and Delivery Problem, with Split loads, Profits and Multiple Time Windows” / F. A. M. Ferri. – São Paulo, 2021

110 p.

Trabalho de Formatura – Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Produção.

1. Vehicle Routing Problem 2. Pickup and Delivery Problem

3. Multiple Time Windows 4. Construction Sector

I. Ferri, Fernando Augusto Martin II. Universidade de São Paulo

Escola Politécnica. Departamento de Engenharia de Produção

III. Título

RESUMO

Este trabalho segue o estudo desenvolvido por Ramdane e Jaballah (2021), orientadores do Trabalho Final iniciado na Mines Nancy e que teve continuidade na Poli-USP, viabilizado pela parceria de duplo diploma entre as duas escolas de Engenharia. Inspirado por um problema do mundo real, o estudo foi levantado por uma parceria público-privada no âmbito de um projeto francês de P&D denominado DILC (Demonstrador de Inovações em Logística para Construções). O projeto foi idealizado por um grupo de organizações da Lorraine, região do nordeste da França, composto por associações locais, empresas, órgãos públicos e laboratórios de pesquisa. Seu objetivo é a criação de uma plataforma logística de partilha (*pooling platform*). Em termos gerais, ela centraliza a entrega de materiais de construção e a coleta de resíduos de canteiros de obras. A plataforma utiliza uma frota limitada e heterogênea que realiza múltiplas viagens sob restrições de tempo e limitação de capacidade. O problema em estudo, denominado *Multi-Trip Pickup and Delivery Problem, with Split Load, Profits and Multiple Time Windows* (MTPDPSPMTW) é uma nova extensão do problema de roteamento de veículos (*vehicle routing problems*) com coleta e entrega (*pickup and delivery*) que considera restrições realistas para a indústria civil. Por exemplo, cada canteiro de obras possui uma prioridade na sua demanda de entrega e/ou coleta, que é medida através de um lucro trazido para a plataforma logística por realizar este serviço. Além disso, cada canteiro de obras pode ter várias janelas temporais nas quais os veículos podem comparecer para realizar seu serviço de coleta/entrega. Para resolver este problema, Ramdane e Jaballah (2021) desenvolveram uma heurística construtiva. O objetivo do presente trabalho foi resolver o mesmo problema usando um modelo exato por meio da programação linear inteira mista. Assim, através de abordagens diferentes, foi possível avaliar resultados e comparar a relevância das soluções. Este trabalho apresenta uma introdução onde foi descrito o contexto do problema e sua definição por meio de restrições conhecidas da literatura. Em seguida, foi realizada uma revisão bibliográfica com o objetivo de verificar outros artigos que pudessem auxiliar no processo de modelagem. Posteriormente, foi definido o modelo de programação linear inteira mista, bem como a metodologia da coleta de dados reais. Por fim, os resultados da simulação e uma conclusão completaram o trabalho.

Palavras-chave: Problema de roteamento de veículos. Problema de coleta e entrega. Múltiplas janelas de tempo. Indústria civil**.**

ABSTRACT

This work follows the study developed by Ramdane and Jaballah (2021), supervisors of the capstone project that began at Mines Nancy and that was continued at Poli-USP, made possible through the double degree partnership between both Engineering schools. Their research promoted the first optimization study of multi-site transportation in the construction industry. It allows mutualizing delivery building materials and construction waste removal. Inspired by a real-world problem, the study was raised by a public private partnership in the framework of a French R&D project called DILC (Logistics Innovations Demonstrator in Construction sites). A group of organizations from Lorraine, a northeastern French region, composed by local associations, companies, public agencies and research laboratories were involved to design a pooling logistic platform. In general terms, it centralizes the delivery of building materials and the pickup of construction sites' waste. The platform uses a limited and heterogeneous fleet that performs multiple trips, under time and capacity limitation constraints. The problem under study, called Multi-Trip Pickup and Delivery Problem, with Split loads, Profits and Multiple Time Windows (MTPDPSPMTW) is a new extension of the vehicle routing problem with pickup and delivery that considers realistic constraints aiming the construction industry. For instance, each construction site may have a priority on its delivery and/or its pickup demand, that is measured as a profit brought to the logistic platform. Moreover, each construction site may have several time windows in which vehicles could attend to perform their pickup/delivery service. To solve this problem, Ramdane and Jaballah (2021) developed a constructive heuristic. The present work's goal was to solve the same problem through an exact model by means of mixed integer linear programming. So, through a different approach, it was possible to assess the results and compare their relevance. This work presents an introduction where it was described the context of the problem and its definition by known constraints from the literature. Then, a bibliographic review was performed in order to check for other articles that could help the modelling process. Afterwards, the mixed integer linear programming model was defined, as well as its data collection methodology. Finally, the results of the simulation and a conclusion completed the work.

Key words: Vehicle Routing Problem. Pickup and Delivery Problem. Multiple Time Windows. Construction Sector**.**

LIST OF FIGURES

LIST OF TABLES

SUMMARY

1. INTRODUCTION

This work's aim is to present what has been developed by the student Fernando Augusto Martin Ferri as part of the capstone project in the last year of the Engineering training at Mines Nancy (France) in the Department of Industrial Engineering and Applied Mathematics (Decision and Production Systems Engineering track), and at Escola Politécnica da Universidade de São Paulo (Brazil) in the Department of Production Engineering. The project consists in the mathematical modeling and resolution by solver of a real transport problem from the construction sector, called "Multi-trip pickup and delivery problem, with split loads, profits and multiple time windows". The supervision was done by Wahiba Ramdane Cherif-Khettaf and Atef Jaballah, researchers from Mines Nancy who originally carry out the research on this subject, and also by Débora Pretti Ronconi, researcher from Poli-USP.

To this end, the subject will be presented and explained in the introduction through the work context and the problem definition. Then there will be a literature review, where a state-of-the-art was carried out on the relevant problems addressed by researchers nowadays. Then the mathematical modeling will be described, where this problem will be formalized through a mixed integer linear programming (MILP) formulation. Afterwards, the data collection methodology will be explained, in which instances with actual parameters were constructed. Then, a simulation of this model will be shown using *Xpress IVE*, optimization software. Finally, the conclusion will address the results of this project, outlining what could have been done differently, with suggestions for model changes and next steps.

1.1. Work context

This work's subject was created under the *Démonstrateur Innovations Logistiques Chantiers – DILC* project (Logistics innovations demonstrator in construction sites), funded by the French *Agence de la transition écologique – ADEME* (Ecological Transition Agency). This project general goal is to develop a tool to optimize the delivery of construction materials and the collection of construction waste, which would allow energy recovery of waste from construction sites in Lorraine, a French region. This would be achieved by the creation of a pooling logistic platform that gathers different raw materials for the construction industry. It allows the pooling of inbound and outbound logistics for construction sites as well as storage of waste and materials. This modular and transportable platform would be adaptable to any size of operation and it would be reused for other purposes, depending on the needs of the territory concerned.

This Research and Development (R&D) program carry social and circular economy values, since it targets the reduction of the emission of greenhouse effect gases through the logistics optimization. In that manner, the building companies would contribute to the economy in a more circular and inclusive way. The DILC aims to optimize the inbound (materials), outbound (wastes) of construction sites, as well as internal logistic of ecocity the building programs. By means of the industrial and territorial ecology process, this target will be met by the flow pooling through concentration of the flow platform. The ultimate goal of the project is to improve the rate of recycling wastes from the construction site through inclusive devices designed for non-qualified workforce and with limited financial impact for the building companies.

In summary, the DILC designed an innovative logistic pooling platform which includes modular, removable and transportable concentration of the materials' flow; as well as digital tools for the logistic management and planning; and finally, a range of services for the companies working on the construction site. Firstly, the project will be managed in the regional context of Lorraine and then in the country level with the goal of spreading it specially for ecocity and eco-district development operations.

When it comes to sustainability, the project can be related to the Triple Bottom Line, which refers to economic, environmental, and social value of an investment and is related to the concept of sustainable development (HAMMER; PIVO, 2016). In the economical scope, the DILC aims to reduce waste management costs. In the social extent, the construction of the logistic platform would create jobs and inclusion. And finally, in the environmental scope, it would reduce greenhouse gases emissions and the pressure on resources.

To carry out this work, a group of multidisciplinary actors was formed. This public-private partnership finds all its relevance in the complementarity of the skills associated with it. The main institutions/companies involved in this project can be found in the list below:

- *PTCE Florange e2i*: territorial poles of economic cooperation, specialized in the coordination of industrial and territorial ecology projects
- *ECOTA Conseil:* design office, expert in the field of R&D in industrial ecology
- *LORIA*: laboratory specialized in fundamental and applied research in computer sciences
- *GEORESSOURCES*: laboratory specialized in the engineering of decision and production systems.
- *CCPHVA*: local community collective that supports innovative projects.
- EPA Alzette-Belval: public planning establishment, that promotes Alzette-Belval[1] national interest operation projects, such as an eco-city that would need an innovative logistic platform for new construction sites.

Figure 1: Project partners

Source: DILC reports

Each organization had a role in the project. The following table presents the main task that they performed on DILC:

[1] Border area between France and Luxembourg that encompasses the river Alzette valley. It includes the French city of Audun-le-Tiche and the Luxemburgish city of Belval.

Table 1: Institutions of the project and their roles

Institution	Main role in the project
CCPHVA	Searching for an experimentation field/area
ECOTA	Identification of pooling potential Technical and economic feasibility analysis Life cycle analysis and scenarios choosing
EPA Alzette-Belval	Incentive measures to get construction companies to join the system
GEORESSOURCES	Physical platform design
LORIA	Software platform design

Source: DILC reports

The present work is associated to *LORIA - Laboratoire lorrain de recherche en informatique et ses applications* (Lorrain Laboratory for research in computer science and its applications). This laboratory is part of University of Lorraine, that encompasses the Engineering Mines Nancy School. The supervisors of this work, Ramdane and Jaballah, are involved in projects of LORIA and Mines Nancy as well. Therefore, their role in the project was to develop optimization algorithms to the pooling platform concerning the logistics between construction sites.

Besides creating a physical mutualization platform, the project targets the creation of a software platform to perform the operation and management of materials. Within this context, there is the logistics flow optimization, which allowed the conception of the present work.

There are 20 construction companies involved in this project. In order to design the platform according to their demands, interviews were performed with each one of them. In that way, their needs concerning raw materials delivery and waste collection were translated to project requirements, whether to the physical or software platform.

Therefore, 6 modules were designed for the software platform, addressing different functions. The following table describes each one of them:

Table 2: Modules' description of the DILC software platform

Module	Role
1. Mobile app for construction site managers to order services	• Just-in-time waste rotations • Deliveries of materials and equipment to apartment doors • Waste containers for pickup at sites • Cleanliness of life bases, cleaning of construction sites and security • Delegation of temporary staff
2. Regulatory reporting on waste management	• Waste tracking forms • Waste registers by site • Recycling certificate • Reporting on the waste treatment method
3. Corporate Social Responsability (CSR) reporting	Measures the positive impacts generated by site.
4. Engineering for sorting at source	Helps companies organize sorting at source and publish customizable awareness materials (labels, procedures, templates)
5. Material exchanges	Facilitate the reuse of outgoing materials of reusable sites.
6. Optimization of flows and resources	Making it possible to optimize storage, inbound and outbound logistics for construction sites, as well as manpower.

Source: DILC reports

This work addresses the sixth module, where optimization techniques were applied to leverage operations involving inbound and outbound flow between construction sites and the logistic platform. Hence, a heuristic was made by Ramdane and Jaballah (2021) following all constraints and requirements of this project. They used as instances actual data provided by DILC partners. Even though the logistics platform was still being designed, they came up with real values of which distances they would need to deal with, how many pallets of pickup and delivery demands they would have, and so on. After concluding the heuristics, Ramdane and Jaballah (2021) wanted to test their solutions with a linear-programming solver solution. In order to do so, they would need to model the problem mathematically, solving it and then comparing the results to check if the heuristics has performed properly.

Thus, the following work has the objectives described in the past paragraph, as follows:

- Comprehend the real case scenario and all its requirements;

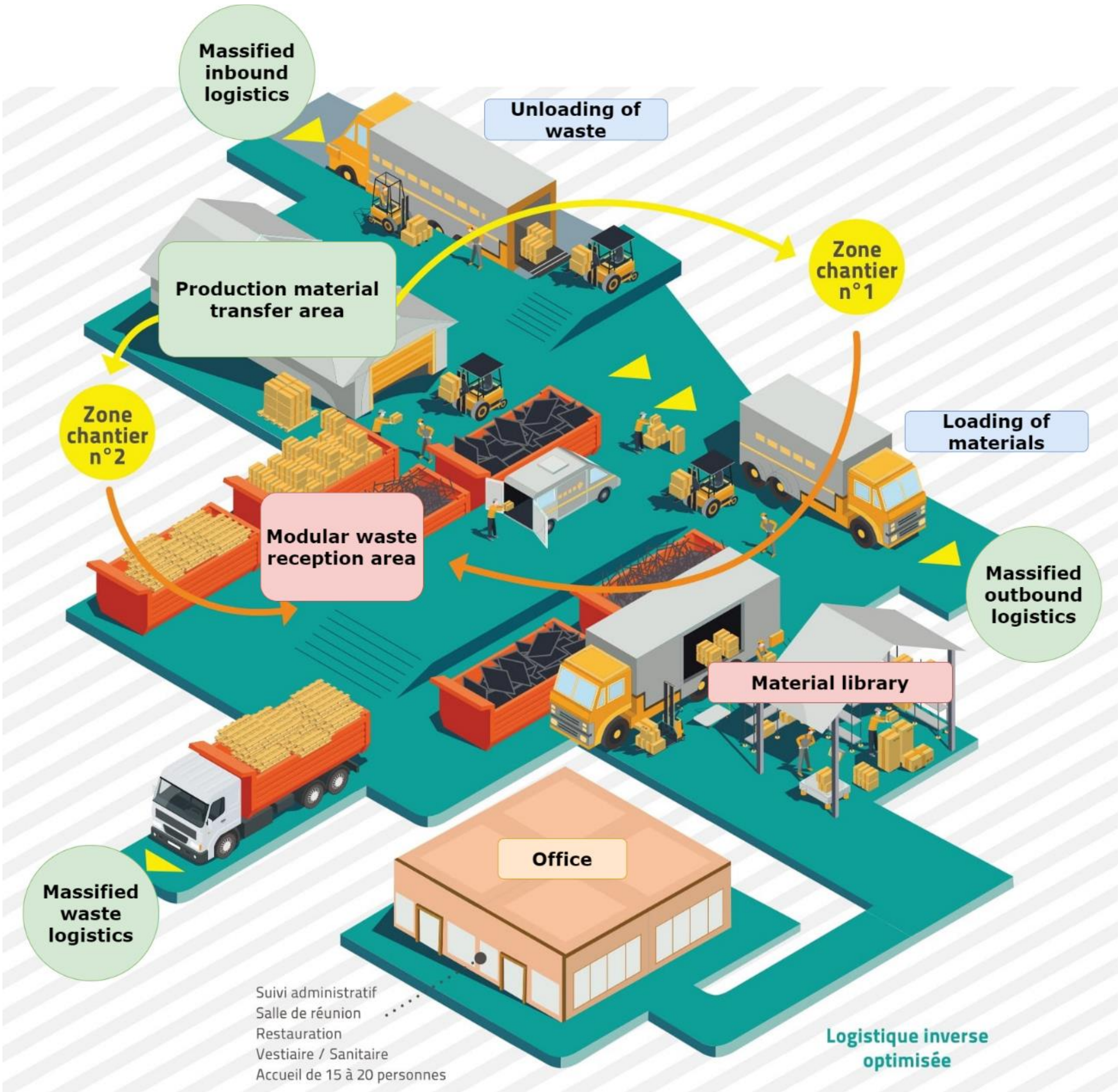
- Transform project needs into a mathematical model by mixed integer linear programming;
- Collect data with actual parameters of the problem;
- Solve the problem by an optimization software;
- Compare results with the constructive heuristics previous developed by Ramdane and Jaballah (2021)

Hence, the role of this work is to validate the heuristics designed to leverage the pooling platform logistics flow of waste and materials.

With the purpose of better describing the problem, the physical platform will now be addressed. The software platform exists to manage and operate the physical structures of the pooling platform. The platform makes it possible to pool the logistics entering and leaving the sites as well as the storage (waste, materials). This modular and transportable platform can adapt to any size of operation and be reused after construction sites are done, according to the needs of the territory concerned.

The following diagram illustrates the physical platform:

Figure 2: Pooling platform illustration diagram



Source: DILC reports – adapted

The platform encompasses zones that perform different roles, as follows:

- Office: area responsible for administrative follow, composed of meeting rooms, catering, cloakroom, sanitary.
- Unloading of waste area: massified inbound logistics, where the reception of big-bag waste from construction sites happens.
- Modular waste reception area: waste sorting zone, where waste is separated into categories such as recycling, reusable, or disposable waste.

- Material library: place where recycling or reusable waste are recovered and sorted, so it could be resaled (mostly at solidarity prices). By doing this waste energy recovering, the pooling platform operation avoids them to go directly to landfills.
- Production material transfer area: zone of reception and sorting of raw materials into palletized kits.
- Loading of materials: massified outbound logistics, where the delivery of raw material kits to construction sites starts.

The DILC project has been designed inside a bigger project. Since 2014, there is a conception of an Eco-Industrial Park (EIP) in the north of Lorraine. An EIP is a park of industries that collaborate by reusing waste and by the energy-efficient use of resources with less impact on the environment (AMATO et al., 2018). Therefore, the industries are connected in closed loops through the reuse and recycling of materials and waste. Then, the pooling platform of materials and waste for construction sites are integrated within an EIP.

There are 7 initiatives for this EIP in northern Lorraine. The "New Flow" initiative encompasses the operation of this work's pooling platform. That is to say that the DILC is the project that envisages the implementation of the New Flow platform.

In total, the EIP counts with 523 institutions, recycling/reusing 4701 tons of waste, saving 1319 tons of CO_2 (FE2i, 2020). To illustrate one operating initiative, the following figure presents a center of hard plastic windows recycling:

Figure 3: Example of one EIP initiative

Source: FE2i website

To conclude, there is a territory that is mapped to be used for the construction of the physical pooling platform, that is shown in the following image:

Figure 4: Experimental ground which may happen the construction of the pooling platform

Source: DILC reports

1.2. Problem definition

The construction sector is a key driver of the economy. According to INSEE, this sector obtained an investment of € 210.1 billion in 2016 in France (INSEE, 2018). A large portion of this budget is dedicated to transportation, as the delivery of construction materials and waste collection at the sites. The amount of materials consumed by this industry is very large and generates a large quantity of waste, resulting in high transportation needs, significant social costs and a whole series of negative impacts on the environment. In order to improve costs and reduce these impacts, the DILC project proposes to find innovative concepts of flow massing platform.

A logistic platform can be defined as a specialized area with the infrastructure and services required for co-modal transportation and added value services, where different agents coordinate their activities to benefit the competitiveness of the products making use of the infrastructure (LEAL, 2009). The kind of logistics platform proposed by the project centralizes the delivery of construction materials and the removal of waste from construction sites, using a limited and heterogeneous fleet of vehicles. This type of logistics problem is addressed on the literature as a “Pickup and delivery problem”, which is a kind of “Vehicle Routing Problem” (VRP).

The transport model resulting from this study is a new extension of the problems of vehicle trips with split collection and delivery, which takes into account new realistic constraints specific to the construction sector. These additional constraints have not been taken into account in the models and algorithms of the literature, which is going to be addressed later on this work.

Unlike direct transport from suppliers to construction sites, the aim of the pooling platform is to bring together numerous delivery equipment from different suppliers, by receiving them in pallets according to a schedule corresponding to the progress of construction works on sites. From the construction materials received, ready-to-use kits are prepared on the platform, stored and delivered in pallets to construction sites. The kit represents the supply unit, that is, a kit must be delivered in full. It is not possible to split the kit into several deliveries. Each kit is characterized by its ID, the number of pallets it contains and its weight (RAMDANE; JABALLAH, 2021).

The delivery demand of a construction site may involve different kits, and the quantity of materials required for delivery sometimes exceed the truck’s capacity, which requires providing the site several times. So, the delivery demand division is allowed simply put, which is called “Split loads constraint”. The quantities of waste are relatively smaller than deliveries but can also be distributed in different vehicles (RAMDANE; JABALLAH, 2021).

The logistic platform must also manage the removal of waste from the construction sites to the platform. It is noted that there are two types of waste: big-bag waste and dumpster waste. Big-bag waste is packed on pallets and concerns small and medium quantities of waste such as soft plastic, hard plastic and cardboard. Tipper/dumpster waste refers to waste produced in large quantities such as wood and metals. The focus of this work is directed only on big-bag waste because its removal can be shared with the delivery of construction materials using a limited and heterogeneous lift truck fleet, whose capacity is expressed in pallets (RAMDANE; JABALLAH, 2021).

The vehicles can make multiple trips between the platform and the construction sites. The vehicles are loaded of kits in the platform, then they make deliveries to the construction sites, collecting also their waste. Finally, at the end of a trip, the vehicles dump the waste into the recycling center located on the logistic platform. So, the “Multi-trip constraint” consists in the fact that each vehicle can make several trips from the warehouse (logistic platform) during its working day (RAMDANE; JABALLAH, 2021).

It should also be noted that operational planning within the platform is not relevant to this study. The problem addressed here is the optimization of the trips between the platform and the construction sites to combine the delivery of equipment and the removal of waste, with specific constraints (RAMDANE; JABALLAH, 2021).

These constraints include also the so-called “Profit constraint”, where each construction site is assigned to a certain priority over its delivery demand or its removal demand (or both) thanks to the profit that this transport would bring to the platform. This comes along with a non-mandatory meeting of demands, that is to say that the model will choose which job sites are more interesting to attend in terms of the profit to the pooling platform. By doing it, this work assures the outlook from the pooling platform, in which it will be optimized its profit, the trips of its vehicle fleet, etc. (RAMDANE; JABALLAH, 2021).

Moreover, there is the constraint called “Multiple time windows”, meaning that each construction site may have several time windows during the work day reserved for receiving materials or loading waste. An import remark is the fact that vehicles should not wait for time windows to open in order to perform a service. That is to say that a vehicle cannot arrive at a job site before the opening of its time window service. This problem constraint was defined because in the real case scenario, companies expressed that there is not enough physical space for vehicles to park near construction sites so they could wait for a time window to open and perform its transport service. Nonetheless, vehicles can have a waiting time between two trips, since there is enough space for them to stay at the depot. That can be advantageous given that time windows are usually spaced out during the day. For instance, it is possible to have three time windows, being one of them in the morning (before jobs sites begin their working day), then another at noon (during lunch time), and the last one at night (after the working day is done). So, time windows can be very spaced from each other, and it could be interesting for vehicles to stay waiting at the depot between trips so they could serve more construction sites (RAMDANE; JABALLAH, 2021).

The following table summarizes the definitions previously explained:

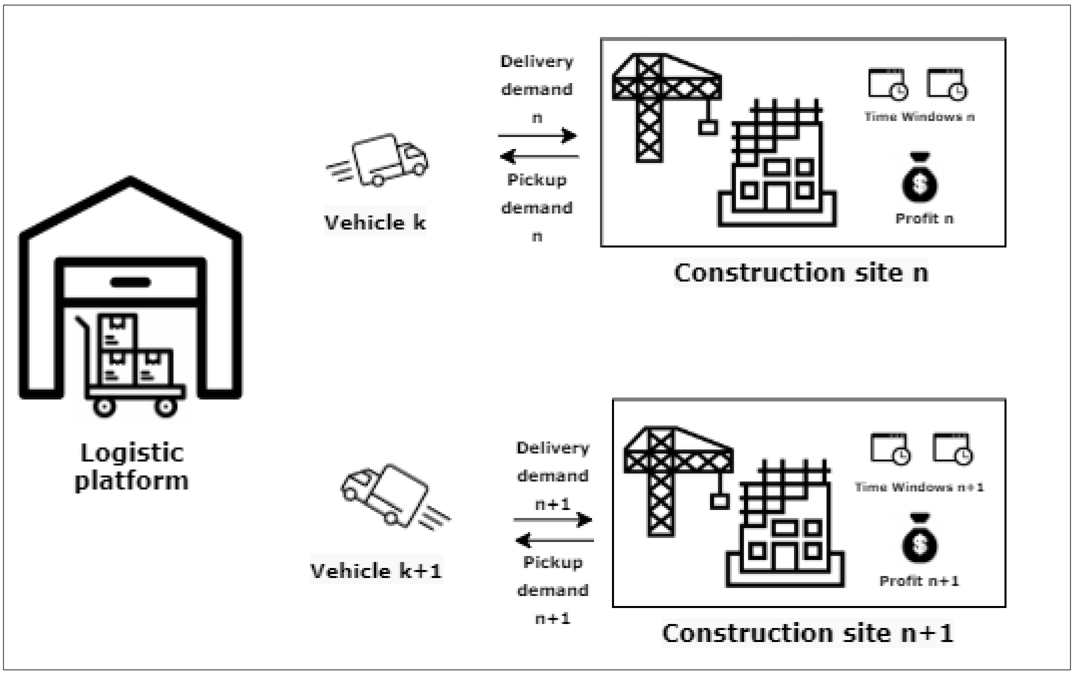
Table 3: Summary of the problem and its constraints

Title	Definition
Vehicle routing problem (VRP)	Procedure for finding the optimal solution for vehicles serving a set of customers by minimizing the total cost of travel
Pickup and delivery	VRP where customers have two different demands: a delivery and a pickup one. For this study, it is the delivery of raw materials and the collection of waste
Multi-trip	The problem allows a vehicle to make multiple trips (customer drop off) during its working day
Split loads	The problem allows the demand of a job site to be divided in several vehicles if we respect the non-division of kits/big-bags (whose supply unit is composed of a pallet)
Profits	Each demand is composed of a respective profit. The mathematical model will allow a preference for customers where profit is greater, since not all demands will be met
Multiple time windows	Each customer will have multiple time windows where they can receive materials and dispose waste

Source: author

The following diagram illustrates the problem and its constraints:

Figure 5: Diagram illustrating the problem



Source: author

Even though the figure contributes to understanding the problem, there are some constraints that are not so explicit. It is possible to reinforce that multiple trips can be carried out by vehicles and also it is possible to perform the splitting of delivery and pickup demands.

Hence, the problem entitled "Multi-trip pickup and delivery problem, with split loads, profits and multiple time windows" (MTPDPSPMTW) deals with a real case scenario of the construction sector brought by French construction companies and the Ecological Transition Agency. This study wants to generally minimize transportation costs and greenhouse gas emissions, maximizing likewise the profits of the logistic platform involved. To this end, it presents a state-of-the-art of the literature on the subject, showing then a way to mathematically model it, as well as a simulation on an optimization software.

2. LITERATURE REVIEW

To find what has already been developed in the literature, a bibliographical study was carried out. The aim was to find other VRP-type problems with similar constraints, and draw inspiration from them to carry out the modelling and simulation. In addition, the goal was also to show the relevance of this study since there are no other articles addressing this specific problem.

Data from the literature review were collected from *Web of Science* and *Scopus*, two bibliographic databases which cover all scientific fields. Moreover, the state-of-the-art has also relied on pre-selected articles by Ramdane and Jaballah (2021), supervisors of this project, composing an important base of articles.

A number of studies have been conducted to address similar subjects. Initially, the results obtained during the research were analyzed to verify the correlation with the main theme. In other words, the summary of all research papers found on the bibliographic databases was read in order to perform a data screening. In addition, refinements were used for each database to advance the process of verifying the link to the desired domain.

Thus, the articles were extracted using as research topics "Vehicle routing problem", "Pickup and delivery", "Time windows" and "Split loads", which compose the most common constraints of this study. Indeed, the authors, title, source and abstract were extracted from each relevant article. Although refinements were used, it was still necessary to read each abstract to verify the consistency of the article with the theme.

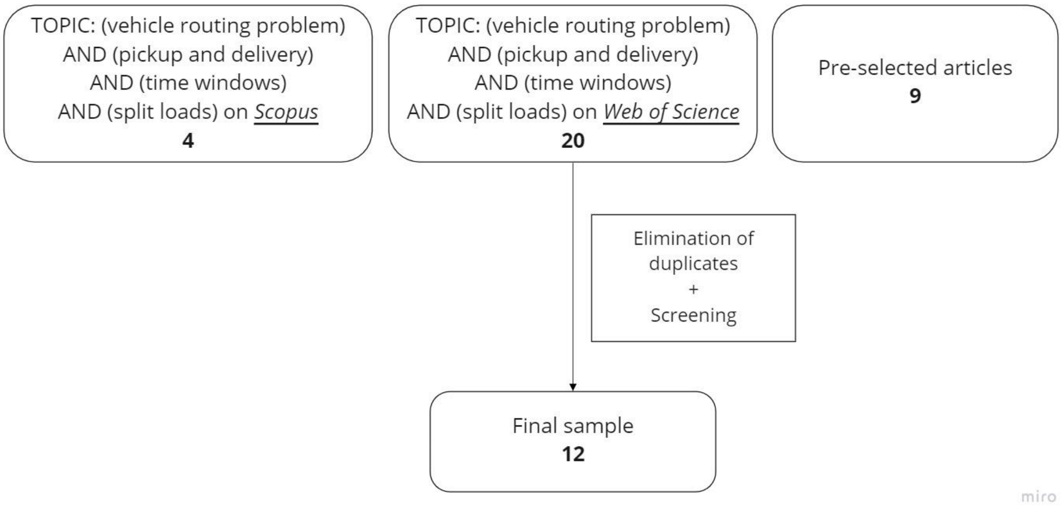
The criteria used to perform the systematic literature review in order to screen the articles found was based on the following standards:

- Relation with the theme (VRP approach)
- Presenting a mathematical model
- Similarity of constraints

However, this sort of items did not include checking duplicates between databases. It was then decided to eliminate the articles found in *Scopus* and *Web of Science* that were already in the database of articles from the previous research from Ramdane and Jaballah (2021). Additionally, there were other articles suggested by my Poli-USP's supervisor, Débora Pretti Ronconi.

The following figure presents the number of articles of each sample found along the data screening:

Figure 6: Search terms, number of articles found with data screening



Source: author

The following table presents the set of articles found:

Table 4: Set of articles found in the literature review

Author	Title
Nowak, 2005	The Pickup and Delivery Problem with Split Loads
Wang, 2013	Vehicle Routing Problem: Simultaneous Deliveries and Pickups with Split Loads and Time Windows
Wassan and Nagy, 2014	Vehicle Routing Problem with Deliveries and Pickups: Modelling Issues and Meta-heuristics Solution Approaches
Wassan and Nagy, 2015	The Vehicle Routing Problem with Divisible Deliveries and Pickups
Chen, 2014	Model and algorithm for an unpaired pickup and delivery vehicle routing problem with split loads
Li, 2016	Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests

Nguyen et al., 2016	Multi-trip pickup and delivery problem with time windows and synchronization
Chentli et al., 2018	A SELECTIVE ADAPTIVE LARGE NEIGHBORHOOD SEARCH HEURISTIC FOR THE PROFITABLE TOUR PROBLEM WITH SIMULTANEOUS PICKUP AND DELIVERY SERVICES
Zhiyuan and Wantao, 2015	The Vehicle Scheduling Problem of Third-Party Passenger Finished Vehicle Logistics Transportation: Formulation, Algorithms, and Instances
Ríos-Mercado, 2013	A GRASP for a multi-depot multi-commodity pickup and delivery problem with time windows and heterogeneous fleet in the bottled beverage industry
Cattaruzza, Absi et Feillet, 2016	Vehicle routing problems with multiple trips
Yoshizaki and Belfiore, 2006	Scatter search for vehicle routing problem with time windows and split deliveries

Source: author

Afterwards, the articles were analyzed in order to sort the information that could have been used from each. On a sheet, it was retrieved the title, parameters, data, constraints and methods of each article, showing also its applications and results in its outline:

Figure 7: Table with the sorting of relevant information for each article

Article	Parameters/Data	Constraints	Objective/application	Method	Results	Comments
The Pickup and Delivery Problem with Split Loads (Nowak, 2005)	Demand between the origin i and the destination j Distance between nodes i and j	Pickup & Delivery Split loads	Delivery/transportation sector (case study of FedEx)	Mathematical modeling Heuristics	Average CPU time (minutes) : - 3.4 (75 transportation requests, no route limit, 0.11 - 0.2 load range) - 286.3 (125 transportation requests, no route limit, 0.51 - 0.6 load range)	Complex mathematical formulation
Vehicle Routing Problem : Simultaneous Deliveries and Pickups with Split Loads and Time Windows (Wang, 2013)	Number of permutations (used to increase the diversity of the initial routes) Travel speed (time dependent) Max/min number of iterations Number of costumers	Simultaneous pickup & delivery Split loads Time Windows	Delivery/transportation sector (Modified Solomon Data Sets)	Mathematical modeling Heuristics: - Purposed by the author - Hybrid heuristic method (HHM) - Construction heuristic algorithm (CHA) - Reactive tabu search algorithm (RTSA)	Heuristic created by the author has the best performance	Mandatory simultaneous pickup and delivery
Vehicle Routing Problem with Deliveries and Pickups: Modelling Issues and Meta-heuristics Solution Approaches (Wassan and Nagy, 2014)	Distance between locations i and j Delivery demand of customer i Backhaul (pickup) demand of customer i Vehicle capacity	Pickup & Delivery Split loads (divisible deliveries)	Delivery/transportation sector (Christofides-Eilon VRP instances)	Mathematical modeling (Integer Linear Programming ILP) Software: IBM ILOG CPLEX	CPU time : - from 1 second to 1440 minutes (du modèle le plus simple jusqu'à celui avec plus de contraintes et paramètres plus importants)	Use of constraints slightly different from Split Loads of our work
Model and algorithm for an unpaired pickup and delivery vehicle routing problem with split loads (Chen, 2014)	Number of vertices Number of materials Vehicle duration Vehicle capacity Vehicle speed Location of vertex i Inventory level Demand	Pickup & Delivery Split loads	Delivery/transportation sector (Instances originelles)	Mathematical modeling Heuristics Software: IBM ILOG CPLEX	CPU time : - from 1 second to 6420 minutes (du modèle le plus simple jusqu'à celui avec plus de contraintes et paramètres plus importants)	Use of constraints slightly different from Split Loads of our work

Source: author

Figure 8: Table with the sorting of relevant information for each article

The Vehicle Routing Problem with Divisible Deliveries and Pickups (Wassan and N.Nagy, 2015)	Distance between locations i and j Delivery demand of customer i Vehicle capacity	Pickup & Delivery Split loads (divisible)	Delivery/transportation sector (Instances adaptées de Christofides and Eilon)	Mathematical modeling (ILP) Heuristics Software: IBM ILOG CPLEX	CPU time : - from 5 seconds to 5.8 minutes (du modèle le plus simple jusqu'à celui avec plus de contraintes et paramètres plus importants)	Comparaison between different constraints of Split-loads
Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests (Li, 2016)	Requests Number of vehicles Vehicles capacity Time windows Travel cost Travel time	Pickup & Delivery Time windows Profits Reserved requests	Delivery/transportation sector (Euclidean instances)	Mathematical modeling Heuristics: - Adaptive large neighborhood search (ALNS) - Sequential insertion heuristic (SIH) Software: IBM ILOG CPLEX	CPU time : - from 3.1 seconds to 600 minutes (du modèle le plus simple jusqu'à celui avec plus de contraintes et paramètres plus importants)	-
Multi-trip pickup and delivery problem with time windows and synchronization (Nguyen et al., 2016)	Pickup/Delivery costumers demand [X,Y] coordinates Supply points	Pickup & Delivery Time Windows Multi-trip	Delivery/transportation sector (TMZTVRPTW instances of Crainic et al. (2009).)	Mathematical modeling Heuristics: - Tabu search	CPU time : - from 3.1 seconds to 600 minutes (du modèle le plus simple jusqu'à celui avec plus de contraintes et paramètres plus importants)	
A SELECTIVE ADAPTIVE LARGE NEIGHBORHOOD SEARCH HEURISTIC FOR THE PROFITABLE TOUR PROBLEM WITH SIMULTANEOUS PICKUP AND DELIVERY SERVICES (Chentli et al., 2018)	Number of costumers Number of vehicles Capacity bound Pickup/delivery demand Profit Traveling cost	Pickup & Delivery Profit	Delivery/transportation sector (sALNS 117 instances with 50–199 customers)	Mathematical modeling Heuristics: - Adaptive large neighborhood search (ALNS) Software: IBM ILOG CPLEX	CPU time : - from 0.1 seconds to 22.4 minutes (du modèle le plus simple jusqu'à celui avec plus de contraintes et paramètres plus importants)	

Source: author

With this sort, it was possible to observe what would be taken as inspiration for the model of this project. There are three articles that have contributed the most to this project. They were made by Wang (2013), Cattaruzza, Absi and Feillet (2016) and Nowak (2005). In addition, it is very important to note that there is also an article developed by the supervisors of this work, Ramdane and Jaballah (2021), which configures as the most important source since this project targets the continuation of their research.

Finally, through Ramdane and Jaballah (2021), the characterization of the model of this work's model was made, being necessary for the taking of parameters, variables, and some constraints and objective function. Then, the mathematical formalization was inspired by that of Wang (2013) and that of Cattaruzza, Absi and Feillet (2016) through the adaptation of their work to our specific constraints. Finally, Nowak (2005)'s article was used at the general theoretical consultation level because its article is long and dense, with many concepts.

In summary, Ramdane and Jaballah (2021) were responsible for defining the problem, its target and its parameters. Their scope was to develop a constructive heuristic to solve the problem, while the scope of this work was to model the problem mathematically and solve it by an optimization software. It is remarkable that this problem has not been addressed by the literature before, as the state-of-the-art proved. It is also important to state that the work from Ramdane and Jaballah (2021) did not contain a mathematical formulation of the problem.

3. MATHEMATICAL MODELING

First, we will present a model that takes into account a Pickup and Delivery Problem with Split Loads, Profits and Multiple Time Windows. Then we will present a model adding the Multi-trip constraint. That strategy was used because the addition of the Multi-trip constraint provokes an increase of complexity in the model by attaching one more index to the variables.

As already stated, many constraints were inspired by those of Wang (2013) and Cattaruzza, Absi and Feillet (2016), taking into account the specifications of this work's problem from Ramdane and Jaballah (2021). Additionally, many binary modelling strategies were used according to Winston (2003).

In the following sections the parameters of each model will be described, then its variables, then the objective function and finally the restrictions. Finally, there will be a section concerning the validation of the models through the simulation with instances composed by estimated parameters.

3.1. Pickup and Delivery Problem with Split Loads, Profits with Multiple Time Windows (PDPSPMTW)

The problem in question is assigned as a mixed integer linear programming (MILP) model. That is to say that the variables required to formalize the problem mathematically consisted of integer, binary or real variables (which is a mix of types of variables). They were needed so it would be possible to account things properly. For instance, the number of pallets delivered to a job site needs to be an integer value, since it is not possible to deliver half of a pallet. Or still, a beginning time of service at a job site could be any time, which means that a real variable is necessary. And finally, in order to keep up with transports performed, binary variables were required so it would be possible to check which vehicle was assigned to perform which displacement.

The following sections will address the parameters, variables, objective function and restrictions of the problem

3.1.1. Parameters

Sets:

$\boldsymbol{V}$:	$\{0, 1, \ldots, \boldsymbol{n}\}$	$set\ of\ construction\ sites, being\ 0\ the\ warehouse;$
$\boldsymbol{V} \setminus \{\boldsymbol{0}\}$:	$\{1, \ldots, \boldsymbol{n}\}$	$set\ of\ construction\ sites;$
$\boldsymbol{E}$:	$\{(\boldsymbol{i}, \boldsymbol{j}) : \boldsymbol{i}, \boldsymbol{j} \in \boldsymbol{V}, \boldsymbol{i} \neq \boldsymbol{j}\}$	$set\ of\ possible\ roads\ between\ sites;$
$\boldsymbol{H}$:	$\{1, \ldots, \boldsymbol{m}\}$	$set\ of\ vehicles;$
$\boldsymbol{A}$:	$\{1, \ldots, \boldsymbol{a}\}$	$set\ of\ time\ windows;$

Capacities:

$\boldsymbol{Q_k}$:	$vehicle\ capacity\ (in\ pallets)$	$\boldsymbol{k} \in \boldsymbol{H};$
$\boldsymbol{T_k}$:	$maximal\ working\ time\ by\ vehicle\ (in\ hours)$	$\boldsymbol{k} \in \boldsymbol{H};$
$\boldsymbol{D}_k$:	$maximum\ distance\ that\ vehicle\ \boldsymbol{k}\ can\ travel\ (in\ kilometers)$	$\boldsymbol{k} \in \boldsymbol{H};$

Demands:

$\overrightarrow{\boldsymbol{p_i}}$: $pickup\ demand\ (in\ pallets)\ of\ the\ construction\ site\ \boldsymbol{i} \in \boldsymbol{V} \setminus \{0\},$
$composed\ of\ \boldsymbol{u}\ demands\ of\ different\ bigbags\ (bigbag_1, \ldots, bigbag_u);$

$\overrightarrow{\boldsymbol{d_i}}$: $delivery\ demand\ (in\ pallets)\ of\ the\ construction\ site\ \boldsymbol{i} \in \boldsymbol{V} \setminus \{0\},$
$composed\ of\ \boldsymbol{u}'\ demands\ of\ different\ bigbags\ (kit_1, \ldots, kit_{u'});$

$\boldsymbol{Q}(\overrightarrow{\boldsymbol{p_i}})$: $total\ pickup\ demand\ (in\ pallets), being\ \boldsymbol{i} \in \boldsymbol{V} \setminus \{0\};$

$\boldsymbol{Q}(\overrightarrow{\boldsymbol{d_i}})$: $total\ delivery\ demand\ (in\ pallets), being\ \boldsymbol{i} \in \boldsymbol{V} \setminus \{0\};$

Profits:

$\boldsymbol{pp_i}$: $unit\ profit\ by\ pallet,$
$associated\ with\ the\ pickup\ demand\ of\ the\ job\ site\ \boldsymbol{i} \in \boldsymbol{V} \setminus \{0\};$

$\boldsymbol{pd_i}$: $unit\ profit\ by\ pallet,$
$associated\ with\ the\ pickup\ demand\ of\ the\ job\ site\ \boldsymbol{i} \in \boldsymbol{V} \setminus \{0\};$

Time windows:

$$\boldsymbol{TW}_i = \{[\boldsymbol{e_i^1}, \boldsymbol{l}_i^{\boldsymbol{1}}], \ldots, [\boldsymbol{e_i^\alpha}, \boldsymbol{l}_i^{\boldsymbol{\alpha}}], \ldots, [\boldsymbol{e_i^a}, \boldsymbol{l}_i^{\boldsymbol{a}}]\}$$

$time\ windows\ of\ the\ job\ site\ \boldsymbol{i}, where\ \boldsymbol{e_i^\alpha}\ is\ the\ earliest\ time\ of\ the$
$time\ window\ \boldsymbol{\alpha}\ to\ begin\ service\ at\ the\ job\ site\ \boldsymbol{i},$
$and\ \ \boldsymbol{l}_i^{\boldsymbol{\alpha}}\ is\ the\ latest\ time\ of\ the\ time\ window\ \boldsymbol{\alpha}\ to\ finish\ service\ at$
$the\ job\ site\ \boldsymbol{i}, with\ \boldsymbol{\alpha} \in \boldsymbol{A}, being\ \boldsymbol{a}\ \ the\ total\ number\ of\ time$
$windows\ of\ the\ job\ sites;$

$\boldsymbol{TW}_0 =$ $[\boldsymbol{e_0}, \boldsymbol{l}_0]\ is\ the\ time\ window\ of\ the\ warehouse;$

Durations:

$\boldsymbol{t_{i,j}}$: $transport\ time\ between\ vertices\ \boldsymbol{i}\ and\ \boldsymbol{j}, with\ (\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{E};$

$\boldsymbol{s}_i$: $service\ time\ (loading\ and\ unloading\ vehicles)\ at\ the\ construction$
$site\ \boldsymbol{i}, with\ \boldsymbol{i} \in \boldsymbol{V} \setminus \{0\};$

Distance:

$\boldsymbol{d_{i,j}}$: $distance\ between\ vertices\ \boldsymbol{i}\ and\ \boldsymbol{j}, with\ (\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{E};$

3.1.2. Variables

$\boldsymbol{v_k}$:	$\mathbf{1}\ if\ vehicle\ \boldsymbol{k}\ is\ chosen,$ $\mathbf{0}\ otherwise$	$\boldsymbol{k} \in \boldsymbol{H}$
$\boldsymbol{x_{i,j,k}}$:	$\mathbf{1}\ if\ vehicle\ \boldsymbol{k}\ \ fulfill\ the\ transport\ from\ client\ \boldsymbol{i}\ to\ client\ \boldsymbol{j},$ $\mathbf{0}\ otherwise$	$\boldsymbol{k} \in \boldsymbol{H}$ $(\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{E}$
$\boldsymbol{\theta_{i,k}^\alpha}$:	$\mathbf{1}\ if\ the\ time\ window\ \boldsymbol{\alpha}\ is\ used\ by\ vehicle\ \boldsymbol{k}\ to\ serve\ \boldsymbol{i},$ $\mathbf{0}\ otherwise$	$\boldsymbol{i} \in \boldsymbol{V} \setminus \{\mathbf{0}\}$ $\boldsymbol{\alpha} \in \boldsymbol{A}$ $\boldsymbol{k} \in \boldsymbol{H}$
$\boldsymbol{y_{i,k}}$:	$number\ of\ pallets\ delivered\ by\ vehicle\ \boldsymbol{k}\ at\ client\ \boldsymbol{i}$	$\boldsymbol{i} \in \boldsymbol{V} \setminus \{\mathbf{0}\}$ $\boldsymbol{k} \in \boldsymbol{H}$
$\boldsymbol{z_{i,k}}$:	$number\ of\ pallets\ picked\ up\ by\ vehicle\ \boldsymbol{k}\ at\ client\ \boldsymbol{i}$	$\boldsymbol{i} \in \boldsymbol{V} \setminus \{\mathbf{0}\}$ $\boldsymbol{k} \in \boldsymbol{H}$

$\boldsymbol{y'}_{\boldsymbol{i,k}}$: $number\ of\ pallets\ on\ vehicle\ \boldsymbol{k}\ \ when\ it\ leaves\ client\ \boldsymbol{i}$ $\qquad \boldsymbol{i} \in \boldsymbol{V}$

$\boldsymbol{k} \in \boldsymbol{H}$

$\boldsymbol{b}_{\boldsymbol{i,k}}$: $begginning\ time\ of\ service\ fulfilled\ by\ vehicle\ \boldsymbol{k}\ at\ client\ \boldsymbol{i}$ $\qquad \boldsymbol{i} \in \boldsymbol{V}$

$\boldsymbol{k} \in \boldsymbol{H}$

3.1.3. Objective function

The objective function was given by Ramdane and Jaballah (2021). Since the goal of the present work is to simulate the problem by an optimization solver and compare its result with the heuristics previously created, it was necessary to employ the same objective function. Nevertheless, at a first time, the mathematical model was created with simpler objective functions. Through the constructive heuristics, Ramdane and Jaballah (2021) were able to apply a more complex model. For instance, with the mixed integer linear programming of the present work, it was necessary to consider plenty of modeling strategies to linearize constraints through strategies involving binary variables.

Therefore, this section will address the main objective function although it was not the first modeling strategy applied to the creation of the models. These other objective function strategies will be addressed in the last part of the modelling section, the validation of models.

The objective function of Ramdane and Jaballah (2021) is the following:

$$\min \left(\frac{\sum_{k=1}^{m} \sum_{i=0}^{n} \sum_{j=0}^{n} \boldsymbol{d}_{i,j}\, \boldsymbol{x}_{i,j,k}}{\sum_{k=1}^{m} \boldsymbol{D}_{k}} \right) - \left(\frac{\sum_{i=1}^{n} \left[\boldsymbol{pd}_{i} \left(\sum_{k=1}^{m} \boldsymbol{y}_{i,k} \right) + \boldsymbol{pp}_{i} \left(\sum_{k=1}^{m} \boldsymbol{z}_{i,k} \right) \right]}{\sum_{i=1}^{n} \left[\boldsymbol{pd}_{i}\, \boldsymbol{Q}\left(\overrightarrow{\boldsymbol{d}_{i}} \right) + \boldsymbol{pp}_{i}\, \boldsymbol{Q}\left(\overrightarrow{\boldsymbol{p}_{i}} \right) \right]} \right)$$

It minimizes the total distance travelled by the vehicle set and maximizes profits by satisfying pickup and delivery demands. The first part of the expression concerns the distance, while the second the profit. In the first one, the total distance travelled by the set of vehicles is in the numerator, whereas the maximum distance that can be travelled by the set of vehicles is in the denominator. Then, in the second part, the profit achieved by meeting demands is in the numerator whilst the total profit that can be made is in the denominator. Both expressions indicate a fraction of a total, allowing a relation between them by means of a subtraction. Therefore, it is possible to minimize distance and maximize profit at the same time. As already stated, this modeling strategy was applied by Ramdane and Jaballah (2021). It was necessary to follow the same objective function so this work can fulfill its goal. An important observation to make is that their work addressed this objective function through the constructive heuristics, and not by a mathematical model composed by mixed integer linear programming. All modelling components were created by the present work, that were validated by all supervisors.

3.1.4. Constraints

$$(1) \quad \sum_{j=1}^{n} x_{0,j,k} \leq 1 \qquad \forall k \in H$$

$$(2) \quad \sum_{i=0}^{n} x_{i,u,k} - \sum_{j=0}^{n} x_{u,j,k} = 0 \qquad \forall u \in V \setminus \{0\} \quad \forall k \in H$$

$$(3) \quad \sum_{i=1}^{n} x_{i,0,k} \leq 1 \qquad \forall k \in H$$

$$(4) \quad y'_{0,k} = \sum_{i=1}^{n} y_{i,k} \qquad \forall k \in H$$

$$(5a) \quad y'_{j,k} - \left(y'_{i,k} + z_{j,k} - y_{j,k}\right) \leq M\left(1 - x_{i,j,k}\right) \qquad \forall (i,j) \in E \quad \forall k \in H$$

$$(5b) \quad y'_{j,k} - \left(y'_{i,k} + z_{j,k} - y_{j,k}\right) \geq M\left(x_{i,j,k} - 1\right) \qquad \forall (i,j) \in E \quad \forall k \in H$$

$$(6) \quad y'_{i,k} \leq Q_k \qquad \forall i \in V \quad \forall k \in H$$

$$(7) \quad \sum_{k=1}^{m} y_{i,k} \leq Q(\overrightarrow{d_i}) \qquad \forall i \in V$$

$$(8) \quad \sum_{k=1}^{m} z_{i,k} \leq Q(\overrightarrow{p_i}) \qquad \forall i \in V$$

$$(9) \quad \sum_{i=0}^{n}\sum_{j=0}^{n}(t_{i,j} + s_i)\, x_{i,j,k} \leq T_k \qquad \forall k \in H$$

$$(10) \quad \sum_{i=0}^{n}\sum_{j=0}^{n} d_{i,j}\, x_{i,j,k} \leq D_k \qquad \forall k \in H$$

$$(11) \quad \sum_{i=0}^{n} x_{i,j,k} \leq 1 \qquad \forall j \in V, \ \forall k \in H$$

$$(12a) \quad y_{i,k} + z_{i,k} \geq \sum_{j=0}^{n} x_{i,j,k} \qquad \forall i \in V, \ \forall k \in H$$

$$(12b) \quad y_{i,k} + z_{i,k} \leq M \sum_{j=0}^{n} x_{i,j,k} \qquad \forall i \in V, \ \forall k \in H$$

$$(13a) \quad b_{0,k} \geq v_k\, e_0 \qquad \forall k \in H$$

$$(13b) \quad b_{i,k} + s_i + t_{i,0} - l_0 \leq M(1 - x_{i,0,k}) \qquad \forall i \in V, \ \forall k \in H$$

$$(14) \quad \sum_{\alpha=1}^{a} \theta_{i,k}^{\alpha}\, e_i^{\alpha} \leq b_{i,k} \leq \sum_{\alpha=1}^{a} \theta_{i,k}^{\alpha}\, (l_i^{\alpha} - si) \qquad \forall i \in V \setminus \{0\}, \ \forall k \in H$$

$$(15) \quad \sum_{\alpha=1}^{a} \theta_{i,k}^{\alpha} \leq 1 \qquad \forall i \in V \setminus \{0\} \quad \forall k \in H$$

$$(16a) \quad b_{i,k} + s_i + t_{i,j} - b_{j,k} \leq M(1 - x_{i,j,k}) \qquad \forall i \in V \quad \forall j \in V \ \{0\} \quad \forall k \in H$$

$$(16b) \quad b_{i,k} + s_i + t_{i,j} - b_{j,k} \geq M(x_{i,j,k} - 1) \qquad \forall i \in V \quad \forall j \in V \setminus \{0\} \quad \forall k \in H$$

$$(17a) \quad \sum_{i=0}^{n} \sum_{j=0}^{n} x_{i,j,k} \geq v_k \qquad \forall k \in H$$

$$(17\text{b}) \quad \sum_{i=0}^{n} \sum_{j=0}^{n} x_{i,j,k} \leq \text{M}\, v_k \qquad \forall k \in H$$

$$(18a) \quad v_k = \sum_{i=1}^{n} x_{i,0,k} \qquad \forall k \in H$$

$$(18\text{b}) \quad v_k = \sum_{j=1}^{n} x_{0,j,k} \qquad \forall k \in H$$

$$(19) \quad \sum_{j=0}^{n} x_{i,j,k} = \sum_{\alpha=1}^{a} \theta_{i,k}^{\alpha} \qquad \forall i \in V \setminus \{0\} \quad \forall k \in H$$

$$(20) \quad x_{i,i,k} = 0 \qquad \forall i \in V \quad \forall k \in H$$

$$(21) \quad x_{i,j,k}, v_k, \theta_{i,k}^{\alpha} \in \{0, 1\} \qquad \forall (i,j) \in E$$

$$y_{i,j,k}, y'_{i,j,k}, z_{i,j,k} \in \mathbb{Z}^{+} \qquad \forall k \in H$$

$\boldsymbol{b_{j,k}} \in \mathbb{R}^{+}$ $\forall \boldsymbol{l} \in \boldsymbol{V}\ \{\boldsymbol{0}\}$

$\forall \boldsymbol{\alpha} \in \boldsymbol{A}$

Description of constraints:

(1) Every vehicle begins its journey at the warehouse

(2) Every vehicle will leave a certain costumer to serve another one until it finally returns to the depot. This constraint ensures that the trips will have a logical sequence.

(3) Every vehicle shall complete its trip at the depot.

(4) The quantity of pallets in the vehicle **k** leaving the depot is worth the quantity of products it will deliver in its complete trip.

(5) Constraint notion: $\boldsymbol{y'_{j,k} = y'_{i,k} + z_{j,k} - y_{j,k}}\ \forall(\boldsymbol{i,j}) \in \{\boldsymbol{E} \mid \boldsymbol{x_{i,j,k} = 1}\}, \forall \boldsymbol{k} \in \boldsymbol{H}.$ The quantity of pallets in the vehicle **k** leaving a site **j** is equal to the quantity of pallets before it reaches it (that is, the quantity of pallets that the vehicle **k** has when it leaves site **i**) plus the quantity collected in **j** minus the quantity delivered in **j**. **(5a)** and **(5b)** are constraints that contain modelling strategies involving binary variables combined with the use of a big-enough parameter **M**, where the condition $\mathbf{x_{i,j,k} = 1}$ checks the fulfilment of the constraint, and $\mathbf{x_{i,j,k} = 0}$ cancels it.

(6) The vehicle **k** cannot exceed its capacity (in pallets) if all its loads/unloads are considered.

(7) The total quantity of pallets delivered in **j** must be less than or equal to the total demand for delivery in **j**.

(8) The total quantity of pallets picked up in **j** must be less than or equal to the total pickup demand in **j**.

(9) The trip of the vehicle **k** shall not exceed its maximum working time.

(10) The trip of the vehicle **k** shall not exceed its maximum distance.

(11) Each job site may be visited at most once per trip of a vehicle **k**.

(12) Constraint notion: $\boldsymbol{If\ y_{i,k} > 0 \lor z_{i,k} > 0 \rightarrow \sum_{j}^{n} x_{i,j,k} = 1}\ \forall \boldsymbol{i} \in \boldsymbol{V}, \forall \boldsymbol{k} \in \boldsymbol{H}.$ This constraint makes it possible to give a value to the variable **x** if site **i** is served (that is to say if **z** or **y** are greater than **0**). The big-enough **M** with binary variables approach is used here.

(13) Constraint notion **(13b)**: $\boldsymbol{b_{i,k} + s_i + t_{i,0} \leq l_0}\ \ \forall \boldsymbol{i} \in (\boldsymbol{V} \mid \boldsymbol{x_{i,0,k} = 1}), \forall \boldsymbol{k} \in \boldsymbol{H}.$ Each vehicle **k** must arrive at the depot before the end of its time window, or at the same time it ends (if and only if **k** has been chosen), from a given site **i**. To check if this vehicle had left this site, we use the $\mathbf{x_{i,0,k}}$ binary with the big-enough **M**. If it is **1**, then the beginning time at **i** plus the service duration in **i** plus the travel duration from **i** to the depot may not exceed the end of its time window. If it is **0**, then the expression will leave the variable $\mathbf{b_{i,k}}$ free (because **k** does not necessarily arrive at the warehouse after going through **i**, it may go through other job sites, or even not at all go through **i**).

(14) The service must be started at **i** after one of its time windows is opened, or at the same time it opens (if and only if **k** was one of the vehicles selected to serve **i**). The service must be completed at **i** before one of its time windows is closed, or at the same time it does (if and only if **k** was one of the vehicles selected to serve **i**). If the vehicle **k** has never passed **i**, the expression will be cancelled, and $\mathbf{b_{i,k}}$ will be **0**, since the service has not been performed. Verify that at most one time window will be used.

(15) This constraint ensures that only one time window $\boldsymbol{\alpha}$ is taken for the service of the vehicle **k** in **i**.

(16) Constraint notion: $\boldsymbol{b_{i,k} + s_i + t_{i,j} = b_{j,k}}\ \forall(\boldsymbol{i,j}) \in \{\boldsymbol{E} \mid \boldsymbol{x_{i,j,k} = 1}\}, \forall \boldsymbol{k} \in \boldsymbol{H}$. The beginning time of service at **j** ($\mathbf{b_{j,k}}$), must start after the end of service in **i** ($\mathbf{b_{i,k} + s_i}$), taking also into account the transport duration from **i** to **j** ($\mathbf{t_{i,j}}$) (if and only if **k** was chosen to carry out this transport, i.e., $\mathbf{x_{i,j,k} = 1}$). The constraints **(16a)** and **(16b)** use the big-enough **M** to validate this condition of transport from **i** to **j**. If it is true, then **(16)** is true too. Otherwise, the variables are clear.

(17) To check if the vehicle **k** was selected, the constraints **(17a)** and **(17b)** are used, which link the variables **v** and **x**. The constraints use the big-enough **M** combined with binary variables to implement this verification.

(18) Constraints **(18a)** and **(18b)** are responsible for the exit and arrival of the vehicle **k** at the depot. If the vehicle **k** has been selected, that is, the $\mathbf{v_k = 1}$, then there is a $\mathbf{x_{0,j,k} = 1}$ and a $\mathbf{x_{i,0,k} = 1}$ that are true.

(19) Constraint that makes the relationship between the variables **x** and $\boldsymbol{\theta}$. If the vehicle **k** has never passed through site **i**, it means that no $\boldsymbol{\theta}$ is **1**. Otherwise (if the vehicle **k** passes through **i**), then there is a $\boldsymbol{\theta}$ that is **1**.

(20) There is no route from **i** to **i**.

(21) Definition of binary variables and positive real variables.

3.1.5. Remarks

The model developed did not distinguish the demand for different kits and big-bags because variables **y** and **z** would need an additional index to assign a quantity of pallets of each type of kit/big-bag. This would increase the complexity of the model as well as the simulation. It was preferable not to distinguish them. In addition, the later version will already use an additional index to address the Multi-trip constraints.

Vehicles do not wait for the earliest time of time windows, that means that the service of loading/unloading a vehicle should only begin if it arrives at the job site after the opening of its time window (or at the same time it does). This was an assumption made by Ramdane and Jaballah (2021) in their work, based on real scenarios where construction sites do not have enough space for vehicles to park and wait.

As stated before, the objective function applied was the same as the one used by the constructive heuristic developed by Ramdane and Jaballah (2021). Nevertheless, it was necessary to create simpler objective functions before simulating with the one previously presented. This validation of model will be addressed later on this work.

Finally, it was considered that the warehouse has an infinite capacity regarding the amount of raw material available and waste reception space.

3.2. Multi-trip Pickup and Delivery Problem with Split Loads, Profits with Multiple Time Windows (MTPDPSPMTW)

Like on PDPSPMTW, the problem in question was formulated as a MILP. Parameters, variables, objective function and restrictions are described as follows.

3.2.1. Parameters

The parameters are exactly the same as the ones previously used. The following set is the only parameter that was added to the model:

$\boldsymbol{R}$:	$\{1, \dots, \boldsymbol{c}\}$	$set\ of\ trips\ that\ a\ vehicle\ can\ perform;$
$\boldsymbol{R} \setminus \{\boldsymbol{c}\}$:	$\{1, \dots, \boldsymbol{c-1}\}$	$set\ of\ trips\ that\ a\ vehicle\ can\ perform,$ $excluding\ the\ last\ one;$

3.2.2. Variables

$\boldsymbol{v_k^r}$:	$\mathbf{1}\ if\ vehicle\ \boldsymbol{k}\ fulfills\ trip\ \boldsymbol{r},$ $\mathbf{0}\ otherwise$	$\boldsymbol{k} \in \boldsymbol{H}$ $\boldsymbol{r} \in \boldsymbol{R}$
$\boldsymbol{x_{i,j,k}^r}$:	$\mathbf{1}\ if\ vehicle\ \boldsymbol{k}\ fulfill\ the\ transport$ $from\ client\ \boldsymbol{i}\ to\ client\ \boldsymbol{j}\ on\ its\ trip\ \boldsymbol{r},$ $\mathbf{0}\ otherwise$	$\boldsymbol{k} \in \boldsymbol{H}$ $(\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{E}$ $\boldsymbol{r} \in \boldsymbol{R}$
$\boldsymbol{\theta_{i,k}^{\alpha,r}}$:	$\mathbf{1}\ if\ the\ time\ window\ \boldsymbol{\alpha}\ is\ used\ by\ vehicle\ \boldsymbol{k}\ to\ serve\ \boldsymbol{i}$ $on\ its\ trip\ \boldsymbol{r},$ $\mathbf{0}\ otherwise$	$\boldsymbol{i} \in \boldsymbol{V} \setminus \{\mathbf{0}\}$ $\boldsymbol{\alpha} \in \boldsymbol{A}$ $\boldsymbol{k} \in \boldsymbol{H}$ $\boldsymbol{r} \in \boldsymbol{R}$
$\boldsymbol{y_{i,k}^r}$:	$number\ of\ pallets\ delivered\ at\ client\ \boldsymbol{i}\ by\ vehicle\ \boldsymbol{k}$ $on\ its\ trip\ \boldsymbol{r}$	$\boldsymbol{i} \in \boldsymbol{V} \setminus \{\mathbf{0}\}$ $\boldsymbol{k} \in \boldsymbol{H}$ $\boldsymbol{r} \in \boldsymbol{R}$

$\boldsymbol{z^r_{i,k}}$: $number\ of\ pallets\ picked\ up\ by\ vehicle\ \boldsymbol{k}\ at\ client\ \boldsymbol{i}$ $\boldsymbol{i} \in \boldsymbol{V} \setminus \{\boldsymbol{0}\}$
$on\ its\ trip\ \boldsymbol{r}$ $\boldsymbol{k} \in \boldsymbol{H}$
$\boldsymbol{r} \in \boldsymbol{R}$

$\boldsymbol{y'^{\,r}_{i,k}}$: $number\ of\ pallets\ on\ vehicle\ \boldsymbol{k}\ when\ it\ leaves\ client\ \boldsymbol{i}$ $\boldsymbol{i} \in \boldsymbol{V}$
$on\ its\ trip\ \boldsymbol{r}$ $\boldsymbol{k} \in \boldsymbol{H}$
$\boldsymbol{r} \in \boldsymbol{R}$

$\boldsymbol{b^r_{i,k}}$: $begginning\ time\ of\ service\ fulfilled\ by\ vehicle\ \boldsymbol{k}$ $\boldsymbol{i} \in \boldsymbol{V}$
$on\ its\ trip\ \boldsymbol{r}\ at\ client\ \boldsymbol{i}$ $\boldsymbol{k} \in \boldsymbol{H}$
$\boldsymbol{r} \in \boldsymbol{R}$

3.2.3. Objective function

The objective function is analogous to the one of the previous model, then its description is identical. The difference is that the variables now use one more index (**r**).

$$\min\left(\frac{\sum_{k=1}^{m}\sum_{r=1}^{c}\sum_{i=0}^{n}\sum_{j=0}^{n} d_{i,j}\, x^r_{i,j,k}}{\sum_{k=1}^{m} D_k}\right) - \left(\frac{\sum_{i=1}^{n}\left[pd_i\left(\sum_{r=1}^{c}\sum_{k=1}^{m} y^r_{i,k}\right) + pp_i\left(\sum_{r=1}^{c}\sum_{k=1}^{m} z^r_{i,k}\right)\right]}{\sum_{i=1}^{n}\left[pd_i\, Q(\overrightarrow{d_i}) + pp_i\, Q(\overrightarrow{p_i})\right]}\right)$$

3.2.4. Constraints

$$(1) \qquad \sum_{j=1}^{n} x^r_{0,j,k} \le 1 \qquad \forall k \in H,\ \forall r \in R$$

$$(2) \qquad \sum_{i=0}^{n} x^r_{i,u,k} - \sum_{j=0}^{n} x^r_{u,j,k} = 0 \qquad \forall u \in V \setminus \{0\},\ \forall k \in H,\ \forall r \in R$$

$$(3) \qquad \sum_{i=1}^{n} x^r_{i,0,k} \le 1 \qquad \forall k \in H,\ \forall r \in R$$

$$(4)\qquad {y'}^{r}_{0,k} = \sum_{i=1}^{n} y^{r}_{i,k} \qquad \forall k \in H \quad \forall r \in R$$

$$(5a)\qquad {y'}^{r}_{j,k} - \left({y'}^{r}_{i,k} + z^{r}_{j,k} - y^{r}_{j,k}\right) \le M\left(1 - x^{r}_{i,j,k}\right) \qquad \forall (i,j) \in E \quad \forall k \in H \quad \forall r \in R$$

$$(5b)\qquad {y'}^{r}_{j,k} - \left({y'}^{r}_{i,k} + z^{r}_{j,k} - y^{r}_{j,k}\right) \ge M\left(x^{r}_{i,j,k} - 1\right) \qquad \forall (i,j) \in E \quad \forall k \in H \quad \forall r \in R$$

$$(6)\qquad {y'}^{r}_{i,k} \le Q_k \qquad \forall i \in V \quad \forall k \in H \quad \forall r \in R$$

$$(7)\qquad \sum_{k=1}^{m}\sum_{r=1}^{c} y^{r}_{i,k} \le Q\left(\overrightarrow{d_i}\right) \qquad \forall i \in V$$

$$(8)\qquad \sum_{k=1}^{m}\sum_{r=1}^{c} z^{r}_{i,k} \le Q\left(\overrightarrow{p_i}\right) \qquad \forall i \in V$$

$$(9)\qquad \sum_{r=1}^{c}\sum_{i=0}^{n}\sum_{j=0}^{n} \left(s_i + t_{i,j}\right) x^{r}_{i,j,k} \le T_k \qquad \forall k \in H$$

$$(10)\qquad \sum_{r=1}^{c}\sum_{i=0}^{n}\sum_{j=0}^{n} d_{i,j}\, x^{r}_{i,j,k} \le D_k \qquad \forall k \in H \quad \forall r \in R$$

$$(11)\qquad \sum_{j=0}^{n} x^{r}_{i,j,k} \le 1 \qquad \forall k \in H \quad \forall r \in R$$

$$(12a) \quad y_{i,k}^{r} + z_{i,k}^{r} \geq \sum_{j=0}^{n} x_{i,j,k}^{r} \qquad \forall\, i \in V \quad \forall k \in H \quad \forall r \in R$$

$$(12b) \quad y_{i,k}^{r} + \; z_{i,k}^{r} \leq M \sum_{j=0}^{n} x_{i,j,k}^{r} \qquad \forall\, i \in V \quad \forall k \in H \quad \forall r \in R$$

$$(13a) \quad b_{0,k}^{r} \geq v_{k}^{r}\, e_{0} \qquad \forall k \in H \quad \forall r \in R$$

$$(13b) \quad b_{i,k}^{r} + s_{i} + t_{i,0} - l_{0} \;\leq M\left(1 - x_{i,0,k}^{r}\right) \qquad \forall i \in V \setminus \{0\} \quad \forall k \in H \quad \forall r \in R$$

$$(14) \quad \sum_{\alpha=1}^{a} \theta_{i,k}^{\alpha,r} e_{i}^{\alpha} \;\leq\; b_{i,k}^{r} \;\leq\; \sum_{\alpha=1}^{a} \theta_{i,k}^{\alpha,r}\, (l_{i}^{\alpha} - si) \qquad \forall i \in V \setminus \{0\} \quad \forall k \in H \quad \forall r \in R$$

$$(15) \quad \sum_{\alpha=1}^{a} \theta_{i,k}^{\alpha,r} \leq 1 \qquad \forall i \in V \setminus \{0\} \quad \forall k \in H \quad \forall r \in R$$

$$(16a) \quad b_{j,k}^{r} - \left(b_{i,k}^{r} + s_{i} + t_{i,j}\right) \leq M\left(1 - \; x_{i,j,k}^{r}\right) \qquad \forall i \in V \quad \forall j \in V \;\; \{0\} \quad \forall k \in H \quad \forall r \in R$$

$$(16b) \quad b_{j,k}^{r} - (b_{i,k}^{r} + s_{i} + t_{i,j}) \;\geq M\left(x_{i,j,k}^{r} - 1\right) \qquad \forall i \in V \quad \forall j \in V \setminus \{0\} \quad \forall k \in H \quad \forall r \in R$$

$$(\mathbf{17})\qquad \boldsymbol{b}_{0,k}^{r+1} - \left(\boldsymbol{b}_{i,k}^{r} + \boldsymbol{s}_{i} + \boldsymbol{t}_{i,0}\right) \le \boldsymbol{M}\left(\mathbf{1} - \boldsymbol{x}_{i,0,k}^{r}\right) + \boldsymbol{M}\left(\mathbf{1} - \boldsymbol{v}_{k}^{r+1}\right) \qquad \forall \boldsymbol{i} \in \boldsymbol{V} \setminus \{\mathbf{0}\},\ \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R} \setminus \{\boldsymbol{c}\}$$

$$(\mathbf{18a})\qquad \sum_{i=0}^{n}\sum_{j=0}^{n} \boldsymbol{x}_{i,j,k}^{r} \ge \boldsymbol{v}_{k}^{r} \qquad \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R}$$

$$(\mathbf{18b})\qquad \sum_{i=0}^{n}\sum_{j=0}^{n} \boldsymbol{x}_{i,j,k}^{r} \le \mathbf{M}\, \boldsymbol{v}_{k}^{r} \qquad \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R}$$

$$(\mathbf{19a})\qquad \boldsymbol{v}_{k}^{r} = \sum_{i=1}^{n} \boldsymbol{x}_{i,0,k}^{r} \qquad \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R}$$

$$(\mathbf{19b})\qquad \boldsymbol{v}_{k}^{r} = \sum_{j=1}^{n} \boldsymbol{x}_{0,j,k}^{r} \qquad \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R}$$

$$(\mathbf{20})\qquad \sum_{j=0}^{n} \boldsymbol{x}_{i,j,k}^{r} = \sum_{\alpha=1}^{a} \boldsymbol{\theta}_{i,k}^{\alpha,r} \qquad \forall \boldsymbol{i} \in \boldsymbol{V} \setminus \{\mathbf{0}\},\ \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R}$$

$$(\mathbf{21})\qquad \boldsymbol{v}_{k}^{r} \ge \boldsymbol{v}_{k}^{r+1} \qquad \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R} \setminus \{\boldsymbol{c}\}$$

$$(\mathbf{22})\qquad \boldsymbol{x}_{i,i,k}^{r} = \mathbf{0} \qquad \forall \boldsymbol{i} \in \boldsymbol{V},\ \forall \boldsymbol{k} \in \boldsymbol{H},\ \forall \boldsymbol{r} \in \boldsymbol{R}$$

$$(23) \quad x_{i,j,k}^r, v_k^r, \theta_{u,k}^{\alpha,r} \in \{0,1\} \qquad \forall (i,j) \in E$$

$$y_{i,k}^r, y'^{\,r}_{i,k}, z_{i,k}^r \in \mathbb{Z}^+ \qquad \forall u \in V \setminus \{0\}$$

$$b_{i,k}^r \in \mathbb{R}^+ \qquad \forall k \in H$$

$$\forall r \in R$$

$$\alpha \in A$$

The description of the constraints is analogous to that of before, but now using the concept of trips and journeys (where a journey is the set of trips made by a vehicle). To facilitate the understanding of the model, the descriptions are updated below with this concept, having also the addition of two new constraints compared to the previous model, **(17)** and **(21)**.

Description of constraints:

(1) Every vehicle begins its journey at the warehouse

(2) Every vehicle will leave a certain costumer to serve another one until it finally returns to the depot. This constraint ensures that the trips will have a logical sequence.

(3) Every vehicle shall complete its trip at the depot.

(4) The quantity of pallets in the vehicle **k** leaving the depot is worth the quantity of products it will deliver in its complete trip.

(5) Constraint notion: $y'^{\,r}_{j,k} = y'^{\,r}_{i,k} + z_{j,k}^r - y_{j,k}^r \ \forall (i,j) \in \{E \mid x_{i,j,k}^r = 1\}, \forall k \in H$, $\forall r \in R$. The quantity of pallets in the vehicle **k** (on its trip **r**) leaving a site **j** is equal to the quantity of pallets before it reaches it (that is, the quantity of pallets that the vehicle **k** has when it leaves site **i**) plus the quantity collected in **j** minus the quantity delivered in **j**. **(5a)** and **(5b)** are constraints that contain modelling strategies involving binary variables combined with the use of a big-enough parameter **M**, where the condition $\mathbf{x^r_{i,j,k} = 1}$ checks the fulfilment of the constraint, and $\mathbf{x^r_{i,j,k} = 0}$ cancels it.

(6) The vehicle **k** cannot exceed its capacity (in pallets) if all its loads/unloads on its trip **r** are considered.

(7) The total quantity of pallets delivered in **j** must be less than or equal to the total demand for delivery in **j**.

(8) The total quantity of pallets picked up in **j** must be less than or equal to the total pickup demand in **j**.

(9) The journey of the vehicle **k** (that is, the set of its trips) shall not exceed the maximum working time.

(10) The journey of the vehicle **k** shall not exceed the maximum distance.

(11) Each job site may be visited at most once per trip of a vehicle **k**.

(12) Constraint notion: $\boldsymbol{If\ y^r_{i,k} > 0 \lor z^r_{i,k} > 0 \rightarrow \sum_j^n x^r_{i,j,k} = 1\ \forall i \in V, \forall k \in H}$, $\forall \boldsymbol{r} \in \boldsymbol{R}$. This constraint makes it possible to give a value to the variable **x** if site **i** is served (that is to say if **z** or **y** are greater than **0**). The big-enough **M** with binary variables approach is used here.

(13) Constraint notion **(13b)**: $\boldsymbol{b^r_{i,k} + s_i + t_{i,0} \leq l_0\ \forall i \in (V \mid x^r_{i,0,k} = 1), \forall k \in H}$, $\forall \boldsymbol{r} \in \boldsymbol{R}$. Each vehicle **k** must arrive at the depot before the end of its time window, or at the same time it ends (if and only if **k** has been chosen), from a given site **i**. To check if this vehicle had left this site, we use the $\mathbf{x^r_{i,0,k}}$ binary with the big-enough **M**. If it is **1**, then the beginning time at **i** plus the service duration at **i** plus the travel duration from **i** to the depot may not exceed the end of its time window. If it is **0**, then the expression will leave the variable $\mathbf{b^r_{i,k}}$ free (because **k** does not necessarily arrive at the warehouse after going through **i**, it may go through other job sites, or even not at all go through **i**).

(14) The service must be started at **i** after one of its time windows is opened, or at the same time it opens (if and only if **k** was one of the vehicles selected to serve **i** on its trip **r**). The service must be completed at **i** before one of its time windows is closed, or at the same time it does (if and only if **k** was one of the vehicles selected to serve **i** on its trip **r**). If the vehicle **k** has never passed **i** on its trip **r**, the expression will be cancelled, and $\mathbf{b^r_{i,k}}$ will be **0**, since the service has not been performed. Verify that at most one time window will be used.

(15) This constraint ensures that only one time window **α** is taken for the service of the vehicle **k** at **i** on its trip **r**.

(16) Constraint notion: $\boldsymbol{b^r_{j,k} = b^r_{i,k} + s_i + t_{i,j}\ \forall (i,j) \in \{E \mid x^r_{i,j,k} = 1\}, \forall k \in H, \forall r \in R}$. The beginning time of service at **j** ($\mathbf{b^r_{j,k}}$), must start after the end of service in **i** ($\mathbf{b^r_{j,k}}$ + $\mathbf{s_i}$), taking also into account the transport duration from **i** to **j** ($\mathbf{t_{i,j}}$) (if and only if **k** was chosen to carry out this transport on its trip **r**, i.e., $\mathbf{x^r_{i,j,k} = 1}$). The constraints **(16a)** and **(16b)** use the big-enough **M** combined with binary variables to validate this condition of transport from **i** to **j**. If it is true, then **(16)** is true too. Otherwise, the variables are clear.

(17) Constraint notion: $\boldsymbol{b^{r+1}_{0,k} = b^r_{i,k} + s_i + t_{i,0}\ \forall i \in \{V \setminus \{0\} \mid x^r_{i,0,k} = 1, v^{r+1}_k = 1\}}$, $\forall \boldsymbol{k} \in \boldsymbol{H}, \forall \boldsymbol{r} \in \boldsymbol{R} \setminus \{\boldsymbol{c}\}$. The beginning time of a given trip **r+1** at the depot must be

greater than or equal to the time the vehicle arrives at the depot at the end of the previous trip **r**. The set **R \ {c}** is needed because there is an **r+1** index addressed in this constraint, which would be extrapolated if the set were not limited.

(18) To check if the vehicle **k** was selected, the constraints **(18a)** and **(18b)** are used, which link the variables **v** and **x**. The constraints use the big-enough **M** combined with binary variables to implement this verification.

(19) Constraints **(19a)** and **(19b)** are responsible for the exit and arrival of the vehicle **k** at the depot. If the vehicle **k** has been selected and it fulfills the trip **r**, that is, the $v^r_k = 1$, then there is a $x^r_{0,j,k} = 1$ and a $x^r_{i,0,k} = 1$ that are true.

(20) Constraint that makes the relationship between the variables **x** and **θ**. If the vehicle **k** has never passed through site **i** on its trip **r**, it means that no **θ** is **1**. Otherwise (if the vehicle **k** passes through **i** on its trip **r**), then there is a **θ** that is **1**.

(21) This constraint assures that there is a logical sequence between trips in a journey. If a given trip **r+1** happens (that is, $v^{r+1}_k = 1$) that means that a previous trip **r** happened too.

(22) There is no route from **i** to **i**.

(23) Definition of binary variables and positive real variables.

3.2.5. Remarks

The remarks of the previous model are valid for this one too. So, it is considered that vehicles do not wait for the earliest time of a time window at a job site. At the end of a trip, when the vehicle returns to the warehouse, this study takes into account possible breaks in order to perform the next trip. Finally, vehicles only serve a job site one time per trip, but a job site can be served by several vehicles (and also by the same vehicle but on different trips).

3.3. Validation of models

The creation of models was an iterative process. At first, the problem was conceived without the Multi-trip constraint, that would increase the complexity of it. Then, in order to test if the modelling strategies worked, it was necessary to simulate it. Every time a simulation error was pointed out, it was possible to find if something was wrong with the model.

The simulation of the developed models was performed on *Xpress IVE*, a solver from FICO, an American data analysis company. At a first moment, the data used to create instances was obtained by an estimated database from the supervisors, whose access was easy through Excel. The real parameters, obtained by the DILC project by contacting 20 companies from the construction sector, had a specific format, which would need a data processing. This procedure was performed after the model was validated.

Thus, the present section aims to present what was obtained as previous results to validate both models presented before. Firstly, the estimated instances were simulated with different objective functions. In that way, it was possible to validate the model and its optimization opportunities. Secondly, after validating the model, it was possible to check if the objective function employed by Ramdane and Jaballah (2021) was a feasible application in a mixed integer linear programming context, since they combine two goals (the one of increasing profits and the one of reducing distances).

3.3.1. Objective function analysis

With the goal of testing different objective functions to assess the model functionality and its optimization opportunities, it was developed the following objective functions:

$$\min \boldsymbol{F_1} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{m}} \sum_{\boldsymbol{r=1}}^{\boldsymbol{c}} \boldsymbol{v_k^r}$$

$$\min \boldsymbol{F_2} = \sum_{\boldsymbol{r=1}}^{\boldsymbol{c}} \sum_{\boldsymbol{k=1}}^{\boldsymbol{m}} \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \sum_{\boldsymbol{j=0}}^{\boldsymbol{n}} (\boldsymbol{t_{i,j}} + \boldsymbol{s_i})\, \boldsymbol{x_{i,j,k}^r}$$

$$\min \boldsymbol{F_3} = \sum_{\boldsymbol{r=1}}^{\boldsymbol{c}} \sum_{\boldsymbol{k=1}}^{\boldsymbol{m}} \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \sum_{\boldsymbol{j=0}}^{\boldsymbol{n}} \boldsymbol{d_{i,j}}\, \boldsymbol{x_{i,j,k}^r}$$

$$\max \boldsymbol{F_4} = \sum_{i=1}^{n} \left[\boldsymbol{pd_i} \left(\sum_{k=1}^{m} \sum_{r=1}^{c} y_{i,k}^{r} \right) + \boldsymbol{pp_i} \left(\sum_{k=1}^{m} \sum_{r=1}^{c} \boldsymbol{z_{i,k}^{r}} \right) \right]$$

The objective functions above were presented in the Multi-trip model format (MTPDPSPMTW), with the **r** index. Their goals are presented as follows:

(F1): Minimize total number of trips performed

(F2): Minimize total transport time for the vehicle set

(F3): Minimize total distance travelled by the vehicle set

(F4): Maximize profits by satisfying pickup and delivery demands

These objective functions were possible formulations that the model could use according to distinct optimization strategies. The one that serves as an engine to the model was F4, since the problem addresses the outlook from the logistic platform. As the platform can choose which clients to serve, it is reasonable to apply the profit as the model's engine.

A combination of objective functions was studied in order to test if it was possible to achieve different results depending on what is wanted to be optimized. For example, one initial approach to maximize profit by minimizing the travelled distance was taking (F4 – F3) as an objective function. However, that strategy is not ideal since these two parameters have different orders of magnitude. This kind of approach was applied simply to test the model in distinct ways, so it could be possible to check for possible errors on its modelling process.

Two database files were created to test the developed code: "Data_v1" and "Data_v2". Besides, a copy of "Data_v1" was made containing a few parameters that could allow time constraints to be relaxed, which is going to be addressed later. It was called "Data_v1_relaxed".

Therefore, these instances allowed the validation of the model in different steps. Firstly, after the mathematical modelling was done, the model was simulated with these instances so it would be possible to search for inconsistencies. Secondly, it was performed the objective function analysis of the present section. Finally, these instances allowed a pre-simulation process using the same objective function as the constructive heuristic from Ramdane and Jaballah (2021).

The parameters of these initial instances were obtained by assumptions made by Ramdane and Jaballah (2021) when they were testing their heuristic as well. The following tables present the data from these databases:

Table 5: Description and values from databases

Parameter	Description	Value of *Data_v1*	Value of *Data_v1_relaxed*	Value of *Data_v2*
V	Set of job sites from **0** to **n**, being 0 the warehouse	[0 1 2 3 4]	[0 1 2 3 4]	[0 1 2 3 4 5 6 7]
H	Set of vehicles	[1 2 3]	[1 2 3]	[1 2 3 4 5 6 7]
V0	Set of job sites without warehouse	[1 2 3 4]	[1 2 3 4]	[1 2 3 4 5 6 7]
A	Set of time windows	[1 2 3]	[1 2 3]	[1 2 3]
R	Set of trips (c)	[1 2 3]	[1 2 3]	[1 2 3]
Rc	Set of trips excluding the last one (c-1)	[1 2]	[1 2]	[1 2]
Qk	Vehicle capacity in pallets	[15 15 20]	[15 15 20]	[15 15 15 15 20 20 20]
Dk	Vehicle capacity in kilometers	[600 600 700]	[600 600 700]	[600 600 600 600 700 700 700]
Tk	Vehicle capacity in working hours	[9 9 6.5]	[12 12 9]	[9 9 9 9 6.5 6.5 6.5]
Dij	Distance matrix between construction sites	Table 6	Table 6	Table 6
QDi	Delivery demand by site	[15 20 24 28]	[15 20 24 28]	[15 20 24 28 8 21 11]
QPi	Pickup demand by site	[6 4 4 10]	[6 4 4 10]	[6 4 4 10 5 5 6]

TWe	Time matrix of earliest times of different time windows by job site (depot included)	[7 0 0 7 12 17 7 0 17 7 12 17 7 0 0]	[7 0 0 7 0 0 7 0 0 7 0 0 7 0 0]	Table 7
TWl	Time matrix of beginning times of different time windows by job site (depot included)	[19 0 0 8 13 18 8 0 18 8 13 18 8 0 0]	[19 0 0 19 0 0 19 0 0 19 0 0 19 0 0]	Table 7
Tij	Transport time matrix between sites	Table 8	Table 8	Table 8
Si	Service duration at job sites (depot included)	[1 0.5 0.5 0.5 0.5]	[1 0.5 0.5 0.5 0.5]	[1 0.5 0.5 0.5 0.5 0.5 0.5 0.5]
PDi	Unit delivery profit associated to each site's demand	[50 50 20 50]	[50 50 20 50]	[50 50 20 50 20 20 20]
PPi	Unit pickup profit associated to each site's demand	[20 20 10 20]	[20 20 10 20]	[20 20 10 20 10 10 10]

Source: (RAMDANE; JABALLAH, 2021)

The parameters that were modified to create the “Data_v1” copy (“Data_v1_relaxed”) were highlighted in pale orange in the previous table. They were used with the purpose of relaxing time constraints, that were blocking results in some cases, which will be addressed later.

Table 6: Matrix of distances between job sites (km)

	Platform	Site 1	Site 2	Site 3	Site 4	Site 5	Site 6	Site 7
Platform	0,0	120,3	90,5	140,0	196,2	60,0	95,3	117,0
Site 1	120,3	0,0	50,0	156,6	103,0	163,8	110,0	213,0
Site 2	90,5	50,0	0,0	75,0	155,0	19,0	122,0	69,0
Site 3	140,0	156,6	75,0	0,0	70,3	147,0	68,0	132,0
Site 4	196,2	103,0	155,0	70,3	0,0	45,0	180,0	28,0
Site 5	60,0	163,8	19,0	147,0	45,0	0,0	80,0	111,0
Site 6	95,3	110,0	122,0	68,0	180,0	80,0	0,0	158,9
Site 7	117,0	213,0	69,0	132,0	28,0	111,0	158,9	0,0

Source: (RAMDANE; JABALLAH, 2021)

Table 7: Time windows

Platform	07:00-19:00
Site 1	07:00-8:00 ,12:00-13:00, 17:00-18:00
Site 2	07:00-8:00,17:00-18:00
Site 3	07:00-8:00 ,12:00-13:00, 17:00-18:00
Site 4	07:00-8:00
Site 5	07:00-19:00
Site 6	07:00-8:00 ,12:00-13:00, 17:00-18:00
Site 7	07:00-8:00, 17:00-18:00

Source: (RAMDANE; JABALLAH, 2021)

Table 8: Matrix of estimated transport time between job sites

	Platform	Site 1	Site 2	Site 3	Site 4	Site 5	Site 6	Site 7
Platform	0,0	2,2	1,6	2,5	3,6	1,1	1,7	2,1
Site 1	2,2	0,0	0,9	2,8	1,9	3,0	2,0	3,9
Site 2	1,6	0,9	0,0	1,4	2,8	0,3	2,2	1,3
Site 3	2,5	2,8	1,4	0,0	1,3	2,7	1,2	2,4
Site 4	3,6	1,9	2,8	1,3	0,0	0,8	3,3	0,5
Site 5	1,1	3,0	0,3	2,7	0,8	0,0	1,5	2,0
Site 6	1,7	2,0	2,2	1,2	3,3	1,5	0,0	2,9
Site 7	2,1	3,9	1,3	2,4	0,5	2,0	2,9	0,0

Source: (RAMDANE; JABALLAH, 2021)

It is possible to note that the first database consists of a simplified instance with 4 construction sites and 3 vehicles, while the second one is a more complex instance with 7 construction sites and 7 vehicles. Some estimations were made through actual parameters. For example, the travel time was calculated with the given distances between sites and the average speeds of each vehicle (worth 55 km/h). The demand for the different kits and big-bags has been assembled on a single pickup and delivery demand. When it comes to service duration, an estimate was also made, as it usually depends on the quantity of pallets involved in each service, with a different vehicle loading time than that of unloading. So, it was used an average estimated service duration of how much time it would take to load/unload the trucks according to the average demand, which resulted in half an hour.

The detailed results from this objective function analysis are displayed on the Annex 1 of this work. In general terms, it allowed the model to be assessed in different ways according to possible outcomes that could be wanted. That is to say that an objective function analysis was performed through simulations using the four different functions described before. The goal was to search possible objective formulations for the problem concerning the logistic platform's needs, such as the minimization of the number of trips performed by vehicles, the minimization of total distance or transport time, and the maximization of profits.

The objective function analysis proved that the models could find feasible solutions on simpler objective functions. Additionally, it showed the need of a different approach when combining objective functions. There are some strategies addressing multi-objective models, but in the scope of this work it will be only addressed the normalization of different functions since it was the strategy adopted by Ramdane and Jaballah (2021). Therefore, the objective function analysis was important as it allowed us to simulate models with a simpler feasible objective function just as F4. In that way, the models were validated to the pre-simulation, in which a more complex objective function will be used.

3.3.2. Pre-simulation

After validating the model with simpler objective functions, the simulation was run with the actual objective function conceived by Ramdane and Jaballah (2021) and the estimated instances. In that way, the validation of the model would be complete.

The results are displayed according to each model (PDPSPMTW or MTPDPSPMTW, abbreviated as MTPDP and PDP) and each instance ("Data_v1", "Data_v1_relaxed" or "Data_v2"). The main results collected of each model encompasses some key performance indicators, such as the computing time, total profit, total distance and percentage of satisfied demands.

Table 9: Pre-simulation results

	Solution		Computing time (s)		Total profit		Total distance (km)		Satisfied demands	
	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP
Data_v1	Optimal	Optimal	0,0	0,0	87	87	240,6	240,6	18,9%	18,9%
Data_v1 relaxed	Optimal	Optimal	0,2	0,9	294	319	828,1	828,1	66,7%	71,2%
Data_v2	Optimal	Optimal	0,4	0,8	197	197	831,2	831,2	49,7%	49,7%

Source: author

The results indicate a good performance of the models. They were able to find only optimal solutions in a very short time.

Concerning the performance related to percentage of satisfied demands (that is also related to profit and distance), there are a few remarks to make. Regarding "Data_v1", both models presented a low rate of demands that were met. This can be explained due to time windows and time capacity parameters, as "Data_v1_relaxed" instance had these parameters relaxed and presented a better solution. Thus, a possible interpretation to make out of this comparison is that these parameters can interfere severely with the results. Moreover, as this simulation was run with mostly estimated parameters, it could be possible that they were not quite accurate. Therefore, this can be a focal point to the simulation with real parameters. If the instances conceived by the construction companies do not allow the simulation code to obtain expressive results, it would be necessary to reassess the problem definition so some time constraints could be reformulated.

Comparing the performances of PDPSPMTW and MTPDPSPMTW, there was only a slight difference of a 4,5% increase in satisfied demands in “Data_v1_relaxed”. That was led by the fact that MTPDPSPMTW performed two trips of a vehicle with greater capacity than the others. Analyzing the results in detail, PDPSPMTW used all its 3 vehicles, while MTPDPSPMTW used 2 vehicles, having one of them (third vehicle on Table 5) performing 2 trips. Even though, in total, both models completed 3 trips in “Data_v1_relaxed”, MTPDPSPMTW chose to use a vehicle with greater capacity twice, being able to satisfy more demands.

Although time windows and time capacities were relaxed in “Data_v1_relaxed”, the optimized solutions did not satisfy all clients. For PDPSPMTW, a possible hypothesis could be the fact that there were not enough vehicles to obtain more profit (since each one performs a single trip). However, MTPDPSPMTW obtained almost the same solution. That can be explained because of the distance expression used in the objective function. Hence, the models will always return results that try to travel as little as possible obtaining as much profit as possible. The PDPSPMTW model had already satisfied enough demands so the solution would be optimal when it comes to the distance-profit trade-off.

Regarding “Data_v2”, both models were able to satisfy around half of the demands. Although MTPDPSPMTW could perform multiple trips, the optimal result did not change from PDPSPMTW. This could be explained by the fact that time windows and time capacities constraints were possibly overestimated or not quite accurately estimated. In order to see an improvement of the results obtained by MTPDPSPMTW, it would be necessary to relax time constraints by changing its time parameters, just as it has been made for “Data_v1_relaxed”.

In summary, both models performed well with estimated values. They were able to find feasible solutions for previous test instances. The use of Multi-trips was proven worthwhile in one of the instances as it led to an increase on models’ performance indicators. However, a focal point found by this pre-simulation was the fact that some time constraints (such as the time capacity of vehicles and time windows of job sites) could impact severely on the solution of the problem. As discussed before, in the case of hard time parameters, the use of multiple trips was not useful since the vehicles could not travel anyway. By simulating with real parameters, it is expected that this issue would not happen and that performing multiple trips would be advantageous.

4. DATA COLLECTION METHODOLOGY

The first step to simulate was the data recovery to compose the real parameters of the model. As stated previously, there are 20 companies of the construction sector that are involved in the DILC project. They provided data so the software of the pooling platform could be developed. Thus, Ramdane and Jaballah (2021) were able to develop their constructive heuristic. They provided their database so the simulation of this work could be performed with actual parameters from the industry.

However, the instances provided by them were in a specific format that *Xpress IVE* could not read easily. There were some parameters to be retrieved and treated before use, such as the demands, which were expressed by kits, being each kit expressed in pallets. So, it was necessary to assemble each job site demand in a single palletized delivery/pickup demand. Moreover, there were data that was unnecessary to the scope of this work.

An important process that was required to process data was due the fact that there was data available concerning demands that were delayed. That is to say that, if the platform did not serve a given job site in a specific day D, this demand would be accumulated for the next day D+1. However, its priority might change. For instance, if a demand on day D had not a high priority for the platform (translated in a high profit), the solver might not deal with it. On the following day, this demand could be treated as a high priority demand since it was delayed. So, a given job site could have two demands with two priorities: the one from the past day (generally with a high priority), and the one from the actual day (with either a high or low priority). Thus, it was necessary to find a way of how to deal with delayed demands without changing parameters and variables from the model. The solution found was creating a fictional client when a given client had two types of demand (delayed and actual demands). This client's parameters would be copied into a new fictional client, except for the demand and profit parameters, which would be assigned with the ones from the delayed demand. Every step from this data recovery process is commented on the code presented in Annex 2.

Therefore, a *Python* program was developed to perform the data processing. Firstly, the program needed to open the original instance file, then it was read and treated like a string. Secondly, the program went through all lines of this string, checking if a specific data were to be found. When a parameter matched the search, then the program retrieved it, treated it if necessary, and then collected it by saving in a new ".*dat*" file (*Xpress IVE* database file). Two *Python* files were developed in order to treat data that would be used to PDPSPMTW and MTPDPSPMTW models. The full code can be observed in the Annex 2 of this work. It was decided to let only the MTPDPSPMTW code, as it would be repetitive to put on this document PDPSPMTW's code, which is very similar.

After running the data processing codes, it was possible to obtain 20 instances. Half of them is composed of 5 construction sites and 2 vehicles, and the other half by 10 construction sites and 3 vehicles. Unfortunately, the MTPDPSPMTW model was not able to run with the 10 construction sites instances due to solver's capacity. The parameters of these instances are presented in the following table. Since it would get difficult to display all data, it was preferred to let intervals or a set of values found for each parameter.

Table 10: Instances parameters

	Construction sites	Vehicles	Capacity in pallets	Capacity in time (h)	Capacity in distance (km)	Pickup demand (pallets)	Delivery demand (pallets)	Pikcup profit by pallet	Delivery profit by pallet	Time windows	Transport time (h)	Service duration (h)	Distance between sites (km)
Instance\Parameter	V	H	Qk	Tk	Dk	QPi	QDi	PPi	PDi	TWe TWl	Tij	Si	Dij
1 - 10	5	2	16	[12 16]	[500 800]	[0 9]	[5 9]	{1, 2}	{2, 5}	*Depot:* [05 21] Sites: Up to 4 TWs [06 09] [10 13] [14 16] [17 20]	[00 1.6]	[0.27 0.42]	[00 98.71]
101 - 110	10	3	16	[12 16]	[500 800]	[1 9]	[5 8]	{1, 2}	{2, 5}	*Depot:* [05 21] Sites: Up to 4 TWs [06 09] [10 13] [14 16] [17 20]	[00 1.75]	[0.27 0.42]	[00 99.4]

Source: DILC instances

5. SIMULATION

As stated before, the simulation of the developed models was performed on *Xpress IVE*. This was an iterative process, where the model had to be adapted according to the simulation responses, which sometimes pointed out mathematical mistakes on the model. Thus, the constraints had to be reformulated a few times.

The *Xpress* codes were divided into eight parts, concerning the creation of sets, parameters and variables, then the recovery of data (from a separate database file), then the creation of objective functions and constraints, and finally the launch of objective functions and the printing of results. They have been tested for the databases listed above. The full *Xpress IVE* code (in *Mosel* programming language) is exhibited on Annex 3 of this work. As the data processing code, it was decided to let in this work only the MTPDPSPMTW simulation code, since it would be repetitive to put on this document the PDPSPMTW code too, which is very similar.

The simulation was run with the instances previously presented in Table 10. In order to assess the results, performance indicators of the solution were calculated for each instance. The results are displayed in Table 11:

Table 11: Simulation results

	Solution		Computing time (s)		Total profit		Total distance (km)		Satisfied demands		Clients fully served		Clients partially served		Clients unserved	
	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP	PDP	MTPDP
1	Optimal	Optimal	0,4	2,9	147	147	344,27	344,27	100,0%	100,0%	5/5	5/5	0/5	0/5	0/5	0/5
2	Optimal	Optimal	0,3	5,0	181	252	269,15	459,81	64,4%	87,7%	1/5	4/5	1/5	0/5	3/5	1/5
3	Optimal	Optimal	0,3	12,5	168	234	216,35	362,81	61,1%	100,0%	1/5	5/5	2/5	0/5	2/5	0/5
4	Optimal	Optimal	0,4	1,9	166	166	397,02	397,02	100,0%	100,0%	5/5	5/5	0/5	0/5	0/5	0/5
5	Optimal	Optimal	0,2	4,2	145	214	375,80	665,23	68,0%	98,7%	3/5	4/5	2/5	1/5	0/5	0/5
6	Optimal	Optimal	0,4	1,7	145	145	390,35	335,86	100,0%	100,0%	5/5	5/5	0/5	0/5	0/5	0/5
7	Optimal	Optimal	0,3	2,3	129	200	305,67	503,52	62,7%	98,7%	2/5	4/5	2/5	1/5	1/5	0/5
8	Optimal	Optimal	1,1	253,4	198	334	352,06	597,83	55,7%	88,7%	1/5	3/5	1/5	1/5	3/5	1/5
9	Optimal	Optimal	0,8	5,4	169	273	86,59	272,55	54,3%	100,0%	2/5	5/5	2/5	0/5	1/5	0/5
10	Optimal	Optimal	0,2	5,2	161	222	315,45	551,21	71,6%	100,0%	3/5	5/5	1/5	0/5	1/5	0/5
101	Optimal	Best	8,7	7200,0	248	539	270,30	838,46	34,9%	97,6%	3/10	1/10	2/10	9/10	5/10	0/10
102	Optimal	Best	7,7	7200,0	213	235	455,48	610,39	89,2%	100,0%	7/10	10/10	2/10	0/10	1/10	0/10
103	Best	Best	7200,0	7200,0	240	280	437,91	575,20	83,3%	100,0%	7/10	10/10	3/10	0/10	0/10	0/10
104	Optimal	Best	63,7	7200,0	265	306	666,33	840,61	80,2%	96,6%	6/10	9/10	3/10	1/10	1/10	0/10
105	Optimal	Best	23,5	7200,0	228	264	410,08	586,32	83,7%	100,0%	7/10	10/10	2/10	0/10	1/10	0/10
106	Best	Best	7200,0	7200,0	274	312	337,49	440,35	80,3%	99,1%	6/10	9/10	2/10	1/10	1/10	0/10
107	Optimal	Best	70,9	7200,0	261	316	478,13	645,76	81,5%	99,1%	6/10	9/10	3/10	1/10	1/10	0/10
108	Best	Best	7200,0	7200,0	266	375	419,04	503,60	67,1%	97,9%	6/10	6/10	3/10	4/10	1/10	0/10
109	Optimal	Best	604,4	7200,0	268	292	388,05	445,00	88,5%	100,0%	8/10	10/10	1/10	0/10	1/10	0/10
110	Best	Best	7200,0	7200,0	285	467	408,03	748,89	55,5%	98,8%	6/10	8/10	1/10	2/10	3/10	0/10

Source: author

The solutions are displayed above through their performance indicators. Firstly, there is the type of solution, that shows whether it was an optimal solution or only the best one found during the simulation time. Secondly, there is the computing time, that measures the duration that the code was run to find a solution. If it did not find a solution in 2 hours, the code was stopped and the best solution found was retrieved from it. Thirdly, there is the total profit and total distance obtained by the solution. Then, the percentage of satisfied demands, which represents how many pallets were delivered/picked up from the total costumer's demand. And finally, there is the degree of service of clients, shown on the last three columns of Table 11. They were extracted so it would be possible to translate the percentage of client satisfaction into an actual number. If a given job site had all its demands supplied (either its delivery or its pickup demand), it means it was fully served. Meanwhile, if a job site had some of its demands satisfied, then it was partially served. Lastly, if a construction site were not attended by any vehicles, it was assigned as an unserved client.

The following paragraphs will address possible interpretations of the results presented above. Firstly, instances 1-10 results will be described, then the ones from instances 101-110.

Concerning the instances 1-10, which have 5 construction sites, all results obtained were optimal. In general, MTPDPSPMTW results obtained slightly higher computing times than PDPSPMTW's. Regarding the percentage of satisfied demands, PDPSPMTW were able to achieve 100% only for three instances (1, 4 and 5), while MTPDPSPMTW obtained this result to almost all instances. That shows the importance of performing multiple trips in order to optimize the resolution of the problem. Even though instances 1-10 had fewer clients, the use of multi-trips was proven useful. Taking instance 9 as an example, PDPSPMTW were able to achieve only 54,3% of demand satisfaction, meanwhile MTPDPSPMTW could supply 100% of the job sites. The indicators of profit and distance highlighted the same conclusion, since they are all linked.

When it comes to instances 101-110, which have 10 construction sites, it was necessary to set a fixed time of simulation as MTPDPSPMTW would take too long to find an optimal solution. Hence, it was chosen a 2 hours as maximum simulation time, which was defined by Ramdane and Jaballah (2021). Then, computing times for MTPDPSPMTW increased significantly. Again, concerning the satisfaction of clients' demands, the multiple trips model was proven better, achieving all results nearing the totality (100%), whilst PDPSPMTW had varied results, ranging from 34,9% to 89,2%. However, in order to obtain better results, it was necessary a considerable amount of time to run the code. Therefore, there is a trade-off related to how much is wanted to tune our resolution and how much time we are willing to employ to it.

After running the model with 20 different instances, it is now possible to better asses the issue addressed in the pre-simulation where PDPSPMTW and MTPDPSPMTW obtained the same results. Two possible causes for this issue were raised: the fact that time parameters were underestimated (in the sense that they restrained time constraints), and/or they were not quite accurate. After simulating with real instances, MTPDPSPMTW presented better results than PDPSPMTW in general, so this issue was ruled out. Hence, by performing a deep-dive analysis of time capacities and time windows parameters of real instances and estimated instances, it is possible to say that the two explanations are true: vehicles time capacities and the duration of time windows were underestimated (which restrained time constraints). For estimated instances, time windows had only a one-hour duration, while for actual instances they considered three hours. Time capacities for estimated instances ranged from 6 to 9 hours, whilst for actual instances from 12 to 16 hours.

Finally, the Table 12 displays a comparison between the results from MTPDPSPMTW and from the constructive heuristic developed by Ramdane and Jaballah (2021):

Table 12: Results from the heuristic and the MILP

	Solution		Objective		Computing time (s)		Total profit			Total distance (km)			Satisfied demands	
	Heuristic	MTPDP	Heuristic	MTPDP	Heuristic	MTPDP	Heuristic	MTPDP	Gap	Heuristic	MTPDP	Gap	Heuristic	MTPDP
1	Best	Optimal	-0,06	-0,74	0,1	2,9	147	147	0%	395,47	344,27	15%	100,0%	100,0%
2	Best	Optimal	-0,34	-0,56	0,1	5,0	285	252	13%	680,10	459,81	48%	100,0%	87,7%
3	Best	Optimal	0,08	-0,74	0,1	12,5	234	234	0%	528,29	362,81	46%	100,0%	100,0%
4	Best	Optimal	-0,02	-0,64	0,1	1,9	166	166	0%	504,27	397,02	27%	100,0%	100,0%
5	Best	Optimal	-0,05	-0,51	0,1	4,2	136	214	36%	580,68	665,23	13%	62,7%	98,7%
6	Best	Optimal	-0,15	-0,74	0,1	1,7	145	145	0%	452,34	335,86	35%	100,0%	100,0%
7	Best	Optimal	-0,04	-0,65	0,1	2,3	131	200	35%	513,28	503,52	2%	64,5%	98,7%
8	Best	Optimal	-0,25	-0,50	0,1	253,4	359	334	7%	923,01	597,83	54%	98,0%	88,7%
9	Best	Optimal	-0,34	-0,79	0,1	5,4	196	273	28%	251,50	272,55	8%	71,0%	100,0%
10	Best	Optimal	-0,02	-0,63	0,1	5,2	150	222	32%	549,46	551,21	0%	67,5%	100,0%
101	Best	Best	-0,01	-0,51	0,1	7200,0	461	539	14%	1001,58	838,46	19%	83,6%	97,6%
102	Best	Best	-0,06	-0,66	0,1	7200,0	235	235	0%	875,01	610,39	43%	100,0%	100,0%
103	Best	Best	-0,18	-0,74	0,1	7200,0	278	280	1%	777,35	575,20	35%	99,3%	100,0%
104	Best	Best	-0,19	-0,51	0,1	7200,0	312	306	2%	1044,26	840,61	24%	98,4%	96,6%
105	Best	Best	-0,09	-0,70	0,1	7200,0	260	264	2%	870,20	586,32	48%	98,5%	100,0%
106	Best	Best	-0,19	-0,70	0,1	7200,0	306	312	2%	592,44	440,35	35%	97,1%	99,1%
107	Best	Best	-0,14	-0,69	0,1	7200,0	317	316	0%	1004,59	645,76	56%	100,0%	99,1%
108	Best	Best	-0,16	-0,72	0,1	7200,0	300	375	20%	781,89	503,60	55%	78,3%	97,9%
109	Best	Best	0,08	-0,74	0,1	7200,0	292	292	0%	619,22	445,00	39%	100,0%	100,0%
110	Best	Best	-0,20	-0,64	0,1	7200,0	367	467	21%	922,46	748,89	23%	77,5%	98,8%

Source: author and (RAMDANE; JABALLAH, 2021)

The performance indicators displayed are similar to the ones of the Table 11. The only difference is that it was calculated the gap of the profit and the distance obtained between the MTPDPSPMTW and the heuristic. This gap represents a percentage of how distant the heuristic results are from the optimality. Besides, the Table 11 presents also the result obtained from the objective function of each model.

Concerning the type of solution found, MTPDPSPMTW was able to find optimal solutions only to small instances (1-10). Regarding the computing time, the heuristic shown itself much more efficient. However, it is not known when the simulation of MTPDPSPMTW actually find the best solution which was shown in the table, since it was set a maximum solving time on *Xpress*. Hence, even though it is not quite possible to compare computing times, it is still clear that the heuristic finds results more efficiently (in less than a second every time).

When it comes to the percentage of satisfied demands, both models performed similarly. Nonetheless, the travelled distance indicator highlighted a focus point because the models performed differently. The heuristic's total distance was in average 30% higher, which was calculated from the average of the gaps between them. Although it might seem considerable, there is always an acceptance of how much the decision maker is willing to be away from the optimality by using a more efficient model as the heuristic.

Regarding the profit, the heuristic performed properly by obtaining a 10% average gap. There are only a few instances where the profit obtained for both models was not almost the same (5, 7, 9, 10, 108 and 110). If we consider them as outliers, the average gap decreases up to 4%. However, since it occurred in several instances it would not be quite possible to classify them as outliers.

In summary, the constructive heuristic developed by Ramdane and Jaballah (2021) obtained similar results concerning some model indicators, such as the profit and the satisfaction of demands. The heuristic obtained results much faster than the MILP model. However, there is a considerable gap between models regarding the indicator of the total distance travelled. In general, it is possible to affirm that the heuristic was validated by this work. Even though there were some gaps, the heuristic is invariably more efficient in computing time. Most importantly, both models obtained consistent results. A possible outcome from this analysis would be reassessing the heuristic algorithm in order to search what could possibly affect the total distance. Or still, it would be conceivable to re-examine if both models took the same assumptions in all 23 constraints (even though this was already made several times).

6. CONCLUSION

It is important to note that the construction sector plays a significant role in the economy, and that it has a complex logistics network. This can lead to very high costs and significant greenhouse gas emissions, which this work wants to minimize. In this way, this subject is well qualified in the context of a capstone project for an engineering student specialized in optimization and supply chain.

The DILC project was responsible for the proposition of this work. It is characterized by a multidisciplinary taskforce involving several actors from France, such as non-profit organizations, government agencies and the private sector. This partnership intended to create the logistic platform discussed in this study, which consolidates raw materials of different suppliers in a distribution center by sorting them into palletized kits to deliver construction sites. The platform also has a center of waste treatment, collecting waste from job sites and sorting it into recycling, reusable, or disposable waste. In order to perform this pickup and delivery problem, the platform has its own vehicle fleet, whose route and cargo are optimized by this work.

Thus, the problem was addressed according to known constraints from the literature, which were not combined all together until now by Ramdane and Jaballah (2021). These constraints involve the multiple trips constraint, involving the possibility of trucks to perform several trips in a single working day; the split loads constraint, which allows the demand of a jobs site to be delivered/picked up partially (by more than one vehicle if necessary); the multiple time windows constraint, that limit vehicles services in the construction sites to happen only during certain times of the day. Therefore, a Multi-Trip Pickup and Delivery Problem with Split loads, Profit and Multiple Time Windows (MTPDPSPMTW) was designed to address the problem.

Through Ramdane and Jaballah's work, the present study was designed. They had developed a constructive heuristic to solve this problem. In a form of validation of their work, it was necessary to simulate the problem through mixed integer linear programming. In that way, by using small instances, the present work was designed so it would be possible to check if the results of both approaches were similar enough. This would prove that the constructive heuristic is validated, and it is usable to bigger instances (which this work could not address with a solver).

After fulfilling a state-of-the-art, it was noted that the MTPDPSPMTW problem is new in the literature. It was also possible to find relevant articles to the modeling and simulation of this problem.

The process of modeling counted with many parameters, variables and constraints, consisting of a complex mixed integer linear problem. Firstly, it was presented a simpler model, PDPSPMTW, which did not consider the multiple trips constraint since it would increase considerably the degree of complexity of the problem. Then, a complete model was presented, the MTPDPSPMTW. It was necessary to use several binary approaches in order to fulfill causality constraints during the modelling process, inspired by (Winston, 2003). The models were validated with 3 instances containing estimated parameters.

Afterwards, it was necessary to collect data to produce instances for the simulation. That was done by the DILC project by contacting 20 companies of the construction sector involved in this partnership. In that way, Ramdane and Jaballah (2021) already had databases containing real data to simulate the models. However, this database was in a specific format, which could not be used in *Xpress IVE* without performing data processing/cleansing. Thus, these processes were done in *Python*.

After validating the model and collecting data, the simulation with 20 instances was run. The main conclusion from the simulation was the validation of Ramdane and Jaballah (2021)'s constructive heuristic, which performed well in comparison to an exact model such as the one of this work. But there is still an opportunity to solve a few considerable gaps between models concerning the total distance travelled. To that end, some possible tasks would be reassessing the heuristic, comparing the use of all parameters, reviewing constraints assumptions, etc. Though this distance gap, their algorithm is very well suited to solve this problem for bigger instances.

To conclude, it is important to list which would be the next steps to improve this research. Firstly, it would possible to look for other parameters, which were not taken into account. For example, on this model it is feasible to not satisfy all demands, but there is no penalty to be charged if we do so. It would be interesting to add a stockout cost as a penalty for shortage of inventory.

Another example is that the model has a constant $\mathbf{t_{i,j}}$, while the vehicle fleet is heterogeneous. This would not be quite reasonable because of the fact that different vehicles have different average speeds. This issue would be easily adjustable if we add an extra index to **t** in order to assign a travel time between two job sites for a specific vehicle ($t_{i,j,k}$). However, it is true that this parameter is highly dependent on the level of loading of the vehicle, which changes on each trip. Hence, it could be necessary to transform this parameter in something proportional to the level of load of a vehicle. Nevertheless, this could imply in non-linear constraints by having a floating $t_{i,j,k}$, that depends on the $y'_{i,k,r}$ (vehicle loading level on each transport) . There are constraints that could end up in a non-linearity context, such as **(9)**, that would imply a multiplication between $t_{i,j,k}$, $y'_{i,k}$ and $x_{i,j,k}$.

Another parameter that is not fixed and it was necessary to approximate its values is the service duration (s_i). Such as $t_{i,j}$, s_i depends on a variable. The duration of the service is proportional to the number of pallets being delivered/picked-up at a job site. So, it would be necessary to associate s_i to $y_{i,k}$ and $z_{i,k}$, which would cause a non-linearity context to some constraints, such as **(9)** or **(14)**.

Concerning the objective function applied on this work, there are a few remarks to be done. As previously discussed, the function was conceived in a way that it could be possible to tackle two fronts: the maximization of profits and the minimization of distances. That choice was made by Ramdane and Jaballah (2021), since the DILC project designed the logistic platform as a sustainable business model. The approach used in this work to act on both goals was the normalization of the optimizing expressions that calculate profit and distance, allowing us to compare them. A similar approach would be assigning a certain weight to one of these expressions, so we could give priority whether to profit or to distance.

However, there are some different strategies to address multi-objective models on the literature. For instance, a possible approach would be the epsilon-constraint method (CHANKONG; HAIMES, 1983). The Pareto set concept is used in this strategy, which is the set of optimal solutions considered equally good, where none of the objective functions can be improved in value without degrading some of the other objective values. As this set could be possibly infinite, there is a goal of finding a representative set of Pareto optimal solutions, addressing trade-offs in satisfying different objective or finding a single solution that satisfies the subjective preferences of a given person (decision maker) (LAUMANNS et al., 2006). The epsilon-constraint method is defined as a form of addressing this multi-objective issue, where a single objective function is chosen to be minimized and the others are converted into inequality constraints. The bound of these constraints is a given constant (epsilon) which will be iteratively increased until the feasibility of the problem allows it (CHANKONG; HAIMES, 1983). Thereby, this strategy could be applied in this work as a form of addressing more than one objective without mixing them in the same objective function.

Finally, it would be interesting to simulate the model with instances of real examples of other sectors beyond the construction sector since the DILC project foresees the use of the logistic platform with different industries in the future. Therefore, it would be necessary to simulate the model, compare its results with the construction sector and verify its relevance. In this manner, it would be possible to note the logistics optimization of other realistic scenarios.

7. REFERENCES

(1) RAMDANE, JABALLAH. **Multi-trip pickup and delivery problem, with split loads, profits and multiple time windows to model a real case problem in the construction industry**. 2021.

(2) HAMMER, PIVO. **The Triple Bottom Line and Sustainable Economic Development Theory and Practice**. 2016.

(3) AMATO NETO, CAMPOS-SILVA, SOUSA MELO. **Innovation in eco-industrial parks: a comparative review of case studies.** 2018.

(4) FE2i. **Actualités > Découvrez nos actions coopératives en vidéo**. 2020. Available at: < https://www.fe2i.fr/decouvrez-notre-nouvelle-video/> (Accessed 08/09/2021)

(5) INSEE. **Tableaux de l'économie française**. 2018. Available at: <https://www.insee.fr/fr/statistiques/3303618?sommaire=3353488#graphique-T18F184G3> (Accessed 03/12/2020)

(6) NOWAK. **The Pickup and Delivery Problem with Split Loads**. 2005.

(7) WANG. **Vehicle Routing Problem: Simultaneous Deliveries and Pickups with Split Loads and Time Windows**. 2013.

(8) WASSAN, NAGY. **Vehicle Routing Problem with Deliveries and Pickups: Modelling Issues and Meta-heuristics Solution Approaches**. 2014.

(9) WASSAN, NAGY. **The Vehicle Routing Problem with Divisible Deliveries and Pickups**. 2015.

(10) CHEN. **Model and algorithm for an unpaired pickup and delivery vehicle routing problem with split loads**. 2014.

(11) LI. **Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests**. 2016.

(12) NGUYEN. **Multi-trip pickup and delivery problem with time windows and synchronization**. 2016.

(13) CHENTLI. **A selective adaptive large neighborhood search heuristic for the profitable tour problem with simultaneous pickup and delivery services**. 2018.

(14) ZHIYUAN, WANTAO. **The Vehicle Scheduling Problem of Third-Party Passenger Finished Vehicle Logistics Transportation: Formulation, Algorithms, and Instances.** 2015.

(15) RÍOS-MERCADO. **A GRASP for a multi-depot multi-commodity pickup and delivery problem with time windows and heterogeneous fleet in the bottled beverage industry**. 2013.

(16) CATTARUZZA, ABSI, FEILLET. **Vehicle routing problems with multiple trips.** 2016.

(17) YOSHIZAKI, BELFIORE. **Scatter search for vehicle routing problem with time windows and split deliveries**. 2006.

(18) LEAL, PÉREZ SALAS. **Logistic platforms: conceptual elements and the role of the public sector**. 2009.

(19) WINSTON. **Operations Research Applications and Algorithms, 4th. Edition.** p477-564. 2003.

(20) LAUMANNS, THIELE, ZITZLER. **An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method.** 2006.

(21) CHANKONG, HAIMES. **Multiobjective Decision Making Theory and Methodology**, Elsevier. p274-276. 1983.

8. ANNEX

8.1. Annex 1: Results obtained from the validation of models

First of all, before analyzing the results, a few remarks must be made. The simulations that did not present such different results by varying the objective functions were not graphically represented. Moreover, the version of *Xpress IVE* used did not allow to perform the simulation of MTPDPSPMTW on "Data_v2", where the complexity is greater than the solver's capacity.

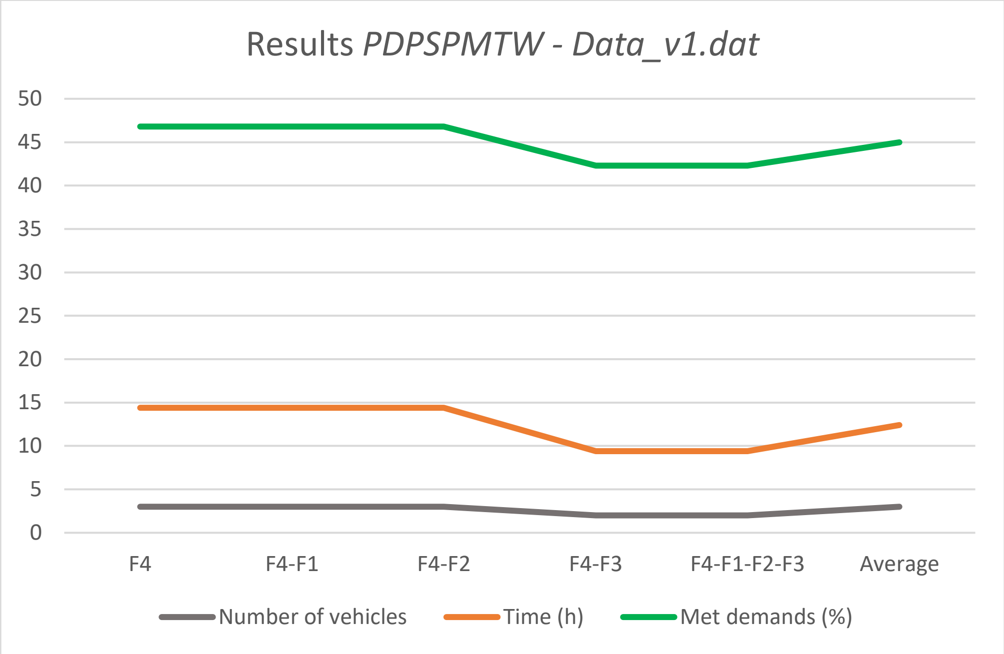
The results are presented according to the model (PDPSPMTW or MTPDPSPMTW), the database ("Data_v1", "Data_v2", or "Data_v1_relaxed") and the objective functions (F4, F3, F2 and F1).

Table 13: Simulation results from PDPSPMTW - Data_v1

Parameter	F4	F4-F1	F4-F2	F4-F3	F4-F1-F2-F3	Average
Number of vehicles	3	3	3	2	2	**3**
Time (h)	14,4	14,4	14,4	9,4	9,4	**12,4**
Distance travelled (km)	800,6	800,6	800,6	520,6	520,6	**688,6**
Profit (€)	1390	1390	1390	1310	1310	**1358**
Met demands (%)	46,8	46,8	46,8	42,3	42,3	**45**

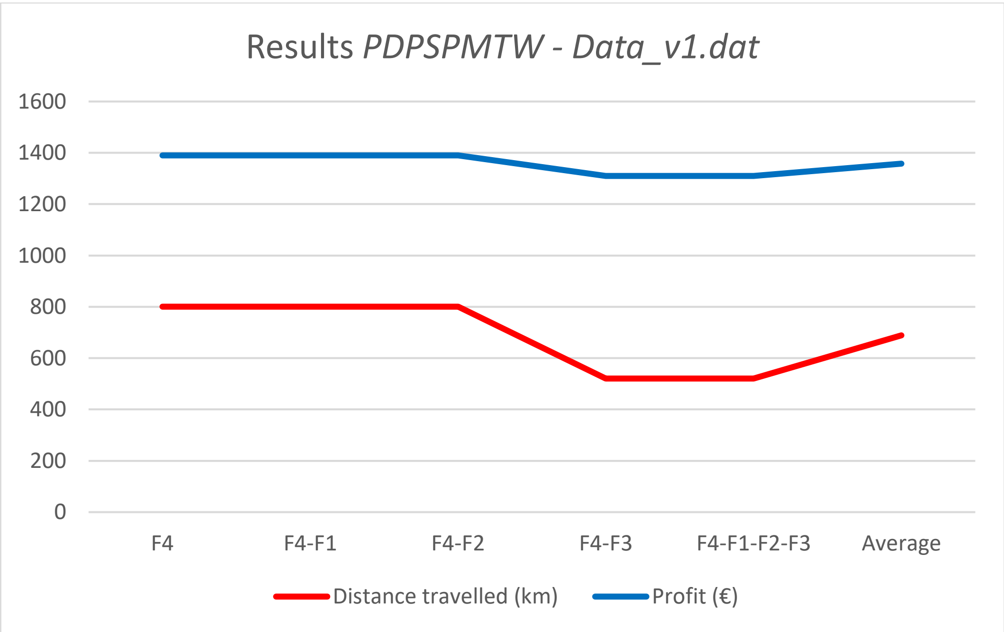
Source: author

Figure 9: Simulation results from PDPSPMTW - Data_v1



Source: author

Figure 10: Simulation results from PDPSPMTW - Data_v1



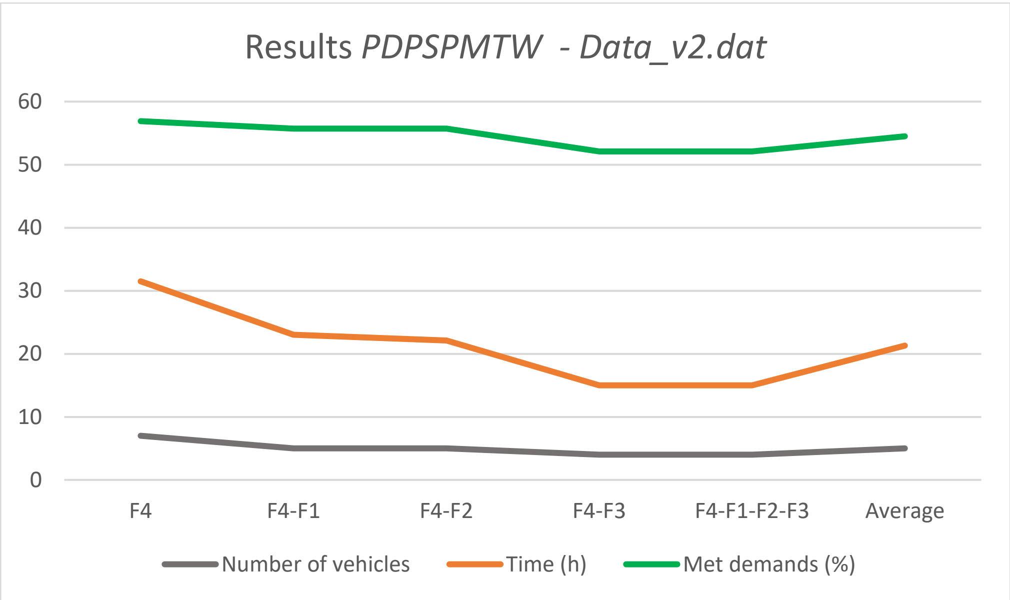
Source: author

Table 14: Simulation results from PDPSPMTW - Data_v2

Parameter	F4	F4-F1	F4-F2	F4-F3	F4-F1-F2-F3	Average
Number of vehicles	7	5	5	4	4	**5**
Time (h)	31,5	23	22,1	15	15	**21,32**
Distance travelled (km)	1747,1	1271,2	1226,5	831,2	831,2	**1181,44**
Profit (€)	2070	2070	2070	1970	1970	**2030**
Met demands (%)	56,9	55,7	55,7	52,1	52,1	**54,5**

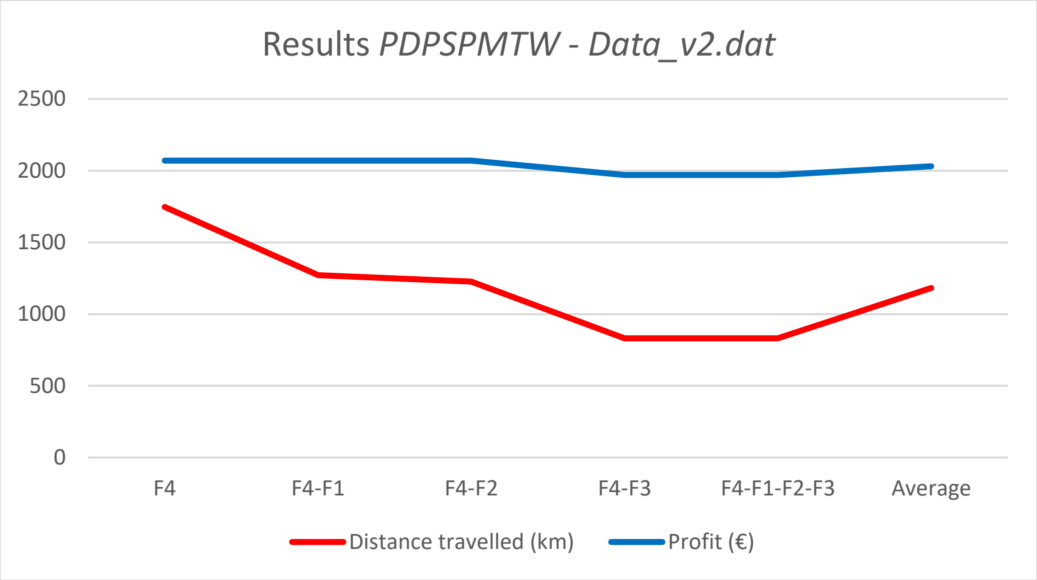
Source: author

Figure 11: Simulation results from PDPSPMTW - Data_v2



Source: author

Figure 12: Simulation results from PDPSPMTW - Data_v2



Source: author

Table 15: Simulation results from PDPSPMTW - Data_v1_relaxed

Parameter	F4	F4-F1	F4-F2	F4-F3	F4-F1-F2-F3	Average
Number of vehicles	3	3	3	3	3	**3**
Time (h)	20,1	19,5	15	15	15	**16,92**
Distance travelled (km)	1107	1065	828,1	828,1	828,1	**931,26**
Profit (€)	2940	2940	2940	2940	2940	**2940**
Demands met (%)	69,4	69,4	69,4	69,4	69,4	**69,4**

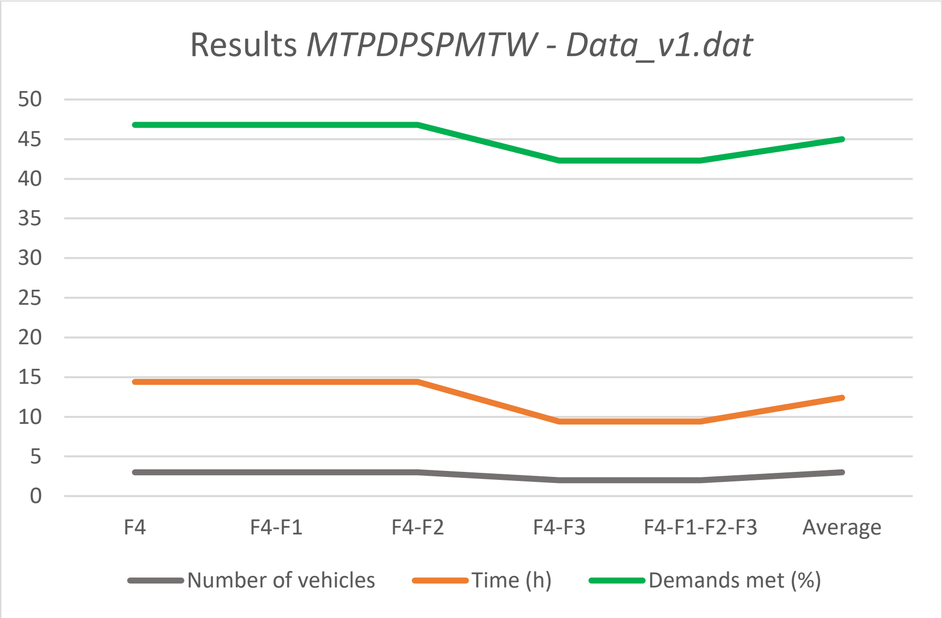
Source: author

Table 16: Simulation results from MTPDPSPMTW - Data_v1

Parameter	F4	F4-F1	F4-F2	F4-F3	F4-F1-F2-F3	Average
Number of vehicles	3	3	3	2	2	**3**
Time (h)	14,4	14,4	14,4	9,4	9,4	**12,4**
Distance travelled (km)	800,6	800,6	800,6	520,6	520,6	**688,6**
Profit (€)	1390	1390	1390	1310	1310	**1358**
Demands met (%)	46,8	46,8	46,8	42,3	42,3	**45**

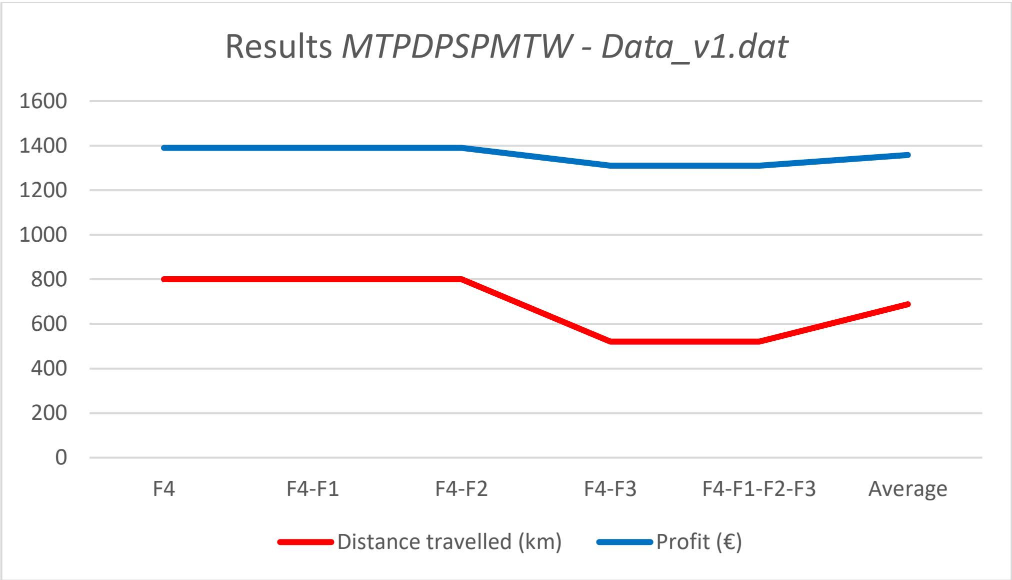
Source: author

Figure 13: Simulation results from MTPDPSPMTW - Data_v1



Source: author

Figure 14: Simulation results from MTPDPSPMTW - Data_v1



Source: author

Table 17: Simulation results from MTPDPSPMTW - Data_v1_relaxed

Parameter	F4	F4-F1	F4-F2	F4-F3	F4-F1-F2-F3	Average
Number of vehicles	4	4	4	4	4	**4**
Time (h)	23	23	22,7	22,7	22,7	**22,82**
Distance travelled (km)	1267,8	1267,8	1246	1246	1246	**1254,72**
Profit (€)	3630	3630	3630	3630	3630	**3630**
Demands met (%)	83,8	83,8	83,8	83,8	83,8	**83,8**

Source: author

Table 18: Comparison between models' averages

	PDPSPMTW Data_v1	MTPDPSPMTW Data_v1	PDPSPMTW Data_v1_relaxed	MTPDPSPMTW Data_v1_relaxed
Number of vehicles	3	3	3	4
Time (h)	12,4	12,4	16,9	22,8
Distance travelled (km)	688,6	688,6	931,3	1254,7
Profit (€)	1358	1358	2940	3630
Demands met (%)	45	45	69,4	83,8

Source: author

Concerning the results, it is possible to make an assessment of the use of different objective functions. The purpose of mixing them was to optimize different axes of the problem, such as profit and distance. However, this assessment was not quite possible to be made due to differences between orders of magnitude. It would be desirable that profit be maximized by minimizing either distance travelled, total working time, number of vehicles, or all previous options at once.

We can note that there were very slight changes between the use of F4, F4 – F1 and F4 – F2. This is due to the fact that the order of magnitude of F4 is larger than those of F1 and F2. This also applies to F4 – F3 and F4 – F1 – F2 – F3, which do not show different results according to the previous charts. Thus, it would be interesting to simulate these objective functions combined by normalizing them, just as the original objective function from (RAMDANE; JABALLAH, 2021).

In return, different results are obtained when comparing F4 and F4 – F3. If we combine them, we will have bigger changes. That is again due to the order of magnitudes, because F3 obtains higher values. For example, according to the data for PDPSPMTW "Data_v1", on F4 and F4 – F3 functions, we have a 35% decrease in time and distance travelled, while we have only a 5,8% decrease in profit. Indeed, if we look at the data for PDPSPMTW "Data_v2", on F4 and F4 – F3 functions we have a 52,4% decrease in time and travelled distance, while we have a 4,8% decrease in profit. And finally, for MTPDPSPMTW "Data_v1", there is a reduction of about 35% in distance and time, and 5,8% in profit if we compare the results of F4 and F4 – F3. Even though to an optimization model the use of F4 – F3 would not be convenient, this points out to the possibility of not damaging the profit result by using a decreasing factor. That is desirable to the model because the minimization of the distance plays an important role to the DILC project, where the sustainability is a strong factor. By reducing it, there could be a better connection between profits and significant transportation costs (either monetary or environmental costs).

It is also possible to observe differences in the use of Multi-trips by comparing the results from PDPSPMTW and MTPDPSPMTW at the same instance.

First of all, it is important to note the interest of relaxing the simulation parameters so that we can verify the impact of performing several trips. The capacity in working hours of each vehicle and the short time windows prevent the vehicles from performing more than a trip through the non-relaxed bases. Thus, identical results for different models at a same instance are seen in "PDPSPMTW - Data_v1" and "MTPDPSPMTW - Data_v1", since the vehicles failed to make several trips.

So, by checking "PDPSPMTW - Data_v1_relaxed" and "MTPDPSPMTW - Data_v1_relaxed", it is noticeable an increase of a trip. This leads to an increase of about 35% of the time and distance travelled by also increasing in 23,5% the profit, having 14,4% in addition to demands that were met.

It would be interesting to compare the results of the "Data_v2" for the Multi-trips model, because it is a more complete instance, having more job sites and vehicles. Unfortunately, due to the model complexity and the version of the solver, it was not possible to achieve it.

8.2. Annex 2: *Python* code developed for the data processing

MTPDPSPMTW file: “*data_processing – multi-trip.py*”

```
# -*- coding: utf-8 -*-
"""
@author: Fernando Augusto Martin Ferri
Capstone project Poli-USP
Wahiba Ramdane
Debora Ronconi
"""



def main():
    #Opening instance
    #Creating database to use in Xpress IVE
    file_name = input('Name of the instance: ')
    f = open(file_name,"r")
    inst = open("MT" + file_name + ".dat","w+")
    f = f.readlines()

    for x in f:
        if "NumberOfSites" in x:
            #V = list of job sites considering the depot
            V = []
            for word in x.split():
                    if word.isdigit():
                        for i in range(int(word)+1):
```

```
                    V.append(i)

            #V0 = list of job sites without considering the depot

            V0 = []

            for i in range(1,len(V)):

                V0.append(i)


        elif "VEHICULE_SECTION" in x:

            #H = list of vehicles

            H = []

            #Qk = capacity in pallets

            Qk = []

            #Tk = capacity in time

            Tk = []

            #Dk = capacity in distance

            Dk = []

            speed = 0

            number_of_vehicles = 0

            y = f.index(x)+1

            while ("Kits_SECTION" not in f[y]) and ("Ampliroll" not in
f[y]):

                draft_list = []

                for word in f[y].split():

                    draft_list.append(word)

                if draft_list[1] == '2': #choosing only palletized
vehicles, type 2

                    H.append(int(draft_list[0])+1)

                    Qk.append(float(draft_list[4]))

                    Tk.append(round(float(draft_list[-1]),2))

                    Dk.append(int(draft_list[5]))
```

```
            speed += float(draft_list[-3])

            number_of_vehicles += 1

        y += 1


    #finding average speed to calculate Ti,j

    average_speed = speed/number_of_vehicles


    #A = list of time windows

    test = False

    while test == False:

        try:

            A = int(input('Number of time windows (by default 
4): '))

            test = True

        except ValueError:

            print('\nType integer numbers only')

            test = False


    #R = list of trips

    test = False

    while test == False:

        try:

            R = int(input('Maximum number of trips (by default 
3): '))

            test = True

        except ValueError:

            print('\nType integer numbers only')

            test = False
```

```
        elif "PICK_UP_BIGBAG_SECTION" in x:
            #QPi = pickup demand in pallets
            QPi = []
            #PPi = pickup profit
            PPi = []
            y = f.index(x)+1
            while ("PICK_UP_TIPPER_SECTION" not in f[y]):
                draft_list = []
                for word in f[y].split():
                    draft_list.append(word)
                total_demand = 0
                for i in range(1,len(draft_list)-1): #excludes first
column (index column) and last column (priority column)
                    total_demand += int(draft_list[i])
                if draft_list[-1] == "True":
                    PPi.append(2)
                elif draft_list[-1] == "False":
                    PPi.append(1)
                else:
                    PPi.append(0)
                QPi.append(total_demand)
                y += 1


        elif "Kits_SECTION" in x:
            #finding how many pallets each kit has
            pallets = []
```

```
            y = f.index(x)+1
            while ("LOADING_UNLOADING_TIME_PALETT_TIPPER_SECTION" not
in f[y]):
                draft_list = []
                for word in f[y].split():
                    draft_list.append(word)
                pallets.append(int(draft_list[3]))
                y += 1



        elif "TimeWindow PlateForme" in x:
            TWe_warehouse = x.split()[2][0:2]
            TWl_warehouse = x.split()[2][6:8]



        elif "DEMAND_KIT_SECTION" in x:
            #QDi = delivery demand in pallets
            QDi = []
            #PDi = delivery profit
            PDi = []
            #TWe = earliest time of time windows
            TWe = []
            TWee = []
            TWee.append(TWe_warehouse)
            for i in range(1,A):
                TWee.append(0)
            TWe.append(TWee)
            #TWl = latest time of time windows
```

```
            TWl = []
            TWll = []
            TWll.append(TWl_warehouse)
            for i in range(1,A):
                TWll.append(0)
            TWl.append(TWll)
            y = f.index(x)+1
            while ("PICK_UP_BIGBAG_SECTION" not in f[y]):
                draft_list = []
                for word in f[y].split():
                    draft_list.append(word)
                #finding the demand of which kits the job sites needs
                kits = []
                z = 1
                while draft_list[z].isdigit():
                    kits.append(int(draft_list[z]))
                    z += 1
                #finding if the job site has a priority to be served
                for i in draft_list:
                    if i == "True":
                        PDi.append(5)
                    elif i == "False":
                        PDi.append(3)
                #combining the demand of different kits into a single
demand in pallets for a job site
                total_demand_i = 0
                for i in range(len(kits)):
                    total_demand_i += kits[i]*pallets[i]
```

```
                QDi.append(total_demand_i)

                #finding time windows

                for i in range(len(draft_list)):

                    if draft_list[i] == "True" or draft_list[i] ==
"False":

                        number_of_time_windows = int(draft_list[i+1])

                        TW = []

                        TWee = []

                        TWll = []

                        if number_of_time_windows == 1 and
draft_list[i+2] == '0-0': #if there is no time constraint for the job
site

                            TWee.append('0')

                            TWll.append('24')

                            for i in range(1,A):

                                TWee.append('0')

                                TWll.append('0')

                            TWe.append(TWee)

                            TWl.append(TWll)

                        else:

                            for j in range(i+2,
i+2+2*number_of_time_windows):

                                if len(draft_list[j]) == 11: #do not
consider flexible time windows "30-0"

                                    TW.append(draft_list[j])

                            for i in range(0,number_of_time_windows):

                                TWee.append(TW[i][0:2])

                                TWll.append(TW[i][6:8])

                            for i in range(0, A-
number_of_time_windows):
```

```
                    TWee.append('0')

                    TWll.append('0')

                TWe.append(TWee)

                TWl.append(TWll)

    y += 1


elif "DELAY_IN_PICK_UP_BIGBAG" in x: #retrieving delayed
pickup demands

    QPii=[]  #list of delayed pickup demands

    priority_delayed_pickup = []

    profit_delayed_pickup = []

    y = f.index(x)+1

    while ("DELAY_IN_PICK_UP_TIPPER" not in f[y]):

        draft_list = []

        for word in f[y].split():

            draft_list.append(word)

        qp=0

        for i in range(1,len(draft_list)-2):

            qp+= int(draft_list[i])

        QPii.append(qp)

        priority_delayed_pickup.append(draft_list[-2])

        if draft_list[-1] == "True":

            profit_delayed_pickup.append(2)

        elif draft_list[-1] == "False":

            profit_delayed_pickup.append(1)

        y += 1
```

```
        elif "DELAY_IN_DELIVERY" in x and "DELAY_IN_DELIVERY_Tipper" not in x: #retrieving delayed delivery demands
            QDii=[]  #list of delayed delivery demands
            priority_delayed_delivery = []
            profit_delayed_delivery = []
            y = f.index(x)+1
            while ("DELAY_IN_PICK_UP_BIGBAG" not in f[y]):
                draft_list = []
                for word in f[y].split():
                    draft_list.append(word)
                qd=0
                for i in range(1,len(draft_list)-2):
                    qd+= int(draft_list[i])*int(pallets[i-1])
                QDii.append(qd)
                priority_delayed_delivery.append(draft_list[-2])
                if draft_list[-1] == "True":
                    profit_delayed_delivery.append(5)
                elif draft_list[-1] == "False":
                    profit_delayed_delivery.append(3)
                y += 1


        elif "MATRIX_DISTANCE" in x:
            #Dij = distance matrix
            Dij = []
            #Tij = distance matrix
            Tij = []
            y = f.index(x)+1
```

```
            while ("WASTE_WEIGHT_SECTION" not in f[y]):
                dij = []
                for word in f[y].split():
                    dij.append(round(float(word),2))
                Dij.append(dij)
                y += 1
            #Calculating time matrix (Tij) by using average speed
            for i in Dij:
                tij =[]
                for j in i:
                    tij.append(round(j/average_speed,2))
                Tij.append(tij)


        elif "LOADING_UNLOADING_TIME_PALETT_TIPPER_SECTION" in x:
            #Si = service time in job site i
            Si = []

            loading_time = [] #loading time per pallets (given in
minutes)
            unloading_time = [] #unloading time per pallets (given in
minutes)

            y = f.index(x)+1
            while ("MATRIX_DISTANCE" not in f[y]):
                draft_list = []
                for word in f[y].split():
                    draft_list.append(word)
                loading_time.append(float(draft_list[1]))
```

```
                unloading_time.append(float(draft_list[2]))

                y += 1




    #Service time assumption:

    Qk_avg = sum(Qk)/len(Qk)

    Si.append(str(round(((Qk_avg*loading_time[0] +
Qk_avg*unloading_time[0])/2)/60,2))) #approximated service time to the
warehouse

    for i in range(len(QPi)):

        Si.append(round((loading_time[i+1]*QPi[i] +
unloading_time[i+1]*Qk_avg)/60,2))

        #QPi is used because usually the pickup demand is small, while
Qk is used because usually the trucks are loaded in full capacity to
deliver raw materials


    #Treating delayed demands

    for i in range(len(QDii)):

        if priority_delayed_delivery[i] == "True":   #if the delayed
demand should be treated

            if profit_delayed_delivery[i] == PDi[i]: #if normal and
delayed demands have the same profit, assemble them

                QDi[i] = int(QDii[i])+ int(QDi[i])

            else: #creating fictional client with same parameters
except delivery demand and profit

                V.append(V[-1]+1)

                V0.append(V0[-1]+1)

                QPi.append(0)

                QDi.append(QDii[i])

                PPi.append(0)
```

```
                PDi.append(profit_delayed_delivery[i])
                TWe.append(TWe[i+1])
                TWl.append(TWl[i+1])
                for j in range(len(Tij)):
                    Tij[j].append(Tij[j][i+1])
                Tij.append(Tij[i+1])
                Si.append(Si[i])
                for j in range(len(Dij)):
                    Dij[j].append(Dij[j][i+1])
                Dij.append(Dij[i])
    for i in range(len(QPii)):
        if priority_delayed_pickup[i] == "True":   #if the delayed 
demand should be treated
            if profit_delayed_pickup[i] == PPi[i]: #if normal and 
delayed demands have the same profit, assemble them
                QPi[i] = int(QPii[i])+ int(QPi[i])
            else:
                V.append(V[-1]+1)
                V0.append(V0[-1]+1)
                QPi.append(QPii[i])
                QDi.append(0)
                PPi.append(profit_delayed_pickup[i])
                PDi.append(0)
                TWe.append(TWe[i])
                TWl.append(TWl[i])
                for j in range(len(Tij)):
                    Tij[j].append(Tij[j][i+1])
                Tij.append(Tij[i+1])
                Si.append(Si[i])
```

```
            for j in range(len(Dij)):
                Dij[j].append(Dij[j][i+1])
            Dij.append(Dij[i])


#Printing a new .dat file with instance's parameters
list_V = ["V: ["]
for i in range(len(V)):
    list_V.append(str(V[i]))
    list_V.append(" ")
list_V.pop(-1)
list_V.append("]")
inst.writelines(list_V)


list_H = ["\nH: ["]
for i in range(len(H)):
    list_H.append(str(H[i]))
    list_H.append(' ')
list_H.pop(-1)
list_H.append("]")
inst.writelines(list_H)


list_V0 = ["\nV0: ["]
for i in range(len(V0)):
    list_V0.append(str(V0[i]))
    list_V0.append(" ")
list_V0.pop(-1)
list_V0.append("]")
inst.writelines(list_V0)
```

```
list_A = ["\nA: ["]
for i in range(1,A+1):
    list_A.append(str(i))
    list_A.append(' ')
list_A.pop(-1)
list_A.append("]")
inst.writelines(list_A)

list_R = ["\nR: ["]
for i in range(1,R+1):
    list_R.append(str(i))
    list_R.append(' ')
list_R.pop(-1)
list_R.append("]")
inst.writelines(list_R)

list_Rc = ["\nRc: ["]
for i in range(1,R):
    list_Rc.append(str(i))
    list_Rc.append(' ')
list_Rc.pop(-1)
list_Rc.append("]")
inst.writelines(list_Rc)

list_Qk = ["\nQk: ["]
for i in range(len(Qk)):
    list_Qk.append(str(Qk[i]))
```

```
    list_Qk.append(' ')
list_Qk.pop(-1)
list_Qk.append("]")
inst.writelines(list_Qk)

list_Tk = ["\nTk: ["]
for i in range(len(Tk)):
    list_Tk.append(str(Tk[i]))
    list_Tk.append(' ')
list_Tk.pop(-1)
list_Tk.append("]")
inst.writelines(list_Tk)

list_Dk = ["\nDk: ["]
for i in range(len(Dk)):
    list_Dk.append(str(Dk[i]))
    list_Dk.append(' ')
list_Dk.pop(-1)
list_Dk.append("]")
inst.writelines(list_Dk)

list_QPi = ["\nQPi: ["]
for i in range(len(QPi)):
    list_QPi.append(str(QPi[i]))
    list_QPi.append(' ')
list_QPi.pop(-1)
list_QPi.append("]")
inst.writelines(list_QPi)
```

```
list_QDi = ["\nQDi: ["]
for i in range(len(QDi)):
    list_QDi.append(str(QDi[i]))
    list_QDi.append(' ')
list_QDi.pop(-1)
list_QDi.append("]")
inst.writelines(list_QDi)

list_PPi = ["\nPPi: ["]
for i in range(len(PPi)):
    list_PPi.append(str(PPi[i]))
    list_PPi.append(' ')
list_PPi.pop(-1)
list_PPi.append("]")
inst.writelines(list_PPi)

list_PDi = ["\nPDi: ["]
for i in range(len(PDi)):
    list_PDi.append(str(PDi[i]))
    list_PDi.append(' ')
list_PDi.pop(-1)
list_PDi.append("]")
inst.writelines(list_PDi)

list_TWe = ["\nTWe: ["]
for i in TWe:
    for j in i:
```

```
        list_TWe.append(str(j))
        list_TWe.append(' ')
    list_TWe.append('\n')
list_TWe.pop(-1)
list_TWe.pop(-1)
list_TWe.append("]")
for i in list_TWe:
    inst.writelines(i)

list_TWl = ["\nTWl: ["]
for i in TWl:
    for j in i:
        list_TWl.append(str(j))
        list_TWl.append(' ')
    list_TWl.append('\n')
list_TWl.pop(-1)
list_TWl.pop(-1)
list_TWl.append("]")
for i in list_TWl:
    inst.writelines(i)

list_Tij = ["\nTij: ["]
for i in Tij:
    for j in i:
        list_Tij.append(str(j))
        list_Tij.append(' ')
    list_Tij.append('\n')
list_Tij.pop(-1)
```

```
    list_Tij.pop(-1)
    list_Tij.append("]")
    for i in list_Tij:
        inst.writelines(i)

    list_Si = ["\nSi: ["]
    for i in range(len(Si)):
        list_Si.append(str(Si[i]))
        list_Si.append(' ')
    list_Si.pop(-1)
    list_Si.append("]")
    inst.writelines(list_Si)

    list_Dij = ["\nDij: ["]
    for i in Dij:
        for j in i:
            list_Dij.append(str(j))
            list_Dij.append(' ')
        list_Dij.append('\n')
    list_Dij.pop(-1)
    list_Dij.pop(-1)
    list_Dij.append("]")
    for i in list_Dij:
        inst.writelines(i)

    inst.close()
main()
```

8.3. Annex 3: *Mosel* code developed for the simulation process

```
!@encoding CP1252

model MTPDPSPMTW

uses "mmxprs", "mmsvg"; !gain access to the Xpress-Optimizer solver


declarations

      ! Ensembles

V: set of integer                  ! i ou j (n chantiers + depot)

H: set of integer                  ! k (H vehicules)

V0:set of integer                  ! i ou j (n chantiers, sans le
depot)

A: set of integer                  ! alpha (nombre de fenetres
horaires)

R: set of integer                  ! r (nombre de tournees)

Rc:set of integer                  ! r (nombre de tournees moins 1)


      ! Parametres

Qk: array (H) of real        ! capacite en palettes des k vehicules

Tk: array (H) of real              ! capacite en Tij des k vehicules

Dk: array (H) of real              ! capacite en km des k vehicules


QPi: array (V0) of integer         ! demande de pickup en palettes
pour chaque chantier i

QDi: array(V0) of integer          ! demande de delivery en palettes
pour chaque chantier i


PPi: array (V0) of integer         ! profit de pickup de chaque
chantier
```

```
PDi: array (V0) of integer         ! profit de delivery de chaque
chantier


TWe: array (V,A) of real           ! horaire de debut de la fenetre
temporelle du chantier i

TWl: array (V,A) of real           ! horaire de fin de la fenetre
temporelle du chantier i


Tij: array(V,V) of real            ! Tij en heures du deplacement
entre deux chantiers

Si: array(V) of real               ! Tij en heures de
chargement/dechargement en i


Dij: array(V,V) of real            ! distance en km entre les
chantiers


      ! Variables

v: array (H,R) of mpvar            ! binaire montrant si le vehicule k
a realise la tourne r

x: array (V,V,H,R) of mpvar        ! binaire montrant si le vehicule k
realise le transport de i vers j dans sa tournee r

theta: array(V0,H,A,R)of mpvar     ! binaire montrant quelle fenetre
temporelle alpha le vehicule k utilise au chantier i dans sa tournee r

y: array (V0,H,R) of mpvar         ! reel de la quantite de palettes
livrees en i par le vehicule k dans sa tournee r

z: array(V0,H,R) of mpvar          ! reel de la quantite de palettes
ramassees en j par le vehicule k dans sa tournee r

yprime: array(V,H,R) of mpvar ! reel de la quantite restante de
palettes dans le vehicule k dans sa tournee r partant de i

b: array(V,H,R) of mpvar           ! reel de l'horaire de debut du Si
en j par le vehicule k dans sa tournee r
```

```
end-declarations


      ! Donnees
initializations from "MT110.dat" ! CHANGEZ ICI
      V H V0 A R Rc Qk Tk Dk QPi QDi PPi PDi TWe TWl Tij Si Dij
end-initializations


      ! Fonctions objectif
FO1:= sum(k in H, r in R)v(k,r)
                  ! min vehicules
FO2:= sum(k in H, i,j in V, r in R|i<>j)((Tij(i,j)+Si(i))*x(i,j,k,r))
      ! min temps
FO3:= sum(k in H, i,j in V, r in R|i<>j)Dij(i,j)*x(i,j,k,r)
      ! min distance
FO4:= sum(i in V0)PDi(i)*(sum(k in H, r in R)y(i,k,r)) +  sum(i in
V0)PPi(i)*(sum(k in H, r in R)z(i,k,r))

                                    ! max profit


FO:= (sum(k in H, i,j in V, r in R|i<>j)Dij(i,j)*x(i,j,k,r))/(sum(k in
H)Dk(k)) - (sum(i in V0)PDi(i)*(sum(k in H, r in R)y(i,k,r))+sum(i in
V0)PPi(i)*(sum(k in H, r in R)z(i,k,r)))/((sum(i in V0)PDi(i)*QDi(i))
+ (sum(i in V0)PPi(i)*QPi(i)))


      ! Contraintes
forall(k in H, r in R)                          !(1) vehicules ne
partent du depot qu'une fois
sum(j in V0) x(0,j,k,r) <= 1
```

```
forall(u in V0, r in R, k in H)                 !(2) ordre de voyage

(sum(i in V)x(i,u,k,r))-(sum(j in V)x(u,j,k,r)) = 0


forall(k in H, r in R)                          !(3) vehicules ne
retournent au depot qu'une fois

sum(i in V0) x(i,0,k,r) <= 1


forall(k in H, r in R)                          !(4) definition de
yprime

yprime(0,k,r) = sum(i in V0)y(i,k,r)


forall(i in V,j in V0,r in R, k in H|i<>j) do !(5a et 5b) mise a jour
de yprime

yprime(j,k,r) - (yprime(i,k,r) + z(j,k,r) - y(j,k,r)) <= 40*(1-
x(i,j,k,r))

yprime(j,k,r) - (yprime(i,k,r) + z(j,k,r) - y(j,k,r)) >=
40*(x(i,j,k,r)-1)

end-do

!BigM = deux fois la capacite de vehicule la plus grande



! OPTION SANS UTILISER BIG M

(!

forall(i in V,j in V0, r in R, k in H|i<>j)    !(5) mise a jour de
yprime

indicator(1,x(i,j,k,r),yprime(j,k,r) = yprime(i,k,r) + z(j,k,r) -
y(j,k,r))

!)
```

```
forall(i in V, k in H, r in R)                    !(6) capacite du
vehicule

yprime(i,k,r) <= Qk(k)


forall(i in V0)                                         !(7) demande de
QDi

sum(k in H, r in R) y(i,k,r) <= QDi(i)


forall(i in V0)                                         !(8) demande de
QPi

sum(k in H, r in R) z(i,k,r) <= QPi(i)


forall(k in H)                                          !(9) capacite en
heures du vehicule

sum(i,j in V, r in R|i<>j)x(i,j,k,r)*(Tij(i,j) + Si(i)) <= Tk(k)


forall(j in V, k in H, r in R)                    !(10) au maximum une
tournee dans le meme chantier

sum(i in V)x(i,j,k,r) <= 1


forall(k in H)                                          !(11) capacite en
kilometres du vehicule

sum(i,j in V, r in R)x(i,j,k,r)*Dij(i,j) <= Dk(k)


forall(i in V0, k in H, r in R) do       !(12) definition de x

y(i,k,r) + z(i,k,r) >= sum(j in V)x(i,j,k,r)

y(i,k,r) + z(i,k,r) <= 40*sum(j in V)x(i,j,k,r)

end-do

!BigM = deux fois la capacite de vehicule la plus grande
```

```
forall(i in V0, k in H, r in R) do       !(13) horaire fin Si au depot

b(0,k,r) >= v(k,r)*TWe(0,1)

b(i,k,r)+Si(i)+Tij(i,0)-TWl(0,1) <= 23*(1-x(i,0,k,r))

end-do

!BigM = horaire de fermeture du depot



! OPTION SANS UTILISER BIG M

(!


forall(i in V0, k in H, r in R) do       !(13) horaire fin Si au depot

b(0,k,r) >= v(k,r)*TWe(0,1)

indicator(1,x(i,0,k,r),b(i,k,r)+Si(i)+Tij(i,0) <= TWl(0,1))

end-do

!)



forall(i in V0,r in R,k in H) do                    !(14) fenetres
temporelles

sum(a in A)theta(i,k,a,r)*TWe(i,a) <= b(i,k,r)

b(i,k,r) <= sum(a in A)(TWl(i,a)-Si(i))*theta(i,k,a,r)

end-do


forall(i in V0,r in R,k in H)                       !(15) Prendre une
seule fenetre

sum(a in A)theta(i,k,a,r) <= 1


forall(i in V,j in V0,r in R,k in H|i<>j)do    !(16) horaire debut en
i versus en j
```

```
b(i,k,r)+Si(i)+Tij(i,j)-b(j,k,r)<= 23*(1-x(i,j,k,r))

b(i,k,r)+Si(i)+Tij(i,j)-b(j,k,r)>= 23*(x(i,j,k,r)-1)

end-do

!BigM = horaire de fermeture du depot + 1 (obtenu empiriquement)



! OPTION SANS UTILISER BIG M

(!

forall(i in V,j,r in R,k in H|i<>j)      !(16) horaire debut en i
versus en j

indicator(1,x(i,j,k,r),b(j,k,r)=b(i,k,r)+Si(i)+Tij(i,j))

!)



forall(i in V0, k in H, r in Rc) do      !(17) mise a jour de b dans
chaque tournee

b(0,k,r+1) - (b(i,k,r)+Si(i)+Tij(i,0)) <= 100*(1-x(i,0,k,r)) + 100*(1-
v(k,r+1))

!b(0,k,r+1) - (b(i,k,r)+Si(i)+Tij(i,0)) >= 100*(x(i,0,k,r)-1) +
100*(v(k,r+1)-1)

end-do

!BigM = ...


forall(k in H, r in R) do                      !(18) lien entre v et x

v(k,r)     <= sum(i,j in V)x(i,j,k,r)

v(k,r)*25  >= sum(i,j in V)x(i,j,k,r)

end-do

!BigM = nombre de chantiers + depot de l'instance
```

```
forall(k in H, r in R) do                    !(19) arrivee et sortie
du vehicule au depot

sum(i in V0)x(i,0,k,r) = v(k,r)

sum(j in V0)x(0,j,k,r) = v(k,r)

end-do


forall(i in V0,r in R, k in H)               !(20) obliger ï¿½
prendre une seule fenetre temporelle

sum(j in V)x(i,j,k,r) = sum(a in A)theta(i,k,a,r)


forall(k in H, r in Rc)                      !(21) ordre de tournees

v(k,r) >= v(k,r+1)


forall(i in V, k in H, r in R)               !(22) pas de route de i
vers i

x(i,i,k,r)=0


forall(i,j in V,k in H,a in A, ii in V0,r in R) do !(23) variables
binaires

x(i,j,k,r) is_binary

v(k,r) is_binary

y(ii,k,r) is_integer

z(ii,k,r) is_integer

yprime(ii,k,r) is_integer

theta(ii,k,a,r) is_binary

end-do


      ! Lancement des fonctions objectif
```

```
! Tests pour d'autres objectifs
!maximise(FO4)
!maximise(FO4 - FO1)
!maximise(FO4 - FO2)
!maximise(FO4 - FO3)
!maximise(FO4 - FO1 - FO2 - FO3)

minimize(FO)



! Trouver les meilleures solutions obtenues pendant 120 minutes
(!
! Display progress log
 setparam('XPRS_VERBOSE',true)

! Solve the problem
 setparam("XPRS_TUNERMAXTIME", 120)
 setparam("XPRS_TUNEROUTPUTPATH", string(expandpath("TuneOut")))
 minimize(XPRS_TUNE, FO)
!)

      ! Imprimer les resultats
writeln("Objective function result: ", getobjval)
writeln("Number of trips: ", getsol(FO1))
writeln("Elapsed time: ", getsol(FO2)," h")
writeln("Distance travelled: ", getsol(FO3)," km")
writeln("Profit: ", getsol(FO4))
```

```
demandes_satisfaites:=(sum(i in V0, k in H, r in R)(y(i,k,r) +
z(i,k,r)))/(sum(i in V0)(QDi(i)+QPi(i)))*100

writeln("Supplied demand: ",
strfmt(getsol(demandes_satisfaites),0,1)," %")

writeln("")


forall(k in H, r in R)

writeln(" v(", k,',',r, "): ", getsol(v(k,r)))




forall(i,j in V, k in H, r in R)

      if getsol(x(i,j,k,r))>0 then

            writeln(" x(",i,',',j,',',k,',',r, "): ",
getsol(x(i,j,k,r)))

      end-if


forall(i in V0,r in R, k in H)

      if getsol(y(i,k,r))>0 then

            writeln(" y(",i,',',k,',',r, "): ", getsol(y(i,k,r)))

      end-if


forall(i in V0,r in R, k in H)

      if getsol(z(i,k,r))>0 then

            writeln(" z(",i,',',k,',',r, "): ", getsol(z(i,k,r)))

      end-if


forall(i in V, k in H, r in R)

      if getsol(yprime(i,k,r))>0 then
```

```
            writeln(" yprime(",i,',',k,',',r, "): ",
getsol(yprime(i,k,r)))

      end-if


forall(j in V, k in H, r in R)

      if getsol(b(j,k,r))>0 then

            writeln(" b(",j,',',k,',',r, "): ", getsol(b(j,k,r)))

      end-if


forall(i in V0,r in R, k in H, a in A)

      if getsol(theta(i,k,a,r))>0 then

            writeln(" theta(",i,',',k,',',a,',',r, "): ",
getsol(theta(i,k,a,r)))

      end-if


writeln("")

forall(i in V0) do

      if (sum(r in R, k in H)getsol(y(i,k,r)))/QDi(i) = 0 then

            writeln(" Client ",i," was not served")

      end-if

      if (sum(r in R, k in H)getsol(y(i,k,r)))/QDi(i) > 0 and (sum(r
in R, k in H)getsol(y(i,k,r)))/QDi(i) < 1 then

            writeln(" Client ",i," was partially served")

      end-if

      if (sum(r in R, k in H)getsol(y(i,k,r)))/QDi(i) = 1 then

            writeln(" Client ",i," was completely served")

      end-if

end-do

end-model
```