MATHEMATICAL PROGRAMMING FOR TACTICAL TRANSPORTATION PLANNING IN A MULTI-PRODUCT SUPPLY CHAIN

Simon Belieres

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ FÉDÉRALE TOULOUSE MIDI-PYRÉNÉES

Délivré par :

l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue le *06/11/2019* **par :**

SIMON BELIERES

MATHEMATICAL PROGRAMMING FOR TACTICAL TRANSPORTATION PLANNING IN A MULTI-PRODUCT SUPPLY CHAIN

JURY

FRANÇOIS CLAUTIAUX	Professeur des Universités Université de Bordeaux	Examinateur
BERNARD GENDRON	Full Professor Université de Montréal	Rapporteur
MARIE-JOSÉE HUGUET	Professeur des Universités INSA Toulouse	Présidente du Jury
NICOLAS JOZEFOWIEZ	Professeur des Universités Université de Lorraine	Directeur de Thèse
IVANA LJUBIC	Professeur ESSEC Business School	Examinatrice
OLIVIER PÉTON	Professeur IMT Atlantique	Rapporteur
FRÉDÉRIC SEMET	Professeur des Universités Centrale Lille	Directeur de Thèse

---

École doctorale et spécialité :

EDSYS : Informatique 4200018

Unité de Recherche :

Laboratoire d'analyse et d'architecture des systèmes

Directeur(s) de Thèse :

Nicolas JOZEFOWIEZ et *Frédéric SEMET*

Rapporteurs :

Bernard GENDRON et *Olivier PÉTON*

Acknowledgments

First and foremost, my sincerest gratitude goes to my supervisors, Nicolas Jozefowiez and Frédéric Semet. It has been a pleasure and an honor to be their PhD student during these three years. I appreciate their contributions of time and ideas that made my PhD experience productive and stimulating. They trusted me through this journey, supported me during tough times, and allowed me to grow as a researcher.

I would like to acknowledge the French National Research Agency that financially supported my work through the Pi-Comodalité project, under the grant ANR-15-CE22-0012.

I am specially grateful to all the members of my jury: François Clautiaux, Bernard Gendron, Marie-Josée Huguet, Ivana Ljubic and Olivier Péton. I appreciate their insightful comments and suggestions that will help me to improve my work and writings.

I express my deep gratitude to Tom van Woensel and Alysson Costa, who kindly received me, respectively at the Eindhoven University of Technology and the University of Melbourne. Working with them has been an extremely enriching and pleasant experience.

I offer my warmest thanks to Mike Hewitt, who guided me through this thesis, shared with me his invaluable insights and provided me unflinching encouragements. He has been a tremendous support in so many ways, and I am indebted to him for that.

These three years were made enjoyable in large part due to the members of the ROC team. I will miss the delightful work atmosphere, as well as the cohesion of the team, in, and outside the office. Special thanks to Estèle, Idir, Mikaël and Tom for their friendship.

Lastly, but certainly not least, I would like to thank my family and friends for all their love and support. There are far too many names to list individually, but I thank you all for our time together. To my partner Amandine; thank you for having brought so much joy and love into my life. To my parents; thank you for your unconditional love. I am blessed to have you by my side.

Contents

List of Figures

List of Tables

Introduction

Freight transportation in supply chains

Freight transportation is a pillar of our economy [Crainic 2000a]. Indeed, with globalization and growth in international trade, production processes are often fragmented in different regions in order to take advantage of differences in the cost and quality of services. As a result, the average distance travelled by goods is globally increasing [Esnouf 2013]. In 2018 in OECD countries, the road usage for freight transportation [1] is estimated to be more than 1,374 billion tonne-kilometers, which represents an increase of 4.1% compared to 2017. Thus, the success of various business fields highly relies on an effective logistics management.

Moreover, air pollution and climate changes are major concerns of our era as they put humanity in danger. It turns out that the sector of freight transportation has a major environmental impact, as it is responsible for more than 7% of global carbon emissions [2]. For this reason, innovation and solutions in logistics management are more urgent than ever.

A major share of logistics movements takes place within supply chains. A supply chain is a network of partner companies that include suppliers, manufacturers, warehouses, distribution centers, or retailers. These companies work together to create products and satisfy the consumer market. To get products to the customers, the supply chain process involves a series of stages, such as the acquisition of raw materials or the transformation of raw materials into finished products.

From the delivery of raw materials to manufacturers to the distribution of finished products to the end-user, freight transportation supports the whole supply chain process. As efficient freight transportation is necessary for a supply chain to be profitable, related problems must be solved efficiently. However, optimization problems for planning the transportation operations in a supply chain are very challenging to solve in practice. Due to their scale, real-life supply chains often yield models that cannot be solved in reasonable amount of time by commercial mixed integer programming (MIP) solvers.

This thesis was carried out with the help of a Third-Party Logistics (3PL), DHL Supply Chain, and focuses on a distribution problem that arises in supply chain management. More specifically, the problem addressed in this thesis is inspired by the collaboration between DHL and a large French chain of restaurants that outsources part of its logistics. In that partnership, DHL must plan the transportation operations that take place within the supply chain to fulfill the restaurant orders for products over a planning horizon. To do so, DHL designs a transportation plan that determines the routes followed by the products, from the suppliers to the restaurants. In practice, the transportation plans are defined "manually" by a team of

[1] Source: OECD International Transport Forum
[2] Source: OECD International Transport Forum

experienced logistics managers, according to a set of decision policies. To improve the quality of its services and stay ahead of the competition with other 3PLs, DHL seeks to develop optimization tools for the design of its transportation plans.

Research questions

Supply chain optimization problems encompass three levels of decisions that have different consequences on the planning horizon. The strategic level involves long-term decisions that are related to the acquisition, the location and the sizing of major infrastructures such as production units, distribution centers, plants, vehicles, etc. The tactical level seeks to determine the utilization of available resources on a mid-term horizon. Tactical decisions include fleet vehicle management, crew management, production planning and inventory level management. Finally, the operational level involves short-term decisions that cannot be planned well in advance as they rely on short notice conditions. Operational decisions are decentralized and define daily schedules by adapting tactical planning services to current conditions.

Most optimization problems for supply chain management studied in the literature are defined at the strategic level, and cover long-term planning horizons that are discretized in monthly or yearly periods. Less studies focus on the tactical planning of the supply chain. To the best of our knowledge, existing tactical models for supply chain optimization disregard merge-in-transit and shipment consolidations. However, shipment consolidations allow the reduction of vehicle utilization and is thus an effective strategy to improve the profitability of the transportation operations. Thus, we believe that existing models for supply chain management are not necessarily suitable for planning transportation operations over a mid-term horizon.

A fundamental tactical problem for planning transportation operations is the *Service Network Design Problem* (SNDP) [Crainic 2000a, Wieberneit 2008], which determines paths for shipments in a network and allocate the necessary services. Although the SNDP enables shipment consolidations, it makes no presumptions regarding the considered distribution network. On the other hand, supply chains have specific structures that can and should be exploited algorithmically. In addition, in the SNDP the origin and destination location of each freight to be transported is known in advance. In supply chains, customers order products that may be manufactured and shipped from multiple locations.

The literature lacks tactical studies on supply chains distribution planning. To fill this gap, we introduce the *Logistics Service Network Design Problem* (LSNDP) which aims to design transportation operations within a supply chain over a finite time horizon. As customers request products that may be offered at multiple locations, the LSNDP determines the origin location for each transported product request as well as the routes followed by the products. The purpose of this thesis is to solve instances of the LSNDP that reflects the operations of DHL. To do this, we propose multiple solution algorithms.

Outline of the manuscript

In Chapter 1 we define the industrial context of this thesis. We provide details about the chain of restaurants, its logistics and its partnership with DHL. In Chapter 2, we introduce a problem description and a mathematical formulation of the LSNDP. We also review the relevant literature on supply chain optimization problems and service network design problems.

The scientific contributions of this thesis are three solution algorithms for the LSNDP. These algorithms are designed to solve large-scale instances in a reasonnable amount of time. In Chapters 3, 4 and 5, we present these methods and test them on random instances related to the logistics operations of the restaurant chain to assess their performances.

Chapter 3 introduces an enhanced Benders strategy. In the proposed Benders decomposition, the subproblem intends to route customers demands based on a vehicle allocation determined in the master. To improve the quality of these vehicle allocations, we reinforce the master with artificial information based on products aggregation. The resulting master allocates vehicle capacities on the network in order to satisfy customer demands of super-product. To improve the convergence of our Benders strategy, we also strengthen the master with a priori valid inequalities and generate heuristic primal solutions from unfeasible subproblems. The computational study shows that the increase in the number of products has little effect on our method.

In Chapter 4 we extend this idea and propose a dynamic Benders strategy, wherein the information strengthening the master is refined during the run of the algorithm. In a spirit of exploration and exploitation, the number of super-products considered in the master problem is increased at each iteration. Thus, we demonstrate that the master presented in Chapter 3 can be extended to multiple super-products obtained by partitioning the set of products. As the partitioning of the product set impacts the quality of the bounds produced by the master problem, we propose a metric that evaluates whether a pair of products is worth aggregating or not. Based on that metric, we implement clustering strategies for partitioning the product set at each main iteration of the algorithm.

Chapter 5 presents a method inspired from the Dynamic Discretization Discovery, a recent algorithm proposed for solving the Continuous Time SNDP. The LSNDP is defined on an underlying graph. Thus, the size of the model is partially determined by the number of nodes and arcs in that graph. To be able to solve large-scale industrial instances, we propose a heuristic that constructs a subgraph based on a subset of the nodes and arcs in the original graph. The computational study shows that our heuristic builds subgraph orders of magnitude smaller than the original graph. Solving the LSNDP associated with the subgraph allows to identify high-quality solutions in a small amount of time, even for instances defined on large-scale networks.

In Chapter 6 we describe real-life LSNDP instances that reflect the actual logistics network of the restaurant chain. To solve these extremely challenging instances

we propose a matheuristic that combines the solution methods presented in Chapters 3, 4 and 5. We first assess the performance of our hybrid method. Then, we analyze in detail the solutions obtained and discuss our management insights.

Chapter 1

Industrial Context

Contents

This chapter presents the context of the thesis and describes the industrial collaboration between DHL Supply Chain and a large French chain of restaurants that we cannot name for confidentiality reasons. General information about the chain of restaurants and its logistics operations are given in sections 1.1 and 1.2. Then, the details of the industrial collaboration are described in section 1.3. Finally the objective of this thesis is developped in section 1.4.

1.1 Description of the Chain of Restaurants

The first restaurants are created in the seventies. In the years following, the chain has considerably expanded. Today it counts 239 standardized restaurants, located in multiple European countries. The chain belongs to a major restaurant group that holds an important market share of the French foodservice industry, with more than 50 millions of meals served per year.

The restaurants are often located in urban or peri-urban commercial areas. Their customers are mostly people shopping in nearby malls. As a result, the number of people visiting restaurants reaches its peak at noon, when customers have lunch. On average, the number of lunches served per day is between 400 and 600. Furthermore, the number of people visiting restaurants varies according to the time of year. In particular, it increases considerably during winter sales and summer sales, a busy period for shopping centers.

The chain employs more than 10,000 people, with an average of 25 workers per restaurant. Ninety per cent of them are versatile employees and alternate between different tasks as: receiving goods, preparing dishes, replenishing self-service meals, cashiering services, etc. In most situations, versatile employees work part-time and have a high turnover rate. Rest of the staff is composed of managers in charge of the restaurants management. Contrarily to versatile employees, most managers are full-time workers.

1.2 Logistics Operations

The supply of the restaurants is performed through a domestic supply chain managed by a centralized system, which facilitates strong economies of scale. In this section, we present the logistics operations of the restaurant supply chain. We first describe the logistics network that models the supply chain. Then, we describe the main features of the supply chain.

1.2.1 A Tri-Echelon Network

The supply chain involves three types of actors and can be represented by a tri-echelon network. The first set includes the suppliers. Suppliers are business partners that provide the wide range of products required by the restaurants. Most suppliers are large manufacturers or wholesalers who produce their goods continuously, and in large quantities. Figure 1.1 shows the distribution of the suppliers.

The second set includes the warehouses. Warehouses receive products from the suppliers, and redistribute them to restaurants. They enable cross-docking and merge-in-transit. In addition, warehouses can store products, which incurs a cost per day and per unit. The warehouses belong to DHL and another logistics company, STG Group. They can be distinguished into two categories: central warehouses and regional warehouses, which differ in terms of capacity. A product delivered via the distribution network must necessarily pass through a central warehouse before going to a regional warehouse. Figure 1.2 and 1.3 show respectively the central and regional warehouses.

The third set includes the restaurants. To be able to constantly satisfy customer needs, the restaurants order products on a regular basis. These orders are answered several times a week, depending on the delivery schedule specific to each restaurant. A single delivery per week is sufficient for restaurants with lowest volume of customers, while five deliveries per week are required for the busiest restaurants. Restaurants are spread all over the country and are often concentrated in commercial areas around the big cities. Figure 1.4 shows the distribution of the restaurants.

Freight transportation in the supply chain is ensured by a fleet of vehicles. Transporting products from a facility to another incurrs a less-than-truckload (LTL) cost proportional to the amount of freight moved and the distance traveled.

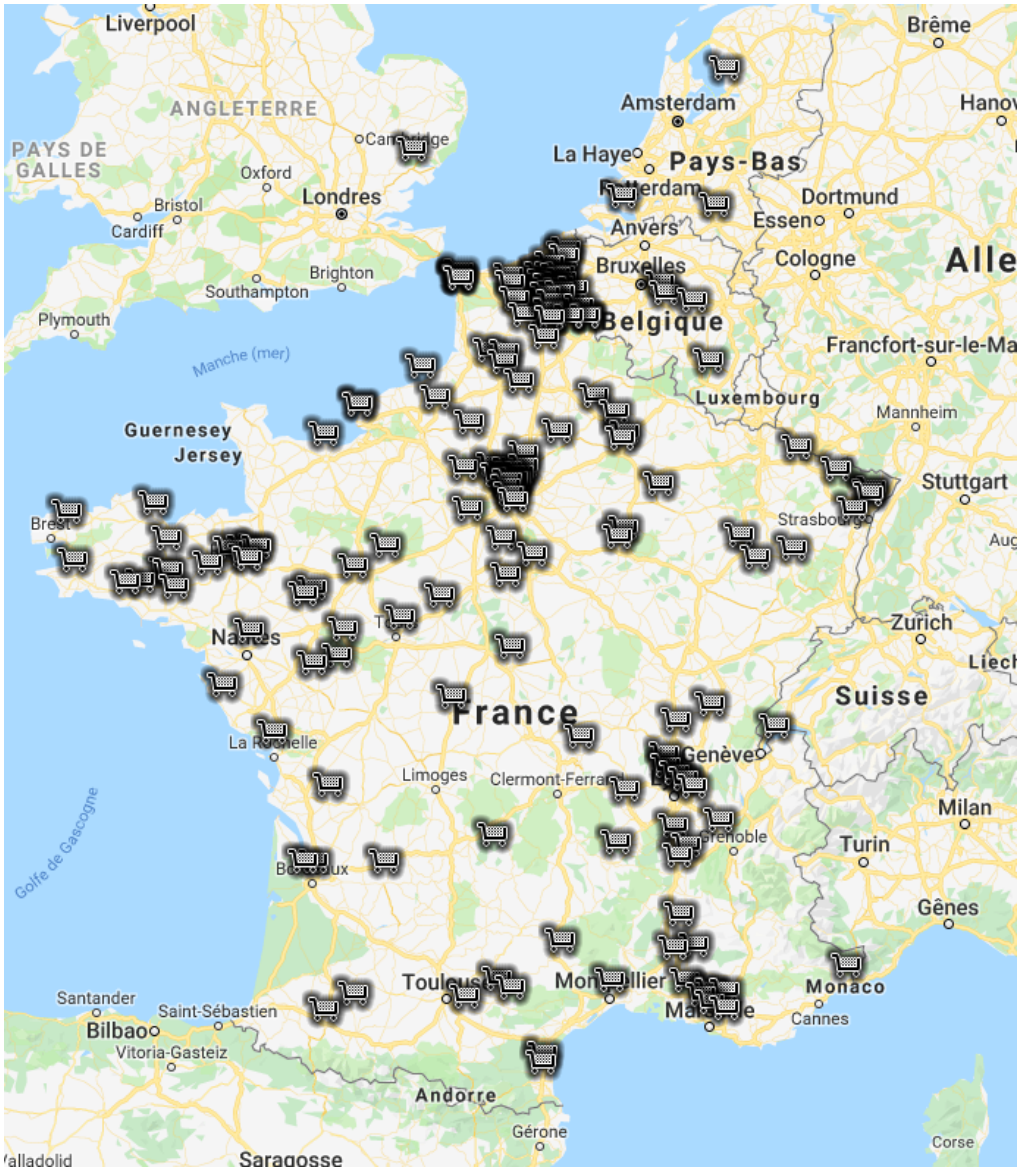

Figure 1.1: Distribution of the suppliers

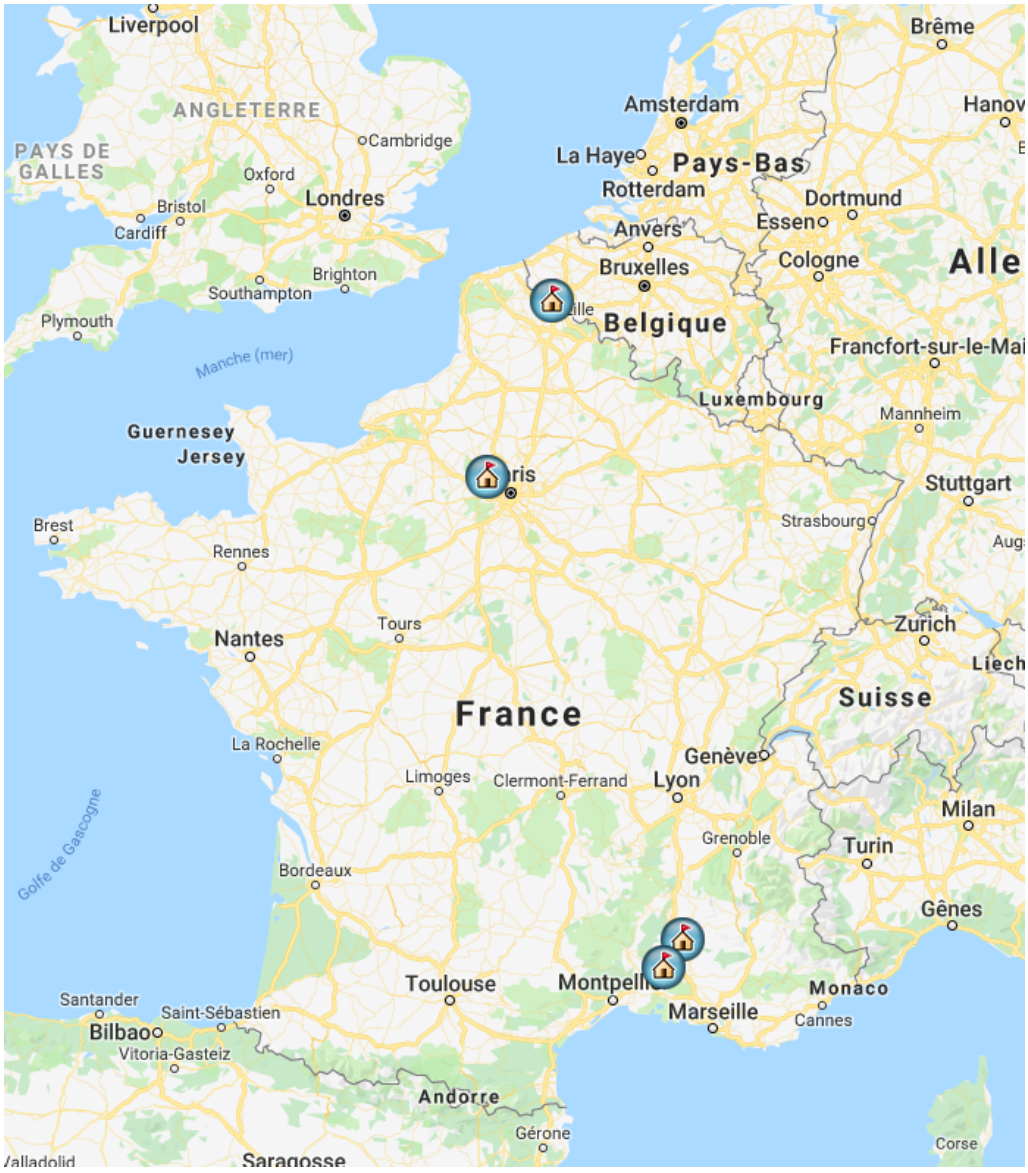

Figure 1.2: Distribution of the central warehouses

Figure 1.3: Distribution of the regional warehouses

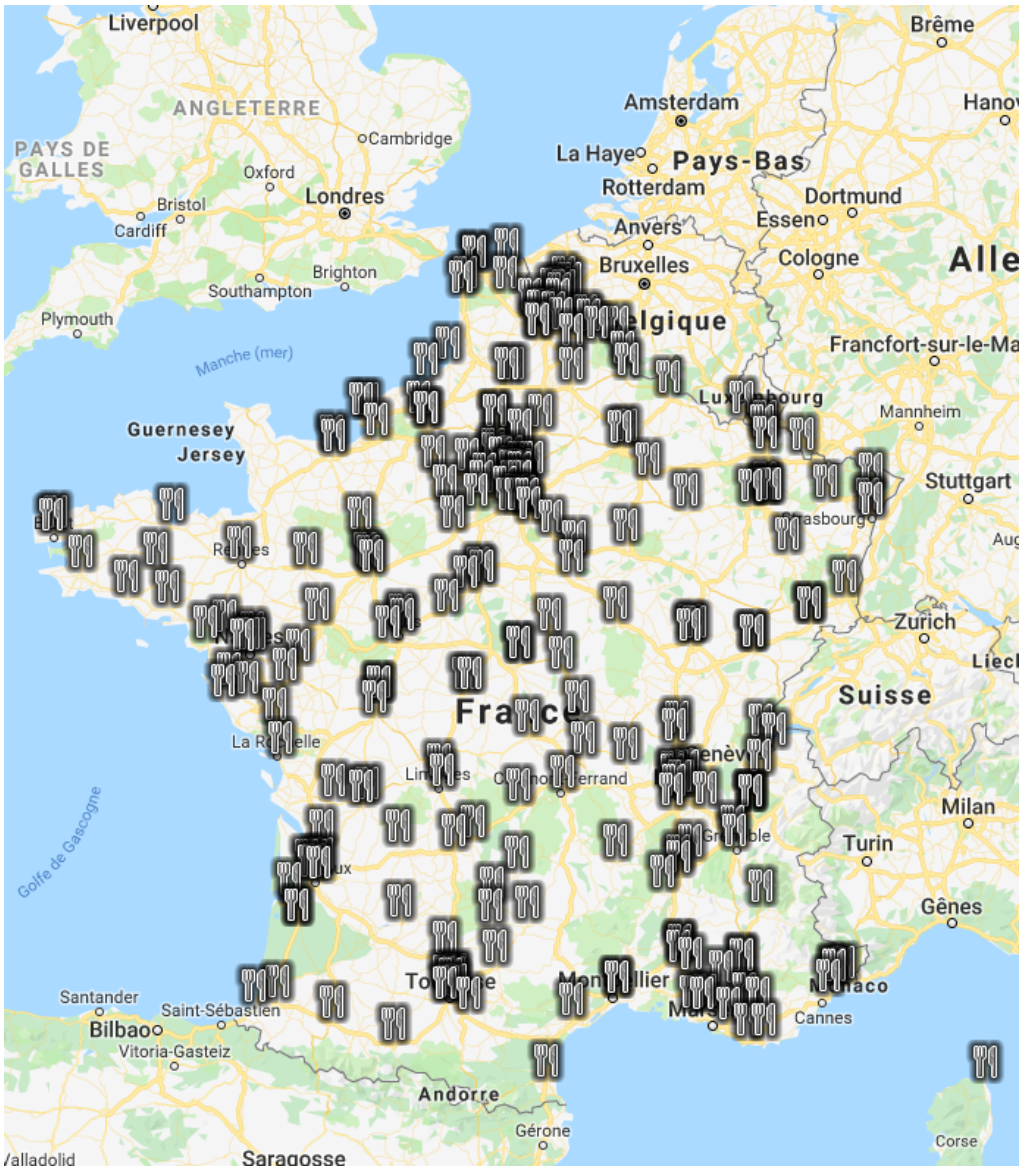

Figure 1.4: Distribution of the restaurants

1.2.2 Products, Demand Management, and Logistics Features

Restaurants use about a thousand type of products. These products are standardized according to several criteria that suppliers must respect. In that way, product quality does not vary from one supplier to another. This favours a similar quality of service from one restaurant to another, and helps the chain to maintain a consistent brand image across the country. Products are classified into five product *families*: frozen products, fresh products, dry products, beverages and non-food products. For example, products as sparkling water and napkins belong respectively to the beverages family and the non-food products family. Product families are classified into two product *categories*: cold products and ambient products. Frozen and fresh products belong to the category of cold products, while other families belong to the category of ambient products. Note that cold products and ambient products are managed independently, as they cannot be transported by the same trucks.

Suppliers are characterized by the products they offer and are specialized in one or multiple product families. A supplier who does not specialize in a given product family does not offer any product in that family. As an example, a supplier that does not specialize in non-food products cannot supply napkins. On the other hand, note that a supplier who specializes in a certain product family may not supply all products in this family. For example, it is possible for a beverage supplier to provide alcoholic beverages and not sodas. Finally, it should also be noted that a product is often offered by multiple suppliers.

Restaurants place orders that are delivered according to their weekly schedules. Table 1.1 shows examples of delivery schedules. An order is defined as a set of products that suppliers must ship to a restaurant. The size of an order is expressed in terms of number of pallets. We consider homogeneous pallets, that contain a single type of product, and always in the same quantity. An order has a unique destination, but can have multiple origins, as its products can come from multiple supplier locations.

Table 1.1: Example of restaurants delivery schedules

Restaurant	Mond.	Tues.	Wed.	Thurs.	Fri.	Sat.	Sun.
Amiens	6:30-7:30	-	-	11:00-12:00	-	-	-
Boulogne	-	8:00-9:00	-	10:15-11:15	-	8:30-9:30	-
Montpellier	5:30-6:30	-	-	6:30-7:30	-	-	-
Paris	6:15-7:15	5:30-6:30	-	-	5:30-6:30	7:30-8:30	-
Toulouse	-	7:00-8:00	-	8:15-9:15	-	10:30-11:30	-

1.3 DHL Supply Chain

The chain of restaurants outsources its logistics operations to DHL Supply Chain. They consist in transporting products from the suppliers to the restaurants. In

the following subsections, we describe the role of DHL Supply Chain, and how the company manages the logistics flows.

1.3.1 A Third-Party Logistics

The Deutsche Post DHL Group is the world's largest international courier service company. The main division of the company, DHL Supply Chain, is a Third-Party Logistics (3PL). Its role is to provide logistics solutions to clients from a variety of sectors, and to coordinate the actors in their supply chains. As part of the collaboration with the chain of restaurants, DHL receives a set of orders from the restaurants, and plans transportation operations to meet these demands over the time horizon. These operations are characterized by a *transporation plan* that extensively describes the itinerary followed by each product. In order to propose attractive offers to its customers, DHL seeks for the most cost-effective transporation plans.

Restaurants place orders to DHL without specifying the origins of the products ordered. Thus, for each order, DHL must determine the supplier locations from which to ship the products requested. Once shipment origins are selected, DHL determines products itineraries from the suppliers to the restaurants. As product pallets are small relative to vehicle capacity, an efficient way to save transportation costs is to reduce the number of vehicles used by consolidating the flows. To do so, it is necessary to coordinate the paths of the different products, in both space and time. For this purpose, DHL can use warehouses to consolidate shipments, which is advantageous if savings achieved by consolidating the flows offset the warehousing costs.

Therefore, the design of the transportation plan involves two levels of decision: selecting the supplier locations from which to ship the products, and routing the products, both in space and in time. These decision-levels make the design of the transportation plan a highly combinatorial problem. In order to facilitate the planning of transportation operations, DHL relies on a set of decision support policies. We describe these policies in the next subsection.

1.3.2 Current Logistics Management

DHL does not have a global optimization tool to elaborate the transportation plan. The planning is carried out "manually" by a team of experienced logistics engineers. These engineers schedule the transporation operations based on their experience of the problem, but also according to a set of decision policies.

This framework limits the number of possible routes from suppliers to restaurants and forces products to follow *centralized paths*. By definition, a centralized path:

- originates from a supplier
- transits through a central warehouse
- transits through a regional warehouse

- ends at a restaurant

Figure 1.5 illustrates centralized paths. In this framework, other delivery paths are not taken into account. For example, it is not possible to ship a product directly from a supplier to a restaurant. Similarly, a product cannot be shipped directly from a supplier to a regional warehouse.

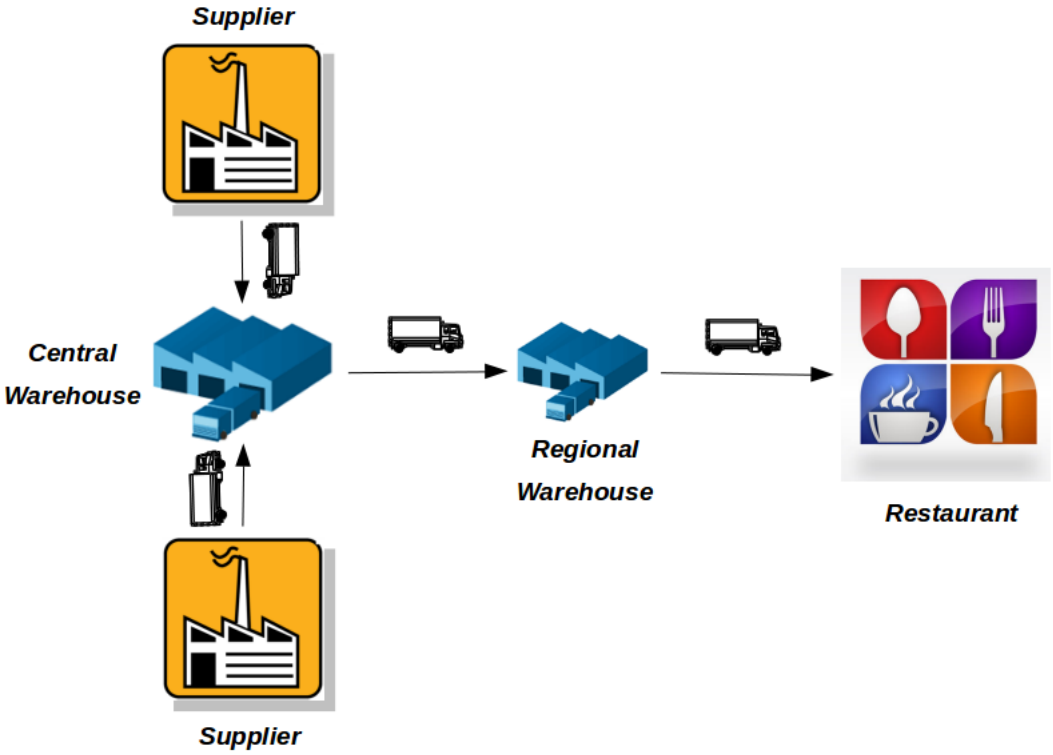

Figure 1.5: Centralized path

In addition, DHL decision policies assigns each restaurant to the nearest regional warehouse. Thus, a restaurant cannot receive products from a regional warehouse it is not assigned to. On the other hand, products can transit from any supplier to any central warehouse, or from any central warehouse to any regional warehouse, without restriction.

According to these decision policies, the logistics engineers at DHL seek to determine the most profitable transportation plan, i.e. a transportation plan that tends to limit the distances covered by the products, minimize storage costs and maximize flow consolidations.

1.4 A Global Optimizer for Planning Transportation Operations

The chain of restaurants has outsourced its logistics operations to DHL for more than ten years. During this collaboration, DHL built a sustainable and efficient

network. Improvements include a better coordination of suppliers-restaurants relationships and a stronger control on the inventory management. In addition, DHL gave visibility on logistics costs by providing a euro cost per tonne associated with each product. This enables the monitoring of the expenses related to the products management.

With the aim of staging ahead of the competition with other 3PLs, DHL constantly seeks to increase the quality of its services. In the management of the chain of restaurants, DHL has identified several areas of improvements that we present in the next subsection. We then describe the global optimization tool DHL wants for planning transportation operations in the supply chain. We also describe the perspectives this tool would open.

1.4.1 Improvement Strategy

For each product, the euro cost per tonne averages the sum of storage and transportation costs over the past periods. The analysis of these costs reveals that DHL logistics management can be improved. Indeed, if the average cost is around 300 euros per ton, for certain products it can reach 800 euros per ton. Such a disparity in logistics costs indicates that transportation operations are not planned optimally for all products.

One improvement is to modify decision policies that limit the possible itineraries. Although these policies have the advantage of simplifying the planning, they also strongly restrict the search space. The emergence of other delivery alternatives may lead to more profitable logistics costs. An appealing idea is to integrate routes that do not necessarily visit a central warehouse and a regional warehouse. Shipping products from suppliers to restaurants by visiting only one warehouse can reduce handling costs. Similarly, it seems interesting to allow direct shipments from suppliers to restaurants.

Another improvement envisaged by DHL is to create new partnerships with small local suppliers, by integrating them into the supply chain. Indeed, a medium-term goal of the chain is to offer local dishes in its restaurants. However, the current decision policies regarding product itineraries do not favor these partnerships. Small local suppliers will be used to ship regional products to restaurants that are geographically close. Thus, for regional products it is clear that direct shipments from suppliers to restaurants are more profitable options than centralized paths, as they avoid unnecessary long travel distances. To limit the logistics costs incurred by shipping from local suppliers, it is therefore essential to allow other delivery alternatives than centralized paths, and develop short distribution channels.

1.4.2 Optimization Tool

DHL wants to develop a decision support tool for the design of its transportation plans. This tool includes an optimization algorithm for a global planning of the transportation operations in the supply chain. The algorithm does not have to be

an exact method, but must provide good solutions in a reasonnable amount of time. As input it takes the following data:

- the list of suppliers and the products they offer;
- the list of warehouses and their storage costs;
- the list of restaurants and their orders over a time horizon;
- travel times, distances and travel costs between actors of the logistics network;
- a design policy.

The aim of the algorithm is to provide a cost-effective transportation plan that respects the constraints and satisfies the restaurants orders.

As explained in subsection 1.3.1, DHL seeks to extend its current design policy. The design policy defines between which actors of the logistics network it is possible to ship products. Thus, it is a crucial input that has a major impact on the transportation plan cost, as it determines the number of feasible delivery paths in the network. To determine the most profitable transportation plan, one has to choose a complete design policy, such that products can transit between any pair of actors. Nevertheless, a complete design policy yields instances that are too large to be solved in a reasonable time. Therefore, the choice of the design policy is an important question that must be answered in advance of the elaboration of the transportation plan.

To assist DHL in the definition of new policies regarding product itineraries, we also develop an exact method for designing the transportation plans. The idea is to execute the exact method on instances that varies only according to the design policy. A comparison of the optimal solutions would enable to quantify the savings achieved through new delivery alternatives. Thus one could select a design policy that incorporates short distribution channels while yielding instances computationally tractable.

The problem of planning transportation operations within the supply chain can be modeled as a mixed integer linear formulation, described in subsection 2.2.2. Instances of sizes relevant to DHL are too large for on-the-shelf optimization solvers. For example, parameters such as the number of products or the size of the logistics network affect greatly the model size. In our case study, there are 177 suppliers, 4 central warehouses, 40 secondary warehouses and 239 restaurants, which constitutes a considerable network. In addition, the logistics network includes 605 cold products and 395 products at room temperature. Finally, for a transportation plan to be relevant, it must be built over a sufficiently long time horizon, at least 15 days. These parameters values lead to models that are computationnally intractable.

More specifically, the problem temporal dimension makes the industrial instances very difficult to solve. A common technique to model the temporal component is discretization; rather than deciding the exact time a product should be shipped (e.g., 5.46 pm), the model decides on a time interval when the shipment

should occur (e.g., between 4pm and 6 pm). By discretizing time, it is no longer possible to capture all the shipment opportunities of the problem in continuous time. Thus, to provide a cost-effective solution, we require a model with a fine granularity for the time discretization [Boland 2018]. On the other hand, the time step has a significant impact on the model size. As a result, providing a cost-effective solution often requires to solve a model that is computationnally intractable.

Therefore, in order to solve the instances proposed by DHL, it is necessary to propose solutions methods that are effective despite the scaling of the previously mentioned parameters. In Chapters 3 and 4, we propose Benders strategies that remain effective even when the number of products increases. In Chapter 5, we propose a heuristic that limits the size of the time-expanded network that models the problem. In Chapter 6, we combine these methods in a hybrid algorithm and solve industrial instances.

Chapter 2

Logistics Service Network Design Problem: Definition and State of the Art

Contents

2.1 Introduction

In this thesis we propose the Logistics Service Network Design Problem (LSNDP), a combinatorial optimization problem for planning transportation operations within a multi-product supply chain. This chapter aims to define the LSNDP, and position it in the freight transportation literature. The remainder of the chapter is organised as follows. In section 2.2, we introduce a problem description and a mathematical formulation of the LSNDP. In section 2.3, we review relevant literature.

2.2 Logistics Service Network Design Problem

2.2.1 Problem statement

We focus on the planning of transportation operations for a 3PL logistics company which delivers products to customers over a fixed planning horizon. Specifically, a customer requests the delivery of a known quantity of product at a predetermined time from the 3PL logistics company. Each product is produced at a supplier facility, and there may be multiple supplier facilities that may supply a given product. However, the customer does not indicate the facility from which the

product should come. In addition, the customer may request that the product be delivered multiple times over the planning horizon. However, the quantity need not be the same in each request, and each delivery need not to come from the same supplier. In the context of supplying restaurants that are part of the same chain, a customer corresponds to an individual restaurant. That restaurant could then request for the next month the delivery of a carton of napkins (the product) on each Friday at 9 in the morning.

We assume that each supplier has a limited product line, but unlimited capacity for the products they do supply. Relatedly, we presume that each customer requests products from more than one supplier. For example, there may be a supplier that specializes in paper products, one that focuses on meat, and one on fruits and vegetables, while a restaurant requests products from each supplier. The 3PL logistics company may plan transportation directly from a supplier facility to a customer location. However, customer order quantities are typically small relative to the vehicle capacity. As a result, the 3PL logistics company may instead transport products through a distribution network that connects supplier facilities with customer locations in order to consolidate orders and increase vehicle fill rates. Terminals within this distribution network are referred to as *Warehouses* and offer both cross-docking and warehousing of products. However, storing product at a warehouse incurs a per-unit, per-unit-of-time cost. We illustrate such a network in Figure 2.1, wherein S_x indicates a supplier facility, C_x indicates a customer location, and W_x indicates a warehouse within the distribution network.

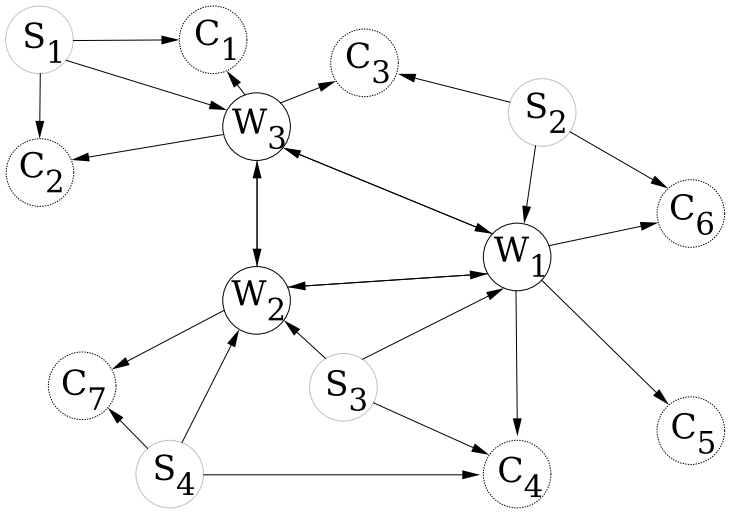

Figure 2.1: Distribution network

We refer to transportation within this network, as well as from a supplier facility or to a customer location, as a *service*. Associated with a service is a departure time

from its origin and an arrival time at its destination. In addition to determining whether a service is executed, the 3PL logistics company can allocate a capacity to that service, each unit of capacity coming at a cost. For example, the 3PL logistics company can determine how many trucks dispatch on a service, each truck providing some extra capacity at a cost. Ultimately, the 3PL logistics company seeks to determine which supplier(s) satisfy customer requests as well as the services and capacities needed to support those deliveries in order to minimize the overall cost.

2.2.2 Mathematical formulation

We model the supply chain by the network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, wherein the node set $\mathcal{N}$ contains nodes that represent supply locations $\mathcal{S}$, customer locations $\mathcal{C}$, and warehouses $\mathcal{W}$, and the set $\mathcal{A}$ contains arcs that represent transportation between the locations. The set $\mathcal{P}$ represents all the products, the set $\mathcal{P}^i$ denotes the products provided by supplier i. As we only consider suppliers providing products, $\mathcal{A}$ does not contain arcs that model transportation to a supplier. Similarly, as we only consider the delivery of products to customers, $\mathcal{A}$ does not contain arcs that model transportation from a customer. More specifically, $\mathcal{A} \subseteq (\mathcal{S} \times \mathcal{C}) \cup (\mathcal{S} \times \mathcal{W}) \cup (\mathcal{W} \times \mathcal{W}) \cup (\mathcal{W} \times \mathcal{C})$. Associated with each arc $a = (i, j) \in \mathcal{A}$, is a travel time $t_{ij} \in \mathbb{N}^*$, a per unit of flow cost $c_{ij} \in \mathbb{R}^{+*}$, a vehicle capacity $\hat{c}$, and a fixed cost per unit of capacity $f_{ij} \in \mathbb{R}^{+*}$.

We assume that the 3PL logistics company seeks to develop a transportation plan for a fixed planning horizon of length T. Thus, to model the time aspect of the problem, we extend the static network $\mathcal{G}$ to a time-expanded network $\mathcal{G}_\mathcal{T} = (\mathcal{N}_\mathcal{T}, \mathcal{H}_\mathcal{T} \cup \mathcal{A}_\mathcal{T})$. To construct the graph $\mathcal{G}_\mathcal{T}$, each physical node $i \in \mathcal{N}$ is duplicated $|T|$ times. As a result, the set $\mathcal{N}_\mathcal{T}$ contains pairs (i, t) for each $i \in \mathcal{N}$ and $t \in T$. Time-expanded nodes of $\mathcal{N}_\mathcal{T}$ represent time-expanded suppliers $\mathcal{S}_\mathcal{T}$, time-expanded customers $\mathcal{C}_\mathcal{T}$, and time-expanded warehouses $\mathcal{W}_\mathcal{T}$. Arcs in $\mathcal{H}_\mathcal{T}$ represent the storage of products at a warehouse. Thus, for each warehouse $i \in \mathcal{W}$ and each $t \in [1, |T| - 1]$, there is a time-expanded arc $((i, t), (i, t + 1))$ in $\mathcal{H}_\mathcal{T}$. This time-expanded arc has a flow cost c_{ii} that is equal to the storage cost at the warehouse i. Arcs in A_T represent transportation between locations as well as departure and arrival times. To model transportation in the time-expanded network, for each $(i, j) \in \mathcal{A}$ and each time $t \in T$ such that $t + t_{ij} < |T|$, we build a time-expanded arc $((i, t), (j, t + t_{ij}))$. Thus, a transportation arc $((i, t), (j, t + t_{ij}))$ in $\mathcal{A}_\mathcal{T}$ models goods transportation from i to j, leaving at time t and arriving at time $t + t_{ij}$. We note that before creating the network, $\mathcal{G}_\mathcal{T}$, it may be necessary to modify t_{ij} values to ensure that arcs $(i, j) \in \mathcal{A}$ can be mapped to arcs of the form $((i, t), (j, t + t_{ij}))$. For example, if the planning horizon is discretized in time intervals of 2 hours, we modify the travel time of each transportation arc with a travel time lower than 2 hours, and we set it to 2 hours. Given a product $p \in \mathcal{P}$ and a customer $c \in \mathcal{C}$, its demand at time $t \in [1, |T| - 1]$ is denoted by: d^p_{ct}.

We next formulate the Logistics Service Network Design problem defined over a time-expanded network $\mathcal{G}_\mathcal{T}$ (LSNDP). Let $y^{tt'}_{ij}$ be an integer variable associated

with the number of trucks dispatched on transportation arc $((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}$. Continuous variable $x_{ij}^{ptt'}$ represents the quantity of product p that flows along the arc $((i,t),(j,t')) \in \mathcal{H}_\mathcal{T} \cup \mathcal{A}_\mathcal{T}$ (note that j may be equal to i as holding arcs are included). Also note that if $p \notin \mathcal{P}^i$ (i.e. supplier i does not supply product p) then continuous variables $x_{ij}^{ptt'}$ are not defined for all arcs $((i,t),(j,t'))$. Then, the LSNDP is as follows:

$$\text{minimize } z(\mathcal{G}_\mathcal{T}) = \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} + \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ij} x_{ij}^{ptt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ii} x_{ii}^{ptt'} \tag{2.1}$$

subject to :

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{ij}^{ptt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{jl}^{pt't''} = 0, \quad \forall (j,t') \in \mathcal{W}_\mathcal{T}, \forall p \in \mathcal{P} \tag{2.2}$$

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{ij}^{ptt'} \geq d_{jt'}^{p}, \quad \forall (j,t') \in \mathcal{C}_\mathcal{T}, \forall p \in \mathcal{P} \tag{2.3}$$

$$\sum_{p\in\mathcal{P}} x_{ij}^{ptt'} \leq \hat{c} y_{ij}^{tt'}, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{2.4}$$

$$x_{ij}^{ptt'} \in \mathbb{R}^+, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \cup \mathcal{H}_\mathcal{T}, \ \forall p \in \mathcal{P}^i \tag{2.5}$$

$$y_{ij}^{tt'} \in \mathbb{N}^+, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{2.6}$$

The objective function (2.1) minimizes the sum of fixed costs on transportation arcs (first term), variable costs on transportation arcs (second term), and variable costs on holding arcs (third term), i.e. holding costs. The first two constraints ensure the flow feasibility. Constraints (2.2) enforce the flow conservation at each warehouse. Constraints (2.3) impose the respect of each customer demands. Constraints (2.4) ensure that enough trucks are dispatched to transport products. Constraints (2.5) and (2.6) define the variable domains.

2.3 State of the Art

A supply chain is a complex network of facilities working together to effectively produce and deliver products to a market. The stages of the supply chain include the acquisiton of raw materials, the transformation of raw materials into finished products and the distribution of the final products to the end users. In this thesis,

we focus on the supply chain segment in charge of the delivery of final products to customers. The purpose of this section is to present the main publications, models and solution methods for problems related to the planning of transportation operations in supply chains.

Crainic and Laporte [Crainic 1997] classify freight transportation problems into three categories, depending on the level of decision to be dealt with. The strategic level addresses the highest level of planning. It involves long-term decisions that concerns the acquisition and location of major infrastructures. The tactical level adresses mid-term decisions related to the design of the transportation network, i.e. the allocation of existing resources for effectively transporting freight. The operational level adresses short-term decisions such as the planning of day-to-day operations in a highly dynamic environment. The LSNDP falls into the tactical level.

The LSNDP recently appeared in the literature and has not yet received much attention. The most notable LSNDP-related study is proposed by Dufour et al [Dufour 2018]. The problem adressed in their article is inspired by the management of logistics functions in the humanitarian sector. Due to the lack of research on the LSNDP, we position it regarding the existing network design problems. The LSNDP is a network design problem for suply chain optimization [Beamon 1998], such as the Supply Chain Network Design Problem (SCNDP) [Melo 2009] or the Logistics Network Design Problem (LNDP) [Srivastava 2008]. In addition, the LSNDP can be seen as a variant of the Service Network Design Problem (SNDP) [Crainic 1986]. Nevertheless, the LSNDP also differ to these problems in some fundamental ways.

The remainder of the section is divided in two parts. In 2.3.1, we review model for supply chain optimization that are relevant to our study, while 2.3.2 deals with the SNDP.

2.3.1 Models for supply chain optimization

A supply chain is a system of organizations, people and resources that manufactures and distributes products from suppliers to customers. For that purpose, the supply chain involves multiple companies with different tasks working together. Among recurring types of stakeholders in supply chains, there are:

- suppliers, providing raw material;
- plants, manufacturing raw materials into products;
- warehouses, for storing products, grouping orders and cross-docking;
- retailers, selling products to customers in quantities appropriate to their use;
- customers.

The process of manufacturing and distributing products decomposes in multiple stages. A type of stakeholder corresponds to each stage. As a result, a supply chain can be conceived as a multi-echelon network. An echelon groups together

all stakeholders of the same type. The links between the echelons characterize the way the freight is transported within the supply chain, and represents the order in which production and distribution stages take place.

Supply chain management consists in supervising the operations of the supply chain and coordinating the interactions between stakeholders involved, to ensure that the overall process is efficient and cost-effective. It is one of the most studied fields [Erengüç 1999, Mula 2010, Fahimnia 2013] in the scientific literature on operational research. There is a wide range of models for supply chain optimization, and these models cover all levels of decision making. At the strategic level [Vidal 1997], decisions include the selection of production, storage and distribution locations. At the tactical level [Esmaeilikia 2016], decisions include the planning of the production and the distribution, but also the inventory management. Finally, the operational level [Schmid 2013] includes aspects such as replenishment and delivery operations.

Rather than a single generic model for the optimization of the supply chain, there is a large number of models specific to each problem. This is primarily because supply chains structures differ from one industrial context to another. For example, our supply chain does not have plants, as it is not required to manufacture the products provided by suppliers. Similarly, our supply chain does not have retailers. Thus, models for supply chain optimization depends on the structure of the supply chain, but also on the level or levels of decision considered: whether a time horizon is considered or not, whether parameter uncertainty is considered or not, etc. In this section, we identify key scientific publications on supply chain modeling and optimization that are relevant to our problem. As a result, the selected works adress issues that are strongly connected to transportation and to the design of a distribution network.

Most bibliographic references in this section are classified in Tables 2.1 and 2.2. The chosen criteria are inspired by the review of Bravo et al. [Bravo 2013]. In Table 2.1 we show the model characteristics. We examine the structure of the supply chain and the modeling approach. We also indicate if the model is multi-product and/or multi-period. Finally, we report if the model is stochastic. In Table 2.2 we show the decisions adressed by the model. Five types of decisions are considered:

- location of the platforms and capacity allocation;
- production management, i.e. production quantities, quantities of raw materials to purchase and manufacture into products;
- warehouse inventory management;
- product distribution, i.e. flows of raw materials and products along the supply chain;
- management of a vehicle fleet.

We also report whether an exact method or a heuristic is used.

In the following, we review the literature on models for supply chain optimization, respectively for the strategic, tactical and operational level.

Strategic models for supply chain optimization

In the literature, most models for supply chain optimization problems are defined at the strategic level. These models involve long-term decisions and, in most cases, facility location. Facility location is important in multiple situations, for example when operations are extended to new geographical areas. The facilities are usually plants or warehouses, and the location of such facilities generally involve choosing production or storage capacities.

Melo et al. [Melo 2006] consider a four echelon network with a planning horizon over multiple periods. The proposed MILP integrates decisions on the locations of plants and warehouses, as well as on the quantities of products to purchase and on the inventory management. At each period, part or all of the capacity of a facility can be relocated to any other facility. Small and medium-scale random instances are solved with a commercial solver. In a further work [Melo 2012], authors propose a tabu search algorithm for solving large-scale instances that involve up to 50 plants/suppliers, 60 warehouses, 200 customers, 8 periods and 50 products.

Hugo and Pistikopoulos [Hugo 2005] adress a problem involving similar decisions. it differs on the inventory management. They propose a bi-objective model that evaluates both the economic cost as well as the environmental impact of the solutions. Bashiri et al. [Bashiri 2012] consider a four echelon network and develop a MILP including facility location, production planning, inventory management and product distribution. The model is multi-period, and facility capacities can be modified during the planning horizon. As the expansion of the facility capacities is supported by the revenues over time, the induced costs cannot exceed these incomes. In the experimental study, the commercial solver provides high quality solutions for small and medium size instances, but fails to solve large scale instances.

Thanh et al. [Thanh 2008] deal with a four echelon multi-product supply chain. They propose a MILP that includes: opening, closing or enlargement of facilities, inventory management and product distribution. For this problem, Thanh et al. [Thanh 2010] design an iterative LP-rounding heuristic combined with correction procedures. The largest instances involve up to 35 facilities, 300 customers, 5 periods, and 18 products.

In [Altiparmak 2009], Altiparmak et al. study a static MILP involving warehouse location, production planning, and product distribution from a single source. They propose an encoding structure that represents a solution as a transportation tree, and develop a steady-state genetic algorithm. In the computational study, the method is compared with a simulated annealing method, a Lagrangean heuristic and another genetic algorithm. The steady-state genetic algorithm outperforms the heuristics over small, meduim and large instances. Also, the solutions obtained by the steady-state genetic algorithm are compared with those obtained by a commercial solver. For small and medium instances, the steady-state genetic algorithm

provides near-optimal solutions for very small computation times. For large instances (up to 25 plants, 50 warehouses, 300 customers, 3 periods and 3 products) the commercial solver could not reach optimal solutions within a time limit of 10 hours.

In a seminal work, Geoffrion and Graves [Geoffrion 1974] propose a Benders decomposition for a static problem that combines product distribution and location of warehouses between plants and customers. In [Yeh 2005], Yeh formulate a static problem for a network with four echelons. In the model the following decisions are considered: the locations of plants and warehouses, and the distribution of a single product. The author proposes a 2-step algorithm. In the first step, a feasible solution is obtained by a greedy algorithm that iteratively identifies the unsaturated path with minimal cost. In the second step, the solution is improved with Local Search. Amiri [Amiri 2006] describes a related problem that include the location of plants and warehouses. More specifically, multiple levels of capacities are available for the facilities to be built, which differentiates this study from the previous ones. An efficient Lagrangean relaxation based method is proposed. It can solve large instances with up to 20 plants, 30 warehouses and 500 customers.

Eskandarpour et al. [Eskandarpour 2017] consider a static problem with a four echelon supply chain and multiple products. They develop a model that deals with facility location, product distribution and selection of transportation modes. They propose a Large Neighborhood Search Heuristic with removal and repair operators for facility locations, and a greedy heuristic for product distribution. The algorithm provides high-quality solutions for instances involving up to 30 suppliers, 30 plants, 60 warehouses, 300 customers and 5 products. Similarly to Altiparmak et al. [Altiparmak 2009], the computational study shows that commercial solvers are not efficient to solve industry size instances in reasonable computing times.

Many recent models for supply chain optimization either incorporate multi-objective functions, parameters uncertainty, or both aspects. Wang et al. [Wang 2011] present a static problem for a supply chain with suppliers, warehouses and customers. They propose a model that determine warehouse locations, product distribution and an environment protection level in the warehouses. The bi-objective function measures the operations total cost as well as the total carbon emission. Azaron et al. [Azaron 2008] study a tri-echelon supply chain. The problem includes plant location, product distribution and uncertainty associated with the demands, supplies, processing costs, transportation costs, shortage costs and capacity expansion costs. The authors propose a two-stage stochastic model. Decisions related to the plant locations are taken before the realization of the uncertainty, while decisions related to product distribution are considered in the second stage. The model has three objectives, and seeks to minimize the total cost, the variance of second stage costs and the financial risk. An instance with 4 suppliers, 4 plants and 3 customers is solved by a commercial software.

Santoso et al. [Santoso 2005] present a static problem that combines facility location and product distribution, with uncertainty on the demands and the facility capacities. They propose a Sample Average Approximation approach to reduce

the set of scenarios as well as a two-stage stochastic programming model. Based on a subset of the scenarios, the model is solved by a Benders strategy enhanced with acceleration techniques. In the computational study, two industrial instances are solved. The domestic instance involves 380 actors and 13 products, against 104 actors and 29 products for the global instance. The stochastic solutions are compared with the solution of the mean-value problem, i.e. the deterministic problem obtained by averaging the values of the uncertain parameters. The results show that the stochastic solutions are better, on average, for different sets of scenarios. Tsiakis et al. [Tsiakis 2001] present a two-stage stochastic programming model for a similar problem, with demand uncertainty. The authors present a case study of a European supply chain composed of 3 plants, 6 warehouses, 18 customers and 14 products.

In general, strategic models for supply chain optimization seek to locate facilities and estimate the resulting logistics planning on a long-term horizon. Regarding distribution planning, the variables considered are flows aggregated over multiple months or multiple years. Thus, these models are not suitable for designing transportation plans. Some of the instances model large-scale supply chains, but they also involve a small number of periods. Finally, all the methods proposed to solve difficult instances are heuristic.

Tactical models for supply chain optimization

Tactical models for supply chain optimization focus on mid-term decisions, such as production planning, product distribution or inventory management.

Lee et al. [Lee 2008] propose a MILP for a supply chain with plants, two levels of warehouses, and customers. Its aim is to design a replenishment plan for the warehouses and select routes for the products. Fixed capacities limit the flows on transportation arcs. However, the model does not consider vehicle allocation to move products. A decomposition heuristic and a rolling horizon heuristic are implemented to solve instances with up to 12 plants, 29 warehouses, 100 customers and 30 periods.

Yimer and Demirli [Yimer 2010] study a build-to-order problem that does not handle vehicle utilization. Raw materials are shipped from suppliers to plants, where products are manufactured. The products are then distributed to warehouses and retailers. The authors decompose the problem into two sub-problems. The first sub-problem determines how to manufacture the products and distribute them to the retailers. According to the solution of the first subproblem, the second subproblem aims to plan the acquisition of raw materials for the plants. The authors propose a genetic algorithm for the first subproblem and use a commercial solver for the second subproblem. Random instances are generated to represent a supply chain with 4 suppliers, 2 plants, 4 warehouses and 6 retailers. The heuristic provides near-optimal solutions for instances with up to 6 raw materials and 50 products.

Kopanos et al. [Kopanos 2012] describe a problem that applies to the food industry. They consider a two-echelon network composed of plants and warehouses.

They propose a MILP that models the assignement of products to processing units and determines a production schedule, truck loads and products inventories at each time period. In the computational study, two industrial instances are solved with a commercial solver. In the first instance, the supply chain network is composed of a single plant, 3 warehouses and 12 trucks. In the second case it has two plants, 5 warehouses and 20 trucks. Both cases consider 93 products, 23 product families and 9 time periods.

Cintron et al. [Cintron 2010] formulate a distribution network design problem to customer demands in a tri-echelon supply chain. They propose a model to determine the flows of products along the supply chain and allocate the vehicles required to transport products. The model has a multi-objective function that seeks to maximize profit, credit performance, distributors' reputation and minimize lead time. Experiments are conducted on an industrial case study that involves 4 plants, 2 warehouses and 71 customers. One objective of the authors was to assess if direct deliveries are cost-effective options. Results obtained with a commercial solver show that the company can significantly reduce its distribution costs by diminishing indirect deliveries and increasing direct deliveries.

Bilgen [Bilgen 2010] addresses a production and distribution planning problem that arises in a two-echelon network composed of plants and warehouses. The model determines the levels of inventory, the product distribution, and the vehicle utilization. In addition, the production volume of each production lines is determined. To deal with the uncertainty of the production and vehicle capacities, a fuzzy mathematical programming approach is proposed. Similarly, Fazlollahtabar et al. [Fazlollahtabar 2013] study a network composed of suppliers, warehouses and customers. They present a multi-product multi-period problem. The proposed model seeks to determine the distibution planning, the levels of inventory and the number of vehicles traveling from warehouses to customers. As the authors consider uncertain demands, supplying costs, holding costs and truck costs, they propose a fuzzy mathematical programming approach.

Less focus has been placed on tactical models for the optimization of the supply chain. These models generally seek to determine jointly the production planning and the distribution planning. However, decisions related to vehicle utilization are rarely taken into account. Again, commercial solvers are only able to solve small or medium size instances within a reasonnable amount of time. Most existing algorithms for solving large instances are heuristics.

Operational models for supply chain optimization

In the literature, operational issues for supply chain management are primarily extensions of the Vehicle Routing Problem (VRP). These extensions are intended to determine a set of routes to satisfy customer requests, while incorporating production and inventory decisions.

Chandra and Fisher [Chandra 1994] study a two-echelon supply chain with a single plant, multiple customer and multiple products. They propose a multi-period

model, coordinating production, inventories and distribution. Basically, the model combines a VRP with a Lot-Sizing problem. The authors either solve the problem directly, or use a 2-step approach. In the computational study, they generate random instances involving up to 50 customers, 10 time periods and 10 products, and show that gains can be achieved by integrating production and distribution decisions. Adulyasak et al. [Adulyasak 2015] name this problem the Production Routing Problem. They review two formulations of the problem as well as heuristics [Adulyasak 2012] and exact methods [Ruokokoski 2010, Archetti 2011].

Medina [Medina 2016] presents a problem based on an industrial case for sharing transportation resources in a tri-echelon supply chain. The author divides the supply chain in two parts. In the downstream network, long-haul distribution from suppliers to warehouses is modelled by a Service Network Design problem. In the upstream network, regional distribution from warehouses to customers is modelled by a rich VRP. The author presents an integrated model as well as techniques for modelling transfers in the warehouses. A matheuristic that generates columns for both the upstream and downstream network is proposed. The author solves industrial instances that involve up to 10 suppliers, 4 warehouses and 130 customers. Results demonstrate that the joint consideration of the two networks provides significant savings compared with a disjointed approach.

Table 2.1: Model characteristics

References	SC Structure	Modeling approach	Multi-period	Multi-product	Uncertainty
[Adulyasak 2015]	P-C	MILP	X	-	-
[Chandra 1994]	P-R	MILP	X	X	-
[Kopanos 2012]	P-W	MILP	X	X	-
[Geoffrion 1974]	W-C	MILP	-	X	-
[Hugo 2005]	S-P-C	MOMILP	X	X	-
[Santoso 2005]	S-W-C	MILP	-	X	X
[Amiri 2006]	P-W-C	MILP	-	-	-
[Lee 2008]	P-W-C	MILP	X	-	-
[Yeh 2005]	S-P-W-C	MILP	-	-	-
[Altiparmak 2009]	S-P-W-C	MILP	-	X	-
[Cintron 2010]	S-P-W-C	MOMILP	-	-	-
[Thanh 2010]	S-P-W-C	MILP	X	X	-
[Bashiri 2012]	S-P-W-C	MILP	X	X	-
[Melo 2012]	S-P-W-C	MILP	X	X	-
[Eskandarpour 2017]	S-P-W-C	MILP	-	X	-
[Yimer 2010]	S-P-W-R	MILP	X	X	-
[Tsiakis 2001]	P-W-R-C	MILP	-	X	X
Our problem	S-W-C	MILP	X	X	-

S=Supplier; P=Plant; W=Warehouse; R=Retailer; C=Customer;

MILP=Mixed integer linear program; MOMILP=Multi-objective mixed integer linear program;

Table 2.2: Model decisions and solution methods

References	Location	Production	Inventory	Distribution	Vehicles	Exact	Heuristic
[Geoffrion 1974]	X	-	-	X	-	X	-
[Hugo 2005]	X	X	-	X	-	X	-
[Adulyasak 2015]	-	X	X	X	X	X	X
[Chandra 1994]	X	X	X	X	X	-	X
[Tsiakis 2001]	X	X	-	X	-	-	-
[Santoso 2005]	X	-	-	X	-	-	X
[Yeh 2005]	X	-	-	X	-	-	X
[Amiri 2006]	X	-	-	X	-	-	X
[Lee 2008]	-	-	X	X	-	-	X
[Altiparmak 2009]	X	X	-	X	-	-	X
[Cintron 2010]	-	-	-	X	X	-	-
[Thanh 2010]	X	X	X	X	-	-	X
[Yimer 2010]	-	X	X	X	-	-	X
[Bashiri 2012]	X	X	X	X	-	-	-
[Kopanos 2012]	-	X	X	X	X	-	-
[Melo 2012]	X	X	X	X	-	-	X
[Eskandarpour 2017]	X	-	-	X	-	-	X
Our problem	-	-	X	X	X	X	X

Positioning the LSNDP with regard to supply chain optimization literature

In this thesis, we adress a problem with tactical aspects. We do not aim to extend or redesign the supply chain, thus facility location decisions are not considered. In addition, our model does not include production decisions. Indeed, suppliers of the logistics chain are large manufacturers or wholesalers with significant production capacities. Thus, as suppliers are notified in advance of the quantity of products required by DHL, they have time to adjust their production and avoid stockouts. Moreover, production prices are not included in our study. As a result, the choice of the shipping locations does not impact the transportation plan cost.

The problem we study in this thesis aims to compute a transportation plan in order to satisfy customer demands over a mid-term horizon. It faces decisions related to distribution, inventory, and vehicle management. More specifically, product itineraries in both space and time must be determined. The vehicles to transport products between sites must also be allocated.

In the literature, few models for supply chain optimization address vehicle utilization. However, the transportation plan profitability mainly relies on the number of vehicles used. To the best of our knowledge, supply chain models that involve vehicle-related decisions are not suitable to elaborate the transportation plan of our study. Most, if not all strategic and tactical models are based on full-truckload operations and disregard merge-in-transit. The only models that adress flow consolidations are operational models that focus on short term horizons.

2.3.2 Service Network Design Problem

The LSNDP aims to determine a cost-effective plan for transporting multiple products from suppliers to customers through a distribution network. As the amount of products is relatively small compared with the vehicle capacities, an effective strategy for achieving low transportation costs is consolidation. Specifically, routing products from suppliers to customers in a way that maximizes vehicle fill rates. As such, the LSNDP can be viewed as a variant of the Service Network Design Problem (SNDP) [Crainic 2000a, Wieberneit 2008], also named as the Capacitated Multicommodity Network Design Problem (CMNDP) [Gendron 1999], which seeks to determine a plan for transporting shipments through a known network of terminals. Like the LSNDP, the SNDP assumes that shipments do not require full vehicle capacity, and thus consolidation is an option to improve cost efficiency. The SNDP has applications in various fields including road [Kim 1999a], air [Barnhart 2002], maritime [Lo 2013] or multi-modal transportation [Fontaine 2017].

The SNDP is an extension of the Minimum Cost Multi-Commodity Flow Problem [Tomlin 1966] which seeks to route a set of commodities from their origin to their destination. Commodities are transported by means of services (vehicles, ships, drivers, etc.) allocated on the links. Each link is associated with a fixed cost for allocating services, and a variable cost for moving commodities using the allocated services.

The mathematical formulation of the SNDP is based on an underlying network structure. When the underlying network is static, it yields to a model that does not capture shipment timing. This is not suitable when decisions variables are defined on a planning horizon. An usual manner to incorporate temporal aspects is to expand the static network [Ford Jr 1958, Ford 1962] based on a time discretization of the planning horizon, as explained in subsection 2.2.2. Thus, a static solution in the time-expanded network [Skutella 2009] indicates both the locations and time intervals for service allocation and commodity shipments. However, modelling the temporal dimension yields to networks that are significantly larger. As a result, a SNDP defined on a time-expanded network is significantly harder to solve than the SNDP defined on the static version of the network.

We first review the literature on static versions of the SNDP. Then, we focus on the versions of the SNDP that include a planning horizon.

Static Service Network Design

A lot of attention has been paid to static service network designs. Among theses applications, most consider binary design variables, in which case the problem is also known as the *fixed charge* CMNDP. As the fixed charge CMNDP has a weak linear relaxation, Crainic et al. [Crainic 2001] propose techniques for computing effective lower bounds for large-scale instances. They develope bundle and subgradient methods based on two Lagrangean relaxations. The first relaxation is obtained by dualizing the capacity constraints. It is named as the shortest-path relaxation since the resulting problem decomposes into one shortest path problem for each

commodity. The second relaxation is obtained by dualizing the flow conservation constraints. It is named as the knapsack relaxation since the resulting problem decomposes into one knapsack problem for each arc. The computational study shows that the Lagrangean based methods compute high-quality dual solutions, in a very reasonnable time. Holmberg and Yuan [Holmberg 2000] propose a branch-and-bound approach that integrates a Lagrangian Heuristic based on the knapsack relaxation. The Lagrangian Heuristic use a subgradient search to provide dual solutions. To compute primal solutions, it sets values for the design variables and solve the resulting subproblem. The algorithm is tested on the instances of Holmberg and Hellstrand [Holmberg 1998] that involve up to 150 terminals, 1000 services and 282 commodities. Results show that the branch-and-bound outperforms a commercial solver given a time limit of 1 hour.

Chouman et al. [Chouman 2003] propose a cutting-plane algorithm that incorporates three valid inequalities derived from the well-known cutset inequalities. Given a cutset of the network, the authors propose the cover inequalities, the minimum cardinality inequalities and the network custet inequalities, as well as effective separation heuristics and lifting procedures. Promising cutsets of the network are identified by a heuristic. As a benchmark, the authors use a commercial software to solve the strong formulation of the CMNDP. In that formulation, the fixed charge CMNDP is enriched with the strong inequalities [Gendron 1994] that improve significantly the linear relaxation. The computational study shows that the lower bounds obtained by the cutting-plane algorithm dominate those produced by the strong formulation of the CMNDP.

In [Chouman 2016], the authors enhance their cutting-plane algorithm with two extra inequalities: the flow cover inequalities and the flow pack inequalities. They propose two branch-and-cut methods based on that algorithm, and implement a benchmark branch-and-cut based on the strong inequalities. The approaches are tested on the instances described in Crainic et al. [Crainic 2001], that involve up to 30 terminals, 700 services and 400 commodities. The proposed branch-and-cut methods clearly outperform the benchmark in terms of speed, but are not able to solve a lot more instances within the same time limit. Out of the 196 instances, 58 instances were not solved by the proposed branch-and-cut methods after 2 hours of computation, against 61 instances for the benchmark. After 10 hours of computation, the proposed branch-and-cut methods failed to solve 49 instances, against 42 instances for the benchmark. Chouman et al. [Chouman 2018] enhance their branch-and-cut algorithm with filtering methods. At each node of the branch-and-cut tree, the authors derive cuts and reduce the domains of some variables in order to forbid combinations of variable values that yield non optimal solutions. The authors demonstrate that the use of filtering methods improves the performance of the branch-and-cut algorithm as it enables to solve a larger amount of the instances described in Crainic et al. [Crainic 2001].

When considering large scale instances of the fixed charge CMND, it turns out that only heuristics are effective for computing high-quality solutions in a reasonnable amount of time. Crainic et al. [Crainic 2000b] propose a tabu search

for a model including path flow variables and arc design variables. They define two neighborhoods: a continuous neighborhood relative to the flow variables is used in the local search phase, and a discrete neighborhood relative to the design variables is used in the diversification phase. The local move is based on simplex pivots and column generation, while the diversification move sets design variables to a null value. The algorithm is tested on the instances of Gendron and Crainic [Gendron 1994, Gendron 1996], that involve up to 100 terminals, 700 services and 400 commodities. For small instances, the commercial solver outperforms the tabu search. However the metaheuristic provides good primal solutions for instances with more than 200 commodities, while the commercial solver does not even find a feasible solution within 6 hours of computation.

Ghamlouche et al. [Ghamlouche 2003] propose a new neighborhood for local search techniques. The idea behind the cycle-based neighborhood is to identify cycles, i.e. two points in the network that are connected by two different paths. A local move consists in deviating the flow from one path to another. To identify promising cycles, a shortest-path procedure is proposed. The authors demonstrate the effectiveness of cycle-based neighborhoods by comparing tabu search algorithms embedding different local moves. Ghamlouche et al. extend their work [Ghamlouche 2004] and propose an effective path relinking procedure for the fixed charge CMNDP. A tabu search is used to generate an initial set of elite solutions. Then, the path relinking procedure performs local moves based on the cycle neighborhoods to explore paths that connect the initial solutions. The method is tested on the instances of Gendron and Crainic [Gendron 1994, Gendron 1996]. Results show that the path relinking procedure improves the solutions obtained with the cycle-based tabu search.

Crainic et al. [Crainic 2004] present a slope scaling heuristic for the fixed charge CMNDP. Slope scaling consists in constructing a linear approximation of the original formulation that reflects both design and flow costs. The algorithm iteratively solves the approximated linear program and computes a new linearization vector. When the same solution is found in two consecutive iterations, the linearization factors are modified using dual information from a Lagrangean relaxation. The method also integrates long-term memory for diversification and intensification. The computational study shows that the slope scaling heuristic is competitive with the path relinking procedure in [Ghamlouche 2004].

Rodríguez-Martín and Salazar-González [Rodríguez-Martín 2010] propose a heuristic that adapts the local branching method [Fischetti 2003] to the fixed charge CMNDP. Local branching is a branch-and-bound procedure wherein each left node is characterized as a neighborhood of the current best solution. The heuristic is tested on the instances of Gendron and Crainic [Gendron 1994, Gendron 1996], with an imposed time limit of 600 seconds. Overall, it outperforms the cycle-based tabu search [Ghamlouche 2003], the path relinking procedure [Ghamlouche 2004] and the slope scaling heuristic [Crainic 2004]. For 34 out of the 43 instances, the local branching heuristic finds a solution with an objective value equal or lower than that of the best matheuristic solution. Overall, heuristic solutions are of same

quality to those obtained by CPLEX after an hour of computation.

Hewitt et al. [Hewitt 2010] propose an heuristic approach for solving the fixed charge CMNDP. The IP search algorithm provides both a primal solution and a dual bound at each iteration, which enables to assess the quality of the upper bound. Primal solutions are obtained by a local search technique. The method improves the current solution by solving an integer program defined on a subset of the variables. Dual bounds are obtained by solving the linear relaxation of the path-based formulation, enriched with valid inequalities. The IP search algorithm is tested on the instances of Gendron and Crainic [Gendron 1994, Gendron 1996]. For 35 instances out of 37, it outperforms the the cycle-based tabu search [Ghamlouche 2003] and the path relinking procedure [Ghamlouche 2004]. On average, it yields a solution with an objective value 2.76% lower than that of the best matheuristic solution. In addition, the a average optimality gap computed by the IP search algorithm is of 3.96%. The authors also generate larger instances that involve up to 500 terminals, 3000 services and 200 commodities. After 15 minutes of computation, the IP search computes solutions with an average objective value 22.95% lower than that of the solutions found by CPLEX after 12 hours. However, due to high optimality gaps little can be said about the absolute quality of these solutions.

Gendron et al. [Gendron 2018] propose a matheuristics that combines iterative linear programming methods with slope scaling heuristics. The approach is tested on the instances of Gendron and Crainic [Gendron 1994, Gendron 1996] and compared against state-of-the-art heuristics for solving the fixed charge CMNDP. The results show that the matheuristics is very competitive, as for a time limit of 1 hour it is able to find more best solutions than all heuristics in the state of the art.

A lot of research work has been devoted to the static SNDP, and in particular to its simplest version with binary design variables. Even if there exist many techniques for improving the quality of the lower bound, solving large scale instances of the fixed charge CMND remains a difficult task. To that purpose, effective heuristic methods are proposed and compared on the instances of Gendron and Crainic [Gendron 1994, Gendron 1996]. However, there appears to be a lack of efficient exact methods for solving large scale instances.

Service Network Design with planning horizon

Although applications of the SNDP that rely on time-expanded networks are of growing interest, the literature is less abundant on that subject. Here, we review the recent solution algorithms for SNDP variants with planning horizon.

Jarrah et al. [Jarrah 2009] present a SNDP with empty trailer repositioning that applies in the context of large-scale less-than-truckload freight operations. The authors propose a column generation approach, where columns represent freight flow trees, combined with slope scaling. They elaborate load plans for one-week planning horizons. However, the underlying time-expanded network assumes that all shipments from a physical location on a specific day occur at the same time, which significantly overestimate the consolidation opportunities. In the computational

study, five test problems are introduced with 680 commodities, 725 terminals and 31000 services. The authors compare the company base load plans to that computed by their algorithm. The computational study show that the obtained solution are significantly more economic. However, the final optimality gaps are not provided, which makes difficult to assess effectiveness of the algorithm. For a close problem, Erera et al. [Erera 2013] develop a model based on a time-expanded network with an appropriate discretization of time (8 time intervals per day). They propose a matheuristic that combines local search with integer programming.

Andersen et al. [Andersen 2009] introduce the *Service Network Design with Asset Management* (SNDAM), wherein design-balance constraints enforce the number of services entering a terminal to be equal to the number of services leaving this terminal. To elaborate repeatable schedules for the assets, they consider a cyclic time expanded-network. In cyclic time-expanded networks, arcs are added to connect the last nodes of the time horizon with the appropriate first nodes of the time horizon. Andersen et al. present four formulations of the SNDAM that alternate arc-based or path-based formulations relative to the flow variables, and arc-based or cycles-based formulations relative to the design variables. The authors propose a branch-and-price [Andersen 2011] for the path-cycle formulation. They generate instances based on a real-life rail transportation case study that involve up to 10 terminals, 50 services and 60 time periods. For the three largest instances the branch-and-price algorithm provides optimality gaps between 4.9% and 13.5%, after 10 hours of computation.

In the *Service Network Design with Resource Constraints* (SNDRC), Crainic et al. [Crainic 2014b] combine the management of generic assets with the limitations of available resources at terminals. The model is based on the arc-cycle formulation, and the authors propose an iterative matheuristic that combines slope-scaling, column generation, and mathematical programming techniques. Two set of instances are generated, based on the instances in [Andersen 2011]. After 10 hours of computation, the algorithms provides average optimality gaps of respectively 7.79% and 9.62%.

Boland et al. [Boland 2017] adress the *Continuous Time Service Network Design Problem* (CTSNDP) and propose an effective method to overcome large time-expanded networks. In the CTSNDP, the time discretization is sufficiently refined to capture all consolidation opportunities of the problem in continuous time. As such time discretizations yield to extremely large time-expanded networks, Boland et al. propose the Dynamic Discretization Discovery (DDD), an exact method that manipulates a sparse time-expanded network. Under certain conditions, the algorithm converges to a sparse time-expanded network such that solving the associated SNDP provides an optimal solution for the original problem. The authors computationally demonstrate that building that sparse time-expanded network and solving the associated SNDP require much less computational effort than solving the original problem. Instances in the computational study are based on the static instances in [Crainic 2001]. Authors describe a scheme to "time" the static instances, and consider a 1-minute time discretizations. The DDD algorithm overcomes the

time-expanded network sizes, and closes the optimality gap for most instances, in less than 2 hours.

SND problems based on time-expanded networks are of major interest, as a lot of real-life transportation problems consider a planning horizon. However, incorporating the temporal dimension can tremendously impact the underlying network size, as well as the computational effort required to solve an instance. Therefore, the recently proposed DDD opens promising perspectives for solving SNDP variants with planning horizons.

Positioning the LSNDP with regard to Service Network Design literature

Most bibliographic references in this section are classified in Table 2.3. In this table, we indicate if the design variables are binary or integer. We examine whether inventory cost or planning horizon are considered. We report whether shipments to be transported have a fixed origin or not. Finally, we report whether an exact method or a heuristic method is proposed.

Table 2.3: Characteristics of SNDP models

References	Design variables	Planning horizon	Inventory cost	Shipment origin not fixed	Exact	Heuristic
[Crainic 2000a]	Binary	-	-	-	-	X
[Holmberg 2000]	Binary	-	-	-	-	X
[Chouman 2003]	Binary	-	-	-	X	-
[Ghamlouche 2003]	Binary	-	-	-	-	X
[Crainic 2004]	Binary	-	-	-	-	X
[Ghamlouche 2004]	Binary	-	-	-	-	X
[Frangioni 2009]	Integer	-	-	-	X	-
[Jarrah 2009]	Integer	X	X	-	-	X
[Hewitt 2010]	Binary	-	-	-	-	X
[Rodríguez-Martín 2010]	Binary	-	-	-	-	X
[Andersen 2011]	Binary	X	-	-	X	-
[Erera 2013]	Integer	X	X	-	-	X
[Crainic 2016]	Binary	X	-	-	-	X
[Fontaine 2017]	Binary	X	-	X	X	-
[Boland 2017]	Integer	X	-	-	X	-
Our problem	Integer	X	X	X	X	X

While the LSNDP considered in this thesis and the SNDP are similar, they also differ in some fundamental ways. In the LSNDP, products flow from suppliers to customers. Thus, the LSNDP seeks to design a "forward flow" network. The SNDP, on the other hand, makes no presumptions regarding the direction of product/shipment flows, as any terminal can be the origin or destination of a commodity. In a sense, these unidirectional flows make the LSNDP easier to solve than the SNDP, as network structure can be exploited in the algorithms. The SNDP generally presumes that the origin and destination location for each shipment to be transported is specified *a priori*. In the LSNDP, however, customers request delivery of products that may be offered in multiple locations. As a result, the LSNDP also determines the origin location for each transported product request. In this sense, the LSNDP could be more challenging to solve than the SNDP as it considers an additional decision. Finally, in both problems the freight can be stored in

intermediate terminals. However, the LSNDP takes into account the storage costs in the objective function while the SNDP usually does not.

2.4 Conclusions

In this chapter we have defined the LSNDP, and have proposed a mixed-integer linear program. We also have positioned the LSNDP in the existing literature. Because of the network multi-echelon structure and the presence of multiple shipping origins to satisfy customer demands, the LSNDP can be seen as a problem for supply chain optimization. On the other hand, the LSNDP seeks to design a plan for transporting shipments through a distribution network, similarly to the SNDP. We have seen that for both the supply chain optimization problems and the SNDP, commercial solvers are only able to provide high-quality solutions for small and medium size instances. To face the challenge posed by large scale instances, it is necessary to implement effective solution algorithms.

Chapter 3

Managing the Number of Products: An Accelerated Partial Benders Strategy

Contents

3.1 Introduction

As the variables and constraints of the LSNDP are proportional to the number of products, this parameter has a significant impact on the model size. In addition, the larger is the number of products considered, the larger is the number of consolidation

possibilities, which increases the problem combinatorics. For these reasons, on-the-shelf optimization solvers have difficulty solving LSNDP instances with a significant number of products. In this chapter, we propose an exact solution algorithm that remains effective when the number of products increases.

We propose a Benders decomposition-based solution approach [Benders 1962] sharing similarities with the *partial Benders decomposition* approach proposed in [Crainic 2016] for speeding up the solution of scenario-based stochastic programs. In [Crainic 2016], information is derived from the scenarios that define the stochastic program, and used to strengthen the relaxation, namely the *master problem*. Computational results in [Crainic 2016] indicate that the information added to the relaxation greatly strengthens the bound it yields and increases the algorithm's speed of convergence.

Traditional Benders-type methods for deterministic network design problems (see Magnanti et al. [Magnanti 1986], Sridhar and Park, [Sridhar 2000] or Costa [Costa 2005] for applications to the SNDP) solve a master problem where the need to route shipments/products is relaxed, leaving a relaxation whose solution provides a weak bound on the objective function value of the original problem. We propose to strengthen the master problem with variables and constraints that model the need to route a single product that is an aggregation of the different product customer requests. We prove the validity of this new master problem and show with an extensive computational study that it yields significantly stronger bounds than the master problem traditionally solved. By examining the structure of the LSNDP, we derive valid inequalities to reinforce the master problem. Finally, we complement these techniques for strengthening the dual bound with a heuristic for quickly producing high-quality solutions.

The remainder of the chapter is organised as follows. In section 3.2 we present the partial Benders decomposition for the LSNDP. Valid inequalities and heuristic solutions are described resepectively in section 3.3 and section 3.4. In section 3.5, we present and discuss the results of an extensive computational study on the algorithm performance. Finally, we conclude in section 3.6.

3.2 A Partial Benders Decomposition for the Logistics Service Network Design Problem

In this section, we first present the general framework of the Benders algorithm. Then, we describe the straightforward Benders decomposition of the LSNDP. Finally, we propose a partial Benders decomposition of the LSNDP, and a master problem strengthened with information based on product aggregation.

3.2.1 General framework of the Benders decomposition algorithm

Benders decomposition is a solution strategy for large mixed-integer linear problems that decomposes a problem into a master problem and a set of subproblems. As

we consider a single subproblem in our method, we describe the method in that context. The master problem is a relaxation of the original problem that considers a subset of the variables in the original problem and an estimate of the optimal objective function value of the subproblem. Solving the master problem yields a dual bound on the optimal objective function value of the original problem and variable values that are used to formulate the subproblem that determine values for the remaining variables. When the subproblem is feasible, a feasible solution to the original problem can be constructed. This feasible solution yields a primal bound on the optimal objective function value of the original problem. When the objective function value of the subproblem does not agree with the estimate in the master problem, a Benders cut known as an *Optimality cut* is generated. This type of cut is typically generated from an extreme point of the dual polyhedron associated with the subproblem. When the subproblem is not feasible, a Benders cut known as a *Feasibility cut* is generated. This type of cut is typically generated from an extreme ray of the dual polyhedron associated with the subproblem. Generated cuts are added to the master problem, which is then solved again. The process repeats until the primal and dual bounds are within some pre-defined optimality tolerance, ϵ, or, no Benders cuts are generated.

3.2.2 Straightforward decomposition

For the LSNDP, the standard Benders decomposition yields a master problem that allocates trucks on transportation arcs and a subproblem that routes product flows using the capacity allocated by the master. With Ω and Γ representing the extreme rays and extreme points of the subproblem dual polyhedron, this classical master problem, **CMP**, is formulated as follows:

$$\min \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} + z \tag{3.1}$$

$$0 \geq \sum_{(c,t)\in\mathcal{C}_\mathcal{T}} \sum_{p\in\mathcal{P}} d_{ct}^p \rho_{ct}^p, \quad \forall \rho \in \Omega \tag{3.2}$$

$$z \geq \sum_{(c,t)\in\mathcal{C}_\mathcal{T}} \sum_{p\in\mathcal{P}} d_{ct}^p \pi_{ct}^p, \quad \forall \pi \in \Gamma \tag{3.3}$$

$$y_{ij}^{tt'} \in \mathbb{N}^+, \quad \forall((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{3.4}$$

$$z \in \mathbb{R}^+ \tag{3.5}$$

The objective function, (3.1), computes the total vehicle costs and an approximation of the costs associated with routing products. Both constraints (3.2) and (3.3) are traditional Benders cuts added dynamically after solving the subproblem. Constraints (3.2) are feasability cuts, and constraints (3.3) are optimality cuts.

Given an allocation of vehicles $\bar{y}$, the subproblem $\mathbf{SP}(\bar{y})$ is formulated as:

$$\min \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ij} x_{ij}^{ptt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ii} x_{ii}^{ptt'} \tag{3.6}$$

(2.2)- (2.3)

$$\sum_{p\in\mathcal{P}} x_{ij}^{ptt'} \leq \hat{c}\bar{y}_{ij}^{tt'}, \quad \forall((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{3.7}$$

$$x_{ij}^{ptt'} \in \mathbb{R}^+, \quad \forall((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \cup \mathcal{H}_\mathcal{T}, \ \forall p \in \mathcal{P}^i \tag{3.8}$$

Given a vehicle allocation $\bar{y}$, the subproblem seeks to satisfy customer requests for products, while minimizing the routing and storage costs incurred while doing so. Therefore, the subproblem has the same flow conservation constraints (2.2)- (2.3) as the complete model proposed in Chapter 2. Constraints (3.7) ensure that on each transportation arc the total flow cannot exceed the available capacity. It has been recognized that this decomposition leads to poor computational performance [Rahmaniani 2017] since the master problem and subproblem are unbundled. In particular, as the master problem is primarily constrained by the Benders cuts, in the early iterations of the algorithm the solution to the master problem is unlikely to yield a high-quality solution to the original problem.

3.2.3 Strengthening the master with product aggregation

As a result, Crainic et al. [Crainic 2014a, Crainic 2016] propose a *partial Benders decomposition* technique in the context of solving two-stage stochastic programs that strengthens the master problem by adding information to the master that is derived from the subproblem(s). For our problem, we add to the master problem variables and constraints related to the routing of a "super-product", χ that is derived from aggregating all the products $p \in \mathcal{P}$. Therefore for every customer $(c,t) \in \mathcal{C}_\mathcal{T}$, the demand for this "super-product" is obtained by aggregating the demand of all products: $D_{ct}^{\chi} = \sum_{p\in\mathcal{P}} d_{ct}^{p}$. Relatedly, for each arc $((i,t),(j,t'))$ and product p such that a flow variable $x_{ij}^{ptt'}$ is defined in the LSNDP, a super-product flow variable $x_{ij}^{\chi tt'}$ is defined in our master problem. Figures 3.1 and 3.2 illustrate an example, respectively before and after aggregating the products. Such an aggregation induces a loss information, as we cannot restrict suppliers to only ship products they offer. In the aggregated version each supplier offers the "super-product".

Our enhanced master problem (**EMP**) allocates vehicle capacity on transportation arcs in order to satisfy the routing of the super-product. **EMP** is formulated as follows:

$$\min \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} + z \tag{3.9}$$

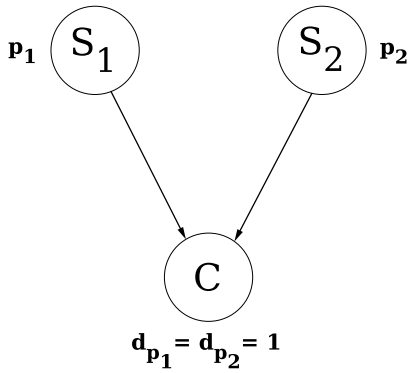

Figure 3.1: Customer requests one unit of each of two products, each of which supplied by a different supplier

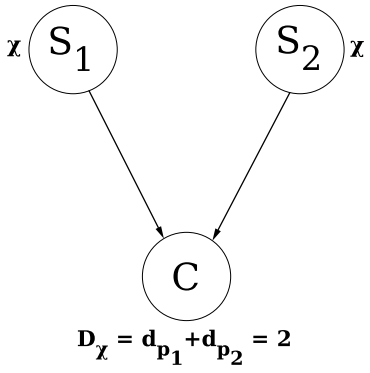

Figure 3.2: Customer requests two units of the "super-product" that is supplied by both suppliers

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{ij}^{\chi tt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{jl}^{\chi t't''} = 0, \quad \forall (j,t') \in \mathcal{W}_\mathcal{T} \tag{3.10}$$

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{ij}^{\chi tt'} \geq D_{jt'}^{\chi}, \quad \forall (j,t') \in \mathcal{C}_\mathcal{T} \tag{3.11}$$

$$x_{ij}^{\chi tt'} \leq \hat{c} y_{ij}^{tt'}, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{3.12}$$

$$z \geq \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} c_{ij} x_{ij}^{\chi tt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} c_{ii} x_{ii}^{\chi tt'} \tag{3.13}$$

(3.2)- (3.3)

$$x_{ij}^{\chi tt'} \in \mathbb{R}^+, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \cup \mathcal{H}_\mathcal{T} \tag{3.14}$$

$$y_{ij}^{tt'} \in \mathbb{N}^+, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{3.15}$$

$$z \in \mathbb{R}^+ \tag{3.16}$$

The objective function remains unchanged. Constraints (3.10) enforce the super-product flow conservation on each warehouse. Constraints (3.11) ensure that each customer demand of super-product is fulfilled. Constraints (3.12) ensure that vehicle capacity is allocated to support the flows of super-product. Constraints

(3.13) bounds the flows cost approximation z. Constraints (3.2) and (3.3) are the Benders cuts generated dynamically.

To ensure that a Benders decomposition-based scheme based on this master problem will converge to an optimal solution of the LSNDP, we must prove that **EMP** is a relaxation of the original problem.

Theorem 1 *The enhanced master problem, **EMP**, is a relaxation of the Logistics Service Network Design problem, LSNDP.*

Proof 1 *We prove this claim by showing that any feasible solution to the LSNDP is also feasible for the **EMP** and has the same objective function value. Let (x, y) be a feasible solution of the LSNDP, and consider a solution (x^χ, y, z) such that:*

$$x_{ij}^{\chi tt'} = \sum_{p\in\mathcal{P}} x_{ij}^{ptt'},\ \forall((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}, \quad z = \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} c_{ij}x_{ij}^{\chi tt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} c_{ii}x_{ii}^{\chi tt'}$$

*It is easy to prove this solution is feasible for the enhanced master problem. By construction, for any variable $x_{ij}^{ptt'}$ in the LSNDP, there is a corresponding variable $x_{ij}^{\chi tt'}$ in **EMP**. We know that for each warehouse $(j,t') \in \mathcal{W}_\mathcal{T}$ and each product $p \in \mathcal{P}$: $\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{ij}^{ptt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{jl}^{pt't''} = 0$. If we sum this expression on the products, we obtain:*

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} \sum_{p\in\mathcal{P}} x_{ij}^{ptt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} \sum_{p\in\mathcal{P}} x_{jl}^{pt't''} = 0$$

$$\Longrightarrow \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{ij}^{\chi tt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{jl}^{\chi t't''} = 0$$

Therefore (x^χ, y, z) respects constraints (3.10). As (x, y) respects constraints (2.3)- (2.4), it is trivial to demonstrate (x^χ, y, z) also respects constraints (3.11)-(3.12). By construction of z, (x^χ, y, z) respects constraints (3.13) which makes it an admissible solution to the enhanced master problem. Let $Q(x, y)$ be the objective value of (x, y) :

$$\begin{aligned} Q(x,y) &= \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} f_{ij}y_{ij}^{tt'} + \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ij}x_{ij}^{ptt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ii}x_{ii}^{ptt'} \\ &= \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} f_{ij}y_{ij}^{tt'} + \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} c_{ij}x_{ij}^{\chi tt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} c_{ii}x_{ii}^{\chi tt'} \\ &= \sum_{\mathcal{A}_\mathcal{T}} f_{ij}y_{ij}^{tt'} + z \ = \ Q(x^\chi, y, z) \end{aligned}$$

Solution (x^χ, y, z) that replicates solution (x, y) by an aggregation of flows, is feasible to the enhanced problem. The two solutions have identical objective function value. Thus EMP is a relaxation of the LSNDP.

There exist multiple state-of-the-art techniques for accelerating the convergence of a Benders algorithm. Among the most common examples, one can tighten the master problem with valid inequalities [Saharidis 2011], develop procedures for effectively computing master problem solutions [Rei 2009, Costa 2012], or put effort to generate strong Benders cuts [Magnanti 1981, Codato 2006, Fischetti 2010]. To accelerate the convergence of our Benders decomposition-based strategy, we develop valid inequalities as well as heuristic solutions. These acceleration techniques are described in the following sections.

3.3 Polyhedral Approach

By formulating the **EMP** with an aggregated product there is still some loss of information, such as which products each supplier can supply. This loss of information can lead the master problem to have an optimal solution which induces an infeasible subproblem. Thus, to prevent that from happening, we reinforce the master problem with three valid inequalities that render infeasible such solutions to the master problem. In this section, we describe these valid inequalities in detail. The validity of each inequality is demonstrated in the Appendix.

3.3.1 Super-Source Inequalities

We illustrate this valid inequality with a static network, but it has a natural analog in a time-expanded network. Specifically, Figure 3.3 illustrates two suppliers, with s_1 providing product p_1 and s_2 providing product p_2. On the demand side, customer c requires one unit of each product. Vehicle capacity is 10. There are transportation arcs (s_1, c) and (s_2, c), but the variable and fixed costs associated with (s_1, c) are less than with (s_2, c).

To formulate the **EMP**, we aggregate products p_1 and p_2 into one super-product χ, which is provided by both s_1 and s_2. Customer c's demand of the super-product is obtained by summing the demands of p_1 and p_2, $D_c^{\chi} = \sum_{p \in \mathcal{P}} d_c^p = d_c^{p_1} + d_c^{p_2} = 2$ Given the cost structure in this instance, the optimal solution to the **EMP** is to route 2 units of the super-product from s_1 to c. This solution is illustrated in Figure 3.4.

However, such a solution to the **EMP** will induce an infeasible subproblem as the vehicle allocation does not provide a path from s_2 to c, which is necessary for c to receive product p_2. To avoid such a solution, for each product $p \in \mathcal{P}$ we add to the network what we refer to as a "super-source" ss_p (see figure 3.5). Then, for each supplier node $s \in \mathcal{N}$ such that $p \in \mathcal{P}^s$, we add to $\mathcal{A}$ the arc (ss_p, s) with zero transit time, linear cost and fixed cost. In addition, we compute the total demand over all customers for each product, $D_p = \sum_{(c,t) \in \mathcal{C}_{\mathcal{T}}} d_{ct}^p$. We then add constraints to **EMP** to ensure that at least D_p units of the super-product is shipped from ss_p and that supplier nodes observe flow conversation with respect to the super-product.

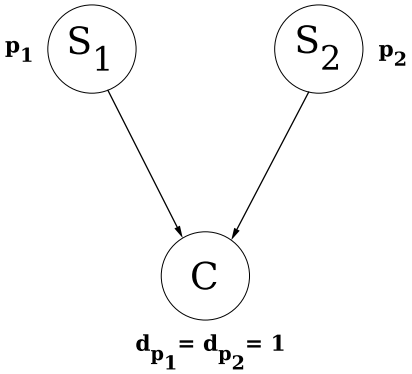

Figure 3.3: LSNDP instance

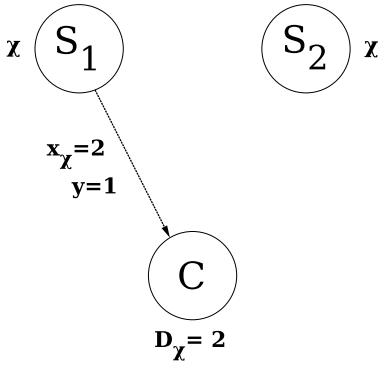

Figure 3.4: **EMP** optimal solution

Returning to our example, as the total demand for each of p_1 and p_2 is one unit, the proposed valid inequalities ensure that both ss_1 and ss_2 ship at least one unit of super-product. As the super-sources have outcoming arcs only to the suppliers that offer their products, in a solution to the **EMP** s_1 and s_2 must receive one unit of super-product respectively from ss_1 and ss_2 (see Figure 3.5). Also, as we enforce flow conservation for s_1 and s_2, any solution to the **EMP** must flow one unit of super-product from s_1 to c and from s_2 to c, meaning the vehicle allocation in the optimal solution to the **EMP** will induce a feasible subproblem (Figure 3.6)

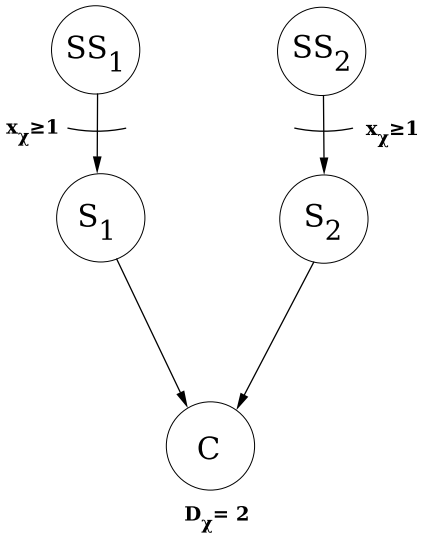

Figure 3.5: Valid inequalities

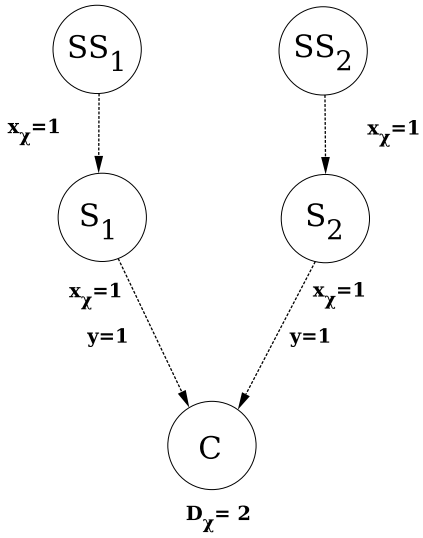

Figure 3.6: **EMP** new optimal solution

Formally, we add the following constraints to the **EMP**:

$$\sum_{((ss_p),(j,t))\in\mathcal{A}_\mathcal{T}} x^{\chi t}_{ss_p j} \geq D^p, \quad \forall p \in \mathcal{P} \tag{3.17}$$

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{ij}^{\chi tt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}} x_{jl}^{\chi t't''} = 0, \quad \forall (j,t') \in \mathcal{S}_\mathcal{T} \tag{3.18}$$

3.3.2 Direct Supply Inequalities

Like the previous valid inequality, we illustrate this inequality with a static network, as in Figure 3.7. We again have that suppliers s_1 and s_2 provide products p_1 and p_2, respectively. Now, however, there are two customers, each of which request one unit of both p_1 and p_2. Because each supplier only makes one of the two products requested, the "direct" arcs (s_1, c_1) and (s_2, c_2) cannot fully satisfy those customer demands. Instead, given this network, any feasible solution to the original problem requires that shipments from s_1 and s_2 be transported through warehouse w. To formulate the **EMP**, the products are aggregated, leaving c_1 and c_2 with the following super-product demands: $D_{c_1}^\chi = \sum_{p\in\mathcal{P}} d_{c_1}^p = 2$ and $D_{c_2}^\chi = \sum_{p\in\mathcal{P}} d_{c_2}^p = 2$. For some cost structures, the optimal solution to the **EMP** will be the solution illustrated in figure 3.8.

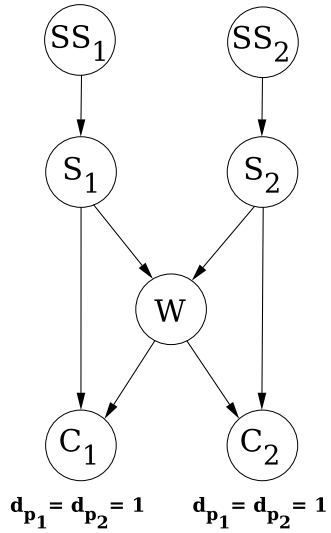

Figure 3.7: LSNDP instance

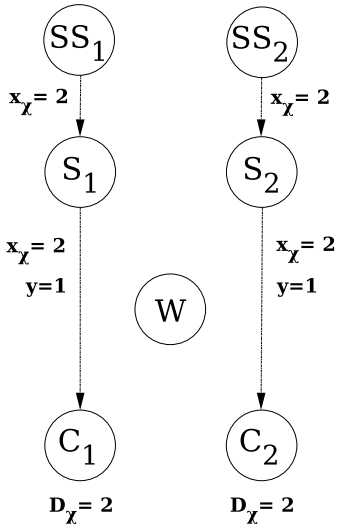

Figure 3.8: **EMP** optimal solution

However, such a solution induces an infeasible subproblem as the vehicle allocation it prescribes does not provide a path from s_1 to c_2 (or from s_2 to c_1). To avoid such an allocation, we use a valid inequality that restricts the flows of super-product on direct arcs. Specifically, given a supplier $s \in \mathcal{S}$ with product set $\mathcal{P}^s$, and a customer $c \in \mathcal{C}$, we compute how much demand c can receive from s: $d_c^s = \sum_{p\in\mathcal{P}^s} d_c^p$. We then restrict the quantity of super-product flow on the direct arc (s, c) to be no greater than d_c^s.

We illustrate these inequalities in Figure 3.9. As s_1 only offers p_1, the flow of super-product on the direct arc (s_1, c_1) cannot exceed $d_{c_1}^{p_1} = 1$. Similarly, the flow of super-product on direct arc (s_2, c_2) cannot exceed $d_{c_2}^{p_2} = 1$. With the inequalities

illustrated in Figure 3.9, an optimal solution to the **EMP** may be the solution illustrated in Figure 3.10, which induces a feasible subproblem.

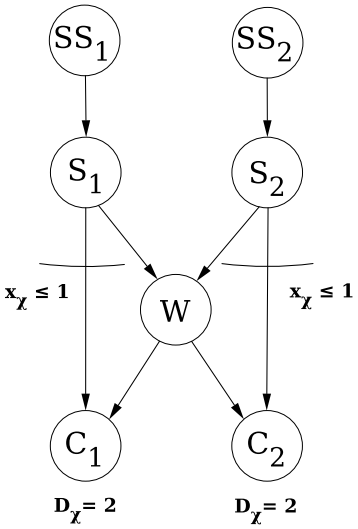

Figure 3.9: Valid inequalities

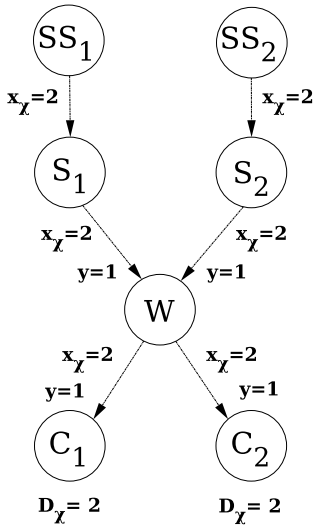

Figure 3.10: **EMP** new optimal solution

In the context of a time-expanded network, given a time-expanded supplier $(s,t) \in \mathcal{S}_\mathcal{T}$ with product set $\mathcal{P}^s$, and a time-expanded customer $(c,t') \in \mathcal{C}_\mathcal{T}$, we denote $d^{st}_{ct'} = \sum_{p \in \mathcal{P}^s} d^p_{ct'}$ as the demand that (c,t') can receive directly from (s,t). Formally, we add the following valid inequality to the **EMP**:

$$x^{\chi tt'}_{sc} \leq d^{st}_{ct'}, \quad \forall((s,t),(c,t')) \in \mathcal{A}_\mathcal{T},\ (s,t) \in \mathcal{S}_\mathcal{T},\ (c,t') \in \mathcal{C}_\mathcal{T} \tag{3.19}$$

3.3.3 Time-Based Super-Source Shipment Inequalities

Unlike the previous two inequalities, this valid inequality considers the timing of shipment activities. Like the previous two inequalities, we explain this inequality with an example. Specifically, Figure 3.11 illustrates a time-expanded network associated with the network depicted in Figure 3.3, wherein the time horizon is 3 days. Customer c's demand is zero for both products at time t_1. However, c requests one unit of each product at times t_2 and t_3. As a result, to formulate the **EMP**, the products are aggregated to yield the following super-product demands:

$$D^\chi_{ct_1} = 0, \ \ D^\chi_{ct_2} = D^\chi_{ct_3} = 2$$

A potential optimal solution to the resulting **EMP** is the solution depicted in Figure 3.12, which does not induce a feasible subproblem as the vehicle allocation does not provide a path from s_2 that arrives at c by day 2, when the delivery of one unit of p_2 is requested. We avoid such allocations in a manner similar to the Super-source inequalities described in subsection 3.3.1, but we now consider the

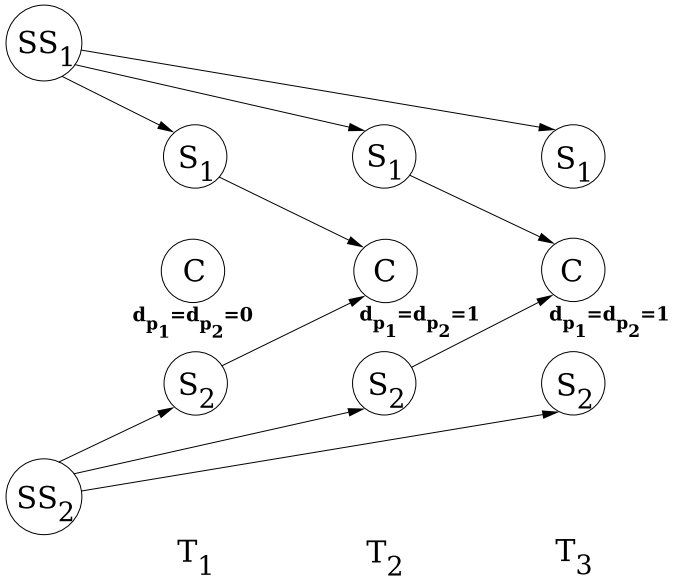

Figure 3.11: LSNDP instance

timing of shipment activities. Specifically, for each product $p \in \mathcal{P}$ and each time $t \in \mathcal{T}$, we sum the demands over all customers and obtain a global demand:

$$D_t^p = \sum_{c \in \mathcal{C}} d_{ct}^p, \quad \forall p \in \mathcal{P}, \forall t \in \mathcal{T}$$

Then, for each product $p \in \mathcal{P}$ and each time $t \in \mathcal{T}$, we sum the global demands requested before time t or at time t and obtain a cumulative global demand:

$$\bar{D_t^p} = \sum_{t' \leq t} D_{t'}^p, \ \forall p \in \mathcal{P}, \forall t \in \mathcal{T}$$

For our example, the global demands and cumulative global demand are as given in the following tables. The left table corresponds to product p_1, the right table corresponds to product p_2. First line shows the global demands, the second line shows the cumulative global demands.

t	1	2	3
$D_t^{p_1}$	0	1	1
$\bar{D_t^{p_1}}$	0	1	2

t	1	2	3
$D_t^{p_2}$	0	1	1
$\bar{D_t^{p_2}}$	0	1	2

Given a period t wherein there is an increase in the cumulative global demand for a product (e.g. day 3 for p_1 in our example), we derive a latest time at which that product can be shipped from the corresponding super-source and be delivered on time. To do so, we determine in $\mathcal{G}$ the shortest-path (in terms of time) between each super-source ($ss_p \in \mathcal{SS}$) and each customer ($c \in \mathcal{C}$). We denote the length

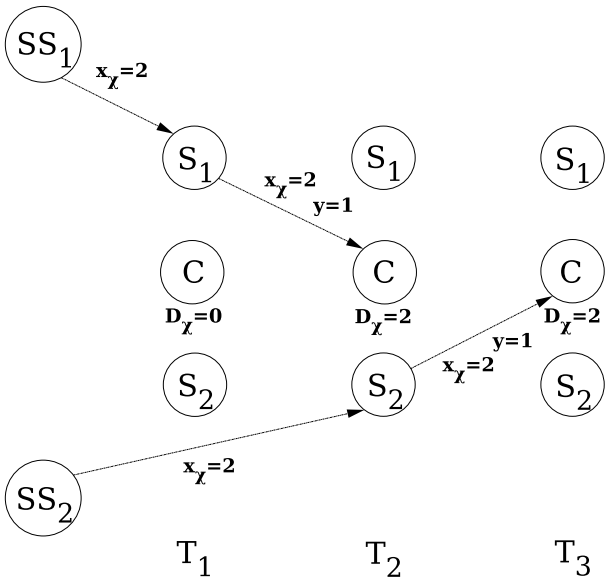

Figure 3.12: **EMP** optimal solution

of this shortest path, in terms of time, by $t^{min}_{ss_pc}$. Then, for each super-source, ss_p, we determine the shortest possible delivery time, $t^{min}_{ss_p} = \min_{\forall c \in \mathcal{C}} t^{min}_{ss_pc}$. This duration indicates the smallest transit time between super-source ss_p and a customer for its product. Thus, given a cumulative global demand $\bar{D}^p_t$ such that $\bar{D}^p_t > \bar{D}^p_{t-1}$, if the total amount of product p shipped from super-source ss_p before $t - t^{min}_{ss_p}$ is strictly less than $\bar{D}^p_t$, the demands of product p at time t cannot be satisfied.

Returning to our example, in Figure 3.11, one can see that the shortest-path duration from ss_{p_2} to c is $t^{min}_{ss_{p_2}c} = 1$. As c is the only customer, we have that $t^{min}_{ss_2} = 1$. Thus, for each $t^* \in \mathcal{T}$ such that $\bar{D}^{p_2}_{t^*} > \bar{D}^{p_2}_{t^*-1}$, we must enforce that the flow of super-product from ss_{p_2} to supply nodes (s,t) with $t \leq t^* - t^{min}_{ss_2}$ is at least $\bar{D}^{p_2}_{t^*}$. For example, as $\bar{D}^{p_2}_{t_2} > \bar{D}^{p_2}_{t_1}$ the super-product flow from ss_2 to (S_2, T_1) must be greater or equal to $\bar{D}^{p_2}_{t_2} = 1$, which is not the case in the solution depicted in Figure 3.12. Similar reasoning can be applied to p_1. We illustrate these valid inequalities in Figure 3.13 and the resulting optimal solution to the **EMP** in Figure 3.14, which will induce a feasible subproblem.

Formally, for each product $p \in \mathcal{P}$ and every time $t^* \in \mathcal{T}$ such that $\bar{D}^p_{t^*} > \bar{D}^p_{t^*-1}$ - we add the following constraint to **EMP**:

$$\sum_{\substack{((ss_p),(j,t)) \in \mathcal{A}_\mathcal{T} \\ t \leq t^* - t^{min}_{ss_p}}} x^{\chi t}_{ss_p j} \geq \bar{D}^p_{t^*}, \quad \forall p \in \mathcal{P},\ \forall t^* \in \mathcal{T},\ \bar{D}^p_{t^*} > \bar{D}^p_{t^*-1} \tag{3.20}$$

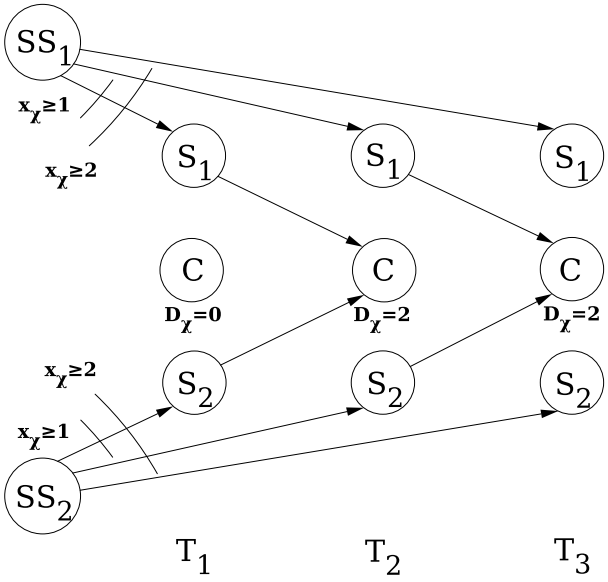

Figure 3.13: Valid inequalities

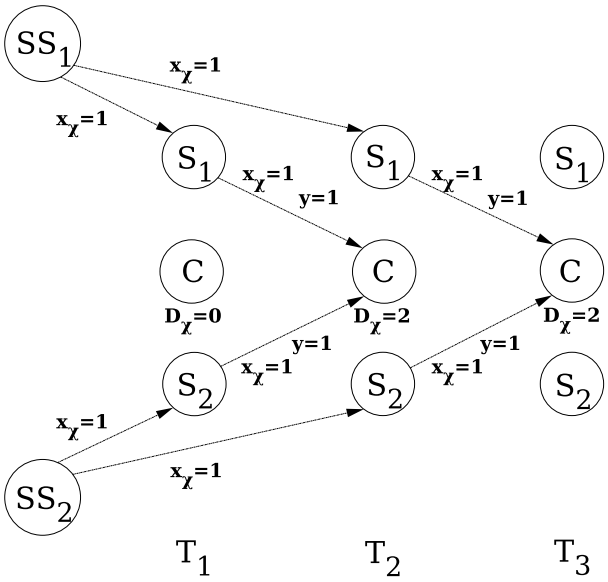

Figure 3.14: **EMP** new optimal solution

3.4 Heuristic Solutions

In a standard Benders decomposition-based solution method, primal solutions are only produced when the subproblem is feasible. Thus, to speed up the search for high-quality primal solutions, we propose a slope-scaling heuristic that derives a feasible primal solution from an unfeasible subproblem. In short, we first determine whether we should attempt to repair the vehicle allocation, $\bar{y}$, and then we repair that allocation. We next describe each step in detail. Algorithm 1 provides a high-level description of the procedure.

Algorithm 1 Build heuristic solution

Require: EMP solution $(\bar{x^{\chi}}, \bar{y})$, threshold $\bar{r}$
  if SP($\bar{y}$) is infeasible **then**
    Add corresponding Benders feasibility cut to the master
    Build SP$_s(\bar{y})$ with slack variables s^p_{jt} for each customer's demand
    Solve SP$_s(\bar{y})$ to obtain $(\dot{x}, \dot{s})$
    Evaluate the percentage, r, of demand quantities, d^p_{jt}, served with slack variables
    if $r < \bar{r}$ **then**
      Determine initial vehicle allocation $\dot{y}$ from $\dot{x}$
      for Each demand d^p_{jt} served by slack variables in decreasing order **do**
        Route demand d^p_{jt} with a slope-scaling linear program
        Update $(\dot{x}, \dot{y})$
      end for
      if $(\dot{x}, \dot{y})$ has a better objective value than the incumbent **then**
        Update the incumbent
      end if
      Solve SP$(\dot{x}, \dot{y})$ and add the resulting Benders optimality cut to the master
    end if
  end if

3.4.1 Identifying Promising Unfeasible Solutions

To determine whether to repair a vehicle allocation, $\bar{y}$, we formulate a subproblem $\mathbf{SP}_s(\bar{y})$ with slack variables to identify how "close" the subproblem is to being feasible given that allocation. The premise is that the closer the subproblem is to being feasible, the more likely a high-quality solution can be derived by making just a few modifications to $\bar{y}$. The subproblem, $\mathbf{SP}_s(\bar{y})$, is formulated as follows:

$$\min \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ij} x^{ptt'}_{ij} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} \sum_{p\in\mathcal{P}} c_{ii} x^{ptt'}_{ii} + \sum_{(i,t)\in\mathcal{C}_\mathcal{T}} \sum_{p\in\mathcal{P}} c^{prohib} s^p_{it} \tag{3.21}$$

(2.2)- (3.7)

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{ij}^{ptt'} + s_{jt'}^{p} \geq d_{jt'}^{p} \quad \forall (j,t') \in \mathcal{C}_\mathcal{T}, \forall p \in \mathcal{P} \tag{3.22}$$

$$x_{ij}^{ptt'} \in \mathbb{R}^{+}, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \cup \mathcal{H}_\mathcal{T}, \forall p \in \mathcal{P}^i \tag{3.23}$$

$$s_{it}^{p} \in \mathbb{R}^{+}, \quad \forall (i,t) \in \mathcal{C}_\mathcal{T} \tag{3.24}$$

This linear program differs from the original subproblem by the extra slack variables, s_{jt}^{p}, which appear in the objective, and the replacement of constraints (2.3) with constraints (3.22). We note that the slack variables guarantee that this suproblem is feasible. In the method we propose, we choose an objective function coefficient, c^{prohib}, for these slack variables that is high enough that an optimal solution to $\mathbf{SP}_s(\bar{y})$ will only assign positive values to the slack variables when the original subproblem is infeasible. Given an optimal solution $(\dot{x}, \dot{s})$ to $\mathbf{SP}_s(\bar{y})$, we compute the percentage of customer demands that cannot be met with the allocation $\bar{y}$:

$$r = \frac{\sum\limits_{(i,t)\in\mathcal{C}_\mathcal{T}} \sum\limits_{p\in\mathcal{P}} \dot{s}_{it}^{p}}{\sum\limits_{(i,t)\in\mathcal{C}_\mathcal{T}} \sum\limits_{p\in\mathcal{P}} d_{it}^{p}}$$

This measure is our indicator of how "close" the allocation of vehicle capacities, $\bar{y}$, is to inducing a feasible subproblem. We compare this percentage with a threshold, $\bar{r}$, to determine whether we should attempt to repair the solution $\bar{y}$ so that it induces a feasible subproblem, $\mathrm{SP}(\bar{y})$.

3.4.2 Slope-Scaling Repair Mechanism

Given a vehicle allocation that is to be repaired, the heuristic determines the minimum vehicle allocation needed to route the product flows, $\dot{x}_{ij}^{ptt'}$, specified by the subproblem. Specifically, the heuristic computes $\dot{y}_{ij}^{tt'} = \left\lceil \frac{\sum_{p\in\mathcal{P}} \dot{x}_{ij}^{ptt'}}{\hat{c}} \right\rceil, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}$. The heuristic then iterates through customer demands served by slack variables in decreasing order of size, $d_{jt'}^{p}$, and finds a route for each demand via a slope-scaling-type ([Kim 1999b, Kim 2000, Crainic 2004, Kim 2006, Zhu 2014]) procedure that we next describe.

The slope-scaling procedure for demand request $d_{jt'}^{p}$ begins by computing the remaining capacity on each arc given the vehicle allocations, $\dot{y}_{ij}^{tt'}$. It does so by computing $res_{ij}^{tt'} = \hat{c}\dot{y}_{ij}^{tt'} - \sum\limits_{p\in\mathcal{P}} \dot{x}_{ij}^{ptt'}, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}$. Then, the procedure determines how many "extra" vehicles are needed on an arc if it is to transport the demand request. Specifically, it calculates for $d_{jt'}^{p}$ the quantity

$extra_{ij}^{tt'} = \max\left(0, \left\lceil \frac{d^p_{jt'} - res_{ij}^{tt'}}{\hat{c}} \right\rceil\right)$. These quantities are then used to compute the terms, $\widetilde{c}_{ij}^{tt'}$, that linearize the fixed costs associated with allocating additional vehicles to arcs. Specifically, the quantities $\widetilde{c}_{ij}^{tt'} = \frac{c_{ij} d^p_{jt'} + f_{ij} extra_{ij}^{tt'}}{d^p_{jt'}} = c_{ij} + \frac{f_{ij} extra_{ij}^{tt'}}{d^p_{jt'}}$ are computed. These terms are used to formulate and solve the following linear program for routing $d^p_{jt'}$.

$$\min \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} \widetilde{c}_{ij}^{tt'} x_{ij}^{tt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} c_{ii}^{tt'} x_{ii}^{tt'} \tag{3.25}$$

subject to

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{ij}^{tt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{jl}^{t't''} = 0, \quad \forall (j,t') \in \mathcal{W}_\mathcal{T} \tag{3.26}$$

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{ij}^{tt'} \geq d^p_{jt'}, \quad \forall (j,t') \in \mathcal{C}_\mathcal{T} \tag{3.27}$$

$$x_{ij}^{tt'} \in \mathbb{R}^+, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \cup \mathcal{H}_\mathcal{T}., p \in \mathcal{P}^i \tag{3.28}$$

The objective function of this linear program computes the total, approximated, routing costs on transportation arcs and storage costs associated with holding arcs. Flow conservation is enforced by (3.26), while the satisfaction of demand d^p_{ct} is enforced by (3.27). Given a solution, $\ddot{x}$ to this linear program, the heuristic updates the overall solution, $(\dot{x}, \dot{y})$ as follows:

$$\dot{y}_{ij}^{tt'} = \left\lceil \frac{\dot{x}_{ij}^{ptt'} + \ddot{x}_{ij}^{tt'}}{\hat{c}} \right\rceil, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T},$$

$$\dot{x}_{ij}^{ptt'} = \dot{x}_{ij}^{ptt'} + \ddot{x}_{ij}^{tt'}, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \cup \mathcal{H}_\mathcal{T}.$$

After executing these steps for each demand served by slack variables, the resulting solution $(\dot{x}, \dot{y})$ is a feasible solution for the original problem. If its objective function value is lower than the current best, the newly-found solution is recorded as being the best known solution. In addition, note that for each heuristic solution $(\dot{x}, \dot{y})$ found, we solve the subproblem associated to the vehicle allocation $\dot{y}$. This enables us to generate a new optimality cut, and thus strengthen the master problem.

3.5 Computational study

In this section, we assess the efficiency of our proposed algorithm through a computational study. We first describe the instances used and then how the study was performed. We then analyze results from that study, first to assess the effectiveness of the approach and then the impact of each of its components.

3.5.1 Instances

The algorithm is tested on a set of instances produced by a random generator inspired by the operations of our industrial partner. One parameter is the size of the node set. Given that size, the generator randomly generates a graph $\mathcal{G}$ on a square area of size 100×100. Based on the supply chain of our industrial partner, we label 30% of the nodes as supplier locations and 50% as customer locations. Amongst the remaining nodes, two are labeled as central warehouses whereas the remaining are labeled as regional warehouses.

Regarding transportation arcs, $\mathcal{A}$ contains an arc from each supplier to the nearest central warehouse and from each central warehouse to each regional warehouse. In addition, there is a transportation arc in $\mathcal{A}$ to each customer from its nearest regional warehouse. A second parameter of the generator is α, a connectivity radius value that is used to determine additional arcs in $\mathcal{A}$. Specifically, a transportation arc is added from each supplier to any regional warehouse/customer in a radius of α units. Similarly, a transportation arc is added from each regional warehouse to any customer in a radius of α units.

The travel times and fixed costs for an arc are set to be proportional to its length. For the travel time, we set a maximum of $\bar{t}_{ij}^{max} = 24h$. We calculate d^{min} and d^{max}, the smallest and largest distances between nodes of V. Then, given two nodes i and j with distance d_{ij}, we set the travel time as: $\bar{t}_{ij} = \bar{t}_{ij}^{max} * \left(\frac{d_{ij}-d^{min}}{d^{max}-d^{min}}\right)$. The truck fixed cost is set to 0.55 per unit of distance: $f_{ij} = 0.55 d_{ij}$. Finally, on each arc we set a flow cost of 0.4 for loading and unloading each pallet of product, yielding $c_{ij} = 0.8$.

The temporal aspect of an instance is determined by two more parameters: (1) D, the number of days in the planning horizon, and, (2) Δ, the time granularity. The time granularity Δ expresses the number of time points per day in the time-expanded graph. For example, if $\Delta = 2$ there are 2 time points per day and each pair of contiguous time points is separated by a time interval of 12 hours. Then, the time horizon for the model is $T = [1, D \times \Delta]$. To express the travel time of an arc in terms of time points, we set $t_{ij} = \lceil \bar{t}_{ij} * \Delta/24 \rceil$, where the original travel time of arc $(n_i, n_j) \in \mathcal{G}$ is $\bar{t}_{ij}$.

The last parameter is the size of the product set, P. Regarding suppliers, each supplier has a 15% chance of offering a product. Regarding customers, each customer has a demand for each product one, two or three times each week with the determination made randomly. These days are chosen randomly. The volume of each demand is randomly chosen in the interval $[0; 5]$. Vehicle capacities are set to

60.

For our experiments, we generate instances based on the following parameter values: $|\mathcal{G}| = \{50\}$, $\alpha = \{10, 30\}$, $D = 30$ days, $\Delta = \{2, 3, 4\}$, and $|P| = \{100, 200, 300, 400, 500\}$. Thus, there are 30 possible combinations of parameter values and for each combination we generated 5 instances, leaving 150 instances in total.

In Figure 3.15 we report the LSNDP growth in terms of variables and constraints, as the number of products increases. Growth is normalised on instances with 100 products, which include an average of 718 156 variables and 111 720 constraints. We see that growth is substantial and quasi-linear, with a factor 5 increase for instances with $|\mathcal{P}| = 500$.

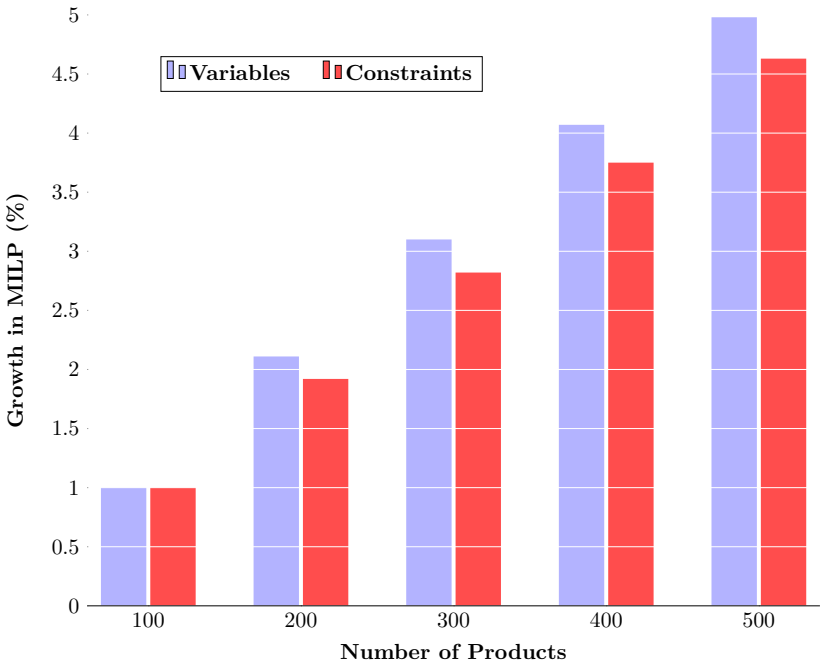

Figure 3.15: Growth in MILP normalized for instances with $|\mathcal{P}| = 100$

3.5.2 Setup of the study

To assess the efficiency of each component of the proposed algorithm, we tested several methods on the instances detailed above. The first method, **SPBD**, is the partial Benders decomposition-based scheme, wherein the Super-Product master problem, **EMP**, is used, but the valid inequalities and heuristic are not used. Then, to test the effectiveness of the valid inequalities, the methods **SPBD1**, **SPBD2** and **SPBD3** are Super-Product Benders Decomposition methods supplemented with the a priori cuts described in subsections 3.3.1 (**SPBD1**), 3.3.2 (**SPBD2**), and 3.3.3 (**SPBD3**). The method **SPBD123** employs all three a priori cuts. The method **SPBD123H** is similar, only it also employs the proposed heuristic.

As benchmarks, we used three other methods. The first method, **CBD** is the Classic Benders Decomposition, wherein none of the enhancements proposed in this paper are used. In the second, **CPLEX**, the complete program is solved with CPLEX's default branch-and-cut. In the last, **CPLEX-Benders**, the complete program is solved with CPLEX's automatic Benders decomposition.

All Benders decomposition-based methods are implemented with the callback framework wherein subproblems are solved within the context of the branch-and-bound tree used to solve the master problem. Specifically, whenever an integral solution is found in the tree, the subproblem is solved. The resulting cut is then embedded in every node of the tree, and may cut-off the incumbent. The process terminates once the optimality gap is closed. We initiate every method with a heuristic solution (x_h, y_h) obtained by setting each vehicle variable, $y_{ij}^{tt'}$ to the ceiling of its value in the optimal solution of the linear relaxation of the LSNDP.

Note that we implemented versions of the Benders decomposition-based methods that generated pareto-optimal cuts ([Magnanti 1981]). Doing so did not improve method performance.

All algorithms are coded in C++ and executed on an Intel Xeon E5-2695 processor with 16 GB of memory under Linux 16.04. Linear and integer programs were solved using Cplex 12.7. All algorithms are executed with a stopping criteria of a proven optimality gap of 1% or less and a maximum run-time of 1.5 hours. For **SPBD123H**, the threshold parameter, $\bar{r}$, is set to 20% after tuning.

3.5.3 Effectiveness of SPBD123H

We first benchmark **SPBD123H** against **CBD**, **CPLEX**, and **CPLEX-Benders** by comparing optimality gaps at termination for each method. We note over the 150 instances, very few were solved within the time limit. More specifically, CPLEX solved to optimality three of the smallest instances. We present in Table 3.1 averages of these gaps over instances with the same number of products $|\mathcal{P}|$. The best values are in bold. We also display distributions (in deciles) of the gaps according to the number of products for $|\mathcal{P}| = 100$ (Figure 3.16), $|\mathcal{P}| = 200$ (Figure 3.17), $|\mathcal{P}| = 300$ (Figure 3.18), $|\mathcal{P}| = 400$ (Figure 3.19) and $|\mathcal{P}| = 500$ (Figure 3.20).

Table 3.1: Optimality gaps comparison of **CBD**, **CPLEX-Benders**, **CPLEX** and **SPBD123H**

$\lvert\mathcal{P}\rvert$	**CBD**	**CPLEX-Benders**	**CPLEX**	**SPBD123H**
	Opt. gap	Opt. gap	Opt. gap	Opt. gap
100	95.75%	54.80%	7.22%	**5.02%**
200	99.89%	57.80%	25.34%	**3.59%**
300	99.90%	54.73%	64.58%	**3.94%**
400	99.99%	56.07%	69.71%	**3.38%**
500	99.99%	51.96%	75.74%	**2.80%**

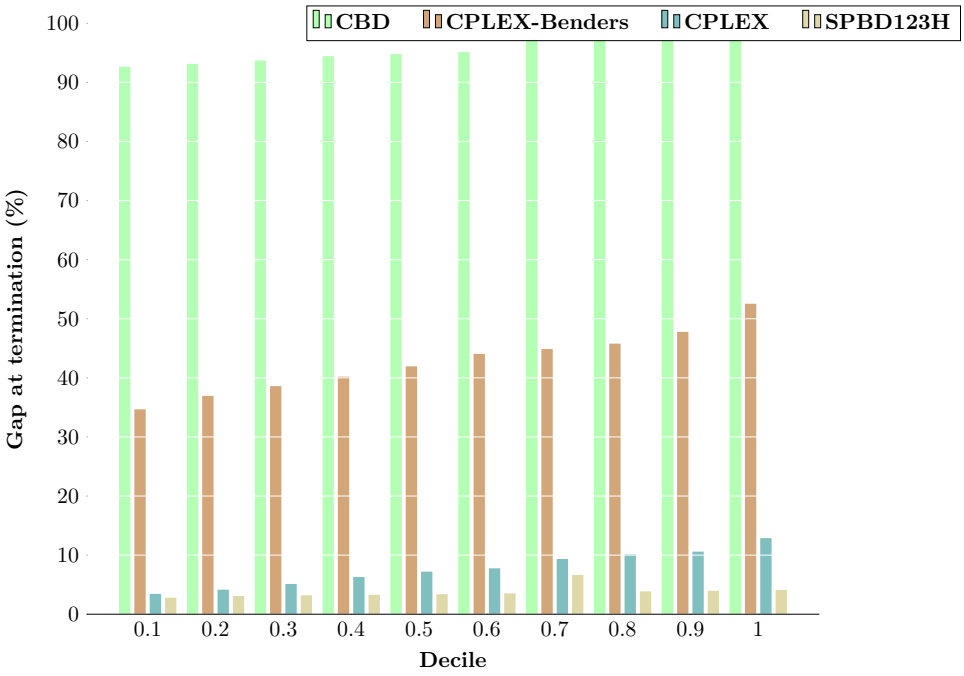

Figure 3.16: Gap at termination distribution for $|\mathcal{P}| = 100$

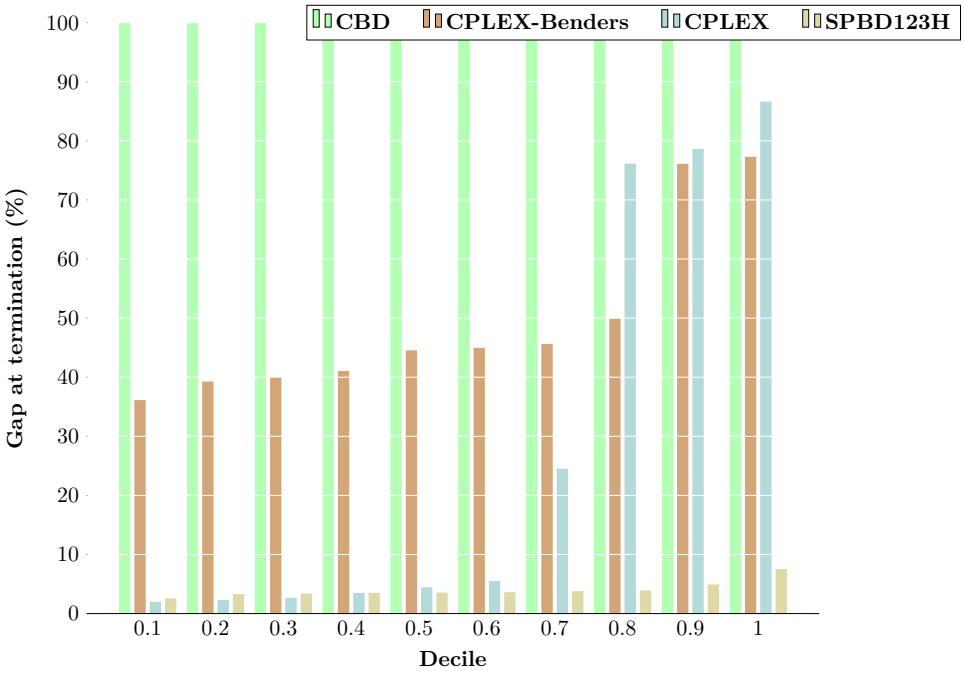

Figure 3.17: Gap at termination distribution for $|\mathcal{P}| = 200$

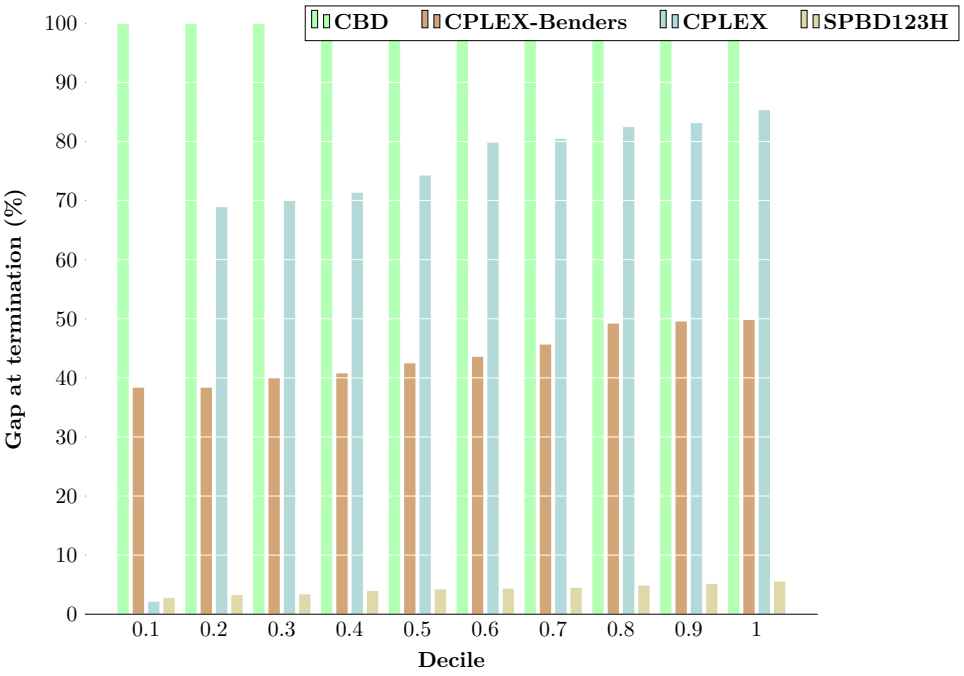

Figure 3.18: Gap at termination distribution for $|\mathcal{P}| = 300$

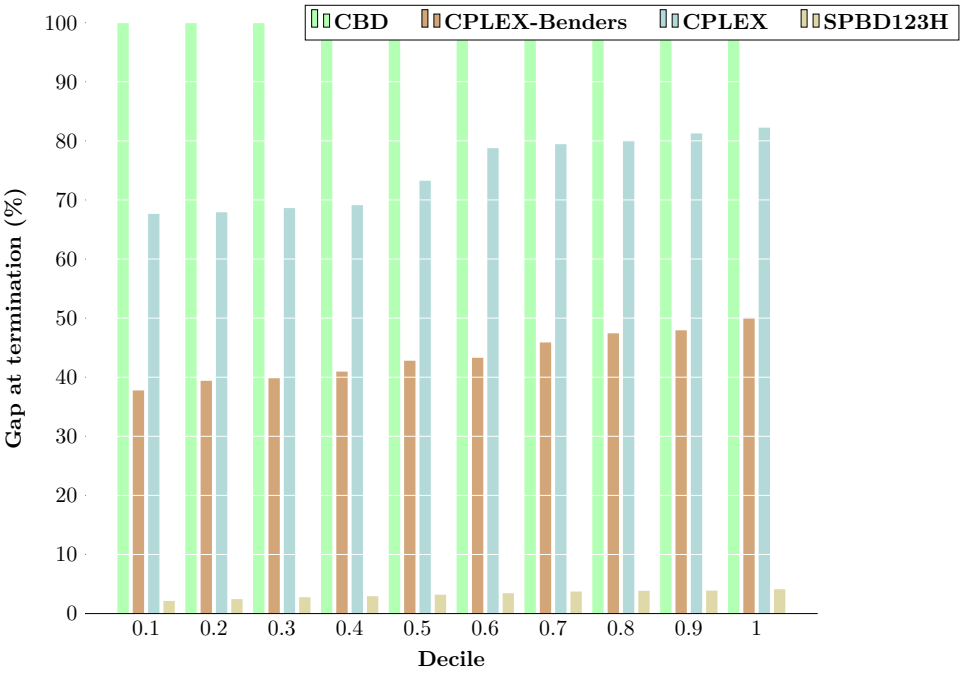

Figure 3.19: Gap at termination distribution for $|\mathcal{P}| = 400$

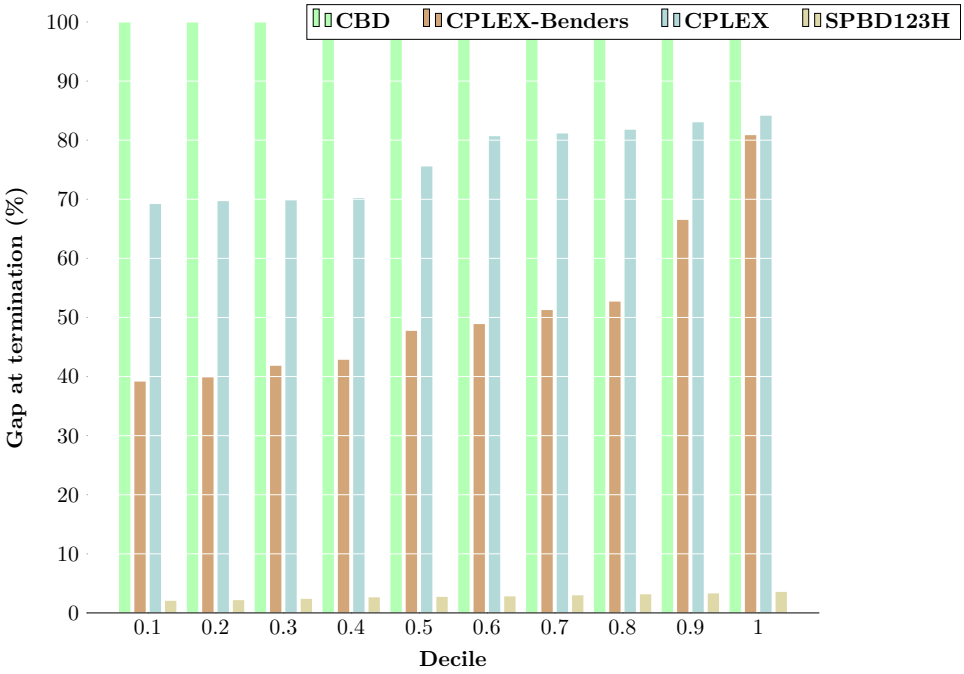

Figure 3.20: Gap at termination distribution for $|\mathcal{P}| = 500$

We observe that **SPBD123H** yields better gaps at termination, on average, than our three benchmark algorithms for every set of instances. The performance of **CPLEX** degrades as the number of products increases. On the other hand, **SPBD123H** remains effective for the larger instances. The average gap at termination reported by **SPBD123h** decreases as the number of products increases. We will analyze why this occurs later. Having established the effectiveness of **SPBD123h**, we next turn our attention to how its features impact its ability to produce a strong lower bound.

3.5.4 Improving the Lower Bound

We first study the impact of using the super-product-based master problem, **EMP**, on the lower bound produced at termination. To do so, we report in Table 3.2 the average lower bound reported by **SPBD** and each of our three benchmarks at termination. We see that **CBD** yields a very weak lower bound, while **CPLEX** and **CPLEX-Benders** produce stronger lower bounds. **SPBD** produces the strongest lower bound, one that is 26.39% greater, on average, in value than the bound produced by the best benchmark (**CPLEX-Benders**). Thus, we conclude that the Benders decomposition scheme based on the super-product master problem is superior to the benchmark methods with respect to the lower bound produced at termination.

Having established the effectiveness of using **EMP** in the context of Benders decomposition, we next assess the impact of the proposed valid inequalities. To

Table 3.2: Average lower bound reported at termination

CBD	CPLEX	CPLEX-Benders	SPBD
1,031	120,316	137,692	187,064

do so, we compare the performance of **SPBD**, **SPBD1**, **SPBD2**, **SPBD3** and **SPBD123** with respect to the average optimality gaps and lower bounds reported at termination as well as the average number of feasibility and optimality cuts generated during execution. We present these results, averaged over all instances, in Table 3.3.

Table 3.3: Gaps, lower bounds and number of Benders cuts found by **SPBD**, **SPBD1**, **SPBD2**, **SPBD3** and **SPBD123**

Method	Opt. gap	Lower bound	Feasibility cuts	Optimality cuts
SPBD	39.26%	187,064	67.19	0.0
SPBD1	37.64%	191,808	39.51	0.0
SPBD2	8.75%	281,211	24.45	0.01
SPBD3	37.39%	192,629	33.91	0.00
SPBD123	5.17%	291,930	14.59	0.16

The use of each valid inequality leads to a decrease in the optimality gap at termination as well as an increase in the lower bound compared to **SPBD**. When considered individually, the first and third valid inequalities have a smaller impact than the second valid inequality. However, amongst these methods, the best results are obtained with **SPBD123**, indicating that all three valid inequalities, together, are the most effective. We also see that the valid inequalities have a significant impact on the number of feasibility cuts generated. Recalling that the valid inequalities are designed to render infeasible vehicle allocations that will not yield a feasible subproblem, this suggests the inequalities are having the intended effect.

We also observe that regardless of the method, the number of optimality cuts generated is very low, which means only a few vehicle allocations result in feasible subproblems. This suggests that the LSNDP is more of a feasibility problem than an optimality problem. Thus, to improve the quality of the upper bound, we implemented a heuristic procedure that derives feasible solutions from infeasible subproblems. We next study the impact of the proposed heuristic.

3.5.5 Improving the Upper Bound

We next analyze the impact the proposed heuristic has on the ability of **SPBD123H** to produce high-quality primal solutions. To do so, we measure for an instance and a method the improvement in the primal solution over that of the

initial heuristic solution, (x_h, y_h), by computing the primal gap:

$$\text{primal-gap}_{UB}^{Method} = \frac{z(x_h, y_h) - \text{UB}_{Method}}{z(x_h, y_h)} \times 100$$

Here, UB_{Method} represents the objective function value of the best primal solution found by the method $Method$ during its execution. We benchmark **SPBD123H** against **CPLEX** and **SPBD123**, and present averages of these primal gaps over instances with the same number of products. In Table 3.4, "% *Method* impr." indicates the percentage of instances for which the considered method managed to improve the initial primal solution. "primal-gap$_{UB}^{Method}$" indicates the average primal gap obtained over instances for which the initial primal solution is improved. The best values are in bold.

Table 3.4: Comparison of upper bounds produced by **CPLEX**, **SPBD123** and **SPBD123H**

$\|\mathcal{P}\|$	% CPLEX impr.	primal-gap$_{UB}^{CPLEX}$	% SPBD123 impr.	primal-gap$_{UB}^{SPBD123}$	% SPBD123H impr.	primal-gap$_{UB}^{SPBD123H}$
100	66.67%	4.08%	16.67%	3.36%	**86.67%**	**6.86%**
200	33.33%	1.68%	0.00%	-	**76.67%**	**2.27%**
300	0.00%	-	0.00%	-	**10.00%**	**0.90%**
400	0.00%	-	0.00%	-	0.00%	-
500	0.00%	-	0.00%	-	**3.33%**	**0.90%**

As the number of products increases, all methods struggle to improve the initial primal solution. In addition, we observe that **CPLEX** has a better performance than **SPBD123** for every set of instances. Thus, implementing the **EMP** and the valid inequalities in the Benders strategy does not allow to obtain better primal solutions than those computed by **CPLEX**. Nevertheless, the situation reverses when we embed the slope-scaling heuristic into the Benders strategy. Indeed, for each set of instances, **SPBD123H** improves the initial primal solution more often than **CPLEX**, and with greater magnitude.

We now compare the primal solutions obtained by **SPBD123H** to those computed by **CPLEX**. For each instance, we compute an improvement rate:

$$\text{impr} = \frac{\text{UB}_{CPLEX} - \text{UB}_{SPBD123H}}{\text{UB}_{CPLEX}} \times 100$$

In Table 3.5, we report the average improvement rates over instances with the same number of products.

Table 3.5: Improvement rates

$\|\mathcal{P}\|$	% impr
100	3.26%
200	0.99%
300	0.09%
400	0.00%
500	0.03%

SPBD123H outperforms **CPLEX** for each set of instances. As both methods

have difficulties to improve the upper bound for large instances, the improvement rate tends to 0 as the number of products grows.

We return our attention to Table 3.1, and the observation that the optimality gap reported by **SPBD123H** at termination decreases as the number of products increases. At the same time, we see that **SPBD123H** has difficulties to improve the initial primal solution as the number of products increases. From these observations, we conclude that the optimality gap decreases as the number of products increases due to stronger lower bounds. We (partially) attribute this to the fact that the size of the super-product master problem, **EMP**, (in terms of number of variables and constraints) is independent of $|\mathcal{P}|$. Thus, solving the master problem does not become more computationally challenging. At the same time, the number of valid inequalities does increase as the number of products increases. As the valid inequalities strengthen the master problem, more of them likely leads to a stronger lower bound. In addition, the demand volumes that must be routed in the master problem increase as the number of products increases. As these increased volumes likely require an increase in vehicle allocations, $y_{ij}^{tt'}$, we hypothesize that they also strengthen the master problem.

3.6 Conclusions

In this chapter, we proposed a Benders decomposition-based solution approach for solving the LSNDP. More specifically, we proposed an algorithm based on the recently-proposed *partial Benders decomposition*, wherein information is retained in the master problem, in order to strengthen the bound it provides and speed up the convergence of the algorithm as a whole. Here, the information retained in the master problem is characterized by a "super-product" that is an aggregation of all the products to be routed. We proved the validity of this new master problem, and computationally demonstrate the effectiveness of solving it in the context of a Benders decomposition-type algorithm compared with a straightforward implementation. We proposed additional speed-up techniques, including valid inequalities and a heuristic for finding high-quality solutions. The resulting is a scalable algorithm that produces solutions that are of provably high-quality, despite the increasing number of products.

Chapter 4

A Dynamic Partial Benders Strategy

Contents

4.1 Introduction

In the previous chapter, each supplier product line is generated randomly. That way, any supplier may provide any product. In the case-study we adress, products are partitioned into families, such as beverages or frozen products, and each supplier is specialized in a subset of product families. Due to the distinction between product families, a possible improvement of the Benders strategy presented in Chapter 3 is

to consider multiple super-products in the master problem. In a partition of the product families, there would be a one to one correspondance between the super-products and the families.

By considering several super-products instead of just one, we assume that the information derived from the subproblem is more accurate. As a result, the master problem with multiple super-products should yield to stronger bounds than the master problem defined with a single super-product. However, strengthening the master is at the cost of an increased complexity. Indeed, having multiple super-products induces additional variables, which may slow down the master solution. It is difficult to assess a priori whether strengthening the master problem makes possible to overcome this increased complexity, and to accelerate the convergence of the Benders strategy.

In this Chapter, we propose a dynamic Benders algorithm based on a partial Benders decomposition that evolves in the course of the optimization process, in a logic of exploration and exploitation. The algorithm starts with the least complex partial Benders decomposition, i.e. with only one super-product in the master. At each iteration, our Benders strategy solves the current decomposition for a certain amount of time. Then the number of super-products is incremented, which strengthens the master.

In section 4.2, we formalize the partial Benders strategy with multiple super-products. We introduce a new master problem, K-EMP, wherein K super-products are obtained by partitioning the products into K subsets. We demonstrate that the acceleration techniques described in sections 3.3 and 3.4 can be extended to the new master problem.

For a fixed value of K, there are many ways to partition the products. However, different partitions lead to different approximations of the original problem. As a result, the product partition affects the quality of each partial Benders decomposition, and so the convergence of the dynamic Benders algorithm. From the characteristics of the LSNDP, we derive rules for partitioning the products effectively. In section 4.3, we describe special cases in which products can be partitioned and aggregated into super-products without loss of information, so the resulting master problem is equivalent to the original problem. From these specific cases, we introduce an indicator that measures whether or not a pair of products should be part of the same subset. In section 4.4, based on this indicator, we present two strategies for partitioning products into K subsets. In section 4.5, we present the dynamic partial Benders strategy. In section 4.6, we assess the performance of our algorithm with an extensive computational study. Finally, we conclude in section 4.7.

4.2 Multiple Super-Products

In this section, we extend the Benders strategy described in Chapter 3. We first introduce a new partial Benders decomposition for the LSNDP, based on a master problem with multiple super-products. Then, we describe how the acceleration

techniques described in sections 3.3 and 3.4 can be extended to the new master problem.

4.2.1 Extending the Enhanced Master Problem

In Chapter 3, we presented a partial Benders decomposition wherein the enhanced master problem **EMP** is strengthened with a single "super-product" χ, derived from the aggregation of all products $p \in \mathcal{P}$. Another possible strategy is to strengthen the master problem with several super-products. Let $\mathcal{P} = \{\mathcal{P}_1, ..., \mathcal{P}_\mathcal{K}\}$ be a $\mathcal{K}$-partition of the products, such that $\forall (i,j) \in \{1, ..., \mathcal{K}\}^2, i \neq j, \mathcal{P}_i \cap \mathcal{P}_j = \emptyset$, and $\bigcup_{k \in \{1,...,\mathcal{K}\}} \mathcal{P}_k = \mathcal{P}$. Let Ξ be the set of super-products. The super-product $\chi_k \in \Xi$ is obtained by aggregating all products in $\mathcal{P}_k$. Therefore for all customers $(c,t) \in \mathcal{C}_\mathcal{T}$, the demand of super-product χ_k is the sum of the demands for products in $\mathcal{P}_k$, i.e. $D_{ct}^{\chi_k} = \sum_{p \in \mathcal{P}_k} d_{ct}^p$. A super-product flow variable $x_{ij}^{\chi_k tt'}$ is defined for each arc $((i,t),(j,t'))$ and for each product $p \in \mathcal{P}_k$ such that a flow variable $x_{ij}^{ptt'}$ is defined in the LSNDP. Thus, for a supplier i that does not supply any product of $\mathcal{P}_k$ (i.e. $\mathcal{P}^i \cap \mathcal{P}_k = \emptyset$), and for all arcs originating from i, the continuous variable $x_{ij}^{\chi_k tt'}$ is not defined. We denote the new master problem by **K-EMP**.

Figures 4.1 and 4.2 illustrate an example. The original problem is depicted in Figure 4.1. A single customer requests one unit of p_1, p_2, p_3 and p_4. Supplier s_1 offers products p_1 and p_2, s_2 offers p_2 and p_3, and s_3 offers p_3 and p_4. Given a 2-partition of the products: $\mathcal{P}_1 = \{p_1, p_2\}, \mathcal{P}_2 = \{p_3, p_4\}$, we aggregate the products into two super-products χ_1 and χ_2. The aggregated problem is depicted in Figure 4.2. As c requests one unit of each p_1 and p_2 in the original problem, c request two units of super-product χ_1 in the aggregated problem. As s_1 and s_2 offer at least one product of $\mathcal{P}_1$ in the original problem, they offer χ_1 in the aggregated problem as well. On the contrary, as s_3 does not offer any product of $\mathcal{P}_1$ in the original problem, it does not offer χ_1 in the aggregated problem. A similar reasoning applies to super-product χ_2 and products p_3 and p_4.

Note that the aggregation of products is still an approximation that may induce a loss of information. Indeed, in the **K-EMP** suppliers may ship products they do not provide. Coming back to our example, as in the aggregated problem s_2 provides all super-products, it can satisfy all super-products demands. However, in the original problem s_2 does not provide p_1 nor p_4, and cannot satisfy the demands of p_1 and p_4 of customer c.

The **K-EMP** allocates vehicle capacity on transportation arcs in order to satisfy the routing of the K super-products. It is formulated as follows:

$$\min \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} + z \tag{4.1}$$

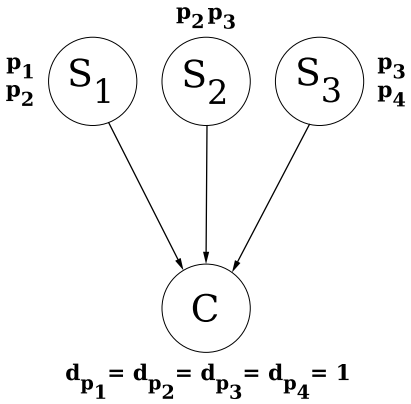

Figure 4.1: Before the aggregation of products

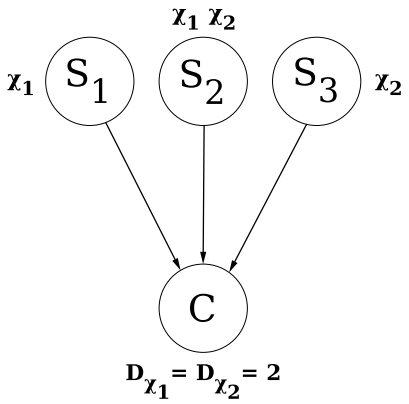

Figure 4.2: After aggregating p_1 with p_2, and p_3 with p_4

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{ij}^{\chi_k tt'} - \sum_{((j,t'),(l,t''))\mathcal{A}_\mathcal{T}\cup\mathcal{H}_\mathcal{T}} x_{jl}^{\chi_k t't''} = 0, \quad \forall (j,t') \in \mathcal{W}_\mathcal{T}, \forall \chi_k \in \Xi \tag{4.2}$$

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{ij}^{\chi_k tt'} \geq D_{jt'}^{\chi_k}, \quad \forall (j,t') \in \mathcal{C}_\mathcal{T}, \forall \chi_k \in \Xi \tag{4.3}$$

$$\sum_{\chi_k\in\Xi} x_{ij}^{\chi_k tt'} \leq \hat{c} y_{ij}^{tt'}, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{4.4}$$

$$z \geq \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} \sum_{\chi_k\in\Xi} c_{ij} x_{ij}^{\chi_k tt'} + \sum_{((i,t),(i,t'))\in\mathcal{H}_\mathcal{T}} \sum_{\chi_k\in\Xi} c_{ii} x_{ii}^{\chi_k tt'} \tag{4.5}$$

$$0 \geq \sum_{(c,t)\in\mathcal{C}_\mathcal{T}} \sum_{p\in\mathcal{P}} d_{ct}^p \rho_{ct}^p, \quad \forall \rho \in \Omega \tag{4.6}$$

$$z \geq \sum_{(c,t)\in\mathcal{C}_\mathcal{T}} \sum_{p\in\mathcal{P}} d_{ct}^p \pi_{ct}^p, \quad \forall \pi \in \Gamma \tag{4.7}$$

$$x_{ij}^{\chi_k tt'} \in \mathbb{R}^+, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \cup \mathcal{H}_\mathcal{T}, \ \forall \chi_k \in \Xi, \mathcal{P}^i \cap \mathcal{P}_k \neq \emptyset \tag{4.8}$$

$$y_{ij}^{tt'} \in \mathbb{N}^+, \quad \forall ((i,t),(j,t')) \in \mathcal{A}_\mathcal{T} \tag{4.9}$$

$$z \in \mathbb{R}^+ \tag{4.10}$$

The objective function is the same as for the **EMP**. Constraints (4.2) enforce the flow conservation of each super-product on each warehouse. Constraints (4.3) ensure that for each super-product, each customer demand is fulfilled. Constraints

(4.4) ensure that enough vehicle capacity is allocated to support the flows of super-products. Constraint (4.5) bounds the flows cost approximation z. Constraints (4.6) and (4.7) are the Benders cuts.

To ensure that a Benders scheme based on this master problem converges to an optimal solution of the LSNDP, we prove that **K-EMP** is a relaxation of the original problem.

Theorem 2 *The $\mathcal{K}$-enhanced master problem, **K-EMP**, is a relaxation of the Logistics Service Network Design problem, LSNDP.*

Proof 2 *We prove this claim by showing that any feasible solution for the LSNDP is also feasible for the **K-EMP** and has the same objective function value. To do so, we let (x, y) be a feasible solution of the LSNDP. Let consider a solution (x^{Ξ}, y, z) such that:*

$$x_{ij}^{\chi_k tt'} = \sum_{p \in \mathcal{P}_k} x_{ij}^{ptt'}, \ \forall((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}} \cup \mathcal{H}_{\mathcal{T}}, \forall k \in \{1, ..., \mathcal{K}\},$$

$$z = \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}}} \sum_{\chi_k \in \Xi} c_{ij} x_{ij}^{\chi_k tt'} + \sum_{((i,t),(i,t')) \in \mathcal{H}_{\mathcal{T}}} \sum_{\chi_k \in \Xi} c_{ii} x_{ii}^{\chi_k tt'}$$

*It is easy to prove that this solution is feasible for the K-enhanced master problem. By construction, for each variable $x_{ij}^{ptt'}$ in the LSNDP, $p \in \mathcal{P}_k$, there is a corresponding variable $x_{ij}^{\chi_k tt'}$ in the **K-EMP**. We know that for each warehouse (j,t') and each product $p \in \mathcal{P}$: $\sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}} \cup \mathcal{H}_{\mathcal{T}}} x_{ij}^{ptt'} - \sum_{((j,t'),(l,t'')) \in \mathcal{A}_{\mathcal{T}} \cup \mathcal{H}_{\mathcal{T}}} x_{jl}^{pt't''} = 0$. If we sum this expression on the products of $\mathcal{P}_k$, we obtain:*

$$\sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}} \cup \mathcal{H}_{\mathcal{T}}} \sum_{p \in \mathcal{P}_k} x_{ij}^{ptt'} - \sum_{((j,t'),(l,t'')) \in \mathcal{A}_{\mathcal{T}} \cup \mathcal{H}_{\mathcal{T}}} \sum_{p \in \mathcal{P}_k} x_{jl}^{pt't''} = 0$$

$$\Longrightarrow \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}} \cup \mathcal{H}_{\mathcal{T}}} x_{ij}^{\chi_k tt'} - \sum_{((j,t'),(l,t'')) \in \mathcal{A}_{\mathcal{T}} \cup \mathcal{H}_{\mathcal{T}}} x_{jl}^{\chi_k t't''} = 0$$

Therefore (x^{Ξ}, y, z) respects Constraints (4.2). As (x, y) respects Constraints (2.3)-(2.4), it is straightforward to demonstrate (x^{Ξ}, y, z) also respects Constraints (4.3)-(4.4). By construction of z, (x^{Ξ}, y, z) respects constraint (4.5) which makes it an admissible solution for the enhanced master problem. Let $Q(x, y)$ be (x, y) objective value:

$$Q(x,y) = \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}}} f_{ij} y_{ij}^{tt'} + \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}}} \sum_{p \in \mathcal{P}} c_{ij} x_{ij}^{ptt'} + \sum_{((i,t),(i,t')) \in \mathcal{H}_{\mathcal{T}}} \sum_{p \in \mathcal{P}} c_{ii} x_{ii}^{ptt'}$$

$$= \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}}} f_{ij} y_{ij}^{tt'} + \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}}} \sum_{k \in \{1,...,\mathcal{K}\}} \sum_{p \in \mathcal{P}_k} c_{ij} x_{ij}^{ptt'} + \sum_{((i,t),(i,t')) \in \mathcal{H}_{\mathcal{T}}} \sum_{k \in \{1,...,\mathcal{K}\}} \sum_{p \in \mathcal{P}_k} c_{ii} x_{ii}^{ptt'}$$

$$= \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}}} f_{ij} y_{ij}^{tt'} + \sum_{((i,t),(j,t')) \in \mathcal{A}_{\mathcal{T}}} \sum_{\chi_k \in \Xi} c_{ij} x_{ij}^{\chi_k tt'} + \sum_{((i,t),(i,t')) \in \mathcal{H}_{\mathcal{T}}} \sum_{\chi_k \in \Xi} c_{ii} x_{ii}^{\chi_k tt'}$$

$$= \sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} f_{ij}y_{ij}^{tt'} + z \;=\; Q(x^\chi, y, z)$$

Solution (x^{Ξ_k}, y, z) *that replicates solution* (x, y) *by an aggregation of flows, is feasible to the K-enhanced master problem. The two solutions have an identical objective function value. Thus,* ***K-EMP*** *is a relaxation of LSNDP.*

Remark 1 *The enhanced master problem,* ***EMP****, is a relaxation of the* $\mathcal{K}$*-enhanced master problem,* ***K-EMP****.*

A similar reasoning can be applied to demonstrate that the master problem with a single super-product, **EMP**, is a relaxation of the master problem with K super-products, **K-EMP**.

4.2.2 Extending the Acceleration Techniques

To apply our Benders strategy to the **K-EMP**, we must ensure that the acceleration techniques described in Chapter 3 work if we consider multiple super-products in the master. It is trivial to prove that the generation of heuristic solutions performs whether a single or multiple super-products are considered. However, some adjustments are necessary in order to adapt the valid inequalities to the **K-EMP**.

To extend the *Super-Sources Inequalities*, we modify the time-expanded graph as we did in subsection 3.3.1. For each product $p \in \mathcal{P}$ we add a super-source ss_p, and for each supplier $s \in \mathcal{S}$ that provides product p, we add an arc from ss_p to s with null transit time, linear cost and fixed cost. Similarly to subsection 3.3.1, for each product $p \in \mathcal{P}$ we compute the total demand $D_p = \sum_{(c,t)\in\mathcal{C}_\mathcal{T}} d_{ct}^p$. As in the **K-EMP** each product p is represented by a single super-product χ_k, we enforce the super-source ss_p to ship at least D_p units of χ_k. We also force the supplier nodes to satisfy the flow conversation with respect to each associated super-product in the **K-EMP**.

Then, we add the following constraints to the **K-EMP**:

$$\sum_{((ss_p),(j,t))\in\mathcal{A}_\mathcal{T}} x_{ss_pj}^{\chi_k t} \geq D^p, \quad \forall k \in \{1,...,\mathcal{K}\},\ \forall p \in \mathcal{P}_k \tag{4.11}$$

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{ij}^{\chi_k tt'} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}} x_{jl}^{\chi_k t't''} = 0, \quad \forall (j,t') \in \mathcal{S}_\mathcal{T},\ \forall k \in \{1,...,\mathcal{K}\}, \mathcal{P}^j \cap \mathcal{P}_k \neq \emptyset \tag{4.12}$$

The *Direct Supply Inequalities* developped in subsection 3.3.2 can also be adapted for the **K-EMP**. For each direct arc from a supplier (s,t) to a customer (c,t'), and each subset $\mathcal{P}_k$ of the K-partition, we compute the demand of customer (c,t') for products $p \in \mathcal{P}_k$ that can be served by (s,t): $d_{ct'}^{kst} = \sum_{p\in\mathcal{P}^s\cap\mathcal{P}_k} d_{ct'}^p$. We then restrict the flow of super-product χ_k on the direct arc $((s,t),(c,t'))$ to be no greater than $d_{ct'}^{kst}$.

We add the following valid inequalities to the **K-EMP**:

$$x_{sc}^{\chi_k tt'} \leq d_{ct'}^{kst}, \quad \forall((s,t),(c,t')) \in \mathcal{A}_{\mathcal{T}},\ (s,t) \in \mathcal{S}_{\mathcal{T}},\ (c,t') \in \mathcal{C}_{\mathcal{T}},\ \forall \chi_k \in \Xi \tag{4.13}$$

To adapt the *Time-Based Super-Source Shipment Inequalities*, we compute the cumulative global demand $\bar{D_t^p}$ and the shortest possible delivery times $t_{ss_p}^{min}$ as we did in subsection 3.3.3. For each subset $\mathcal{P}_k$ of the K-partition, each product $p \in \mathcal{P}_k$, and each time $t \in \mathcal{T}$ such that $\bar{D_t^p} > \bar{D_{t-1}^p}$, we enforce the super-source $\mathcal{SS}_p$ to ship more than $\bar{D_t^p}$ units of super-product χ_k, before time $t - t_{ss_p}^{min}$. To do so, we add the following valid inequalities to the **K-EMP**:

$$\sum_{\substack{((ss_p),(j,t)) \in \mathcal{A}_{\mathcal{T}} \\ t \leq t^* - t_{ss_p}^{min}}} x_{ss_p j}^{\chi_k t} \geq \bar{D_{t^*}^p}, \quad \forall k \in \{1,...,\mathcal{K}\},\ \forall p \in \mathcal{P}_k,\ \forall t^* \in \mathcal{T},\ \bar{D_{t^*}^p} > \bar{D_{t^*-1}^p} \tag{4.14}$$

The partial Benders decomposition presented in Chapter 3 can be extended to multiple super-products. Similarly, the corresponding acceleration techniques can be extended to multiple super-products. Thus, the Benders strategy presented in Chapter 3 can be extended to multiple super-products.

4.3 Products Matching

The super-products model an approximation of the original problem that strengthens the master problem. They are obtained by partitioning the products into K subsets. Yet, for a fixed value of K there are several ways to partition the products, and the definition of the partition impacts the quality of the Benders decomposition. In this section, we analyze the factors to consider to aggregate products effectively. We first describe two specific cases for which product aggregation induces no loss of information. From these particular cases, we establish an indicator that evaluates if it is worth aggregating a pair of products.

4.3.1 Equivalence between EMP and LSNDP

The aggregation of products induces a loss of information about which products each supplier can supply. In the original problem, each supplier can only provide a subset of the products. Thus, a single supplier cannot answer all customers requests. However, in the **EMP**, as we cannot restrict suppliers to only ship products they provide, each supplier provides the super-product. Therefore a single supplier can, by itself, answer all customer requests for the super-product. As requests of super-product χ include demands for all products $p \in \mathcal{P}$, the **EMP** assumes that any supplier can satisfy requests for all products $p \in \mathcal{P}$. Therefore, a feasible solution for the **EMP** may not be convertible to a feasible solution for the LSNDP.

There is a single case for which each **EMP** feasible solution can be converted to a LSNDP feasible solution. Indeed, if all suppliers provide all products, product aggregation induces no loss of information and the **EMP** is equivalent to the LSNDP. Figures 4.3 and 4.4 illustrate an example of equivalence between the **EMP** and the LSNDP. The original problem is depicted in Figure 4.3. All products (p_1 and p_2) are offered by all suppliers (s_1, s_2 and s_3). Aggregating p_1 and p_2 into a single super-product χ induces no loss of information. Indeed, if we aggregate p_1 and p_2 (see Figure 4.3), the demand of super-product χ can be satisfied by s_1, s_2, or s_3. As the demand of super-product χ sums the demands for p_1 and p_2, the aggregated problem assumes that s_1, s_2 or s_3 can satisfy the demands for p_1 and p_2. In the original problem, s_1, s_2 and s_3 offer both p_1 and p_2. Thus, we can aggregate p_1 and p_2 without loss of information. Therefore, the aggregated problem **EMP** illustrated in in Figure 4.4, is equivalent to the LSNDP.

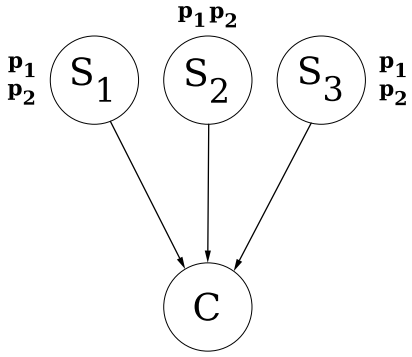

Figure 4.3: Before the aggregation of products

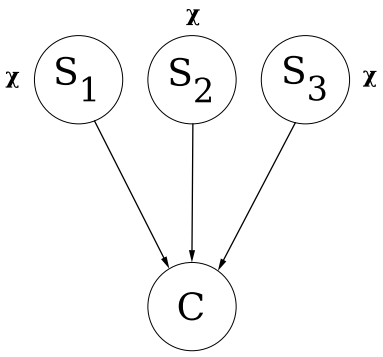

Figure 4.4: After the aggregation of products

Theorem 3 *If all suppliers offer all products (i.e. $\forall s \in \mathcal{S}, \mathcal{P}^s = \mathcal{P}$), then the enhanced master problem **EMP**, is equivalent to the LSNDP.*

Proof 3 *We proved in section 3.2 that the enhanced master problem **EMP** is a relaxation of the LSNDP. We demonstrate now that the LSNDP is a relaxation of the **EMP**, if all suppliers offer all products. In that specific case, a feasible solution for the EMP is necessarily feasible for the original problem. The premise of the proof is as follows. Any solution of the **EMP** can be seen as a set of paths transporting D^{χ}_{ct} units of super-product from suppliers to each customer. Thus, for each customer, the incoming flow of super-product can be divided into $|\mathcal{P}|$ parts of quantity d^{p}_{ct} transported along paths that originates from suppliers (note that d^{p}_{ct} can be null). By converting each part into a flow of product p, one can construct a solution for the LSNDP. Since all suppliers offer all products, the obtained solution is feasible for the LSNDP.*

*Let (x^{χ}, y, z) be a feasible solution of the **EMP**. Let Λ be the set of paths from suppliers to customers in $\mathcal{G}_{\mathcal{T}}$. By definition the flow of super-product x^{χ} originates from suppliers, ends at customers, and respects flow conservation at each warehouse.*

Therefore we can decompose the flow of super-product x^χ into paths of Λ, and obtain an equivalent path-solution γ^χ. We denote γ^χ_λ the quantity of super-product transported along a path $\lambda \in \Lambda$.

*Let Λ_{ct} be the set of paths from suppliers to customer (c,t). As (γ^χ, y, z) respects the **EMP** demand constraints, the sum of flows along paths in Λ_{ct} sustains (c,t) demand of super-product, i.e. $\sum_{\lambda \in \Lambda_{ct}} \gamma^\chi_\lambda = D^\chi_{ct} = \sum_{p \in \mathcal{P}} d^p_{ct}$. Thus, for each customer (c,t), we can partition its incoming path-flow γ^χ into $|\mathcal{P}|$ parts $\{\dot\gamma^1, ..., \dot\gamma^{|\mathcal{P}|}\}$ transiting d^p_{ct} units of super-product each $(\forall p \in \mathcal{P})$. As a result, for each product $p \in \mathcal{P}$ the total flow along paths in the p^{th} subset of the partition equals d^p_{ct}, i.e. $\sum_{\lambda \in \Lambda_{ct}} \dot\gamma^p_\lambda = d^p_{ct}$. In addition, for each path $\lambda \in \Lambda_{ct}$, the sum of flows over the partition equals the total flow of super-product, i.e. $\sum_{p \in \mathcal{P}} \dot\gamma^p_\lambda = \gamma^\chi_\lambda$.*

Based on the partition of γ^χ, we construct a path-solution γ for the LSNDP. We denote γ^p_λ the quantity of product p transiting along path λ. As all suppliers offer all products, product p can transit on any path of Λ_{ct}. Thus, we can set any value for γ^p_λ. For each customer (c,t), each path $\lambda \in \Lambda_{ct}$, and each product $p \in \mathcal{P}$, we set γ^p_λ value to $\dot\gamma^p_\lambda$.

*Let x be the flow-solution equivalent to path-solution γ. We now demonstrate that (x,y) is feasible for the LSNDP. By definition, on a path, flow conservation is ensured. Thus, flow-solution x respects constraints (2.2). For each customer $(c,t) \in \mathcal{C}_\mathcal{T}$, and each product $p \in \mathcal{P}$ we have $\sum_{\lambda \in \Lambda_{ct}} \gamma^p_\lambda = d^p_{ct}$. By extension, the associated flow-solution x respects demand constraints (2.3). As (x^χ, y, z) is a solution of the **EMP**, vehicle allocation y is sufficient to route γ^χ. For all customers (c,t), and all paths $\lambda \in \Lambda_{ct}$, $\sum_{p \in \mathcal{P}} \gamma^p_\lambda = \gamma^\chi_\lambda$ therefore vehicle allocation y is sufficient to route γ, and so x. Thus, (x,y) respects constraints (2.4), and is a feasible solution to the LSNDP.*

Let (x^χ, y, z) objective value be:

$$Q(x^\chi, y, z) = \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y^{tt'}_{ij} + z$$

Due to constraint (3.13) we have:

$$Q(x^\chi, y, z) = \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y^{tt'}_{ij} + \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} c_{ij} x^{\chi tt'}_{ij} + \sum_{((i,t),(i,t')) \in \mathcal{H}_\mathcal{T}} c_{ii} x^{\chi tt'}_{ii}$$

Yet, as γ is obtained by partitioning γ^χ, for each path $\lambda \in \Lambda$ the sum of flows over the p components of γ equals the flow of super-product γ^χ. Regarding flow-solutions x and x^χ, that means on each arc, the sum of flows over the products equals the flow of super-product, i.e. $\sum_{p \in \mathcal{P}} x^{ptt'}_{ij} = x^{\chi tt'}_{ij}$. Therefore:

$$Q(x^\chi, y, z) = \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y^{tt'}_{ij} + \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} \sum_{p \in \mathcal{P}} c_{ij} x^{ptt'}_{ij} + \sum_{((i,t),(i,t')) \in \mathcal{H}_\mathcal{T}} \sum_{p \in \mathcal{P}} c_{ii} x^{ptt'}_{ii}$$

$$= Q(x, y)$$

*Solution (x, y), that replicates solution (x^{χ}, y, z) by partitioning of flows, is feasible to the LSNDP. The two solutions have an identical objective function value. Thus, the LSNDP is a relaxation of the **EMP**. As the **EMP** is also a relaxation of the LSNDP, we demonstrated that the **EMP** and the LSNDP are equivalent if all suppliers offer all products.*

4.3.2 Equivalence between K-EMP and LSNDP

The same approach can be applied to the **K-EMP**. Given a $\mathcal{K}$-partitions of the products $\{\mathcal{P}_1, ..., \mathcal{P}_{\mathcal{K}}\}$, a super-product χ_k is obtained by aggregating the products of subset $\mathcal{P}_k$. In the **K-EMP**, a supplier that provides the super-product χ_k can answer all customer requests of super-product χ_k. As requests of super-product χ_k include demands for all products $p \in \mathcal{P}_k$, the **K-EMP** assumes that suppliers of super-product χ_k can answer requests for all products $p \in \mathcal{P}_k$. However, a supplier that offers super-product χ_k in the aggregated problem, may not offer all products $p \in \mathcal{P}_k$ in the original problem. Therefore, a feasible solution for the **K-EMP** may not be convertible to a feasible solution for the LSNDP.

There is a single case for which each **K-EMP** feasible solution can be converted to a LSNDP feasible solution. If any supplier of a product $p \in \mathcal{P}_k$ also provides all products of $\mathcal{P}_k$, product aggregation induces no loss of information and the **K-EMP** is equivalent to the LSNDP. Figures 4.5 and 4.6 illustrate an example of equivalence between the **K-EMP** and the LSNDP. The original problem is depicted in the left figure. Both products p_1 and p_2 are offered by s_1 and s_2. Aggregating p_1 and p_2 into super-product χ_1 induces no loss of information. Indeed, if we aggregate p_1 with p_2 (see Figure 4.6), the demand of super-product χ_1 can be satisfied by s_1 or s_2. As the demand of super-product χ_1 sums the demands for p_1 and p_2, the aggregated problem assumes that s_1 or s_2 can satisfy the demands for p_1 and p_2. Yet, in the original problem, s_1 and s_2 provide both p_1 and p_2. Thus, we can aggregate p_1 and p_2 without loss of information. Similarly, the aggregation of p_3 and p_4 into super-product χ_2 induces no loss of information. Therefore, the 2-partition $\mathcal{P} = \{\{p_1, p_2\}, \{p_3, p_4\}\}$ yields a **K-EMP** that is equivalent to the LSNDP.

Let us consider again Figures 4.1 and 4.2 where the **K-EMP** is not equivalent to the LSNDP. In this case, we aggregate products p_1 and p_2 into super-product χ_1. In the **K-EMP**, c's demand of super-product χ_1 can be fulfilled by supplier s_2. As the demand of super-product χ_1 sums the demands of p_1 and p_2, the aggregated problem assumes that s_2 can satisfy the demands of p_1 and p_2. Thus, we lose information by aggregating p_1 and p_2 as supplier s_2 offers one but not all products that form χ_1.

Theorem 4 *Let $\mathcal{P} = \{\mathcal{P}_1, ..., \mathcal{P}_{\mathcal{K}}\}$ be a $\mathcal{K}$-partition of the products, and $\chi_k \in \Xi, k \in \{1, ..., \mathcal{K}\}$, the associated super-products. If for each $k \in \{1, ..., \mathcal{K}\}$, and each product $p \in \mathcal{P}_k$, a supplier $s \in \mathcal{S}$ that offers p offers all products of $\mathcal{P}_k$, then the K-EMP is equivalent to the LSNDP.*

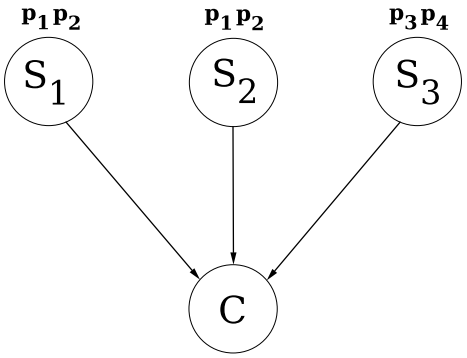

Figure 4.5: Before the aggregation of products

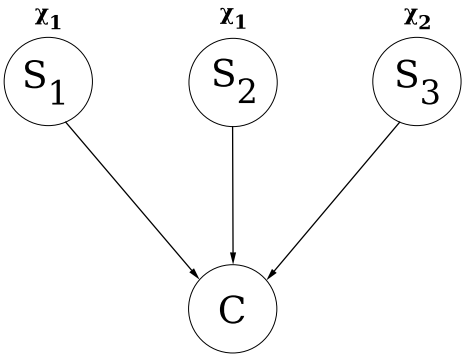

Figure 4.6: After aggregating p_1 with p_2, and p_3 with p_4

Proof 4 *We proved in section 4.2 that the* ***K-EMP*** *is a relaxation of the LSNDP. We demonstrate now that the LSNDP is a relaxation of the* ***K-EMP****, if for each* $k \in \{1, ..., \mathcal{K}\}$*, and each product* $p \in \mathcal{P}_k$*, a supplier* $s \in \mathcal{S}$ *that offers* p *offers all products of* $\mathcal{P}_k$*. The demonstration presented here is similar to the demonstration used for Theorem 3. Indeed, super-products can be considered independently, which leads to the same result.*

Let (x^{Ξ}, y, z) *be a feasible solution of the* ***K-EMP****. Let* Λ *be the set of paths from suppliers to customers in* $\mathcal{G}_{\mathcal{T}}$*. By definition the flow of super-products* x^{Ξ} *originates from suppliers, ends at customers, and respects flow conservation at each warehouse. Therefore we can decompose the flow of super-product* x^{Ξ} *into paths of* Λ*, and obtain an equivalent path-solution* γ^{Ξ}*. We denote* $\gamma_{\lambda}^{\chi_k}$ *the quantity of* k^{th} *super-product transported along a path* $\lambda \in \Lambda$*.*

Let Λ_{ct} *be the set of paths from suppliers to customer* (c, t)*. As* (γ^{Ξ}, y, z) *respects the* ***K-EMP*** *demand constraint, for each super-product* χ_k *the sum of flows along paths of* Λ_{ct} *sustains* (c, t) *demand of* χ_k*, i.e.* $\sum_{\lambda \in \Lambda_{ct}} \gamma_{\lambda}^{\chi_k} = D_{ct}^{\chi_k} = \sum_{p \in \mathcal{P}_k} d_{ct}^{p}$*. Thus, for each a customer* (c, t)*, we can partition its incoming path-flow of super-product* χ_k *into* $|\mathcal{P}_k|$ *parts* $\{\dot{\gamma}^1, ..., \dot{\gamma}^{|\mathcal{P}|}\}$ *transiting* d_{ct}^{p} *units of* χ_k *each* $(\forall p \in \mathcal{P}_k)$*. As a result, for each product* $p \in \mathcal{P}_k$ *the total flow along paths in the* p^{th} *subset of the partition equals* d_{ct}^{p}*, i.e.* $\sum_{\lambda \in \Lambda_{ct}} \dot{\gamma}_{\lambda}^{p} = d_{ct}^{p}$*. In addition, for each path* $\lambda \in \Lambda_{ct}$*, the sum of flows over the partition equals the total flow of super-product* χ_k*, i.e.* $\sum_{p \in \mathcal{P}_k} \dot{\gamma}_{\lambda}^{p} = \gamma_{\lambda}^{\chi_k}$*.*

Based on the partition of γ^{Ξ}*, we construct a path-solution* γ *for the LSNDP. We denote* γ_{λ}^{p} *the quantity of product* p *transiting along path* λ*. For each customer* (c, t)*, each* $k \in \{1, ..., \mathcal{K}\}$*, and each product* $p \in \mathcal{P}_k$*, if* $\dot{\gamma}_{\lambda}^{p} > 0$ *then the supplier of origin manufactures* χ_k *in the* ***K-EMP****. By hypothesis, the supplier of origin manufactures all products of* $\mathcal{P}_k$ *in the original problem. As a result,* $\forall p \in \mathcal{P}_k$ *we can set any value for* γ_{λ}^{p}*. Thus, for each customer* (c, t)*, each* $k \in \{1, ..., \mathcal{K}\}$*, each*

path $\lambda \in \Lambda_{ct}$*, and each product* $p \in \mathcal{P}_k$*, we set* γ_λ^p *value to* $\dot{\gamma}_\lambda^p$*.*

Let x *be the flow-solution equivalent to path-solution* γ*. We now demonstrate that* (x, y) *is feasible for the LSNDP. By definition, on a path, flow conservation is ensured. Thus, flow-solution* x *respects constraints* (2.2)*. For each customer* $(c,t) \in \mathcal{C}_\mathcal{T}$*, each* $k \in \{1, ..., \mathcal{K}\}$*, and each product* $p \in \mathcal{P}_k$ *we have* $\sum_{\lambda \in \Lambda_{ct}} \gamma_\lambda^p = d_{ct}^p$*. By extension, the associated flow-solution* x *respects demand constraints* (2.3)*. As* (x^Ξ, y, z) *is a solution of the* ***K-EMP****, vehicle allocation* y *is sufficient to route* γ^Ξ*. For each customer* (c,t)*, each* $k \in \{1, ..., \mathcal{K}\}$*, and each path* $\lambda \in \Lambda_{ct}$*,* $\sum_{p \in \mathcal{P}_k} \gamma_\lambda^p = \gamma_\lambda^{\chi_k}$ *therefore vehicle allocation* y *is sufficient to route* γ*, and so* x*. Thus,* (x, y) *respects constraints* (2.4)*, and is a feasible solution to the LSNDP.*

Let (x^Ξ, y, z) *objective value be:*

$$Q(x^\Xi, y, z) \; = \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} + z$$

Due to constraint (4.5) *we have:*

$$Q(x^\Xi, y, z) \; = \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} + \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} \sum_{\chi_k \in \Xi} c_{ij} x_{ij}^{\chi_k tt'} + \sum_{((i,t),(i,t')) \in \mathcal{H}_\mathcal{T}} \sum_{\chi \in \Xi} c_{ii} x_{ii}^{\chi_k tt'}$$

By partitioning, for each $k \in \{1, ..., \mathcal{K}\}$*, for each product* $p \in \mathcal{P}_k$ *and for each path* $\lambda \in \Lambda_{ct}$*:* $\sum_{p \in \mathcal{P}_k} \gamma_\lambda^p = \gamma_\lambda^{\chi_k}$*. Regarding flow-solutions* x *and* x^Ξ *associated to* γ *and* γ^Ξ*, that means on each arc, the sum of flows over the products of* $\mathcal{P}_k$ *equals the flow of super-product* χ_k*, i.e.* $\sum_{p \in \mathcal{P}_k} x_{ij}^{ptt'} = x_{ij}^{\chi_k tt'}$*. Therefore:*

$$Q(x^\Xi, y, z) \; = \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} \; + \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} \sum_{k \in \mathcal{K}} \sum_{p \in \mathcal{P}_k} c_{ij} x_{ij}^{ptt'} \; + \sum_{((i,t),(i,t')) \in \mathcal{H}_\mathcal{T}} \sum_{k \in \mathcal{K}} \sum_{p \in \mathcal{P}_k} c_{ii} x_{ii}^{ptt'}$$

$$= \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} f_{ij} y_{ij}^{tt'} \; + \sum_{((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}} \sum_{p \in \mathcal{P}} c_{ij} x_{ij}^{ptt'} \; + \sum_{((i,t),(i,t')) \in \mathcal{H}_\mathcal{T}} \sum_{p \in \mathcal{P}} c_{ii} x_{ii}^{ptt'} = Q(x, y)$$

Solution (x, y) *that replicates solution* (x^Ξ, y, z) *by partitioning of flows, is feasible to the LSNDP. The two solutions have an identical objective function value. Thus, the LSNDP is a relaxation of the* ***K-EMP****. As the* ***K-EMP*** *is also a relaxation of the LSNDP, we demonstrated that the* ***K-EMP*** *and the LSNDP are equivalent if for each* $k \in \{1, ..., \mathcal{K}\}$*, and each product* $p \in \mathcal{P}_k$*, a supplier* $s \in \mathcal{S}$ *that offers* p *offers all products of* $\mathcal{P}_k$*.*

4.3.3 Matching Rate

For a fixed value of $\mathcal{K}$, the most suitable $\mathcal{K}$-partition of products is the one that minimizes the loss of information in the **K-EMP**. In the ideal cases presented in 4.3.1 and 4.3.2, we have seen that any subset of products can be aggregated without

loss of information, if each supplier that offers a product in that subset offers all other products. Based on this feature, we introduce a measurement that evaluates if a pair of products is worth aggregating. For a product p, we denote $\mathcal{S}^p$ as the set of suppliers having p in their product line. We denote the matching rate of a pair of products p_i and p_j as:

$$m(p_i, p_j) = \frac{|\mathcal{S}^{p_i} \cap \mathcal{S}^{p_j}|}{|\mathcal{S}^{p_i} \cup \mathcal{S}^{p_j}|}$$

This matching rate measures the percentage of suppliers of p_i or p_j that offers both products at once, and indicates as to whether p_i or p_j should be aggregated. Two products with a null matching rate have no suppliers in common. On the other hand, a matching rate equal to 1 indicates that both products are provided by the same suppliers.

4.4 Partitioning Strategies

Based on the matching rate, we describe in this section two strategies for partitioning the product set, given a fixed value of $\mathcal{K}$. The first strategy relies on the K-partitioning problem. The second strategy is based on the K-Medoids algorithm.

4.4.1 K-Partitioning Problem

There are multiple MILP formulations for the graph partitioning problem. Recently, Alès et al. [Ales 2016] proposed a MILP formulation called the *K-partitioning problem*, that is stronger than the other formulations. Given a complete graph $G = (V, E)$ and a distance matrix d, the K-partitioning problem aims to determine the K-partition that minimizes the total distance. For each edge $(i, j) \in E$, an integer variable x_{ij} indicates if vertices i and j are in the same cluster. For each vertex i, the binary x_i equals 1 if and only if $i \in V$ is the vertex with the smallest index of its cluster. The K-partitioning problem is formulated as:

$$\min \sum_{ij \in E} d_{ij} x_{ij} \tag{4.15}$$

subject to

$$x_{ik} + x_{jk} - x_{ij} \leq 1, \quad \forall i, j, k \in V, i \neq k, j \neq k, i < j \tag{4.16}$$

$$x_j + x_{ij} \leq 1, \quad \forall i, j \in V, i < j \tag{4.17}$$

$$x_j + \sum_{i=1}^{j-1} x_{ij} \geq 1, \quad \forall j \in V \tag{4.18}$$

$$\sum_{i=1}^{n} x_i = K \tag{4.19}$$

$$x_{ij} \in \{0,1\}, \quad ij \in E \tag{4.20}$$

$$x_i \in [0,1], \quad i \in V \tag{4.21}$$

Objective function (4.15) minimizes the K-partition total weight. Constraints (4.16) ensure that if two incident edges ij and jk are activated then ik is also activated. Constraints (4.17) enforce there is at maximum one representative per cluster. Constraints (4.18) enforce there is at minimum one representative per cluster. Constraints (4.19) set the number of clusters to K. Constraints (4.20) and (4.21) define the variable domains. Note that the binary requirement on the representative variables can be relaxed, as having binary edge variables together with Constraints (4.17) and (4.18) forces the variables x_i to be 0 or 1.

In our context, each vertex $i \in V$ correspond to a product $p \in \mathcal{P}$. As the K-partitioning problem seeks for the K-partition with minimum distance, we set the distance between a pair of products $p, p' \in \mathcal{P}$ as: $d(p,p') = 1 - m(p,p')$.

4.4.2 K-Medoids Partitioning

The K-medoids [Jain 1988, Berkhin 2006] is a greedy algorithm that partitions N objects into K clusters. To do so, it requires a distance between each pair of objects. The algorithm starts with a random cluster configuration and iteratively search for a better neighbor solution. For a given cluster configuration C, K separate objects are designated as medoids (or centers) of the clusters, and each non-center object is assigned to the cluster with closest medoid. The cost of a cluster configuration is the sum of the intra-cluster costs. An intra-cluster cost is the sum of distances between the center of the cluster, and all objects assigned to the cluster.

Here, the objects to partition are the products. Let $C = \{C_1, ..., C_K\}$ be a cluster configuration, and C_i the i^{th} cluster. Each cluster is of the form $C_i = \{p^i_m, p^i_1, ..., p^i_{|C_i|-1}\}$, with p^i_m the medoid product and $\{p^i_1, ..., p^i_{|C_i|-1}\}$ the non-medoid products. Given a cluster configuration, we have: $\forall (i,j) \in \{1, ..., \mathcal{K}\}, i \neq j, C_i \cap C_j = \emptyset$, and $\bigcup_{k \in \{1,...,\mathcal{K}\}} C_k = \mathcal{P}$. As the K-medoids algorithm seeks for the configuration with minimum distance, we set the distance between a pair of products $p, p' \in \mathcal{P}$ as: $d(p,p') = 1 - m(p,p')$. Thus, the intra-cluster cost is:

$$cost(C_i) = \sum_{j=1}^{|C_i|-1} d(p^i_m, p^i_j)$$

The cost of a cluster configuration is:

$$cost(C) = \sum_{i=1}^{\mathcal{K}} cost(C_i)$$

We use the most common realisation of K-medoids clustering: the Partitioning Around Medoids (PAM) algorithm. Algorithm 2 gives an overview of our method.

Algorithm 2 K-medoids algorithm

Require: Partition size $\mathcal{K}$
  Select $\mathcal{K}$ random products as medoids
  Assign each non-medoid product p to medoid p_m^i that minimizes $d(p_m^i, p)$
  Compute cluster configuration cost: $cost(C)$
  $decrease = true$
  while $decrease$ **do**
    $decrease = false$
    for Each medoid product p_m^i **do**
      for Each non-medoid product p **do**
        Swap p_m^i and p
        Assign each non-medoid product p to medoid p_m^i that minimizes $d(p_m^i, p)$
        Compute new cluster configuration cost: $cost(C_{new})$
        if $cost(C_{new}) < cost(C)$ **then**
          $decrease = true$
          $C = C_{new}$
        end if
      end for
    end for
  end while

First, $\mathcal{K}$ random products are designated as medoids. The other products are assigned to the medoid product that minimizes the distance. The initial cluster configuration and its cost are saved. Then, for each pair of medoid and non-medoid products, we swap roles, determine the new cluster configuration and compute the associated cost. If the value found is lower than the cost of the saved cluster configuration, the new cluster configuration and its cost are saved. While a new cluster configuration with lower cost is found, the process is iterated.

The final cluster configuration is converted into a $\mathcal{K}$-partition of the products, wherein elements of the i^{th} subset are: the medoid-product of the i^{th} cluster , i.e. p_m^i, and the non-medoid products assigned to the i^{th} cluster, i.e. $\{p_1^i, ..., p_{|C_i|-1}^i\}$.

4.5 Dynamic Partial Benders Strategy

It is fairly intuitive that, as the number of super-products increases, the information added to the master tends to be more accurate. This yields stronger bounds for the **K-EMP**, at the cost of a more complex branch-and-bound tree. On the other hand, considering less super-products reduces the number of variables in the master problem and augments its computational tractability, but induces looser bounds. To better exploit these characteristics, we developped a dynamic partial Benders

strategy, where the number of super-products evolves as the algorithm progresses. Algorithm 3 gives an overview of the Benders strategy.

Algorithm 3 Dynamic partial Benders strategy

Require: Maximum number of super-products $\mathcal{K}_{max}$, Global time limit t_{max}, Bounds time limit t_{max}^{bounds}
$\mathcal{K} = 1$
while Computation time $< t_{max}$ **AND** optimal solution is not found **do**
Apply partitioning strategy
Build the associated **K-EMP**
if $\mathcal{K} > 1$ **then**
Initiate decomposition **K-EMP-SP** with current bounds
Initiate decomposition **K-EMP-SP** with generated Benders inequalities
end if
if $\mathcal{K} \neq \mathcal{K}_{max}$ **then**
while Computation time $< t_{max}$ **AND** optimal solution is not found **AND** upper/lower bound objective have improved by more than 1% in the last t_{max}^{bounds} seconds **do**
Apply Benders enhanced strategy
end while
else
while Computation time $< t_{max}$ **AND** optimal solution is not found **do**
Apply Benders enhanced strategy
end while
end if
Update best lower bound and best upper bound
Keep in memory the Benders inequalities
$\mathcal{K} = \mathcal{K} + 1$
end while

The rationale is to solve progressively a partial Benders decomposition that gets finer over time, until an optimal solution is found, or the time limit is exceeded. The algorithm starts with a single super-product ($\mathcal{K} = 1$), which yields the master problem with minimal amount of variables, and enables to explore the search space quickly. The corresponding decomposition is solved while there is computation time left, the optimal solution is not found, and the upper and lower bound objective values have been improved by more than 1% in the last t_{max}^{bounds} seconds.

If the upper and lower bound objective values have not been improved by more than 1% in the last t_{max}^{bounds} seconds, we stop solving the decomposition. The current upper/lower bounds as well as the Benders inequalities generated are saved, and $\mathcal{K}$ is increased by one unit. A more accurate master problem **K-EMP** is then determined thanks to one of the partitioning strategies described earlier.

The new Benders decomposition is initiated with the upper and lower bounds, as well as all the Benders inequalities found previously. Since the Benders inequalities only involve the y variables, a Benders inequality obtained for a given **K-EMP**-

SP decomposition, is valid for any **K-EMP-SP** decomposition. By keeping the Benders inequalities from one decomposition to another, we make sure not to revisit a non-optimal solutions generated previously.

The new Benders decomposition is solved until a stopping criterion is met, then the number of super-products $\mathcal{K}$ increases of one unit. This process is repeated until an optimal solution is found, or the time limit is exceeded. To restrict the size of the master problem, a threshold value $\mathcal{K}_{max}$ is set on $\mathcal{K}$.

4.6 Computational Study

The objective is to evaluate the efficiency of the proposed dynamic Benders strategy. We first describe the instances and the setup of the study. We then assess the impact of $\mathcal{K}$, the number of super-products in the master problem, and ϕ, the probability for a supplier to offer a product. Finally, we demonstrate the efficiency of our dynamic Benders strategy.

4.6.1 Instances

In this chapter we modify the random generator presented in Chapter 3, so that a classification of the products is considered. Products are generated from three parameters: the number of product families $|\mathcal{F}|$, the number of products $|\mathcal{P}|$, and a probability ϕ of offering a product. Each product is randomly assigned to a single product family. Are randomly assigned to each supplier one, two or three product families. Each supplier provides each product in these product families with a probability ϕ. For the remaining parameters, instances are produced as in Chapter 3. The instances are based on the following parameter values: the number of nodes $|\mathcal{G}| = \{50\}$, the connectivity radius $\alpha = \{10, 30\}$, the number of days $D = 30$, the time granularity $\Delta = \{2, 3\}$, the number of product families $\mathcal{F} = \{7\}$, the number of products $|\mathcal{P}| = \{100, 200, 300, 400, 500\}$, and the probability $\phi = \{0.25, 0.50, 0.75\}$. We generated 5 instances for the 60 possible combinations of parameter values, for a total of 300 instances.

4.6.2 Setup of the Study

We propose four methods based on the enhanced Benders strategy. The methods **Single**, **Medoids**, and **Random** are direct applications of the enhanced Benders strategy. **Single** considers a master problem that has a single super-product. In **Medoids** and **Random**, the master problem has as many super-products as product families in the instance (i.e. $|\mathcal{F}|$ super-products). In **Medoids**, a partition of the product set is identified with the K-medoids algorithm presented in subsection 4.4.2. In **Random**, a partition of the product set is determined randomly. The method **Dynamic** is our dynamic Benders strategy, where the decompositions are solved with the enhanced Benders strategy. The partitioning strategy used in **Dynamic** is the K-medoids algorithm.

We also implemented the method **K-part**, where the master problem has $|\mathcal{F}|$ super-products obtained from the solution of the K-Partitioning problem presented in subsection 4.4.1. The K-Partitioning problem is solved with a stopping criteria of a proven optimality gap of 1% and a maximum run-time of 5 minutes. For more than half of the instances, the K-Partitioning problem ran out of memory and **K-part** did not provide a feasible solution to the original problem. On the other hand, the K-medoids algorithm performed successfully for all instances. When **K-part** provided a feasible solution to the original problem, **Medoids** reported a lower optimality gap at termination for 90.54% of the instances. These results indicate that the K-medoids algorithm is a better partitioning strategy than solving the K-partitioning problem. Thus, we do not show the results obtained with **K-part** in the computational study.

We compared our four methods with three benchmark methods. In **CPLEX**, the complete program is solved with CPLEX's default branch-and-cut. In **CPLEX-Benders**, the complete program is solved with CPLEX's automatic Benders decomposition. The method **Knowledge** is a direct application of the enhanced Benders strategy, that considers a master problem with $|\mathcal{F}|$ super-products. The super-products are built from the instance structure, i.e. each super-product corresponds to a family of products and aggregates all the products it contains.

The enhanced Benders strategy is implemented with the callback framework where subproblems are solved within the context of the branch-and-bound tree used to solve the master problem. Specifically, whenever an integral solution is found in the tree, the subproblem is solved. The resulting cut is then embedded in every node of the tree, and may cut-off the incumbent. The process terminates once the optimality gap is closed. We initiate every method with a heuristic solution (x_h, y_h) obtained by setting each vehicle variable, $y_{ij}^{tt'}$ to the ceiling of its value in the optimal solution of the linear relaxation of the LSNDP.

All algorithms are coded in C++ and executed on an Intel Xeon E5-2695 processor with 16 GB of memory under Linux 16.04. Linear and integer programs were solved using Cplex 12.7. All algorithms are executed with a stopping criteria of a proven optimality gap of 1% or less and a maximum run-time of 1.5 hours. For **Dynamic** the maximum number of super-products $\mathcal{K}_{max}$ is set to 7. The time limit on the improvement of the bounds t_{max}^{bounds} is set to 770 seconds.

4.6.3 Analyzing the Impact of $\mathcal{K}$ on the Master Problem

We investigate the impact of the number of super-products on the **K-EMP**. For all instances we solve the root relaxation of the LSNDP and that of the **K-EMP**, with $\mathcal{K}$ varying from 1 to 7. We observe two performance indicators: i) the objective value r^{sol} of the root node solution; ii) the computational time r^{time} required to solve the root relaxation.

To compare the **K-EMP** and the LSNDP, we compute a root-gap for each

instance:

$$\text{root-gap} = \frac{r^{sol}_{LSNDP} - r^{sol}_{K-EMP}}{r^{sol}_{LSNDP}}$$

As the **K-EMP** is a relaxation of the LSNDP, the **K-EMP** root solution objective value is lower to that of the LSNDP. Therefore, the root-gap is positive.

For each instance, we also compute a computation time rate:

$$\text{root-time} = \frac{r^{time}_{K-EMP}}{r^{time}_{LSNDP}}$$

In Figure 4.7, we display the root-gap and computation time rate on average over all the instances.

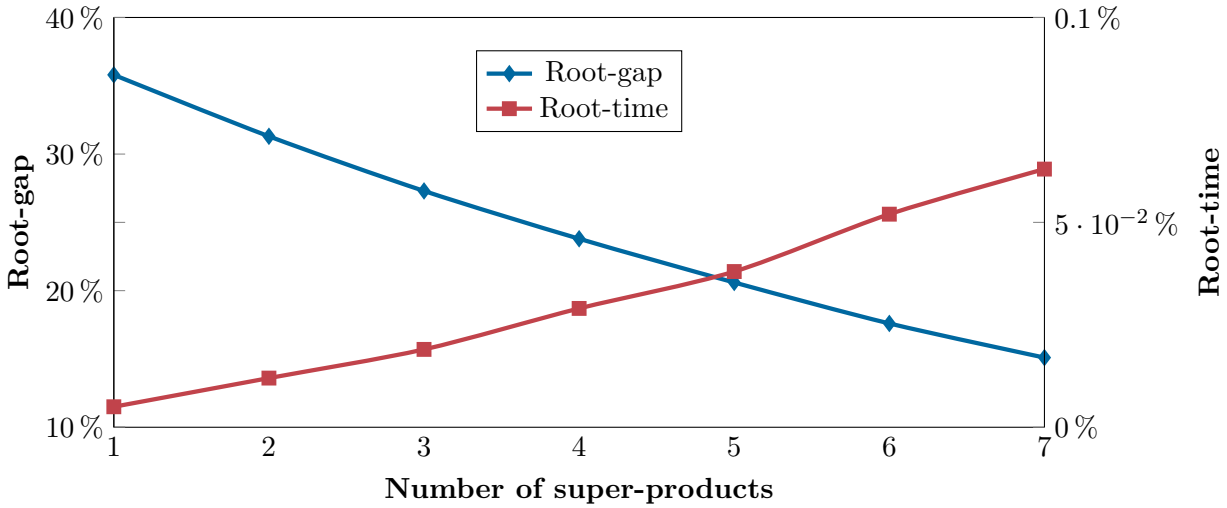

Figure 4.7: **K-EMP** root-gap and computation time rate

The master problem with a single super-product yields an important root-gap of 35.8%. However, we can see that the root-gap decreases gradually as the number of super-products in the master increases. This confirms that considering more super-products in the master allow to better approximate the original problem. This improvement is significant, as the root-gap decreases by more than 20% when we consider 7 super-products in the master. As expected, the computation time for solving the master root relaxation increases with the number of super-products. However, the computation time rates are small. In the worst case, the master root relaxation is solved in 0.063% of the time needed to solve the LSNDP root relaxation.

4.6.4 Analyzing the Impact of ϕ on the Instances

We analyze the impact of parameter ϕ on the instances. As a reminder, a supplier specialized in a family of products has a probability ϕ to offer a product in that

family. In Figure 4.8 we display the root-gap over instances with the same value for parameter ϕ.

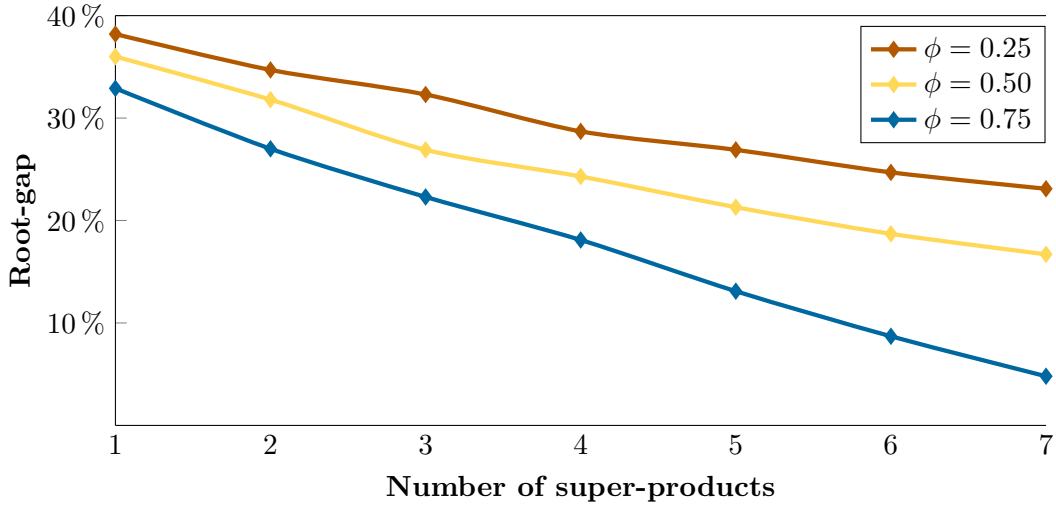

Figure 4.8: **K-EMP** root-gap for different values of ϕ

For each value of parameter ϕ, we observe the same behavior. Indeed, regardless of the set of instances considered, the root-gap decreases with the number of super-products. We note that this reduction is of greater magnitude as the value of ϕ increases. This indicates that increasing the number of super-products in the master is more valuable for instances with high values of ϕ.

This behavior is a direct consequence of the impact of ϕ on the products matching rates. For each instance, we compute the matching rate of each product family. We define the matching rate of a product family as the average matching rate of all pairs of products in that family. As a result, the matching rate of a product family evaluates if it is worth aggregating all the products of the family. In Table 4.1, we report the average matching rates of the product families, over instances with the same values of ϕ.

Table 4.1: Average matching rate of the product families

ϕ	25%	50%	75%
Matching rate	31.52%	41.58%	63.33%

We observe that the average matching rate of the products families raises with ϕ. Thus, when ϕ increases, it is more favourable to aggregate products by families. As a consequence, it is more valuable to increase the number of super-products for instances with high values of parameter ϕ.

4.6.5 Overall Performance

We benchmark our four Benders methods against **CPLEX**, **CPLEX-Benders** and **Knowledge**. To do so, we assess the quality of the primal solution and the dual solution computed by each method. For each instance and each method $Method$, let UB_{Method} be the objective function value of the best primal solution obtained, and let LB_{Method} be the objective function value of the best dual solution obtained. UB_{Best} and LB_{Best} denote the objective function value of the best primal solution and the best dual solution found over all methods. For each instance and each method, we compute two performance indicators:

$$\text{gap}_{UB}^{Method} = \frac{\text{UB}_{Method} - \text{UB}_{Best}}{\text{UB}_{Method}} \qquad \text{gap}_{LB}^{Method} = \frac{\text{LB}_{Best} - \text{LB}_{Method}}{\text{LB}_{Best}}$$

Both indicators are positive. An indicator equals 0 if, over all methods, $Method$ obtained the primal/dual solution with best objective function value. In Table 4.2 we present the average optimality gap at termination and the average performance indicators reported by each method.

Note that over the 300 instances, very few were solved within the time limit. More specifically, CPLEX solved to optimality three of the smallest instances.

Table 4.2: Average gap at termination and average performance indicators

Method	Optimality Gap	gap_{LB}^{Method}	gap_{UB}^{Method}
CPLEX	23.78%	19.86%	2.43%
CPLEX-Benders	63.79%	61.36%	4.30%
Knowledge	3.92%	0.45%	1.14%
Single	4.28%	0.66%	1.31%
Medoids	4.02%	0.37%	1.33%
Random	8.64%	3.82%	2.69%
Dynamic	3.30%	0.55%	0.40%

Overall, both **CPLEX** and **CPLEX-Benders** have lower performances than all other methods. The benchmark method with the best performance is **Knowledge**. The results obtained by **Knowledge** are slightly better than those obtained by static Benders strategies: **Single** and **Medoids**. However, **Knowledge** is outperformed by the **Dynamic** method, that is the most effective overall.

We can see that the performance of **Random** is quite poor compared to the others Benders decompositions based on $|\mathcal{F}|$ super-products: **Knowledge** and **Medoids**. This demonstrates that the partition of the product set has a significant influence on the behaviour of the algorithm. In addition, the fact that the results obtained with **Medoids** are similar to those of **Knowledge** validates our K-medoids procedure for partitioning the product set.

4.6.6 Comparing the Benders Strategies

Now, we analyze the performance of the **Single**, **Medoids** and **Dynamic** methods. In Table 4.3 we report the average optimality gaps at termination and the average performance rates over instances with the same values of ϕ. The best values are in bold.

Table 4.3: Comparison of **Single**, **Medoids** and **Dynamic**

	Opt. gap			gap_{LB}^{Method}			gap_{UB}^{Method}		
ϕ	**Single**	**Medoids**	**Dynamic**	**Single**	**Medoids**	**Dynamic**	**Single**	**Medoids**	**Dynamic**
25%	3.96%	4.37%	**2.90%**	0.54%	**0.30%**	0.46%	1.15%	1.81%	**0.15%**
50%	4.38%	4.71%	**3.46%**	0.62%	**0.42%**	0.55%	1.23%	1.77%	**0.35%**
75%	4.49%	**3.00%**	3.54%	0.80%	**0.39%**	0.66%	1.55%	**0.43%**	0.71%

For each set of instances, the best lower bounds are produced by **Medoids**. This result is not surprising, as we have seen that increasing the number of super-products in the master improves the objective value of the root node solution. We also observe that **Single** and **Dynamic** produce lower bounds comparable in quality with those computed by **Medoids**. On the other hand, we observe that increasing the number of super-products in the master does not necessarily enable to produce better primal solutions. Indeed, for instances $\phi = 25\%$ and $\phi = 50\%$, **Single** yields better upper bounds than **Medoids**. **Medoids** yields better primal solutions solely for instances with $\phi = 75\%$. This can be explained by the fact that these instances are particularily suitable to aggregate products by families, as seen in Subsection 4.6.4.

Dynamic has the best performance overall. In terms of upper bounds, it significantly outperforms both static Benders strategies for instances with $\phi = 25\%$ and $\phi = 50\%$. When $\phi = 75\%$ the primal solutions produced by **Medoids** are slightly better than those computed by **Dynamic**.

Now, we analyze in further detail the **Dynamic** method, and the impact of refining the master problem in the course of the optimization process. Figure 4.4 displays the distribution of the iterations performed by the **Dynamic** method, over all instances. For example, "3 iterations" correspond to instances for which the **Dynamic** method solved three decompositions. We also investigate the improvement in the upper bound performed per iteration. For each decomposition solved, we retain the initial primal solution, and compare it with the best primal solution obtained before refining the master. In Table 4.5, we report the average improvement of the primal solution for each iteration. We also report the average time spent during each iteration. Finally, in Figure 4.9 we display the distribution of the occurrences of improvement for the primal solution in the course of the optimization process. The computation time of 5,400 seconds is divided in 10 intervals of 540 seconds. Thus, a value of 10 for the first interval indicates that the incumbent was improved 10 times within the 540 first seconds of computation. We present results for **Single**, **Medoids** and **Dynamic** over all instances.

Dynamic reaches the maximum number of iterations for a very few number

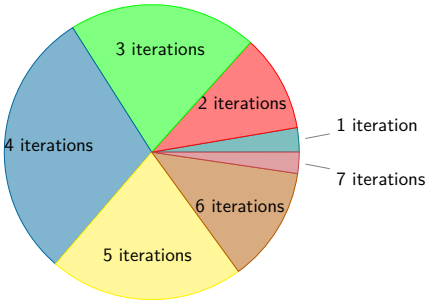

Table 4.4: Distribution of instances by # of iterations performed

Table 4.5: Performance on improving UB

It.	UB Impr.	Time spent
1	2.39%	32.87%
2	0.93%	30.19%
3	0.34%	20.88%
4	0.07%	10.75%
5	0.02%	3.91%
6	0.01%	1.33%
7	0.00%	0.06%

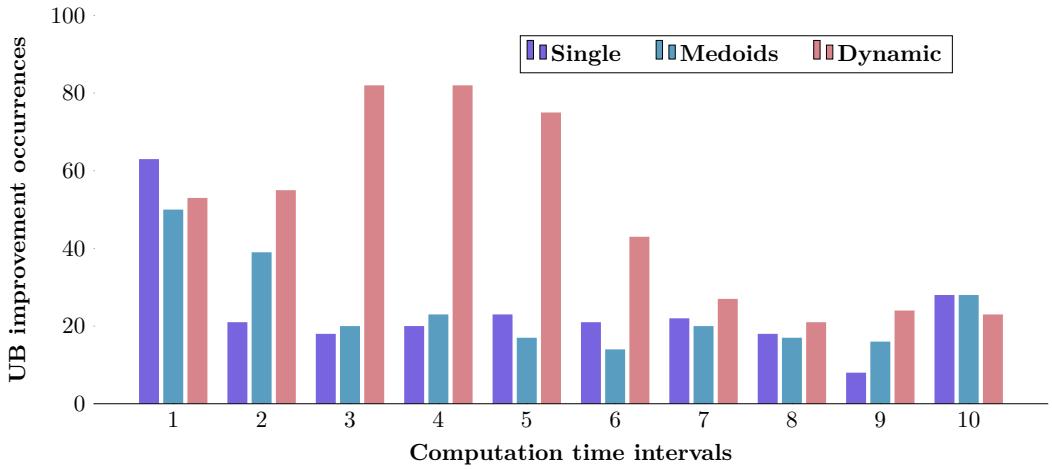

Figure 4.9: Distribution of the occurrences of improvement for the primal solution

of instances. For a majority of instances, the dynamic strategy performs 3, 4 or 5 iterations. Unsurprisingly, the percentage of computation time spent at each iteration is decreasing, as the number of instances for which the i^{th} iteration is performed is necessarily greater or equal than the number of instances for which the $i+1^{th}$ iteration is performed. In addition, we can observe that the percentage of improvement of the primal solution is correlated to the time spent at each iteration.

These results explains why **Dynamic** has a lower performance than **Medoids** for instances with $\phi = 75\%$. Indeed, these instances are suited for aggregating products by families and thus considering 7 super-products in the master. However, the **Dynamic** method spends most of the computation time solving the three firsts decompositions, while it spends a very short time solving the final decomposition with 7 super-products. In that case, the dynamic strategy could be more effective if it spends less time in the early iterations. Nevertheless, these instances are quite singular and the fact that **Dynamic** outperforms **Single** and **Medoids** overall shows the interest of our dynamic approach.

Figure 4.9 shows that, during the course of the optimization, the occurrences of improvement of the primal solution are significantly more numerous for **Dynamic** than for **Single** and **Medoids**. Static methods tend to improve the primal solution during the two first deciles of the computation horizon. After that, the occurrences of improvement of the primal solution are scarce. With the dynamic approach these occurrences are abundant in the six first deciles of the computation horizon, which can explain why **Dynamic** is more effective than **Single** and **Medoids**.

4.7 Conclusions

In this Chapter, we proposed a dynamic partial Benders strategy for solving the LSNDP. Our method solves a partial Benders decomposition where the number of super-products in the master problem increases in the course of the optimization process. To develop this algorithm, we adapted the enhanced Benders strategy proposed in Chapter 3. More specifically, we introduced a Benders master problem with multiple super-products based on a partition of the products, and proved its validity. We also adapted the valid inequalities presented in section 3.3 to this new master problem. To optimize the quality of the master problem at each iteration of the dynamic partial Benders strategy, we proposed strategies to effectively partition the products. The computational study shows that the dynamic Benders strategy is overall more effective than static Benders strategies.

Chapter 5

Managing the Network Size: a Sparse Graph Construction Heuristic

Contents

5.1 Introduction

In Chapter 3, we presented a Benders strategy that remains effective despite scaling up the number of products. In this chapter, we focus on another parameter that directly affects the model computational tractability. The LSNDP is defined over a time-expanded graph $\mathcal{G}_\mathcal{T}$. Since variables and constraints in the LSNDP are based on the nodes and arcs of $\mathcal{G}_\mathcal{T}$, the model size increases directly with the size of the time-expanded graph.

The size of a time-expanded graph is determined by: the static graph $\mathcal{G}$ it reflects, the time horizon and the time discretization. In the time-expanded network,

the physical locations of the static network are replicated at each time interval of the time horizon. The length of these time intervals is a crucial choice, as discretizing time involves a loss of information compared to the continuous time problem. This loss of information may be of great or less impact, depending on the length of the time intervals. Indeed, all travel times in $\mathcal{G}$ that are not multiple of the time intervals, are rounded to the higher multiple in $\mathcal{G}_\mathcal{T}$. For example, given a 1-hour discretization, an arc with a transit time of 1h20 in $\mathcal{G}$ is represented by time-expanded arcs with transit times of 2h in $\mathcal{G}_\mathcal{T}$. These approximations can prevent the model to capture accurately the consolidation opportunities from the continuous time problem [Boland 2017]. To reduce these approximations, and have an accurate model that provides high-quality solutions, one must therefore choose a small duration for the time intervals.

On the other hand, the time discretization significantly impacts the computational tractability. With a 1-hour discretization and a planning horizon of 3 months, each node in the static graph is duplicated 2,160 times in the time-expanded graph. Such time-expanded graph likely yields a LSNDP instance that is too large to get solved in a reasonable amount of time.

To face this challenge in the context of the *Continuous Time Service Network Design Problem* (CTSNDP), Boland *et al.* [Boland 2017] proposed the Dynamic Discretization Discovery (DDD) algorithm. The algorithm proves to be very effective on large time-expanded graphs, in particular when considering fine time discretizations. The main advantage of DDD is that it solves the CTSNDP without building all nodes and arcs in the time-expanded graph.

We seek to solve instances of LSNDP that are inspired by the operations of our industrial partner. As these instances represent distribution networks with many suppliers, warehouses and customers (see Chapter 1), they involve large static networks. The time horizon for planning transportation operations must be sufficiently long, at least 15 days. In addition, to provide high-quality transportation plans it is necessary to consider a sufficiently fine granularity of the temporal discretization. For these reasons, industry-sized instances generate extremely large time-expanded networks, which leads to mathematical programs that are computationally intractable for on-the-shelf optimization solvers.

Because of the differences between the LSNDP and the CTSNDP (see Chapter 2), it is impossible to apply the DDD algorithm to our problem directly. However, in this chapter we propose a very efficient heuristic approach based on the same sparse graph construction principles. This chapter is organized as follows. In section 5.2, we sketch the outlines of DDD algorithm and explain why that method cannot be applied unaltered to the LSNDP. In section 5.3 we describe our heuristic. Its efficiency is evaluated through computational experiments in section 5.4 and conclusions are drawn in section 5.5.

5.2 Dynamic Discretization Discovery and the LSNDP

The DDD adresses the Continuous Time Service Network Design Problem (CT-SNDP), a version of the SNDP wherein time discretization is thin enough to capture every real-life consolidation opportunity. This generally induces extremely large time-expanded graphs and intractable models. The DDD method overcomes this difficulty by manipulating sparse time-expanded graphs. The algorithm rationale is that the time-expanded graph can be very large, but only very limited subsets of nodes and arcs appear in the optimal solution. By an iterative process, the DDD method builds iteratively a sparse time-expanded graph, such that the optimal solution of the integer program defined on the final graph is an optimal solution of the integer program defined on the complete graph.

In order to ease the reading, and since the CTSNDP corresponds to a particular case of the SNDP, we use the term SNDP here. In this section, we sketch the outlines of DDD algorithm and explain what prevents us to use it unaltered to the LSNDP.

5.2.1 Description of the Dynamic Discretization Discovery

DDD is an exact method that solves the SNDP without considering the complete time-expanded graph $\mathcal{G}_\mathcal{T}$. The algorithm first step is to generate a *partially time-expanded graph* $\mathcal{X}_\mathcal{T}$ that contains a subset of the nodes in $\mathcal{G}_\mathcal{T}$, and arcs with underestimated transit times. $\mathcal{X}_\mathcal{T}$ must satisfy a set of properties ensuring that SNDP($\mathcal{X}_\mathcal{T}$) is a relaxation of SNDP($\mathcal{G}_\mathcal{T}$). An optimal solution of SNDP($\mathcal{X}_\mathcal{T}$) provides a lower bound of SNDP($\mathcal{G}_\mathcal{T}$), but may not be valid for the complete time-expanded graph. If so, a repair mechanism modifies $\mathcal{X}_\mathcal{T}$ while maintaining the conditions ensuring that SNDP($\mathcal{X}_\mathcal{T}$) is a relaxation of SNDP($\mathcal{G}_\mathcal{T}$). The DDD method converges when a SNDP($\mathcal{X}_\mathcal{T}$) optimal solution is feasible for the complete time-expanded graph. As refining operations of $\mathcal{X}_\mathcal{T}$ are limited, the process converges in a finite number of iterations. Figure 5.1 presents a high-level overview of the algorithm.

The algorithm convergence relies on the premise that, throughout the process, the SNDP associated with $\mathcal{X}_\mathcal{T}$ remains a relaxation of the SNDP associated with the complete time-expanded graph. Boland *et al.* build an initial $\mathcal{X}_\mathcal{T}$ by selecting nodes, arcs, and arcs travel times as follows. They define an *early-arrival arc* $((i,t)(j,t')) \in \mathcal{A}_\mathcal{T}$ as a time-expanded copy of arc $(i,j) \in \mathcal{A}$, that underestimates real transit time, i.e. $t' - t \leq t_{ij}$. The initial $\mathcal{X}_\mathcal{T}$ satisfies the following properties:

Property 1. For every commodity $k \in \mathcal{K}$ originating from o_k at time t_k and destinated to d_k at time t'_k, the time-expanded nodes (o_k, t_k) and (d_k, t'_k) are in $\mathcal{X}_\mathcal{T}$;

Property 2. Every arc $((i,t),(j,t'))$ in $\mathcal{X}_\mathcal{T}$ is an early-arrival arc;

Property 3. For every arc $a = (i,j)$ in the static network $\mathcal{G}$, and for every node $(i,t) \in \mathcal{X}_\mathcal{T}$, there is a corresponding arc $((i,t),(j,t')) \in \mathcal{X}_\mathcal{T}$.

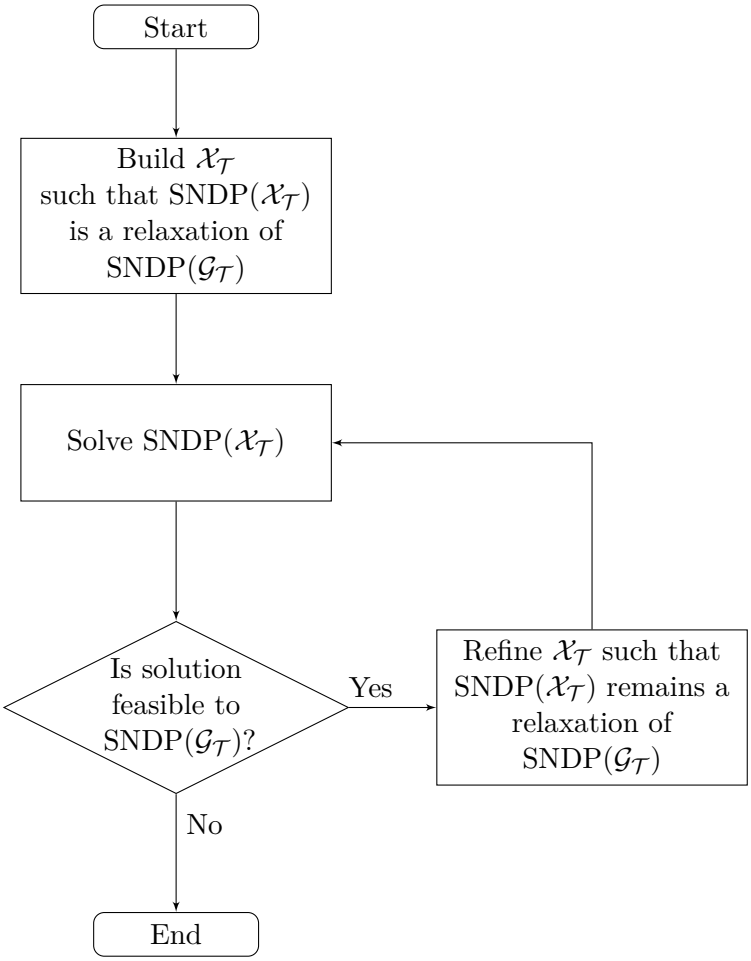

Figure 5.1: DDD global scheme

Given a commodity $k \in \mathcal{K}$, let p be a path in $\mathcal{G}_\mathcal{T}$ from (o_k, t_k) to (d_k, t'_k), that visits distinct physical locations $\{i_1, ..., i_n\}$ at times $\{t_1, ..., t_n\}$. A partially time-expanded graph $\mathcal{X}_\mathcal{T}$ that satisfies Property 1, 2, and 3, necessarily contains an equivalent path p' from (o_k, t_k) to (d_k, t'_k), that visits same succession of physical locations $\{i_1, ..., i_n\}$, at earlier times $\{t'_1, ..., t'_n\}$, i.e. $t'_i \leq t_i \ \ \forall i \in [1, n]$.

So-called *early-arrival paths* are direct consequences of the *early-arrival arcs*, and reproduce each shipment itinerary that is feasible in the original problem. In addition, *early-arrival paths* enable shipments to reach each intermediary node before what is feasible in the original problem, in such a way that $\mathcal{X}_\mathcal{T}$ offers more consolidations opportunities than $\mathcal{G}_\mathcal{T}$. Therefore, SNDP($\mathcal{X}_\mathcal{T}$) provides a lower bound of SNDP($\mathcal{G}_\mathcal{T}$).

A SNDP($\mathcal{X}_\mathcal{T}$) optimal solution may not be valid for the complete time-expanded graph, which happens when flows transit through "too-short" arcs. In that case the durations of such "too-short" arcs are corrected, in a way that the resulting $\mathcal{X}_\mathcal{T}$ again satisfies Properties 1, 2, and 3. This refining mechanism is detailed in subsection 5.3.3.

Figure 5.2 illustrates a static graph with travel times depicted on the arcs. In this example we consider a single commodity k, available at $(A, 1)$ and required at $(C, 4)$. Figure 5.3 is a partially time-expanded network $\mathcal{X}_\mathcal{T}$ that satisfies the properties listed above. The only path from $(A, 1)$ to $(C, 4)$ is an early-arrival path as it requires $((A, 1)(B, 2))$ that underestimates the real transit time from A to B. Hence, the SNDP($\mathcal{X}_\mathcal{T}$) optimal solution is not valid for the original problem, and $\mathcal{X}_\mathcal{T}$ is repaired. Arc $((A, 1)(B, 2))$ is too-short since there is no node in $\mathcal{X}_\mathcal{T}$ that enables to model (A, B) real transit time. Therefore node $(B, 3)$ is added, and $((A, 1)(B, 3))$ replaces $((A, 1)(B, 2))$. Finally, arcs of $\mathcal{X}_\mathcal{T}$ are updated in order to take into account new node $(B, 3)$. Figure 5.4 depicts the updater graph $\mathcal{X}_\mathcal{T}$ after refining operations.

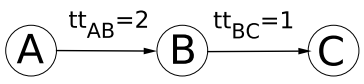

Figure 5.2: Static graph

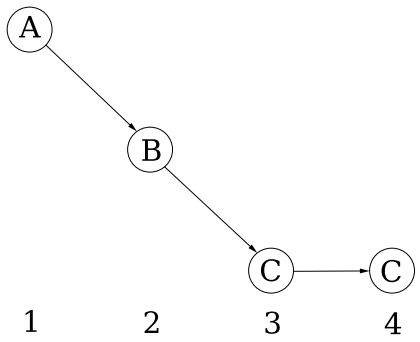

Figure 5.3: $\mathcal{X}_\mathcal{T}$ before repair

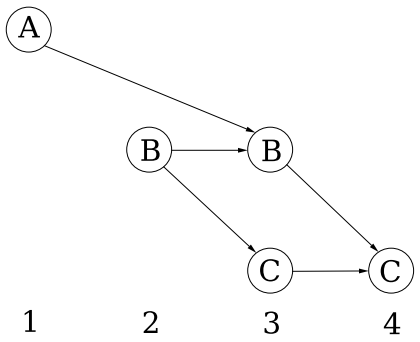

Figure 5.4: $\mathcal{X}_\mathcal{T}$ after repair

5.2.2 Critical Differences between SNDP and LSNDP

Considering storage costs at warehouses in the LSNDP forbid the use of the DDD algorithm as an exact method. Indeed, when we consider storage costs, even if the partially time-expanded graph $\mathcal{X}_\mathcal{T}$ satisfies the properties introduced by Boland *et al.*, the optimal solution of SNDP($\mathcal{X}_\mathcal{T}$) may not be a lower bound of the original problem. Therefore, with storage costs, the DDD algorithm may converge to a sub-optimal solution. We illustrate this issue with the example depicted in Figures 5.5, 5.6 and 5.7.

Let us assume that a single commodity, available at $(A, 1)$, must be routed to $(B, 3)$. The costs of arcs $((A, t), (A, t+1))$, $((A, t), (B, t+1))$ and $((B, t), (B, t+1))$ are 1, 1 and 2 respectively. Figure 5.5 represents the complete time-expanded graph $\mathcal{G}_\mathcal{T}$, and the optimal solution obtained when solving the SNDP($\mathcal{G}_\mathcal{T}$). The objective function of this optimal flow has a value of 2. Figure 5.6 represents the initial partially time-expanded graph $\mathcal{X}_\mathcal{T}$ obtained with Boland *et al.* procedure, as well as the optimal solution obtained when solving the SNDP($\mathcal{X}_\mathcal{T}$). This solution is not feasible for the original problem, as the flow transits along the too-short arc $((A, 1), (B, 1))$. After refining $\mathcal{X}_\mathcal{T}$, we obtain the partially time-expanded graph depicted in Figure 5.7. The optimal solution obtained when solving the SNDP($\mathcal{X}_\mathcal{T}$) is feasible for the original problem. However, it has an objective value of 3. Therefore, on this example the DDD method converges to a sub-optimal solution.

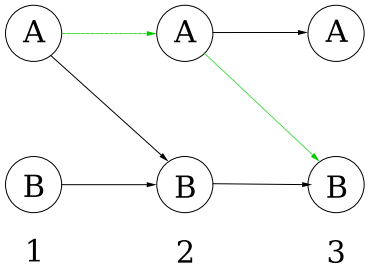

Figure 5.5: $SNDP(\mathcal{G}_\mathcal{T})$ optimal solution

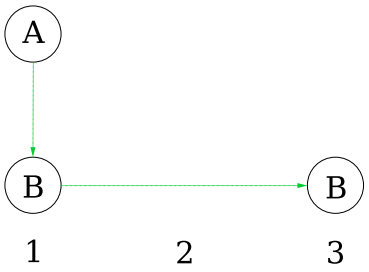

Figure 5.6: Optimal solution of the SNPD associated with initial $\mathcal{X}_\mathcal{T}$

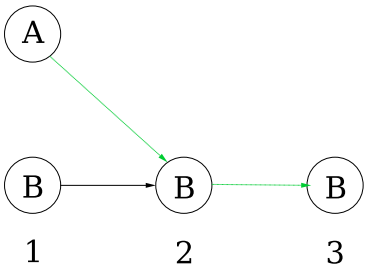

Figure 5.7: Optimal solution of the SNPD associated with refined $\mathcal{X}_\mathcal{T}$

Another reason that motivates us to adapt the DDD algorithm for the LSNDP is the absence of a predefined origin for a customer demand. In the case of the SNDP, it is easy to define a small partially time-expanded graph $\mathcal{X}_\mathcal{T}$ that satisfies the properties of Boland *et al.*. In the case of the LSNDP, a partially time-expanded graph $\mathcal{X}_\mathcal{T}$ that satisfies the properties of Boland *et al.* must contains all suppliers in the network, due to the absence of origins for the demands. As a result, the size of $\mathcal{X}_\mathcal{T}$ becomes rapidly too large for the DDD algorithm to be effective. This issue will be described in details in section 5.3, and we will explain how we handle it.

5.3 Iterated Two Phases Subgraph Generation Heuristic

Although the DDD algorithm cannot be applied directly to the LSNDP, we retain its main concepts to keep its main advantage, i.e. build a subgraph $\mathcal{X}_\mathcal{T}$ of $\mathcal{G}_\mathcal{T}$. However, in our case, we cannot guarantee that we obtain the optimal solution of the LSNDP when we solve it on $\mathcal{X}_\mathcal{T}$.

The method is called the Iterated Two Phases Subgraph Generation Heuristic

(ITPSGH). The general algorithm is presented in 5.3.1 and its steps in the following subsections. A first question addressed in 5.3.2 is how to build the initial partially time-expanded graph $\mathcal{X}_\mathcal{T}$. Then, the method iterates between two phases. The first phase, presented in 5.3.3, is directly inspired from the DDD algorithm. The second phase detailed in 5.3.4 aims to take into account the storage costs. In the remaining of the section, $IP(\mathcal{X}_\mathcal{T})$ and $LP(\mathcal{X}_\mathcal{T})$ denote respectively the optimal value of the LSNDP and its linear relaxation, defined on the partially time-expanded graph $\mathcal{X}_\mathcal{T}$.

5.3.1 General Algorithm

Algorithm 4 Iterated Two Phases Subgraph Generation Heuristic

Build an initial graph $\mathcal{X}_\mathcal{T}$ (see 5.3.2)
repeat
  Refine $\mathcal{X}_\mathcal{T}$ by eliminating too-short arcs (see 5.3.3)
  Refine $\mathcal{X}_\mathcal{T}$ by adding storage costs (see 5.3.4)
until The rounded-up optimal solution of $LP(\mathcal{X}_\mathcal{T})$ is feasible for $IP(\mathcal{G}_\mathcal{T})$
Remove too-short arcs from $\mathcal{X}_\mathcal{T}$
Update storage costs in $\mathcal{X}_\mathcal{T}$
Solve $IP(\mathcal{X}_\mathcal{T})$

The algorithm starts by building an initial graph $\mathcal{X}_\mathcal{T}$ including a subset of nodes of $\mathcal{G}_\mathcal{T}$. The initial arc set of $\mathcal{X}_\mathcal{T}$ is composed of too-short arcs with storage costs set to zero. The procedure to build $\mathcal{X}_\mathcal{T}$ is explained in section 5.3.2. Then, the algorithm iterates between two steps in which we solve linear programs instead of integer programs as in [Boland 2017] to speed-up the algorithm. Similarly to the DDD refining mechanism, the first step consists in eliminating too-short arcs used in the optimal solution of $LP(\mathcal{X}_\mathcal{T})$. The algorithm is described in 5.3.3. Then, in a second phase, the storage costs are updated in $\mathcal{X}_\mathcal{T}$ as explained in 5.3.4. These two steps are iterated until the rounded-up solution of $LP(\mathcal{X}_\mathcal{T})$ is feasible for $IP(\mathcal{G}_\mathcal{T})$. Then, the remaining too-short arcs are removed and all storage costs are set to their original values. Thus, $\mathcal{X}_\mathcal{T}$ becomes a subgraph of $\mathcal{G}_\mathcal{T}$. Finally, $IP(\mathcal{X}_\mathcal{T})$ is solved and the solution is returned. In this chapter, we use a commercial software to solve $IP(\mathcal{X}_\mathcal{T})$. In Chapter 6, we introduce a solution algorithm that integrates the heuristic with the Benders strategy described in Chapter 4.

5.3.2 Building the Initial Graph

When storage costs are not taken into account, it is possible to construct a graph $\mathcal{X}_\mathcal{T}$ such that the optimum of $IP(\mathcal{X}_\mathcal{T})$ is a lower bound for the optimum of $IP(\mathcal{G}_\mathcal{T})$ as in [Boland 2017]. Boland *et al.* [Boland 2017] state that the graph must contain the origin and destination of every commodity. Therefore, given a product, for any origin-destination path in $\mathcal{G}_\mathcal{T}$ there is an equivalent path in $\mathcal{X}_\mathcal{T}$ with a similar cost and passing through the same physical nodes. Without this property, there is no

Algorithm 5 Algorithm to build the initial graph $\mathcal{X}_\mathcal{T}$

$\mathcal{N}_\mathcal{T} \leftarrow \emptyset, \mathcal{A}_\mathcal{T} \leftarrow \emptyset, \mathcal{H}_\mathcal{T} \leftarrow \emptyset$
$\mathcal{N}_\mathcal{T} \leftarrow \{(i,0) : \forall i \in \mathcal{S} \cup \mathcal{W}\}$
$\mathcal{N}_\mathcal{T} \leftarrow \mathcal{N}_\mathcal{T} \cup \{(i,t) : \forall i \in \mathcal{C}, \forall t \in \mathcal{T}, \exists p \in \mathcal{P}, d_{ip}^t > 0\}$
Solve $LP(\mathcal{G}_\mathcal{T})$ to obtain an optimal solution $(\overline{x}, \overline{y})$
for $(i,t) \in \mathcal{N}_\mathcal{T} : \forall i \in \mathcal{S} \cup \mathcal{W}, \exists p \in \mathcal{P}, \overline{x}_{ij}^{pt} > 0$ **do**
    $\mathcal{N}_\mathcal{T} \leftarrow \mathcal{N}_\mathcal{T} \cup \{(i,t)\}$
end for
for $(i,t) \in \mathcal{N}_\mathcal{T}$ **do**
    for $(i,j) \in \mathcal{A} : i \neq j$ **do**
        $\mathcal{A}_\mathcal{T} \leftarrow \mathcal{A}_\mathcal{T} \cup \{((i,t),(j,t+t_{ij})) : t' = \max\{t'' | t'' \leq t + t_{ij}, (j,t'') \in \mathcal{N}_\mathcal{T}\}\}$
    end for
end for
for $(i,t) \in \mathcal{N}_\mathcal{T} : i \in \mathcal{W}$ **do**
    $\mathcal{H}_\mathcal{T} \leftarrow \mathcal{H}_\mathcal{T} \cup \{((i,t),(i,t')) : t' = \min\{t'' | t'' > t, (i,t'') \in \mathcal{N}_\mathcal{T}\}\}$
end for

guarantee that the $IP(\mathcal{X}_\mathcal{T})$ optimal value is a lower bound on the $IP(\mathcal{G}_\mathcal{T})$ optimal value.

In the LSNDP, we are dealing with demands. The destination and delivery time are known, but products can be delivered by several suppliers. If $\mathcal{X}_\mathcal{T}$ does not contain each supplier at every possible period, some origin-destination paths of $\mathcal{G}_\mathcal{T}$ are missing. When storage costs are ignored, the DDD algorithm is optimal for the LSNDP if the initial graph contains every node $(s,t) \in \mathcal{S}_\mathcal{T}$. However, the integer program $IP(\mathcal{X}_\mathcal{T})$ becomes too large and the computation effort of the algorithm becomes prohibitive.

Therefore, we decide to select only a subset of potential suppliers and periods $(s,t) \in \mathcal{S}_\mathcal{T}$. The composition of this subset is essential as it remains the same during the course of the algorithm. If too many potential suppliers are present, the algorithm becomes too time-consuming. Conversely, if too few potential suppliers are selected, the final solution can be very poor. It may even happen that the problem becomes unfeasible. The procedure presented in Algorithm 5 was designed to select the potential suppliers in the initial graph. There is no guarantee that an optimal solution may be found, but it does guarantee that the $\mathcal{X}_\mathcal{T}$ graph we build leads to a feasible solution.

The construction algorithm works as follows. Initially, the sets of nodes and of arcs are empty. First, we follow the same rules as in [Boland 2017]. A node is added for each supplier and warehouse at time 0. Also, a node (i,t) is added for each customer at each time t such that there is a demand for at least one product.

To enlarge the set of potential suppliers, we solve the linear relaxation of $IP(\mathcal{G}_\mathcal{T})$ and obtain a fractional solution $(\overline{x}, \overline{y})$. In this solution, we identify each supplier shipping a positive quantity of product. We suppose that these nodes are more likely to appear in the optimal solution of $IP(\mathcal{G}_\mathcal{T})$, and add them to $\mathcal{X}_\mathcal{T}$. We do the same for warehouses, identifying each warehouse that ships a positive quantity

of product, and adding it to $\mathcal{X}_\mathcal{T}$.

The set of arcs is built as in [Boland 2017]. For each node (i,t) in $\mathcal{X}_\mathcal{T}$ and for each arc (i,j) in the static graph $\mathcal{G}$, we find the node (j,t') with the highest value t' such that $t' \leq t + t_{ij}$ and add the too-short arc $((i,t),(j,t'))$ to $\mathcal{A}_\mathcal{T}$. For each warehouse (i,t) in $\mathcal{H}_\mathcal{T}$ we find the node (i,t') with the smallest t' such that $t' > t$ and add the arc $((i,t),(i,t'))$ to $\mathcal{H}_\mathcal{T}$. In the initial graph, all storage costs are set to 0. In the remaining of the chapter, a storage arc with a cost fixed at 0 will be called a free-storage arc.

Adding these nodes and arcs to $\mathcal{X}_\mathcal{T}$ ensures that any demand can be satisfied, making $IP(\mathcal{X}_\mathcal{T})$ feasible. After the construction of $\mathcal{X}_\mathcal{T}$, the linear relaxation $LP(\mathcal{X}_\mathcal{T})$ is solved. Since $\mathcal{X}_\mathcal{T}$ contains too-short arcs, the obtained solution may be unfeasible for $LP(\mathcal{G}_\mathcal{T})$. Moreover, the $LP(\mathcal{X}_\mathcal{T})$ solution may include free-storage arcs. The next two phases of the algorithm address these issues by expanding and repairing $\mathcal{X}_\mathcal{T}$.

5.3.3 Eliminating Too-Short Arcs

Once $LP(\mathcal{X}_\mathcal{T})$ is solved, the solution may not be feasible for $LP(\mathcal{G}_\mathcal{T})$. This occurs when a non-zero flow is present on at least one too-short arc. Also, the cost of the solution is underestimated if non-zeo flow is present on at least one free-storage arc of $\mathcal{X}_\mathcal{T}$. In that case, a two-phase repair procedure is called. The first phase described in this subsection deals with the too-short arcs while the second phase detailed in the next subsection considers the free-storage arcs.

In the first phase, all too-short arcs of $\mathcal{X}_\mathcal{T}$ with non-zero flows are refined and $LP(\mathcal{X}_\mathcal{T})$ is solved again. In [Boland 2017], the authors developed a refining method such that the resulting $\mathcal{X}_\mathcal{T}$ satisfies the hypothesis that the value of the $IP(\mathcal{X}_\mathcal{T})$ solution is a lower bound of $IP(\mathcal{G}_\mathcal{T})$. Repairing $\mathcal{X}_\mathcal{T}$ consists in replacing a too-short arc $((i,t),(j,t'))$ by the arc $((i,t)(j,t+t_{ij}))$. Therefore to repair the graph, Boland *et al.* add $(j,t+t_{ij})$ to $\mathcal{N}_\mathcal{T}$ and create/delete arcs to take into account this new node. We adapt this method to the LSNDP.

Algorithm 6 is applied to remove the too-short arcs appearing in the optimal solution of $LP(\mathcal{X}_\mathcal{T})$. We refine each too-short arc $((i,t),(j,t'))$ with a non-zero flow. First, we delete $((i,t),(j,t'))$. To insert the arc with the original cost, we need to add a new time-point $(j,t+t_{ij})$ to $\mathcal{N}_\mathcal{T}$. Since suppliers have no in-coming arcs, node j is either a warehouse or a customer. However, customers cannot store or ship products to another location, and $\mathcal{X}_\mathcal{T}$ already contains every time points (i,t) with $i \in \mathcal{C}$ with a positive demand. Thus if j is a customer, adding $(j,t+t_{ij})$ to $\mathcal{N}_\mathcal{T}$ makes no sense. In that case, the refining mechanism stops here.

If j is a warehouse, we add $(j,t+t_{ij})$ to $\mathcal{N}_\mathcal{T}$, and we modify arcs of $\mathcal{X}_\mathcal{T}$ as in [Boland 2017]. To enable the storage of products through $(j,t+t_{ij})$, we must add storage arcs connecting $(j,t+t_{ij})$ to adjacent occurrences of j. We add an arc from the immediately preceding occurrence (j,t_1) to $(j,t+t_{ij})$ with $t_1 < t_{ij}$ such that $\nexists (j,t) \in \mathcal{N}_\mathcal{T}, t_1 < t < t_{ij}$. We also add an arc from $(j,t+t_{ij})$ to the immediately following occurrence (j,t_2) with $t_{ij} < t_2$ such that $\nexists (j,t) \in \mathcal{N}_\mathcal{T}, t_{ij} < t < t_2$.

Algorithm 6 Refining procedure

Require: Arc $((i,t)(j,t')) \in \mathcal{A}_\mathcal{T}$
  $\mathcal{A}_\mathcal{T} \leftarrow \mathcal{A}_\mathcal{T} \setminus \{((i,t)(j,t'))\}$
  if $j \notin \mathcal{C}$ **then**
    $\mathcal{N}_\mathcal{T} \leftarrow \{(j,t_{new}) : t_{new} = t + t_{ij}\}$
    $\mathcal{H}_\mathcal{T} \leftarrow \mathcal{H}_\mathcal{T} \setminus \{((j,t')(j,t'')) :: t'' = \min\{t | t > t', (j,t) \in \mathcal{N}_\mathcal{T}\}\}$
    $\mathcal{H}_\mathcal{T} \leftarrow \mathcal{H}_\mathcal{T} \cup \{((j,t')(j,t_{new}))\}$
    $\mathcal{H}_\mathcal{T} \leftarrow \mathcal{H}_\mathcal{T} \cup \{((j,t_{new})(j,t''))\}$
    for $(l,t)(j,t')) \in \mathcal{A}_\mathcal{T}$ **do**
      if $t_{new} \leq t + t_{lj}$ **then**
        $\mathcal{A}_\mathcal{T} \leftarrow \mathcal{A}_\mathcal{T} \setminus \{(l,t)(j,t'))\}$
        $\mathcal{A}_\mathcal{T} \leftarrow \mathcal{A}_\mathcal{T} \cup \{((l,t)(j,t_{new}))\}$
      end if
    end for
    for $(j,l) \in \mathcal{A}$ **do**
      $\mathcal{A}_\mathcal{T} \leftarrow \mathcal{A}_\mathcal{T} \cup \{(j,t_{new})(l,t')) : t' = \max\{t | t \leq t_{new} + t_{ij}, (j,t) \in \mathcal{N}_\mathcal{T}\}\}$
    end for
  end if

Algorithm 7 Storage costs updating procedure

Require: Solution $(\overline{x}, \overline{y})$ to $LP(\mathcal{X}_\mathcal{T})$
  for $((i,t),(i,t')) \in \mathcal{H}_\mathcal{T}$ with null cost in $\mathcal{X}_\mathcal{T}$ **do**
    if $\exists p \in \mathcal{P} : x_{ii}^{pt} > 0$ **then**
      Set the cost of $((i,t),(i,t'))$ to the initial value in $\mathcal{X}_\mathcal{T}$;
    end if
  end for

Obsolete storage arc $((j,t_1),(j,t_2))$ is then deleted and replaced by $((j,t_1),(j,t+t_{ij}))$ and $((j,t+t_{ij}),(j,t_2))$.

Then, transportation arcs are updated. We replace the deleted too-short arc $((i,t),(j,t'))$ by $((i,t)(j,t+t_{ij}))$. Also, for every transportation arc $((l,t),(j,t'))$, we check if the arc ending at the new time-point $((l,t),(j,t+t_{ij}))$ is a too-short arc. If it is the case, we delete $((l,t),(j,t'))$ and add $((l,t),(j,t+t_{ij}))$. Finally we include out-going arcs from the new time-point. For each arc (j,l) in the static network, we determine node (l,t') in $\mathcal{X}_\mathcal{T}$ with the largest t' such that $t+t_{ij}+t' \leq t+t_{ij}+t_{jl}$ and add the resulting too-short arc $((j,t+t_{ij})(l,t'))$ to $\mathcal{X}_\mathcal{T}$.

5.3.4 Updating Storage Costs

The first phase prevents the presence of too-short arcs in a $LP(\mathcal{X}_\mathcal{T})$ optimal solution. However, the solution may include free-storage arcs. In such a case, the storage costs are set to their initial values for these arcs and $LP(\mathcal{X}_\mathcal{T})$ is solved again. The process is repeated until the $LP(\mathcal{X}_\mathcal{T})$ solution does not includes free-storage

arcs. If the solution contains too-short arcs, the first phase is applied again, and the whole process is reiterated.

5.3.5 Stopping Criterion and Final Solution

The refining process of $\mathcal{X}_\mathcal{T}$ terminates when the solution obtained at the end of the second phase is feasible for $LP(\mathcal{G}_\mathcal{T})$. This occurs after a finite number of iterations. Indeed, the number of arcs to be lengthened in $\mathcal{X}_\mathcal{T}$ is finite and bounded by the number of transportation arcs in $\mathcal{G}_\mathcal{T}$. Similarly, the number of free-storage arcs is finite and bounded by the number of storage arcs in $\mathcal{G}_\mathcal{T}$.

As any $LP(\mathcal{G}_\mathcal{T})$ solution can be transformed into a feasible solution of $IP(\mathcal{G}_\mathcal{T})$ by rounding-up the fractional values, there are $IP(\mathcal{X}_\mathcal{T})$ solutions feasible for $IP(\mathcal{G}_\mathcal{T})$. However, at the end of the iterative process, $\mathcal{X}_\mathcal{T}$ may contain too-short arcs. These arcs are removed to ensure all $IP(\mathcal{X}_\mathcal{T})$ solutions are feasible for $IP(\mathcal{G}_\mathcal{T})$. In addition, $\mathcal{X}_\mathcal{T}$ may contain free-storage arcs. Thus, we set these arcs to their original cost. Finally, we solve $IP(\mathcal{X}_\mathcal{T})$ and obtain a suboptimal solution of $IP(\mathcal{G}_\mathcal{T})$.

5.4 Computational Study

In this section, we assess the efficiency of our heuristic through a computational study. We first describe the instances and how the study was performed. We next analyze results on easy and difficult instances.

5.4.1 Instances

The algorithm is tested on instances produced by the random generator presented in Chapter 3. As our objective is to demonstrate the scalability of our heuristic with respect to the size of the time expanded-graph, we generated new instances based on different parameters values.

Two sets of instances are considered: a set of *easy* instances and a set of *hard* instances. An *easy* instance is defined by $D = 15$ days, $|P| = 10$, $|\mathcal{G}| = \{10, 15, 20, 25, 30\}$, $\Delta = \{2, 3, 4\}$, and $\alpha = \{10, 20, 30\}$. There are 45 possible parameter combinations and 10 randomly instances are generated for each combination. Therefore, we have 450 *easy* instances. A *hard* instance is defined by $D = 30$ days, $|P| = 20$, $|\mathcal{G}| = \{60, 70, 80\}$, $\Delta = \{4, 6, 12, 24\}$, and $\alpha = \{10, 20, 30\}$. There are 36 possible parameter combinations and for each combination 10 randomly instances are generated. Therefore, we have 360 *hard* instances.

5.4.2 Setup of the Study

The performance of our heuristic algorithm is compared with the solution of the full model obtained with Gurobi 7. The MILP solver stops when a proven optimality gap of 0.1% at most is reached, when a CPU-time limit of 7200 seconds is reached or when the solver runs out of memory. When an optimal or near optimal solution

is not obtained, the best solution found is returned. We compute a performance rate between the full MILP solution and the heuristic solution using the formula :

$$\text{Rate} = \frac{\text{H}_{sol} - \text{MILP}_{sol}}{\text{MILP}_{sol}} \times 100.$$

where H_{sol} is the objective value obtained by the heuristic and MILP_{sol} is the objective value returned by Gurobi 7. For a given instance, a positive rate means that the industrial solver found a better solution than the proposed heuristic. This is the case when the solver succeeds in solving the full MILP model. A negative rate means that our heuristic identified a better solution than the solution of the full MILP, which can only occur when the solver does not find the optimal solution of the full model within the time limit or runs out of memory.

The algorithm was coded in C++ and executed on a cluster of 4 Intel Xeon E5-2695 processors with 16Gb under Linux 16.04. Linear and integer programs were solved using Gurobi 7.

5.4.3 Comparison with Optimal Solutions

First, the results on the set of *easy* instances are discussed. Gurobi 7 was able to solve 402 instances out of 450 within the time limit. We restrict our computational result analysis to these instances to compare the heuristic solution values with optimal ones. Figure 5.8 presents the distribution of the performance rate. For 65% of the instances, the rate is below 1% and it is below 3% for more than 90% of the instances. The average performance rate (Rate (%)), the heuristic computation

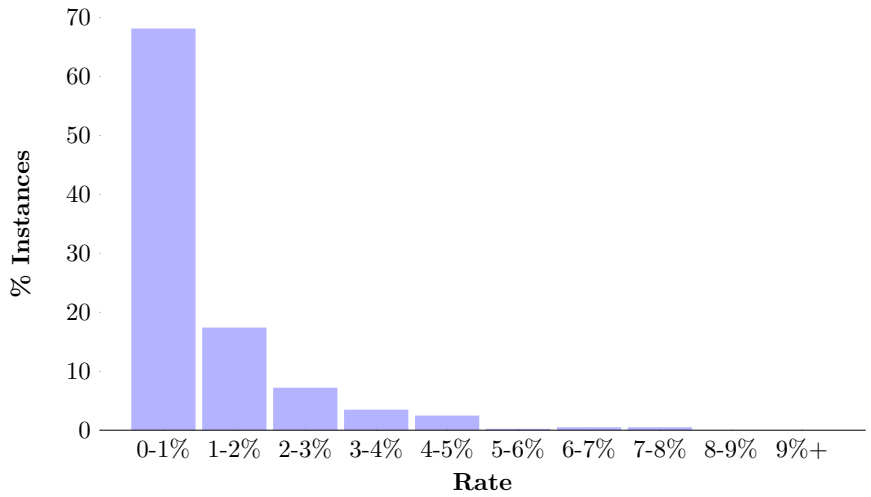

Figure 5.8: Rate distribution

times in seconds (H time (s)) and the full MILP computation times in seconds (MILP time (s)) are reported in Tables 5.1, 5.2 and 5.3. These tables aggregate the

rates and the computation times according to the graph cardinality $|\mathcal{G}|$, the time granularity Δ, and the connectivity radius α respectively.

$\lvert\mathcal{G}\rvert$	Rate (%)	H time (s)	MILP time (s)
10	0.46	0.20	7.34
15	0.37	0.41	17.14
20	0.81	0.92	598.86
25	2.10	2.11	882.17
30	1.06	7.58	1190.25

Table 5.1: Comparison between the proposed approach and the MILP solution for $|\mathcal{G}|$ varying from 10 to 30

Δ	Rate (%)	H time (s)	MILP time (s)
2	0.58	0.83	180.53
3	1.03	1.41	558.92
4	1.15	3.24	750.39

Table 5.2: Comparison between the proposed approach and the MILP solution for $\Delta = \{2, 3, 4\}$

α	Rate (%)	H time (s)	MILP time (s)
Low	0.56	0.77	300.77
Medium	1.08	0.93	507.32
High	1.14	4.67	684.25

Table 5.3: Comparison between the proposed approach and the MILP solution for α ranging from Low to High

Average rates do not exceed 2.10% regardless of the parameter chosen and its value. However, the parameter values have an impact on the computation time. In particular, full MILP computation times increase significantly when the number of nodes, the temporal granularity or the connectivity radius increases. The heuristic computation times also increase, but remain small. This computational time reduction is substantial for the largest values of each parameter. The results for *easy* instances indicate that the heuristic identifies high quality solutions with significant savings in computational times.

5.4.4 Performance on Difficult Instances

Now we turn to the set of *hard* instances. Gurobi 7 is not able to solve the full MILP to optimality for any instance. It only returns suboptimal solutions. Also,

$\|\mathcal{G}\|$	Rate (%)	H OOM	MILP OOM
60	-4.60	0.00%	0.83%
70	-5.79	0.00%	3.33%
80	-8.59	0.00%	7.5%

Table 5.4: Comparison between the proposed approach and the MILP solution for $|\mathcal{G}|$ varying from 60 to 80

note that for these instances neither the full MILP solver nor the heuristic manage to close the optimality gap within the time limit. We display the distributions (in deciles) of the average performance rates according to the number of nodes (Figure 5.9), the network density (Figure 5.10) and the time granularity (Figure 5.11).

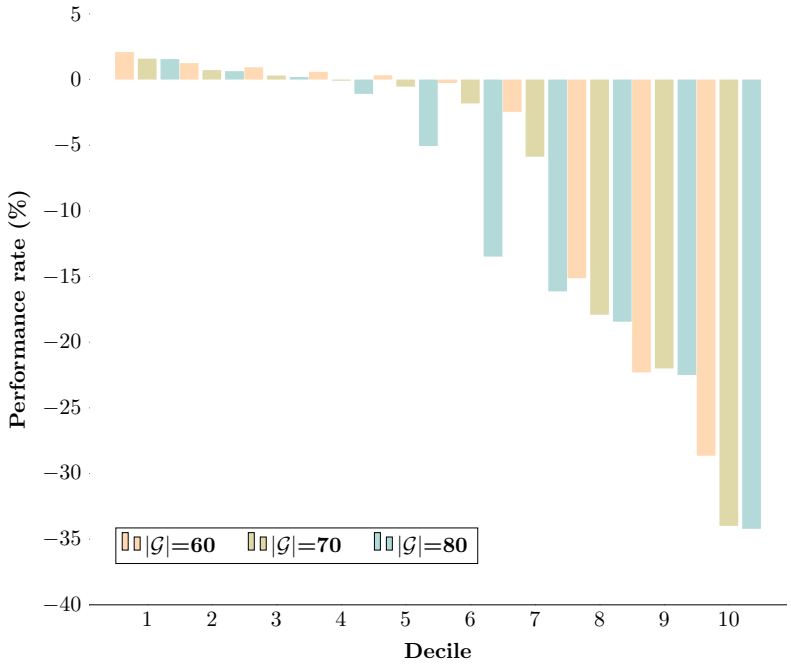

Figure 5.9: Performance rate distribution for $|\mathcal{G}|$.

We observe that whatever the parameter considered, distributions are negatively skewed. For the first deciles, the performance rates are around 0%, which means in the worst case the heuristic solution is very close to the full MILP solution. However, the heuristic significantly outperforms the full MILP solver in the last deciles. To better understand this behavior, we aggregate the instances according to the three parameters in Tables 5.4, 5.5 and 5.6. These tables report the average performance rates (Rate (%)), the percentage of runs for which the heuristic (H OOM (%)) and Gurobi (MILP OOM (%)) exceed the memory limit.

Except for $\Delta = 4$, the average performance rates are always negative. For every parameter, we observe that the average performance rate decreases when the

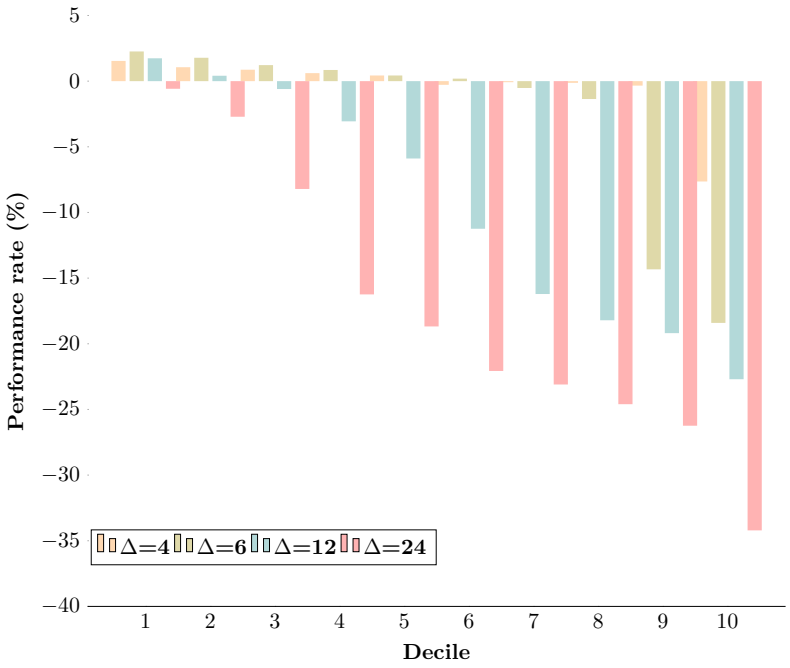

Figure 5.10: Performance rate distribution for Δ.

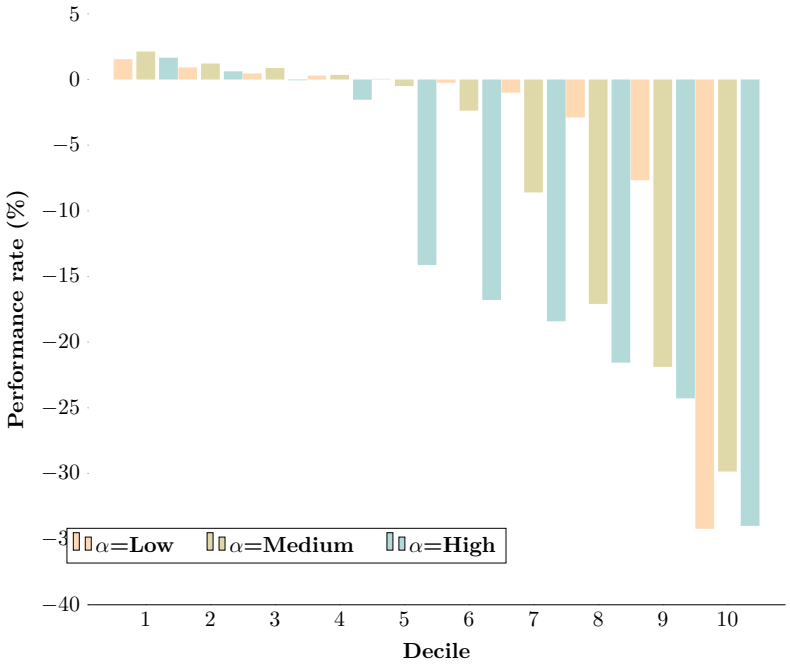

Figure 5.11: Performance rate distribution for α.

Δ	Rate (%)	H OOM	MILP OOM
4	0.35	0.00%	0.00%
6	-1.52	0.00%	0.00%
12	-8.17	0.00%	3.33%
24	-15.97	0.00%	12.22%

Table 5.5: Comparison between the proposed approach and the MILP solution for $\Delta = \{4, 6, 12, 24\}$

α	Rate (%)	H OOM	MILP OOM
Low	-2.27	0.00%	0.00%
Medium	-6.05	0.00%	1.67%
High	-10.66	0.00%	10.00%

Table 5.6: Comparison between the proposed approach and the MILP solution for α ranging from Low to High

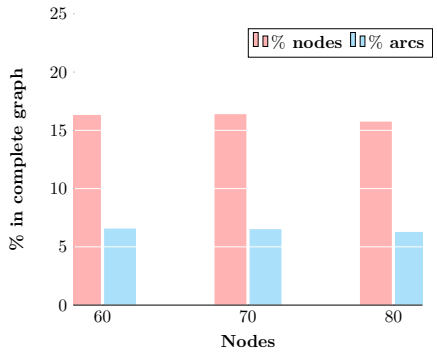

Figure 5.12: X_T relative size for $|\mathcal{G}|$

parameter value increases. One explanation is that Gurobi runs out of memory to solve the full MILP more frequently when the parameter value increases and is not able to identify feasible solutions. It should be noted that the heuristic is able to cope with larger instances and provide feasible solutions without running out of memory.

To better understand these results, we compare the numbers of nodes and arcs in the graphs X_T and G_T. Figures 5.12, 5.14 and 5.13 give the percentage of nodes and arcs of G_T present in X_T. We can see that X_T contains at most 25% of the nodes of G_T. More importantly, the percentage of arcs is very low as it remains lower than 10%. This is a desired behavior since the numbers of variables and constraints in $IP(X_T)$ increase according to the number of arcs in X_T.

The consequences of the reduction of the graph size can be observed in the percentages of variables and constraints of $IP(G_T)$ present in $IP(X_T)$. These values are reported in Figures 5.15, 5.16 and 5.17. The values are aggregated according to

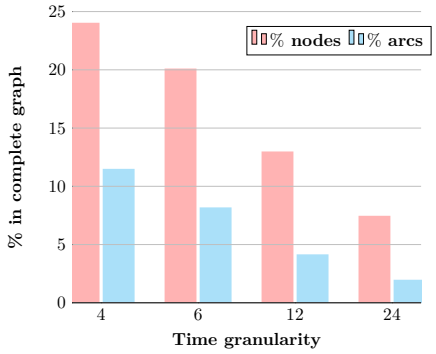

Figure 5.13: X_T relative size for Δ

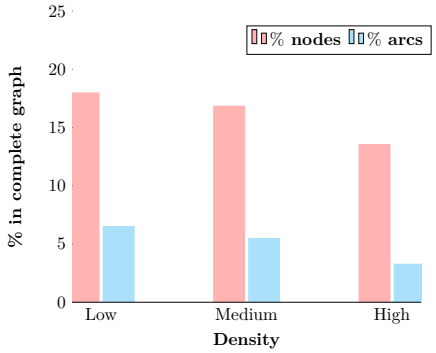

Figure 5.14: X_T relative size for α

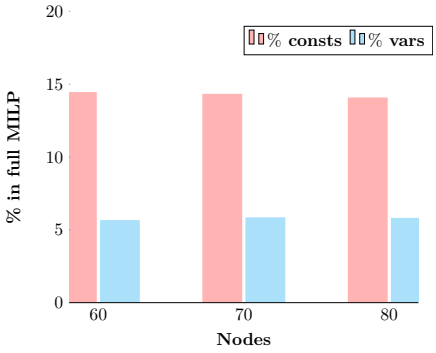

Figure 5.15: IP(X_T) relative size for $|\mathcal{G}|$

the graph cardinality $|\mathcal{G}|$, the value of α and the time granularity Δ respectively.

We observe that the integer programs solved by the heuristic are significantly smaller than the full MILP. In addition, as it can be expected from the graph size comparison, we can see that the higher the network density or temporal granularity, the smaller the relative size of IP(X_T). This observation suggests that the heuristic constructs sparser subgraphs as the network density/time granularity increase. As the network density/time granularity are linked to the shipment opportunities in the network, their increase makes it possible to find better solutions, at the expense

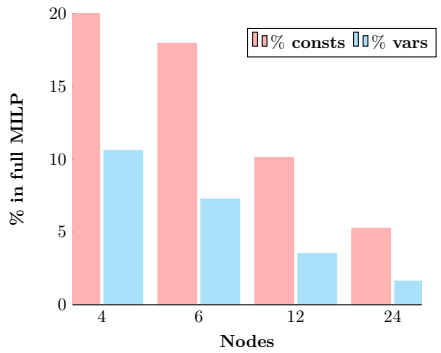

Figure 5.16: IP(X_T) relative size for Δ

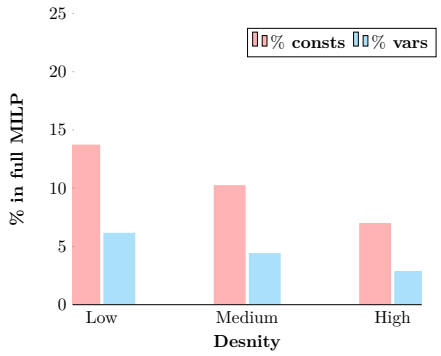

Figure 5.17: IP(X_T) relative size for α

of large, if not intractable models. This justifies the interest of such a heuristic to determine high quality transportation plans on an industrial scale.

5.5 Conclusions

In this chapter we proposed a DDD based algorithm for solving the LSNDP. Our heuristic builds a sparse subgraph, such that solving the associated LSNDP provides a high-quality solution, in a reasonable amount of time. To assess the effectiveness of our method, we compared it with Gurobi solver, on two set of instances.

Results on *easy* instances demonstrate our problem reduction is consistent, as the method computes solutions that are close to the optimum, in a very short amount of time. Results from the *hard* instances show that, for significant time-expanded graphs, the heuristic offers a strict improvement over the solutions obtained using a commercial solver. The heuristic builds subgraphs yielding mathematical models which are significantly smaller than the complete programs. In addition, this difference of magnitude increases when higher densities/time granularities are considered.

A logical perspective is to hybrid the DDD based heuristic with the Benders strategies described in Chapters 3 and 4. As both methods are effective to control

growing difficulty of the instances, with respect to distinct parameters, it appears interesting to combine them. In Chapter 6, we present such an hybrid method.

Chapter 6

Study of an Industrial Case

Contents

6.1 Introduction

In the previous chapters, we proposed solution algorithms for the LSNDP. To assess their effectiveness, these algorithms were tested on random instances related to the logistics operations presented in Chapter 1. In this chapter, we introduce industrial instances that reflect the actual logistics network of the chain of restaurants. Our purpose is twofold.

First, we seek to gain insight into the impact of extending DHL design policy with new shipment opportunities. To this end, we conduct computational experiments on instances based on a subset of the logistics network. In the benchmark instances, we elaborate transportation plans that respect DHL design policy. We also consider instances with extra shipment opportunities. We examine the effects of extending the design policy on the solution costs or the computational tractability.

Secondly, we examine whether or not the algorithms proposed in this thesis manage to scale-up with instances that reflect the whole logistics network. In Chapters 3 and 4, we proposed Benders strategies that overcome instances with significant number of products. In Chapter 5, we developped a DDD-based heuristic to manage the size of the time-expanded network. As the industry-sized instances

involve significant numbers of products, as well as very challenging time-expanded networks, we propose a hybrid matheuristic that combines the graph construction heuristic with the dynamic Benders strategy. The resulting matheuristic should be able to elaborate large-scale transportation plans.

The remainder of the chapter is organised as follows. In section 6.2, we describe the industrial instances. In section 6.3, we present the matheuristic. In section 6.4, we propose a computational study to evaluate the effectiveness of the matheuristic. We also discuss our findings about the effects of extending DHL design policy. Finally, we draw conclusions in section 6.5.

6.2 Industrial Instances

We first describe the data provided by DHL. Then, we explain how the industrial instances are generated. Finally, we present two sets of instances.

6.2.1 Data

The studied logistics network includes 4 stakeholders: the suppliers, the central warehouses, the regional warehouses and the restaurants. In total 460 sites are involved and distributed in four areas of France: North-West (NW), North-East (NE), South-West (SW) and South-East (SE). Table 6.1 shows the distribution of the sites in the different zones.

Table 6.1: Distribution of the logistics facilities

Area	Suppliers	Cent. Warehouses	Reg. Warehouses	Restaurants
NW	36	1	16	51
NE	91	1	9	87
SW	18	1	7	44
SE	32	1	8	57
Total	177	4	40	239

The distance and travel times between each pair of logistics facilities are derived from the shortest route found using Google Maps and considering a light traffic. The transportation cost between each pair of logistics facilities is calculated based on data from the National Road Committee (CNR) [Routier 2019]. These data state that the average transportation cost for a carrier is of 0.519 euro per kilometer. The cost for loading and unloading a pallet of product into a vehicle is 0.8 euros. DHL provides us with the storage cost of each warehouse. The average storage cost for a central warehouse is of 0.65 euro per pallet and per day. The average storage cost for a regional warehouses is of 1.5 euro per pallet and per day.

The chain of restaurants uses a thousand of types of products classified into the following families: frozen products, fresh products, dry products, beverages and non-food products. Product families are classified into two categories of products.

Table 6.2: Demand distribution per product family

			Demand distribution					
			Summer			Winter		
Families	Number of products	% of products	Low	Medium	High	Low	Medium	High
Frozen	315	80			x			x
		20			x	x		
Fresh	290	50			x			x
		30		x			x	
		20			x	x		
Dry	120	50			x			x
		50		x			x	
Beverages	165	50		x			x	
		50			x		x	
Non-Food	110	80	x			x		
		20		x			x	

Frozen and fresh products belong to the category of cold products, while dry products, beverages and non-food products belong to the category of ambient products. The transportation plans designed for ambient and cold products are distinct, as cold products require temperature control for food safety while ambient products do not.

DHL does not provide us with a detailed list of products, but with a distribution of the products in the different families. This distribution takes into account the seasonality of the demands, which varies from product to product. For example, the demand for ice cream tends to be high in summer and low during the rest of the year, while the demand for potatoes is high in winter and medium during the rest of the year. The data related to the products are summarized in Table 6.2.

Also, note that DHL provides us with the product families offered by each supplier. However, we do not have the detailed list of products proposed by each supplier. Finally, for each restaurant DHL provides us a with delivery schedule that indicates the times of the week when products are requested.

6.2.2 Instance Generation

We generated industrial instances based on these data. We proceeded as follows. First, we select a subset of suppliers, warehouses and customers in the logistics network. Similarly to DHL transportation operations, we build a transportation arc from each supplier to each central warehouse, from each central warehouse to each regional warehouse, and to each restaurant from its nearest regional warehouse. Extra transportation arcs are added based on a connectivity radius α. Specifically, a transportation arc is considered from each supplier to any regional warehouse or any restaurant located in a radius of α kilometers. Moreover, a transportation arc is added from each regional warehouse to any restaurant located in a radius of α kilometers. For each transportation arc, the travel time and costs are set based on the data described in the previous section.

At most three types of deliveries are present. In a *centralized delivery*, products are shipped from a supplier to a central warehouse, then to a regional warehouse, and finally to a restaurant. In an *indirect delivery*, products are shipped from a supplier to a regional warehouse, and then to a restaurant. Finally, *direct deliveries* model shipments from a supplier to a restaurant without transit through an intermediate facility. As explained in Chapter 1, DHL design policy only allows centralized deliveries. Thus, we consider instances with $\alpha = 0$ kilometers to elaborate transportation plans that respect DHL design policy. We refer to these instances as the *benchmark instances*. We also consider instances with non-null values for α. These instances correspond to *extended design policies* as they allow centralized deliveries as well as extra indirect and direct deliveries.

The temporal dimension is determined by the number of days in the planning horizon D, and, the time granularity Δ. As explained in Chapter 3, the time granularity expresses the number of time points per day in the time-expanded graph.

Each instance corresponds to a given category of products (i.e. ambient products or cold products) and a season (i.e. summer or winter). We randomly generate the list of products offered by each supplier, based on the product families it provides. Specifically, a supplier that specializes in a product family has a 25% chance to provide a product in that family. Finally, we generate the restaurant demands. For each restaurant, the product deliveries are requested at times that match the corresponding delivery schedule. The volume of each product demand is randomly chosen, based on the product demand seasonality and the season considered.

6.2.3 South-West Instances

We generate a first set of instances that corresponds to the south-western part of the logistics network. The transportation plan to elaborate is for the category of ambient products. Thus, we only consider suppliers specialized in dry products, beverages or non-food products. Figure 6.1 shows the resulting logistics network, which is composed of 67 sites, including 15 suppliers, one central warehouse, 7 regional warehouses and 44 restaurants.

We consider six values for the connectivity radius α: 0, 10, 20, 30, 40 and 50. Figure 6.2 shows the logistics network, with disks of 50 kilometers around the suppliers and regional warehouses. The number of days in the planning horizon is set to 15. We generate instances with a time granularity Δ of 1, 2, 3 and 4, leading to time intervals of respectively 24h, 12h, 8h and 6h for summer and winter seasons. Details on the 48 instances are given in Table 6.3. In Table 6.4 we report the number of transportation arcs between the logistics facilities according to the connectivity radius. $\mathcal{S}$, $\mathcal{W}_1$, $\mathcal{W}_2$ and $\mathcal{C}$ respectively denote the suppliers, the central warehouses, the regional warehouses and the restaurants.

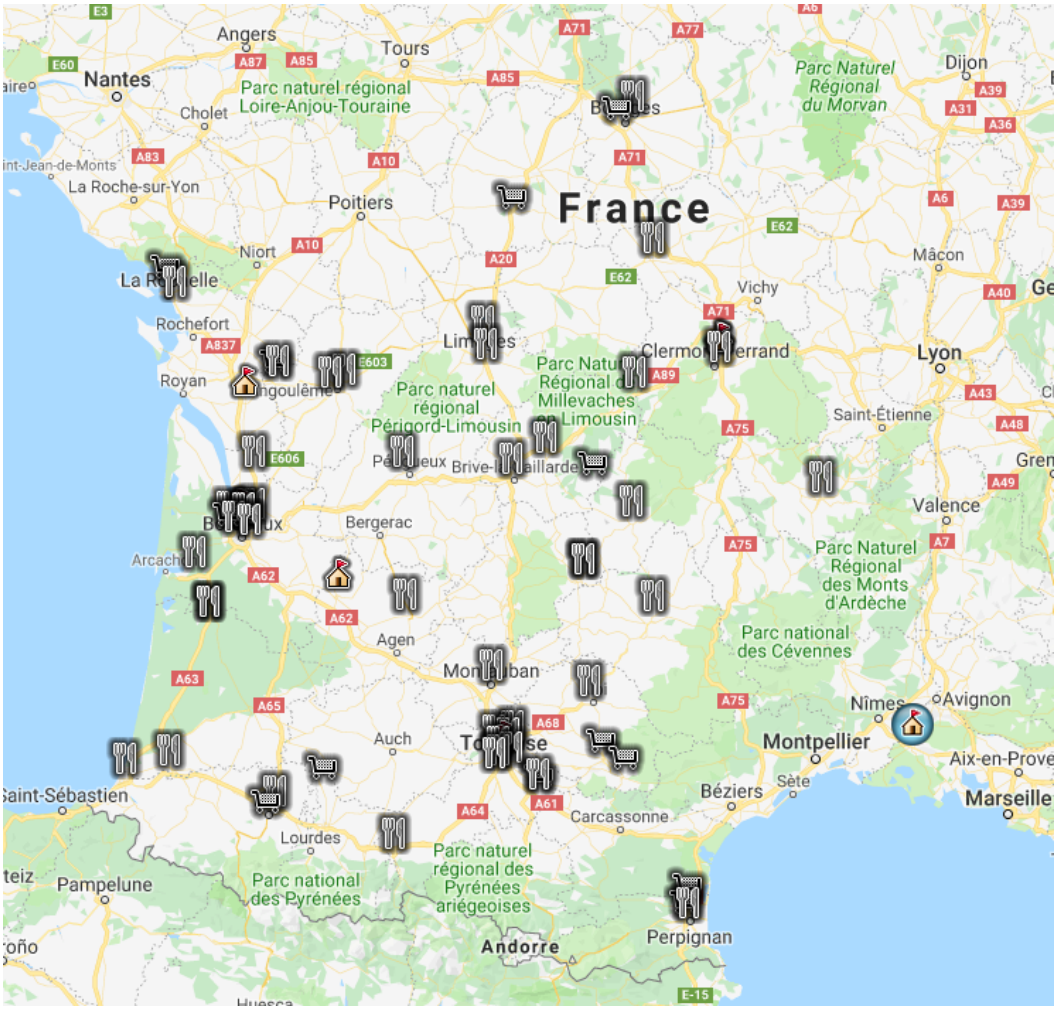

Figure 6.1: South-West region of the logistics network

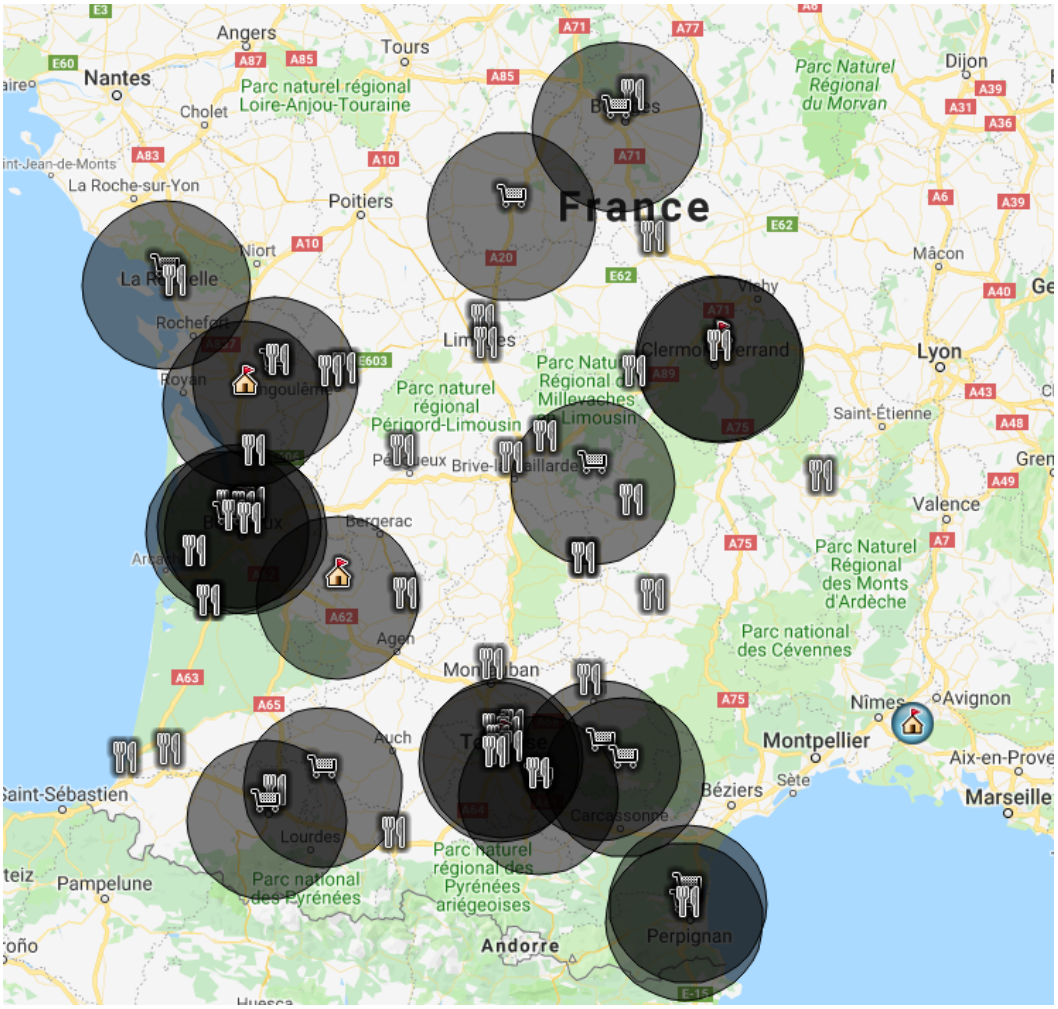

Figure 6.2: The South-West region with a connectivity radius $\alpha = 50$km

Table 6.3: Description of the South-West instances

Instance	Area	Category	α	Δ	Season
1	SW	Ambient	0	1	Summer
2	SW	Ambient	0	1	Winter
3	SW	Ambient	0	2	Summer
4	SW	Ambient	0	2	Winter
5	SW	Ambient	0	3	Summer
6	SW	Ambient	0	3	Winter
7	SW	Ambient	0	4	Summer
8	SW	Ambient	0	4	Winter
9	SW	Ambient	10	1	Summer
10	SW	Ambient	10	1	Winter
11	SW	Ambient	10	2	Summer
12	SW	Ambient	10	2	Winter
13	SW	Ambient	10	3	Summer
14	SW	Ambient	10	3	Winter
15	SW	Ambient	10	4	Summer
16	SW	Ambient	10	4	Winter
17	SW	Ambient	20	1	Summer
18	SW	Ambient	20	1	Winter
19	SW	Ambient	20	2	Summer
20	SW	Ambient	20	2	Winter
21	SW	Ambient	20	3	Summer
22	SW	Ambient	20	3	Winter
23	SW	Ambient	20	4	Summer
24	SW	Ambient	20	4	Winter

Instance	Area	Category	α	Δ	Season
25	SW	Ambient	30	1	Summer
26	SW	Ambient	30	1	Winter
27	SW	Ambient	30	2	Summer
28	SW	Ambient	30	2	Winter
29	SW	Ambient	30	3	Summer
30	SW	Ambient	30	3	Winter
31	SW	Ambient	30	4	Summer
32	SW	Ambient	30	4	Winter
33	SW	Ambient	40	1	Summer
34	SW	Ambient	40	1	Winter
35	SW	Ambient	40	2	Summer
36	SW	Ambient	40	2	Winter
37	SW	Ambient	40	3	Summer
38	SW	Ambient	40	3	Winter
39	SW	Ambient	40	4	Summer
40	SW	Ambient	40	4	Winter
41	SW	Ambient	50	1	Summer
42	SW	Ambient	50	1	Winter
43	SW	Ambient	50	2	Summer
44	SW	Ambient	50	2	Winter
45	SW	Ambient	50	3	Summer
46	SW	Ambient	50	3	Winter
47	SW	Ambient	50	4	Summer
48	SW	Ambient	50	4	Winter

Table 6.4: Distribution of the transportation arcs for South-West instances

α	$\mathcal{S}$->$\mathcal{W}_1$	$\mathcal{S}$->$\mathcal{W}_2$	$\mathcal{S}$->$\mathcal{C}$	$\mathcal{W}_1$->$\mathcal{W}_2$	$\mathcal{W}_2$->$\mathcal{C}$	Total
0	15	0	0	7	44	66
10	15	1	8	7	48	79
20	15	2	17	7	60	101
30	15	3	19	7	67	111
40	15	3	22	7	69	116
50	15	3	31	7	72	128

6.2.4 North-East Instances

We generate a second set of instances that corresponds to the north-eastern part of the logistics network. The transportation plan to elaborate is for the category of cold products. Thus, we only consider suppliers specialized in fresh or frozen products. Figure 6.3 shows the resulting logistics network, which is composed of 126 sites, including 29 suppliers, 1 central warehouses, 9 regional warehouses and 87 restaurants.

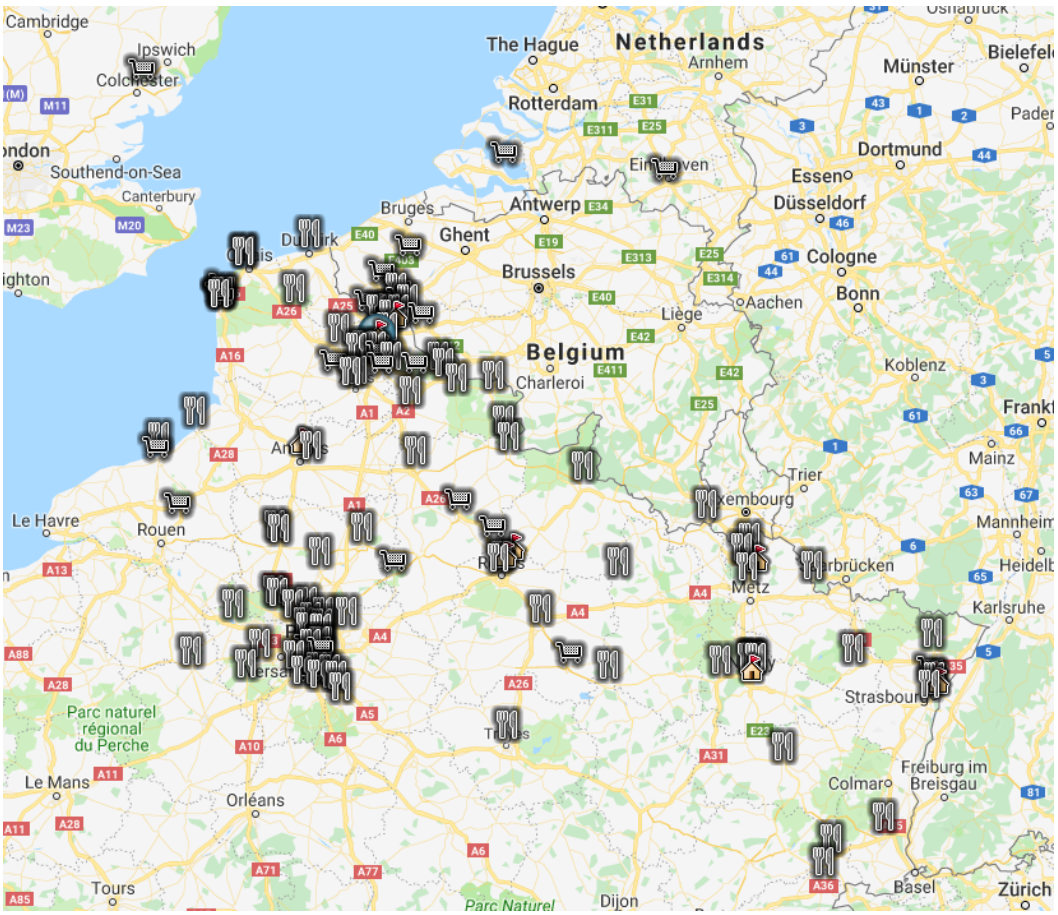

Figure 6.3: North-East region of the logistics network

We consider three values for the connectivity radius α: 0, 25 and 50. Figure 6.4 shows the logistics network, with disks of 50 kilometers around the suppliers and regional warehouses. The number of days in the planning horizon is set to 15. We generate instances with a time granularity Δ of 1, 2 and 4, leading to time intervals of respectively 24h, 12h and 6h for summer and winter seasons. Details on the 18 instances are given in Table 6.5. Table 6.6 gives the number of transportation arcs between the logistics facilities according to the connectivity radius.

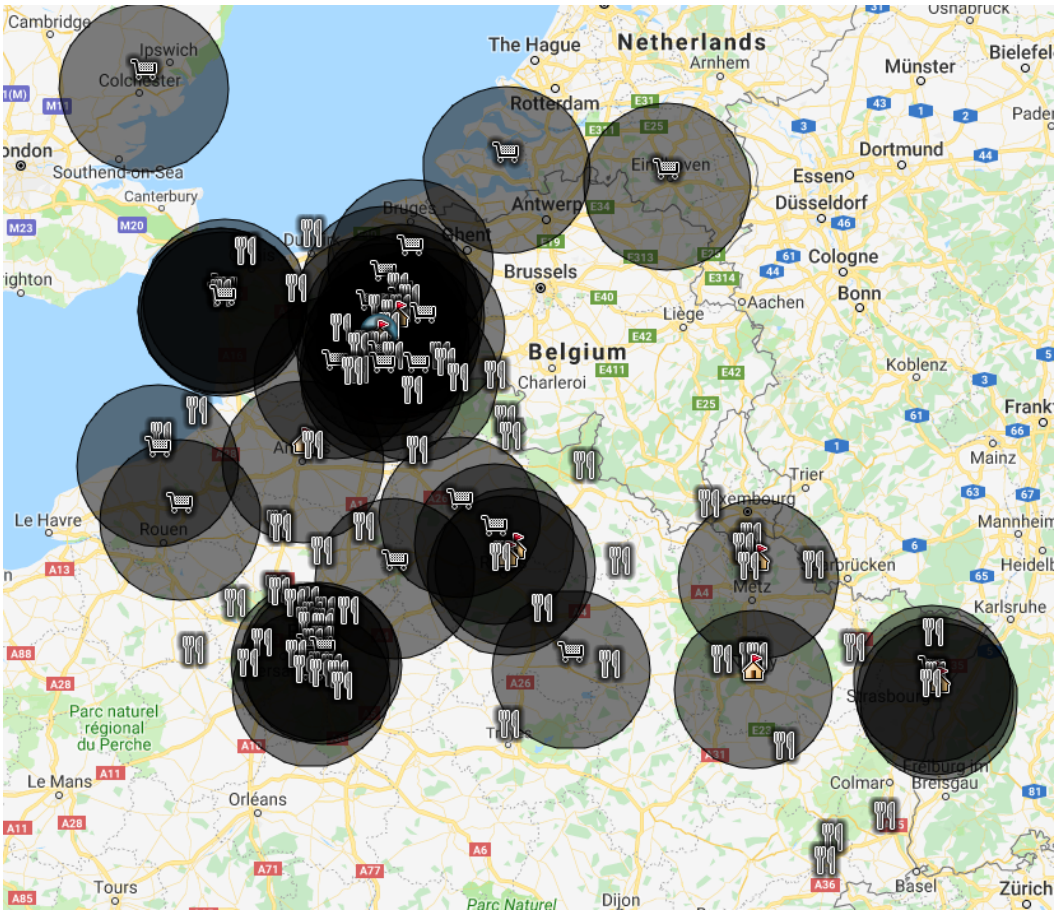

Figure 6.4: The North-East region with a connectivity radius $\alpha = 50$km

Table 6.5: Description of the North-East instances

Instance	Areas	Category	α	Δ	Season
1	NE	Cold	0	1	Summer
2	NE	Cold	0	1	Winter
3	NE	Cold	0	2	Summer
4	NE	Cold	0	2	Winter
5	NE	Cold	0	4	Summer
6	NE	Cold	0	4	Winter
7	NE	Cold	25	1	Summer
8	NE	Cold	25	1	Winter
9	NE	Cold	25	2	Summer
10	NE	Cold	25	2	Winter
11	NE	Cold	25	4	Summer
12	NE	Cold	25	4	Winter
13	NE	Cold	50	1	Summer
14	NE	Cold	50	1	Winter
15	NE	Cold	50	2	Summer
16	NE	Cold	50	2	Winter
17	NE	Cold	50	4	Summer
18	NE	Cold	50	4	Winter

Table 6.6: Distribution of the transportation arcs for North-East instances

α	$\mathcal{S}$->$\mathcal{W}_1$	$\mathcal{S}$->$\mathcal{W}_2$	$\mathcal{S}$->$\mathcal{C}$	$\mathcal{W}_1$->$\mathcal{W}_2$	$\mathcal{W}_2$->$\mathcal{C}$	Total
0	29	0	0	9	87	125
25	29	9	93	9	116	256
50	29	24	210	9	135	407

6.3 A Hybrid Matheuristic

The industrial instances involve a significant number of products, as well as considerable time-expanded networks. In Chapter 4, we proposed a dynamic Benders strategy that is effective for solving instances with a significant number of products. In Chapter 5, we presented a DDD-based heuristic that constructs a sparse time-expanded graph, and we demonstrated that solving the LSNDP defined over that time-expanded graph yields high-quality solutions with less computing effort. To overcome the size of the industrial instances, we propose a matheuristic that combines the graph construction heuristic with the dynamic Benders strategy. The flowchart in Figure 6.5 sketches the matheuristic.

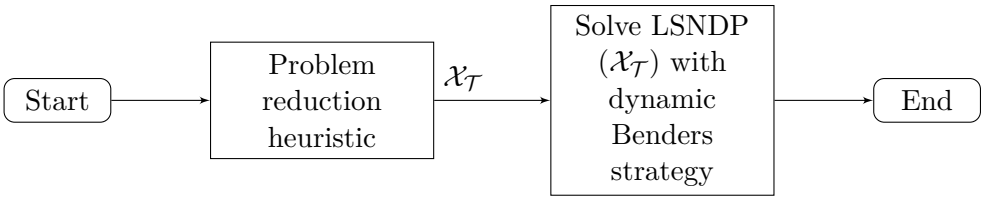

Figure 6.5: Principle of the matheuristic

We first build a sparse time-expanded graph $\mathcal{X}_\mathcal{T}$ that includes a subset of the nodes of the complete graph $\mathcal{G}_\mathcal{T}$, too-short transportation arcs, and free-storage arcs. Similarly to Chapter 5, we iteratively solve LP($\mathcal{X}_\mathcal{T}$) and refine the graph until we obtain a linear solution of the LSNDP in which neither short arcs nor free storage arcs appear. Then, the remaining too-short arcs are removed and all storage costs are set to their original values, such that the resulting graph is a subgraph of $\mathcal{G}_\mathcal{T}$. Last, we apply the dynamic partial Benders strategy to the LSNDP defined over the subgraph, and obtain a suboptimal solution of the original problem.

6.4 Computational Study

We solve the instances described in subsections 6.2.3 and 6.2.4 with two algorithms. The first algorithm is the dynamic Benders strategy presented in Chapter 4. The second algorithm is the matheuristic presented in section 6.3. All experiments are conducted on an Intel Xeon E5-2695 processor with 50 GB of memory under Linux 16.04. Linear and integer programs are solved using Cplex 12.7. Each run terminates if (a) a solution within the 1% optimality gap is found, or (b) the imposed CPU time limit of 24 hours is reached.

In this section, we assess the efficiency of our algorithms on industrial instances. We then analyze the results, and evaluate the impact of the design policy and the time discretization on the transportation plan costs.

Table 6.7: Computational results for South-West instances 1-24

		Dynamic Benders			Hybrid			
	Instance	Objective	Opt. gap	Time	Objective	Opt. gap	Time	Improv.
$\alpha = 0$	1	**125,711**	0.92%	125	126,127	0.76%	24	-0.33%
	2	**95,817**	0.81%	492	96,127	0.92%	32	-0.32%
	3	**117,501**	0.82%	2,226	119,038	0.51%	89	-1.29%
	4	**89,341**	0.97%	4,396	90,084	0.72%	74	-0.82%
	5	**114,724**	0.99%	14,557	117,657	0.95%	117	-2.49%
	6	**86,932**	0.98%	22,365	88,746	0.95%	246	-2.04%
	7	**113,174**	0.99%	64,852	116,857	0.99%	117	-3.15%
	8	**85,843**	0.97%	72,933	88,472	0.90%	316	-2.97%
$\alpha = 10$	9	**118,898**	1.64%	86,400	119,248	1.00%	86,400	-0.29%
	10	90,982	1.58%	86,400	**90,814**	1.11%	86,400	0.19%
	11	114,406	6.31%	86,400	**113,547**	2.36%	86,400	0.76%
	12	87,743	7.52%	86,400	**86,878**	3.55%	86,400	1.00%
	13	114,080	9.61%	86,400	**111,969**	3.76%	86,400	1.89%
	14	85,887	9.38%	86,400	**85,616**	3.79%	86,400	0.32%
	15	112,901	11.22%	86,400	**112,556**	3.81%	86,400	0.31%
	16	85,387	10.66%	86,400	**84,725**	2.57%	86,400	0.78%
$\alpha = 20$	17	118,250	2.94%	86,400	**117,379**	1.15%	86,400	0.74%
	18	89,661	3.21%	86,400	**88,790**	1.14%	86,400	0.98%
	19	112,896	5.90%	86,400	**111,725**	2.91%	86,400	1.05%
	20	86,412	7.95%	86,400	**85,016**	4.21%	86,400	1.64%
	21	112,612	10.09%	86,400	**110,945**	4.57%	86,400	1.50%
	22	85,939	12.72%	86,400	**83,926**	4.61%	86,400	2.40%
	23	111,935	11.34%	86,400	**109,364**	3.10%	86,400	2.35%
	24	84,833	11.66%	86,400	**84,020**	4.68%	86,400	0.97%

6.4.1 Performance of the Algorithms

To assess the performance of the matheuristic proposed in this chapter, we compare its results to those obtained with the dynamic Benders strategy. We first consider the results on the South-West instances and then on the North-East instances.

South-West Instances

Tables 6.7 and 6.8 correspond respectively to the South-West instances 1 to 24 and 25 to 48. For each method, we report the objective value of the obtained solution, the gap at termination and the computation time. Column Improv. displays the percentage of improvement of the matheuristic solution relative to that of the dynamic Benders strategy. A positive improvement indicates that the matheuristic found a solution with lower objective than the Benders strategy.

The first observation that emerges from the analysis of the results is that extending DHL design policy has a strong impact on the instance difficulty. The Benders strategy found an optimal solution within the time limit only for the instances that model the DHL design policy (instances 1 to 8). For the instances

Table 6.8: Computational results for South-West instances 25-48

		Dynamic Benders			Hybrid			
	Instance	Objective	Opt. gap	Time	Objective	Opt. gap	Time	Improv.
$\alpha = 30$	25	**112,563**	3.77%	86,400	112,766	4.73%	86,400	-0.18%
	26	**86,484**	3.77%	86,400	86,915	4.50%	86,400	-0.50%
	27	108,266	9.50%	86,400	**106,344**	2.70%	86,400	1.81%
	28	84,775	12.23%	86,400	**81,916**	4.27%	86,400	3.49%
	29	110,334	15.97%	86,400	**107,166**	6.23%	86,400	2.96%
	30	83,209	17.27%	86,400	**80,419**	5.52%	86,400	3.47%
	31	108,099	18.60%	86,400	**106,854**	5.10%	86,400	1.16%
	32	84,263	20.85%	86,400	**83,581**	8.47%	86,400	0.82%
$\alpha = 40$	33	112,666	3.61%	86,400	**112,472**	3.78%	86,400	0.17%
	34	87,020	5.23%	86,400	**86,545**	2.26%	86,400	0.55%
	35	110,475	13.02%	86,400	**107,174**	3.90%	86,400	3.08%
	36	84,995	14.28%	86,400	**81,476**	3.87%	86,400	4.32%
	37	108,811	14.57%	86,400	**106,510**	5.93%	86,400	2.16%
	38	84,295	19.06%	86,400	**80,048**	5.43%	86,400	5.31%
	39	107,965	18.91%	86,400	**107,689**	6.24%	86,400	0.26%
	40	82,604	17.90%	86,400	**81,690**	7.04%	86,400	1.12%
$\alpha = 50$	41	112,734	5,13%	86,400	**112,090**	2.06%	86,400	0.57%
	42	87,568	6,87%	86,400	**85,488**	1.90%	86,400	2.43%
	43	109,323	10,42%	86,400	**107,047**	4.79%	86,400	2.13%
	44	85,303	14,53%	86,400	**81,868**	5.94%	86,400	4.20%
	45	109,233	18,54%	86,400	**104,658**	5.30%	86,400	4.37%
	46	85,100	22,50%	86,400	**81,098**	7.52%	86,400	4.94%
	47	109,313	19,47%	86,400	**105,207**	5.49%	86,400	3.90%
	48	83,710	22,43%	86,400	**80,057**	6.27%	86,400	4.56%

based on extended design policies, the Benders strategy found suboptimal solutions only. In addition, the matheuristic converges within the time limit only for the first eight instances. For instances based on an extended design policy, the matheuristic does not solve to optimality the LSNDP defined over the subgraph.

The solution found by the Benders strategy is better than that of the metaheuristic for 11 instances out of 48. Among these instances are instances 1 to 8, for which the Benders strategy found a solution within the 1% optimality gap. For these instances, the percentages in the "Improv." column characterize the loss in the quality of the transportation plan by considering the heuristic subgraph instead of the complete graph.

The other instances for which the Benders strategy outperforms the matheuristic are instances with the coarser time discretization (instances 9, 25 and 26). For the 37 remaining instances, the matheuristic outperformed the Benders strategy. Overall, the percentage of improvement of the matheuristic solution relative to that of the dynamic Benders strategy equals 1.26%.

We now compare the matheuristic performance relative to that of the Benders strategy according to the size of the instances. In Table 6.9 we average the ob-

Table 6.9: Computational results according to α

	Dynamic Benders			Hybrid			
α	Objective	Opt. gap	Time	Objective	Opt. gap	Time	Improv.
0	**103,630**	0.93%	22,743	105,389	0.84%	153	-1.68%
10	101,286	7.39%	86,400	**100,669**	2.74%	86,400	0.62%
20	100,317	8.23%	86,400	**98,896**	3.30%	86,400	1.45%
30	97,249	12.80%	86,400	**95,745**	5.19%	86,400	1.63%
40	97,354	13.32%	86,400	**95,450**	4.81%	86,400	2.12%
50	97,786	14.99%	86,400	**94,689**	4.91%	86,400	3.39%

Table 6.10: Computational results according to Δ

	Dynamic Benders			Hybrid			
Δ	Objective	Opt. gap	Time	Objective	Opt. gap	Time	Improv.
1	103,196	3.42%	72,076	**102,897**	2.11%	72,023	0.33%
2	99,286	8.62%	72,592	**97,676**	3.31%	72,033	1.78%
3	98,430	12.64%	75,164	**96,563**	4.55%	72,047	2.06%
4	97,502	13.75%	83,546	**9,6756**	4.56%	72,051	0.84%

jectives, gaps at termination, primal gaps, computation times and percentages of improvement over instances with the same connectivity radius. In Table 6.10 we report the same outputs over instances with the same time granularity.

The matheuristic is more efficient than the Benders strategy for instances with extended design policies. In addition, the matheuristic has a better performance overall than the Benders strategy regardless of the time discretization considered. We also observe that the values in the "Improv." column increase monotonically with the connectivity radius. Therefore, for large-scale instances, it is more effective to solve the LSNDP defined over the sparse subgraph than to solve the complete program. That result validates the interest of combining the sparse graph construction heuristic with the dynamic Benders strategy for industrial matters.

It is worth noting that the industrial instances are more difficult to solve than the random instances. In Chapter 4, the dynamic Benders strategy is tested on random instances of similar scale, and yields an average gap at termination of 3.97%, after 1 hour and 30 minutes of computation. Here, the average final gap is of 9.60% with a time limit of 24 hour. This is due to the quality of the initial heuristic solution which provides a worse primal bound z_h for the industrial instances. For each instance we measure the improvement in the primal solution over that of the initial heuristic solution, by computing the primal gap:

$$\text{primal-gap}_{UB} = \frac{z_h - \text{UB}}{z_h} \times 100$$

Table 6.11: Computational results for North-East instances

		Dynamic Benders			Hybrid			
	Instance	Objective	Opt. gap	Time	Objective	Opt. gap	Time	Improv.
$\alpha = 0$	1	632,388	1.09%	86,400	**631,096**	0.81%	447	0.20%
	2	519,948	1.12%	86,400	**518,809**	0.84%	491	0.22%
	3	**537,070**	0.93%	2,665	537,679	0.88%	769	-0.11%
	4	443,255	1.36%	86,400	**440,991**	0.82%	935	0.51%
	5	499,765	2.94%	86,400	**489,544**	0.23%	2,163	2.09%
	6	402,922	1.11%	86,400	**401,775**	0.17%	1,618	0.29%
$\alpha = 25$	7	521,467	5.78	86,400	521,467	5.57%	86,400	0%
	8	429,715	6.61	86,400	**427,382**	5.18%	86,400	0.55%
	9	456,354	8.39	86,400	456,354	7.43%	86,400	0%
	10	377,673	8.79	86,400	377,673	8.16%	86,400	0%
	11	433,380	12.14	86,400	**428,980**	9.11%	86,400	1.03%
	12	362,504	13.51	86,400	**359,831**	10.92%	86,400	0.74%
$\alpha = 50$	13	447,697	14.56	86,400	447,697	13.04%	86,400	0%
	14	372,335	13.18	86,400	372,335	13.58%	86,400	0%
	15	398,870	18.3	86,400	398,870	15.41%	86,400	0%
	16	334,103	18.57	86,400	334,103	16.68%	86,400	0%
	17	385,315	23.08	86,400	385,315	18.47%	86,400	0%
	18	325,657	24.81	86,400	**318,349**	18.5%	86,400	2.30%

Where UB is the objective value of the final primal solution. For the random instances in Chapter 4, the dynamic Benders strategy yields an average primal gap of 3.82%. For the regional instances, the average primal gap equals 24.69%. As the average final gap obtained for the random instances is lower than the average final gap obtained for the industrial instances, we deduce that the initial heuristic solution is worse in the case of the industrial instances.

North-East Instances

We report the results corresponding to the North-East instances in Table 6.11.

The Benders strategy converged within the time limit for the third instance only. In addition, the Benders strategy only improved the initial heuristic solution for instances that correspond to the DHL design policy. For other instances, no better primal solution is found within the time limit. We observe that the matheuristic is more effective. For 5 out of the 6 instances with $\alpha = 0$, it computes a strictly better solution than the Benders strategy. When $\alpha = 25$, the matheuristic outperforms the Benders strategy and improves the initial heuristic solution for 3 instances out of 6. Finally, the matheuristic only improves the initial heuristic solution for one of the 6 last instances. These results show the limitations of our Benders strategy for solving large scale instances. Although the matheuristic yields a better performance, it also has difficulties to compute good transportation plans for the instances modelling extended design policies. Thus, we conclude that our solution algorithms still require enhancements to cope with instances that reflect the whole

logistics network.

We further investigate the impact of the design policy and the time discretization on the transportation plan costs. In the following subsections, we analyze the results for the SW instances. For each instance, we compare the solution obtained by the Benders strategy with that obtained by the matheuristic, and retain the transportation plan with lowest cost.

6.4.2 Value of extending DHL design policy

We first assess the impact of extending DHL design policy. Note that we obtained optimal solutions for all the benchmark instances, i.e. the instances based on DHL design policy, and suboptimal solutions for the instances based on the extended design policies. Therefore, in the following analysis we underestimate the potential of extending DHL design policy.

We report in Figure 6.6 the average cost savings on the transportation plan achieved by the enhanced design policies compared to the DHL design policy, for a planning horizon of 15 days.

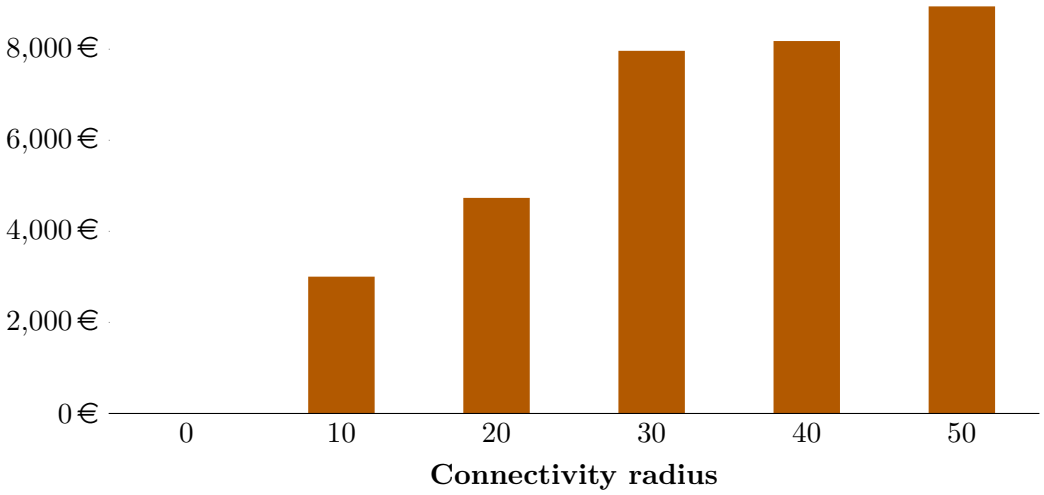

Figure 6.6: Overall cost savings

We observe that the average cost savings are positive for all enhanced design policies. This is not surprising since we extend the shipments opportunities to reduce the transportation plan overall cost. Indeed, increasing the number of delivery opportunities enables to enlarge the solution search space. When we consider a connectivity radius of 50 kilometers, the transportation plan average cost has a value of 94, 689 euros, against 103, 630 euros for the DHL current policy, thus generating savings of more than 8%.

In Figure 6.7, we investigate the distribution of these average cost savings, according to the different variables of our model. **Variable Transp. Savings**, **Inventory Savings** and **Fixed Transp. Savings** shows the average cost savings associated respectively with the transportation flow variables, the storage flow variables and the vehicle allocation variables.

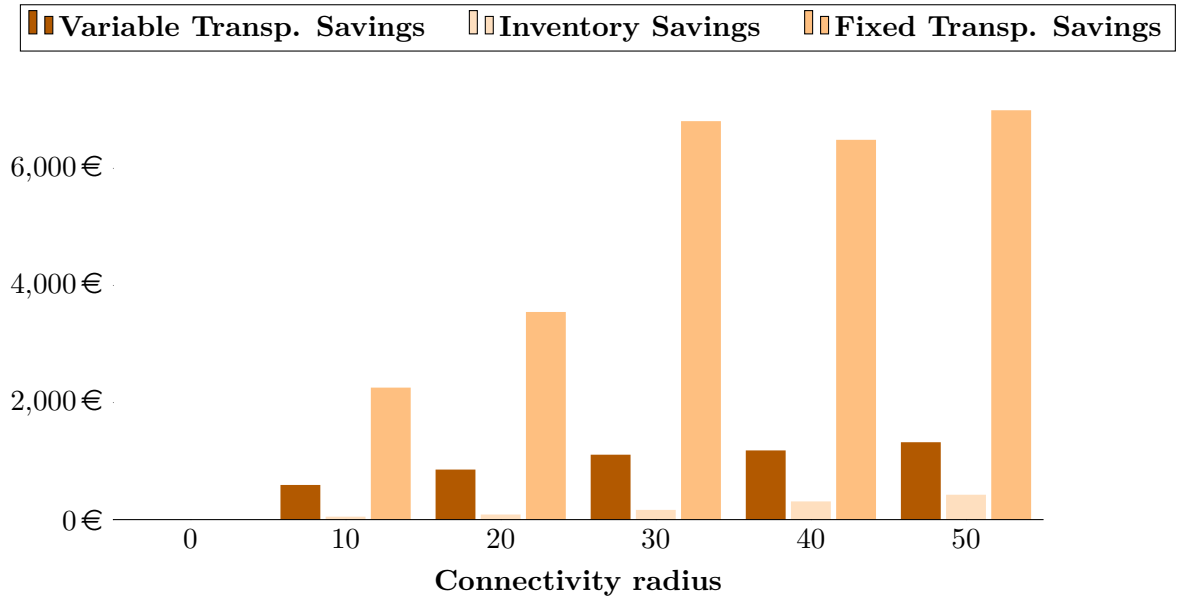

Figure 6.7: Cost savings per type variable

We observe that adding shipment opportunities to the DHL design policy enables costs reductions on transportation. Specifically, most savings are due to a better utilization of the vehicle fleet. This can be attributed to the fact that indirect and direct deliveries require less vehicles than centralized deliveries. In Figure 6.8, for each design policy we report the percentages of products delivered via centralized, indirect and direct paths.

This figure confirms the interest of adding new shipment opportunities to DHL design policy. Indeed, the initial rate of 100% of centralized deliveries decreases gradually as the connectivity radius increases. Although the centralized delivery remains the main option regardless of the design policy considered, this decrease of centralized deliveries indicates that the extra shipment opportunities are successfully exploited.

Even though increasing the number of shipment opportunities theoretically leads to better solutions, the decision-maker must limit the value of the connectivity radius. Indeed, the difficulty of an instance increases with the connectivity radius (see Table 6.9). Thus, to elaborate cost-effective transportation plans in a reasonnable time, it is necessary to limit the number of transportation arcs so as not to prevent the algorithm convergence.

6.4.3 Impact of the time discretization

We now study the impact of time granularity on the overall costs. In Figure 6.9 we average the overall costs of transportation plans over the instances with the same time granularity.

As expected, the choice of the time granularity impacts the overall costs of

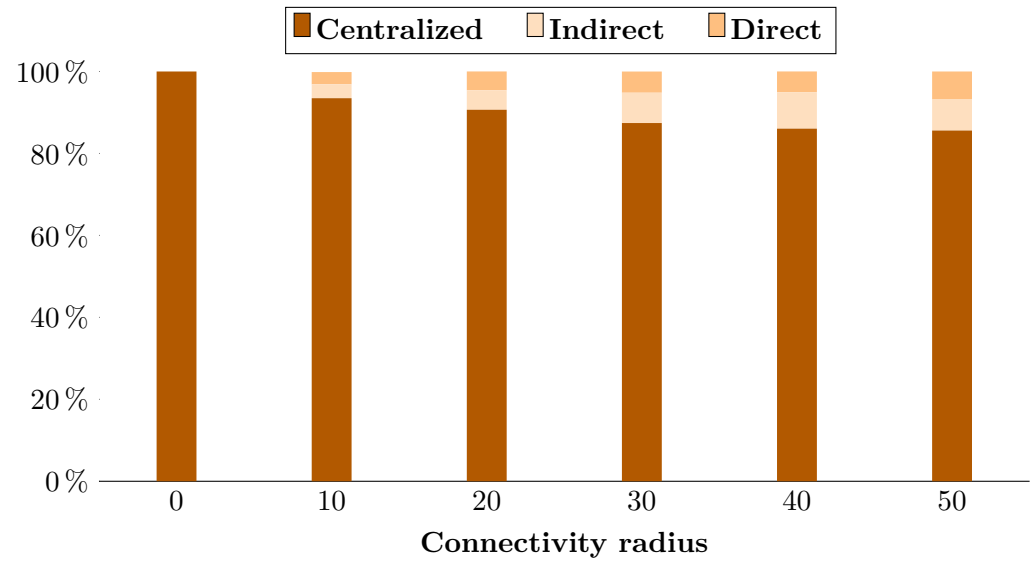

Figure 6.8: Percentages of centralized, indirect and direct deliveries

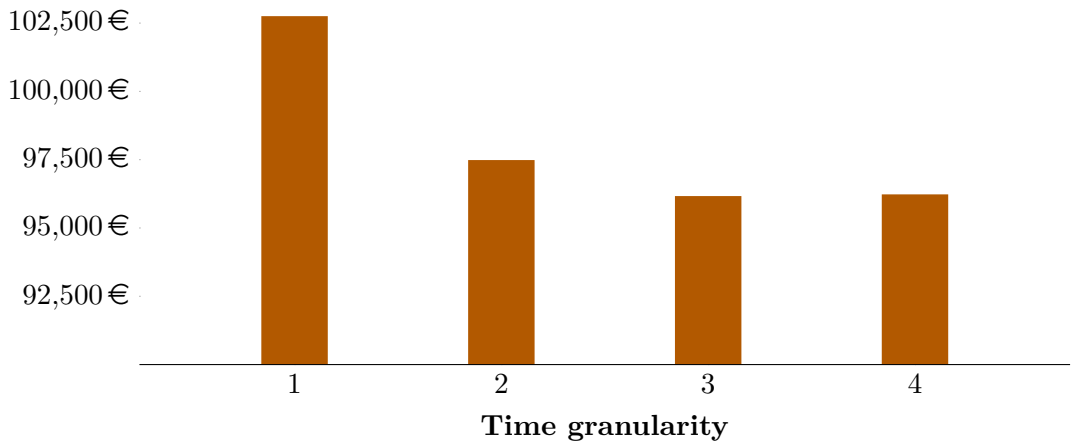

Figure 6.9: Overall costs

transportation plans. More specifically, a refined time granularity allows to compute more cost-effective solutions. With a time granularity value of 1 (i.e. time intervals of 24 hours), the transportation plans have an average overall cost of 102, 754 euros. On the other hand, considering a time granularity value of 4 (i.e. time intervals of 6 hours) enables to elaborate transportation plans with an average overall cost of 96, 230 euros, thus saving more than 6%. To get a better understanding of these savings, we examine the distribution of the costs. In Figure 6.10, we report the average variable transportation costs, inventory costs, and fixed transportation costs.

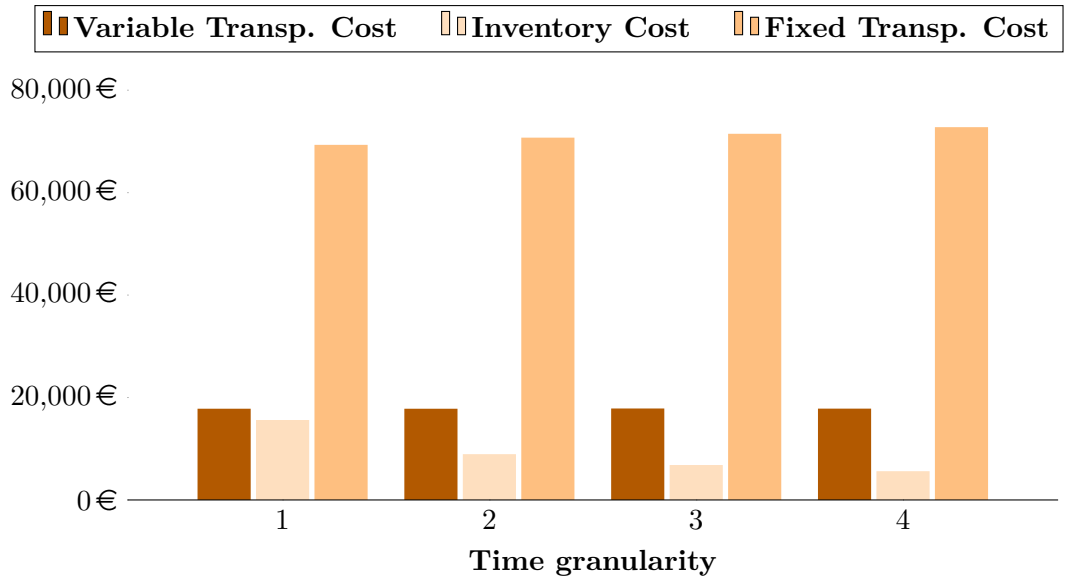

Figure 6.10: Cost per variable

We observe that time granularity does not significantly affect the transportation costs, i.e. the costs related to flow and truck variables. On the other hand, refining the time granularity generates substantial savings in terms of inventory costs. This outcome highlights the effects of a main difference between the LSNDP and the SNDP. In their study of the SNDP, Boland et al. [Boland 2018] exhibited the fact that partitionning the time horizon into long intervals can cause a major loss of quality on the transportation plan. Indeed, the SNDP consider commodities with availability and due dates. As the time windows between available and due dates can be tight, an inaccurate time discretization yields a SNDP model that prevents many feasible consolidation opportunities in the continous time problem. Thus, considering a coarse time discretization for the SNDP results in increased costs in terms of fleet utilization. In the LSNDP, as products do not have available dates, shipping dates are only constrained by the restaurants due dates. As a result, a coarse time discretization poorly affects the number of consolidation opportunities in the LSNDP, which explains why transportation costs are not impacted. However, considering a refined time granularity is advanteageous for the LSNDP, as it permits

to capture storage times more precisely, and reduce inventory costs.

Nevertheless the time granularity has, similarly to the connectivity radius, consequences on the instance difficulty (see Table 6.10). Short time intervals usually result in models that are computationally intractable. Thus, it is not surprising that found gaps increase when the time discretization is refined. Although refining the time discretization enables to reduce overall costs, this result points out that the selection of the time granularity must be the result of a compromise between solution quality and computational tractability.

6.5 Conclusions

In this chapter we introduced industrial instances based on the logistics network of a restaurant chain. To solve these instances, we presented a matheuristic that combines the sparse graph construction heuristic described in Chapter 5 with the dynamic Benders strategy described in Chapter 4.

In the computational study, we showed that extending DHL design policy with extra shipment opportunities enables to reduce the transportation plan overall cost. We also demonstrated that increasing the time granularity allows to gain significant savings, especially on inventory costs. On the other hand, we have seen that extending the design policy or refining the time discretization has a strong impact on the instance size.

We showed that as the instance difficulty increases, the matheuristic yields transportation plans that are more cost-effective than those obtained with the dynamic Benders strategy. We also have seen that the matheuristic has difficulties to solve large instances with extended design policies. This is a direct consequence of the increasing number of transportation arcs in the network. A perspective is to reconsider the way we extend the DHL design policy. Indeed, in this chapter we add a transportation arc from each supplier/warehouse to each warehouse/customer in a radius of alpha kilometers. As the connectivity radius grows, the number of transportation arcs in the network can be quite significant. Nevertheless, it is unliklely that all these transportation arcs contribute to the improvement of the transportation plan. To limit the instance sizes, an improvement would be to develop a solution algorithm that identifies a limited number of transportation arcs to add to the network.

Conclusion

The research presented in this thesis was conducted in partnership with DHL Supply Chain, a logistics solution provider. More specifically, we addressed a tactical transportation problem inspired by DHL's management of a French restaurant chain. We introduced a new problem for planning the transportation operations of a supply chain: the *Logistics Service Network Design Problem* (LSNDP). The aim of our research work was to develop optimization techniques able to provide high-quality solutions to the real-life instances faced by DHL. To this end, we developped solution algorithms that remain effective despite the scaling of two instance parameters: the number of products and the size of the underlying network. The main scientific contributions of this thesis were detailed in chapters 3, 4 et 5.

Contributions

In Chapter 3, we proposed a partial Benders decomposition algorithm for solving the LSNDP. In that decomposition, we strengthen the master problem by considering a super-product derived from the aggregation of the products. We proved the validity of this new master problem and computationally demonstrated the contribution of this aggregated information. To accelerate the convergence of our approach, we enriched the master with three cutset-based valid inequalities. We also integrated a heuristic that derives feasible primal solutions from unfeasible subproblems. The computational study shows the proposed Benders approach produces high-quality primal and dual solutions for large-scale instances. It also shows that the increase in the number of products has little effect on our method. This is mainly due to the fact that the master problem size is independent of the number of products considered.

In Chapter 4, we proposed a dynamic Benders strategy that extends our work from Chapter 3. In that approach, the aggregated information used to strengthen the master problem is refined at each iteration. This refinement is characterized by an increase in the number of super-products in the master. We proved that the Benders decomposition presented in Chapter 3 can be extended to multiple super-products obtained by partitioning the product set. We also demonstrated that the quality of the master problem depends on the partition of the product set. Thus, we introduced a metric to estimate whether or not a pair of product should be aggregated. Based on this metric, we implemented clustering strategies to effectively partition the product set at each main iteration of the algorithm. The results demonstrated the potential of our dynamic method compared with static Benders strategies.

The approaches presented in Chapters 3 and 4 are effective to solve instances with significant numbers of products. Nevertheless, real-life instances also involve extremely large time-expanded graphs. To be able to solve these instances, we pro-

posed in Chapter 5 a network reduction heuristic based on the Dynamic Discretization Discovery. Our heuristic iteratively constructs a subset of the original time-expanded graph, so that solving the LSNDP defined over the subgraph provides a feasible solution to the original problem. We showed that our heuristic generates subgraphs that are orders of magnitude smaller than the original time-expanded graphs. This seems to be particularly true for instances with a fine time discretization and/or many delivery opportunities. Computational experiments show that solving the reduced problem enables to significantly alleviate the computational burden, with a relatively small impact on the solution quality.

Finally, Chapter 6 introduced a matheuristic that combines the problem reduction heuristic with the dynamic Benders strategy. The matheuristic was tested on a set of real-life instances based on regions of the restaurant chain's logistics network. To assess the efficiency of our solution algorithm, we conducted an extensive computational study. Results reveal that our method effectively builds plans for the transportation operations of a supply chain composed of 67 actors, 390 products, over a planning horizon of 15 days discretized in time intervals of 6 hours. From a practical point of view, our experiments suggest that extending the DHL's design policy could yield to significant gains in the transportation plan overall costs.

Perspectives

The matheuristic presented in Chapter 6 proved to be effective for planning the transportation operations of a large-scale supply chain. But some works still need to be done to cope with instances that reflect the logistics network of the restaurant chain. A first perspective would be to readjust our solution algorithms to the industrial instances. The methods proposed in this thesis were designed with respect to random instances, and appear to be less effective in the case of industrial instances. By analyzing in-depth the structure of the industrial instances, we believe that we can identify specific features and exploit them to improve our solution algorithms.

We also see multiple avenues to improve the Benders approaches. We observed that we essentially generate feasibility cuts throughout the iterations. These feasibility cuts are standard, in the sense that they are obtained cutting off unbounded rays in the dual subproblem. However, it is well-known that standard feasibility cuts are not the most effective [Fischetti 2010]. By deriving combinatorial Benders cuts [Codato 2006] from infeasible vehicle allocations, we could considerably reduce the number of iterations of the algorithm. In the dynamic Benders strategy, there are improvements to be done regarding the information refinement process. The algorithm is initiated with a single super-product, but nothing prevents us to start with multiple super-products. Also, the number of super-products is always increased by one unit, whereas we could skip certain configurations. Ideally, the information refinement process should, according to the instance structure, evaluate the optimal number of super-products to consider at each iteration. In this sense, the values of the average matching rates with respect to the number of super-products could

provide clues as to how to proceed.

Finally, we could apply our research to other optimization problems. At first, we could focus on other transportation problems. We introduced a new master problem based on product aggregation that significantly improve the convergence of the Benders scheme. Although we used this idea in the context of the LSNDP, the technique could easily be applied to most network design problems that involve multiple products/commodities. As the number of products/commodities is clearly an obstacle to the solution of large-scale instances, we believe our approach has real relevance. In a larger perspective, the concept of strengthening the master problem with aggregated information that is refined in the course of the optimization process is transferable to all optimization problems. Therefore, a challenging - but not less exciting - avenue for future works would be to develop an abstract dynamic Benders scheme for solving general mixed-integer linear programs.

Appendix A: Validity of the inequalities

Theorem 5 *The proposed inequalities* (3.17)-(3.18) *are valid.*

Proof 5 *We first define the LSNDP over a time-expanded graph with super-sources. We demonstrate that the new problem is equivalent to the original one. Then, we demonstrate that any feasible solution for the new problem is equivalent to a solution for the* ***EMP*** *that respects* (3.17)-(3.18).

Let (x, y) *be a feasible solution for the LSND($\mathcal{G}_\mathcal{T}$). Let us extend* $\mathcal{G}_\mathcal{T}$ *similarly to what was done in section 3.3. For each* $p \in \mathcal{P}$*, we add a super-source* ss_p *to* $\mathcal{G}_\mathcal{T}$*. In addition, for each* $(s,t) \in \mathcal{S}_\mathcal{T}$ *such that* $p \in \mathcal{P}^s$*, we add a time-expanded arc* $((ss_p),(s,t))$ *with null linear cost and null fixed cost to* $\mathcal{G}_\mathcal{T}$*. We name the new time-expanded network as* $\mathcal{G}^+_\mathcal{T}$*. On each time-expanded arc* $((ss_p),(s,t)) \in \mathcal{G}^+_\mathcal{T}$*, let us define a continuous variable* $x^{pt}_{ss_p}$*. Let us add the following constraints to the LSND($\mathcal{G}^+_\mathcal{T}$):*

$$\sum_{((ss_p),(j,t))\in\mathcal{A}_\mathcal{T}} x^{pt}_{ss_pj} \geq D^p, \quad \forall p \in \mathcal{P} \tag{A.1}$$

$$\sum_{((i,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x^{ptt'}_{ij} - \sum_{((j,t'),(l,t''))\in\mathcal{A}_\mathcal{T}} x^{pt't''}_{jl} = 0, \quad \forall (j,t') \in \mathcal{S}_\mathcal{T} \tag{A.2}$$

We now extend a solution for the LSND($\mathcal{G}_\mathcal{T}$) to a solution for the LSND($\mathcal{G}^+_\mathcal{T}$). By construction, for each $(s,t) \in \mathcal{S}_\mathcal{T}$ *and for each* $p \in \mathcal{P}^s$*, the only arc of* $\mathcal{G}^+_\mathcal{T}$ *incoming to* (s,t) *such that a flow variable is defined for product* p *is* $((ss_p),(s,t))$*. Thus, for each* $(s,t) \in \mathcal{S}_\mathcal{T}$*, the only way to satisfy constraint* (A.2) *is to set the flow value of product* p *on arc* $((ss_p),(s,t))$ *to* $\sum_{((s,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x^{ptt'}_{sj}$*. As the original solution satisfies all customer demands, for each* $p \in \mathcal{P}$ *we have* $\sum_{(s,t)\in\mathcal{S}_\mathcal{T}} \sum_{((s,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x^{ptt'}_{sj} \geq D^p$*. Thus, the extended solution satisfies constraint* (A.1) *and is feasible for the LSND($\mathcal{G}^+_\mathcal{T}$).*

Each solution for the LSND($\mathcal{G}_\mathcal{T}$) admits a single corresponding solution for the LSND($\mathcal{G}^+_\mathcal{T}$). In addition, both solutions have identical objective values. Thus, the LSND($\mathcal{G}_\mathcal{T}$) is equivalent to the LSND($\mathcal{G}^+_\mathcal{T}$).

Let $(x,y)^+$ *be a feasible solution for the LSND($\mathcal{G}^+_\mathcal{T}$). Let* (x^χ, y, z) *be the solution for the* ***EMP*** *that replicates* $(x,y)^+$ *by an aggregation of flows. As* $(x,y)^+$ *respects constraints* (A.1) *and* (A.2)*, by construction* (x^χ, y, z) *respects constraints*

(3.17) *and* (3.18)*. Thus, constraints* (3.17) *and* (3.18) *do not cut off* (x^χ, y, z) *that replicates a feasible solution for the LSND*$(\mathcal{G}_\mathcal{T}^+)$*. Inequalities* (3.17) *and* (3.18) *is valid.*

Theorem 6 *The proposed inequality* (3.19) *is valid.*

Proof 6 *Let* (x, y) *be an optimal solution for the LSND*$(\mathcal{G}_\mathcal{T})$*. Let* $((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}$ *such that* $(i,t) \in \mathcal{S}_\mathcal{T}$ *and* $(j,t') \in \mathcal{C}_\mathcal{T}$*. For each product* $p \in \mathcal{P}^i$*,* $x_{ij}^{ptt'}$ *cannot be greater than* $d_{jt'}^p$*, as otherwise* (x, y) *would not be optimal for the LSND*$(\mathcal{G}_\mathcal{T})$*. As a result,* $\sum_{p\in\mathcal{P}^i} x_{ij}^{ptt'} \leq \sum_{p\in\mathcal{P}^i} d_{jt'}^p$*. Let* (x^χ, y, z) *be a solution for the* ***EMP*** *solution that replicates* (x, y) *by an aggregation of flows. By construction, for each* $((i,t),(j,t')) \in \mathcal{A}_\mathcal{T}$ *such that* $(i,t) \in \mathcal{S}_\mathcal{T}$ *and* $(j,t') \in \mathcal{C}_\mathcal{T}$*, we have* $x_{ij}^{\chi tt'} = \sum_{p\in\mathcal{P}^i} x_{ij}^{ptt'} \leq \sum_{p\in\mathcal{P}^i} d_{jt'}^p$*. Thus, constraint* (3.19) *does not cut off* (x^χ, y, z) *that replicates an optimal solution for the LSNDP. Inequality* (3.19) *is valid.*

Theorem 7 *The proposed inequality* (3.20) *is valid.*

Proof 7 *Let* (x, y) *be a feasible solution for the LSND*$(\mathcal{G}_\mathcal{T})$*. Let consider* $p \in \mathcal{P}$ *and* $t^* \in \mathcal{T}$ *such that* $\bar{D_{t^*}^p} > \bar{D_{t^*-1}^p}$*. Thus, there exists a customer* $(c, t^*) \in \mathcal{C}_\mathcal{T}$ *such that* $d_{ct^*}^p > 0$*.* $t_{ss_p}^{min}$ *is the smallest transit time between all supplier of product* p *and a customer for its product. Thus, the total amount of product* p *shipped from suppliers before or at time* $t^* - t_{ss_p}^{min}$ *must be greater or equal than* $\bar{D_{t^*}^p}$*, i.e.:*

$$\sum_{\substack{(s,t)\in\mathcal{S}_\mathcal{T} \\ t\leq t^*-t_{ss_p}^{min}}} \sum_{((s,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{sj}^{ptt'} \geq \bar{D_{t^*}^p}$$

As in Theorem 5, we extend (x, y) *and obtain a feasible solution* $(x, y)^+$ *for the LSND*$(\mathcal{G}_\mathcal{T}^+)$*. By construction, we have:*

$$\sum_{\substack{((ss_p),(s,t))\in\mathcal{A}_\mathcal{T} \\ t\leq t^*-t_{ss_p}^{min}}} {x_{ss_ps}^{pt}}^+ = \sum_{\substack{(s,t)\in\mathcal{S}_\mathcal{T} \\ t\leq t^*-t_{ss_p}^{min}}} \sum_{((s,t),(j,t'))\in\mathcal{A}_\mathcal{T}} x_{sj}^{ptt'} \geq \bar{D_{t^*}^p}$$

Let (x^χ, y, z) *be the* ***EMP*** *solution that replicates* $(x, y)^+$ *by an aggregation of flows. By construction, we have:*

$$\sum_{\substack{((ss_p),(s,t))\in\mathcal{A}_\mathcal{T} \\ t\leq t^*-t_{ss_p}^{min}}} x_{ss_ps}^{\chi t} = \sum_{\substack{((ss_p),(s,t))\in\mathcal{A}_\mathcal{T} \\ t\leq t^*-t_{ss_p}^{min}}} {x_{ss_ps}^{pt}}^+ \geq \bar{D_{t^*}^p}$$

Thus, constraint (3.20) *does not cut off* (x^χ, y, z) *that replicates a feasible solution for the LSND*$(\mathcal{G}_\mathcal{T}^+)$*. Inequality* (3.19) *is valid.*

Bibliography

[Adulyasak 2012] Yossiri Adulyasak, Jean-François Cordeau and Raf Jans. *Optimization-based adaptive large neighborhood search for the production routing problem.* Transportation Science, vol. 48, no. 1, pages 20–45, 2012. (Cited in page 27.)

[Adulyasak 2015] Yossiri Adulyasak, Jean-François Cordeau and Raf Jans. *The production routing problem: A review of formulations and solution algorithms.* Computers & Operations Research, vol. 55, pages 141–152, 2015. (Cited in pages 27 and 28.)

[Ales 2016] Zacharie Ales, Arnaud Knippel and Alexandre Pauchet. *Polyhedral combinatorics of the k-partitioning problem with representative variables.* Discrete Applied Mathematics, vol. 211, pages 1–14, 2016. (Cited in page 75.)

[Altiparmak 2009] Fulya Altiparmak, Mitsuo Gen, Lin Lin and Ismail Karaoglan. *A steady-state genetic algorithm for multi-product supply chain network design.* Computers & Industrial Engineering, vol. 56, no. 2, pages 521–537, 2009. (Cited in pages 23, 24, 27, and 28.)

[Amiri 2006] Ali Amiri. *Designing a distribution network in a supply chain system: Formulation and efficient solution procedure.* European journal of operational research, vol. 171, no. 2, pages 567–576, 2006. (Cited in pages 24, 27, and 28.)

[Andersen 2009] Jardar Andersen, Teodor Gabriel Crainic and Marielle Christiansen. *Service network design with asset management: Formulations and comparative analyses.* Transportation Research Part C: Emerging Technologies, vol. 17, no. 2, pages 197–207, 2009. (Cited in page 33.)

[Andersen 2011] Jardar Andersen, Marielle Christiansen, Teodor Gabriel Crainic and Roar Grønhaug. *Branch and price for service network design with asset management constraints.* Transportation Science, vol. 45, no. 1, pages 33–49, 2011. (Cited in pages 33 and 34.)

[Archetti 2011] Claudia Archetti, Luca Bertazzi, Giuseppe Paletta and M Grazia Speranza. *Analysis of the maximum level policy in a production-distribution system.* Computers & Operations Research, vol. 38, no. 12, pages 1731–1746, 2011. (Cited in page 27.)

[Azaron 2008] A Azaron, KN Brown, SA Tarim and M Modarres. *A multi-objective stochastic programming approach for supply chain design considering risk.* International Journal of Production Economics, vol. 116, no. 1, pages 129–138, 2008. (Cited in page 24.)

[Barnhart 2002] Cynthia Barnhart, Niranjan Krishnan, Daeki Kim and Keith Ware. *Network design for express shipment delivery.* Computational Optimization and Applications, vol. 21, no. 3, pages 239–262, 2002. (Cited in page 29.)

[Bashiri 2012] Mahdi Bashiri, Hossein Badri and Jafar Talebi. *A new approach to tactical and strategic planning in production–distribution networks.* Applied Mathematical Modelling, vol. 36, no. 4, pages 1703–1717, 2012. (Cited in pages 23, 27, and 28.)

[Beamon 1998] Benita M Beamon. *Supply chain design and analysis:: Models and methods.* International journal of production economics, vol. 55, no. 3, pages 281–294, 1998. (Cited in page 21.)

[Benders 1962] Jacques F Benders. *Partitioning procedures for solving mixed-variables programming problems.* Numerische mathematik, vol. 4, no. 1, pages 238–252, 1962. (Cited in page 38.)

[Berkhin 2006] Pavel Berkhin. *A survey of clustering data mining techniques.* In Grouping multidimensional data, pages 25–71. Springer, 2006. (Cited in page 76.)

[Bilgen 2010] Bilge Bilgen. *Application of fuzzy mathematical programming approach to the production allocation and distribution supply chain network problem.* Expert Systems with Applications, vol. 37, no. 6, pages 4488–4495, 2010. (Cited in page 26.)

[Boland 2017] Natashia Boland, Mike Hewitt, Luke Marshall and Martin Savelsbergh. *The continuous-time service network design problem.* Operations Research, vol. 65, no. 5, pages 1303–1321, 2017. (Cited in pages 33, 34, 88, 94, 95, and 96.)

[Boland 2018] Natashia Boland, Mike Hewitt, Luke Marshall and Martin Savelsbergh. *The price of discretizing time: a study in service network design.* EURO Journal on Transportation and Logistics, pages 1–22, 2018. (Cited in pages 16 and 126.)

[Bravo 2013] Juan José Bravo and Carlos Julio Vidal. *Freight transportation function in supply chain optimization models: A critical review of recent trends.* Expert Systems with Applications, vol. 40, no. 17, pages 6742–6757, 2013. (Cited in page 22.)

[Chandra 1994] Pankaj Chandra and Marshall L Fisher. *Coordination of production and distribution planning.* European Journal of Operational Research, vol. 72, no. 3, pages 503–517, 1994. (Cited in pages 26, 27, and 28.)

[Chouman 2003] Mervat Chouman, Teodor G Crainic and Bernard Gendron. A cutting-plane algorithm based on cutset inequalities for multicommodity capacitated fixed charge network design. Université de Montréal, Centre de recherche sur les transports, 2003. (Cited in pages 30 and 34.)

[Chouman 2016] Mervat Chouman, Teodor Gabriel Crainic and Bernard Gendron. *Commodity representations and cut-set-based inequalities for multicommodity capacitated fixed-charge network design.* Transportation Science, vol. 51, no. 2, pages 650–667, 2016. (Cited in page 30.)

[Chouman 2018] Mervat Chouman, Teodor Gabriel Crainic and Bernard Gendron. *The impact of filtering in a branch-and-cut algorithm for multicommodity capacitated fixed charge network design.* EURO Journal on Computational Optimization, vol. 6, no. 2, pages 143–184, 2018. (Cited in page 30.)

[Cintron 2010] Aixa Cintron, A Ravi Ravindran and Jose A Ventura. *Multi-criteria mathematical model for designing the distribution network of a consumer goods company.* Computers & Industrial Engineering, vol. 58, no. 4, pages 584–593, 2010. (Cited in pages 26, 27, and 28.)

[Codato 2006] Gianni Codato and Matteo Fischetti. *Combinatorial Benders' cuts for mixed-integer linear programming.* Operations Research, vol. 54, no. 4, pages 756–766, 2006. (Cited in pages 43 and 130.)

[Costa 2005] Alysson M Costa. *A survey on benders decomposition applied to fixed-charge network design problems.* Computers & operations research, vol. 32, no. 6, pages 1429–1450, 2005. (Cited in page 38.)

[Costa 2012] Alysson M Costa, Jean-François Cordeau, Bernard Gendron and Gilbert Laporte. *Accelerating benders decomposition with heuristic master problem solutions.* Pesquisa Operacional, vol. 32, no. 1, pages 03–20, 2012. (Cited in page 43.)

[Crainic 1986] Teodor G Crainic and Jean-Marc Rousseau. *Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem.* Transportation Research Part B: Methodological, vol. 20, no. 3, pages 225–242, 1986. (Cited in page 21.)

[Crainic 1997] Teodor Gabriel Crainic and Gilbert Laporte. *Planning models for freight transportation.* European journal of operational research, vol. 97, no. 3, pages 409–438, 1997. (Cited in page 21.)

[Crainic 2000a] Teodor Gabriel Crainic. *Service network design in freight transportation.* European journal of operational research, vol. 122, no. 2, pages 272–288, 2000. (Cited in pages 1, 2, 29, and 34.)

[Crainic 2000b] Teodor Gabriel Crainic, Michel Gendreau and Judith M Farvolden. *A simplex-based tabu search method for capacitated network design.* INFORMS journal on Computing, vol. 12, no. 3, pages 223–236, 2000. (Cited in page 30.)

[Crainic 2001] Teodor Gabriel Crainic, Antonio Frangioni and Bernard Gendron. *Bundle-based relaxation methods for multicommodity capacitated fixed charge network design.* Discrete Applied Mathematics, vol. 112, no. 1-3, pages 73–99, 2001. (Cited in pages 29, 30, and 33.)

[Crainic 2004] Teodor Gabriel Crainic, Bernard Gendron and Geneviève Hernu. *A slope scaling/Lagrangean perturbation heuristic with long-term memory for multicommodity capacitated fixed-charge network design.* Journal of Heuristics, vol. 10, no. 5, pages 525–545, 2004. (Cited in pages 31, 34, and 51.)

[Crainic 2014a] Teodor Gabriel Crainic, Mike Hewitt and Walter Rei. Partial decomposition strategies for two-stage stochastic integer programs. CIRRELT, 2014. (Cited in page 40.)

[Crainic 2014b] Teodor Gabriel Crainic, Mike Hewitt, Michel Toulouse and Duc Minh Vu. *Service network design with resource constraints.* Transportation Science, vol. 50, no. 4, pages 1380–1393, 2014. (Cited in page 33.)

[Crainic 2016] Teodor Gabriel Crainic, Walter Rei, Mike Hewitt and Francesca Maggioni. Partial benders decomposition strategies for two-stage stochastic integer programs, volume 37. CIRRELT, 2016. (Cited in pages 34, 38, and 40.)

[Dufour 2018] Émilie Dufour, Gilbert Laporte, Julie Paquette and Marie-Ève Rancourt. *Logistics service network design for humanitarian response in East Africa.* Omega, vol. 74, pages 1–14, 2018. (Cited in page 21.)

[Erengüç 1999] Ş Selçuk Erengüç, Natalie C Simpson and Asoo J Vakharia. *Integrated production/distribution planning in supply chains: An invited review.* European journal of operational research, vol. 115, no. 2, pages 219–236, 1999. (Cited in page 22.)

[Erera 2013] Alan Erera, Michael Hewitt, Martin Savelsbergh and Yang Zhang. *Improved load plan design through integer programming based local search.* Transportation Science, vol. 47, no. 3, pages 412–427, 2013. (Cited in pages 33 and 34.)

[Eskandarpour 2017] Majid Eskandarpour, Pierre Dejax and Olivier Péton. *A large neighborhood search heuristic for supply chain network design.* Computers & Operations Research, vol. 80, pages 23–37, 2017. (Cited in pages 24, 27, and 28.)

[Esmaeilikia 2016] Masoud Esmaeilikia, Behnam Fahimnia, Joeseph Sarkis, Kannan Govindan, Arun Kumar and John Mo. *Tactical supply chain planning models with inherent flexibility: definition and review.* Annals of Operations Research, vol. 244, no. 2, pages 407–427, 2016. (Cited in page 22.)

[Esnouf 2013] Catherine Esnouf, Marie Russel and Nicolas Bricas. Food system sustainability: insights from dualine. Cambridge University Press, 2013. (Cited in page 1.)

[Fahimnia 2013] Behnam Fahimnia, Reza Zanjirani Farahani, Romeo Marian and Lee Luong. *A review and critique on integrated production–distribution planning models and techniques.* Journal of Manufacturing Systems, vol. 32, no. 1, pages 1–19, 2013. (Cited in page 22.)

[Fazlollahtabar 2013] Hamed Fazlollahtabar, Iraj Mahdavi and Amir Mohajeri. *Applying fuzzy mathematical programming approach to optimize a multiple supply network in uncertain condition with comparative analysis.* Applied Soft Computing, vol. 13, no. 1, pages 550–562, 2013. (Cited in page 26.)

[Fischetti 2003] Matteo Fischetti and Andrea Lodi. *Local branching.* Mathematical programming, vol. 98, no. 1-3, pages 23–47, 2003. (Cited in page 31.)

[Fischetti 2010] Matteo Fischetti, Domenico Salvagnin and Arrigo Zanette. *A note on the selection of Benders' cuts.* Mathematical Programming, vol. 124, no. 1-2, pages 175–182, 2010. (Cited in pages 43 and 130.)

[Fontaine 2017] Pirmin Fontaine, Teodor Crainic, Walter Rei and Ola Jabali. Multimodal scheduled service network design with resource management for two-tier city logistics. CIRRELT, Centre interuniversitaire de recherche sur les réseaux d'entreprise, 2017. (Cited in pages 29 and 34.)

[Ford Jr 1958] Lester R Ford Jr and Delbert Ray Fulkerson. *Constructing maximal dynamic flows from static flows.* Operations research, vol. 6, no. 3, pages 419–433, 1958. (Cited in page 29.)

[Ford 1962] LR Ford and DR Fulkerson. *Flows in networks, Princeton Univ.* Press Princeton, 1962. (Cited in page 29.)

[Frangioni 2009] Antonio Frangioni and Bernard Gendron. *0–1 reformulations of the multicommodity capacitated network design problem.* Discrete Applied Mathematics, vol. 157, no. 6, pages 1229–1241, 2009. (Cited in page 34.)

[Gendron 1994] B Gendron and TG Crainic. *Relaxations for multicommodity network design problems.* Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada, 1994. (Cited in pages 30, 31, and 32.)

[Gendron 1996] B Gendron and TG Crainic. *Bounding procedures for multicommodity capacitated network design problems.* Publication crt-96-09, Centre de recherché sur les transports, University of Montreal, vol. 13, 1996. (Cited in pages 31 and 32.)

[Gendron 1999] Bernard Gendron, Teodor Gabriel Crainic and Antonio Frangioni. *Multicommodity capacitated network design.* In Telecommunications network planning, pages 1–19. Springer, 1999. (Cited in page 29.)

[Gendron 2018] Bernard Gendron, Saïd Hanafi and Raca Todosijević. *Matheuristics based on iterative linear programming and slope scaling for multicommodity capacitated fixed charge network design.* European Journal of Operational Research, vol. 268, no. 1, pages 70–81, 2018. (Cited in page 32.)

[Geoffrion 1974] Arthur M Geoffrion and Glenn W Graves. *Multicommodity distribution system design by Benders decomposition.* Management science, vol. 20, no. 5, pages 822–844, 1974. (Cited in pages 24, 27, and 28.)

[Ghamlouche 2003] Ilfat Ghamlouche, Teodor Gabriel Crainic and Michel Gendreau. *Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design.* Operations research, vol. 51, no. 4, pages 655–667, 2003. (Cited in pages 31, 32, and 34.)

[Ghamlouche 2004] Ilfat Ghamlouche, Teodor Gabriel Crainic and Michel Gendreau. *Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design.* Annals of Operations research, vol. 131, no. 1-4, pages 109–133, 2004. (Cited in pages 31, 32, and 34.)

[Hewitt 2010] Mike Hewitt, George L Nemhauser and Martin WP Savelsbergh. *Combining exact and heuristic approaches for the capacitated fixed-charge network flow problem.* INFORMS Journal on Computing, vol. 22, no. 2, pages 314–325, 2010. (Cited in pages 32 and 34.)

[Holmberg 1998] Kaj Holmberg and Johan Hellstrand. *Solving the uncapacitated network design problem by a Lagrangean heuristic and branch-and-bound.* Operations research, vol. 46, no. 2, pages 247–259, 1998. (Cited in page 30.)

[Holmberg 2000] Kaj Holmberg and Di Yuan. *A Lagrangian heuristic based branch-and-bound approach for the capacitated network design problem.* Operations Research, vol. 48, no. 3, pages 461–481, 2000. (Cited in pages 30 and 34.)

[Hugo 2005] André Hugo and Efstratios N Pistikopoulos. *Environmentally conscious long-range planning and design of supply chain networks.* Journal of Cleaner Production, vol. 13, no. 15, pages 1471–1491, 2005. (Cited in pages 23, 27, and 28.)

[Jain 1988] Anil K Jain, Richard C Dubes*et al.* Algorithms for clustering data, volume 6. Prentice hall Englewood Cliffs, 1988. (Cited in page 76.)

[Jarrah 2009] Ahmad I Jarrah, Ellis Johnson and Lucas C Neubert. *Large-scale, less-than-truckload service network design.* Operations Research, vol. 57, no. 3, pages 609–625, 2009. (Cited in pages 32 and 34.)

[Kim 1999a] Daeki Kim, Cynthia Barnhart, Keith Ware and Gregory Reinhardt. *Multimodal express package delivery: A service network design application.* Transportation Science, vol. 33, no. 4, pages 391–407, 1999. (Cited in page 29.)

[Kim 1999b] Dukwon Kim and Panos M Pardalos. *A solution approach to the fixed charge network flow problem using a dynamic slope scaling procedure.* Operations Research Letters, vol. 24, no. 4, pages 195–203, 1999. (Cited in page 51.)

[Kim 2000] Dukwon Kim and Panos M Pardalos. *Dynamic slope scaling and trust interval techniques for solving concave piecewise linear network flow problems.* Networks: An International Journal, vol. 35, no. 3, pages 216–222, 2000. (Cited in page 51.)

[Kim 2006] Dukwon Kim, Xinyan Pan and Panos M Pardalos. *An enhanced dynamic slope scaling procedure with tabu scheme for fixed charge network flow problems.* Computational Economics, vol. 27, no. 2-3, pages 273–293, 2006. (Cited in page 51.)

[Kopanos 2012] Georgios M Kopanos, Luis Puigjaner and Michael C Georgiadis. *Simultaneous production and logistics operations planning in semicontinuous food industries.* Omega, vol. 40, no. 5, pages 634–650, 2012. (Cited in pages 25, 27, and 28.)

[Lee 2008] Byung Ki Lee, Kyung Hwan Kang and Young Hoon Lee. *Decomposition heuristic to minimize total cost in a multi-level supply chain network.* Computers & Industrial Engineering, vol. 54, no. 4, pages 945–959, 2008. (Cited in pages 25, 27, and 28.)

[Lo 2013] Hong K Lo, Kun An and Wei-hua Lin. *Ferry service network design under demand uncertainty.* Transportation Research Part E: Logistics and Transportation Review, vol. 59, pages 48–70, 2013. (Cited in page 29.)

[Magnanti 1981] Thomas L Magnanti and Richard T Wong. *Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria.* Operations research, vol. 29, no. 3, pages 464–484, 1981. (Cited in pages 43 and 55.)

[Magnanti 1986] Thomas L Magnanti, Paul Mireault and Richard T Wong. *Tailoring Benders decomposition for uncapacitated network design.* In Netflow at Pisa, pages 112–154. Springer, 1986. (Cited in page 38.)

[Medina 2016] Juliette Medina. *Modèles et méthodes d'optimisation pour la mutualisation des chaînes logistiques.* PhD thesis, Ecole des Mines de Nantes, 2016. (Cited in page 27.)

[Melo 2006] M Teresa Melo, Stefan Nickel and F Saldanha Da Gama. *Dynamic multi-commodity capacitated facility location: a mathematical modeling framework for strategic supply chain planning.* Computers & Operations Research, vol. 33, no. 1, pages 181–208, 2006. (Cited in page 23.)

[Melo 2009] M Teresa Melo, Stefan Nickel and Francisco Saldanha-Da-Gama. *Facility location and supply chain management–A review.* European journal of operational research, vol. 196, no. 2, pages 401–412, 2009. (Cited in page 21.)

[Melo 2012] MT Melo, S Nickel and F Saldanha-da Gama. *A tabu search heuristic for redesigning a multi-echelon supply chain network over a planning horizon.* International Journal of Production Economics, vol. 136, no. 1, pages 218–230, 2012. (Cited in pages 23, 27, and 28.)

[Mula 2010] Josefa Mula, David Peidro, Manuel Díaz-Madroñero and Eduardo Vicens. *Mathematical programming models for supply chain production and transport planning.* European Journal of Operational Research, vol. 204, no. 3, pages 377–390, 2010. (Cited in page 22.)

[Rahmaniani 2017] Ragheb Rahmaniani, Teodor Gabriel Crainic, Michel Gendreau and Walter Rei. *The Benders decomposition algorithm: A literature review.* European Journal of Operational Research, vol. 259, no. 3, pages 801–817, 2017. (Cited in page 40.)

[Rei 2009] Walter Rei, Jean-François Cordeau, Michel Gendreau and Patrick Soriano. *Accelerating Benders decomposition by local branching.* INFORMS Journal on Computing, vol. 21, no. 2, pages 333–345, 2009. (Cited in page 43.)

[Rodríguez-Martín 2010] Inmaculada Rodríguez-Martín and Juan José Salazar-González. *A local branching heuristic for the capacitated fixed-charge network design problem.* Computers & Operations Research, vol. 37, no. 3, pages 575–581, 2010. (Cited in pages 31 and 34.)

[Routier 2019] Comité National Routier. *http://www.cnr.fr/Indices-Statistiques/Regional-EA/Referentiel-prix-de-revient*, November 2019. (Cited in page 108.)

[Ruokokoski 2010] Mirko Ruokokoski, OGUZ Solyali, Jean-François Cordeau, Raf Jans and Haldun Süral. *Efficient formulations and a branch-and-cut algorithm for a production-routing problem.* GERAD Technical Report G-2010-66, 2010. (Cited in page 27.)

[Saharidis 2011] Georgios KD Saharidis, Maria Boile and Sotiris Theofanis. *Initialization of the Benders master problem using valid inequalities applied to fixed-charge network problems.* Expert Systems with Applications, vol. 38, no. 6, pages 6627–6636, 2011. (Cited in page 43.)

[Santoso 2005] Tjendera Santoso, Shabbir Ahmed, Marc Goetschalckx and Alexander Shapiro. *A stochastic programming approach for supply chain network design under uncertainty.* European Journal of Operational Research, vol. 167, no. 1, pages 96–115, 2005. (Cited in pages 24, 27, and 28.)

[Schmid 2013] Verena Schmid, Karl F Doerner and Gilbert Laporte. *Rich routing problems arising in supply chain management.* European Journal of Operational Research, vol. 224, no. 3, pages 435–448, 2013. (Cited in page 22.)

[Skutella 2009] Martin Skutella. *An introduction to network flows over time.* In Research trends in combinatorial optimization, pages 451–482. Springer, 2009. (Cited in page 29.)

[Sridhar 2000] Varadharajan Sridhar and June S Park. *Benders-and-cut algorithm for fixed-charge capacitated network design problem.* European Journal of Operational Research, vol. 125, no. 3, pages 622–632, 2000. (Cited in page 38.)

[Srivastava 2008] Samir K Srivastava. *Network design for reverse logistics.* Omega, vol. 36, no. 4, pages 535–548, 2008. (Cited in page 21.)

[Thanh 2008] Phuong Nga Thanh, Nathalie Bostel and Olivier Péton. *A dynamic model for facility location in the design of complex supply chains.* International journal of production economics, vol. 113, no. 2, pages 678–693, 2008. (Cited in page 23.)

[Thanh 2010] Phuong Nga Thanh, Olivier Péton and Nathalie Bostel. *A linear relaxation-based heuristic approach for logistics network design.* Computers & Industrial Engineering, vol. 59, no. 4, pages 964–975, 2010. (Cited in pages 23, 27, and 28.)

[Tomlin 1966] JA Tomlin. *Minimum-cost multicommodity network flows.* Operations Research, vol. 14, no. 1, pages 45–51, 1966. (Cited in page 29.)

[Tsiakis 2001] Panagiotis Tsiakis, Nilay Shah and Constantinos C Pantelides. *Design of multi-echelon supply chain networks under demand uncertainty.* Industrial & Engineering Chemistry Research, vol. 40, no. 16, pages 3585–3604, 2001. (Cited in pages 25, 27, and 28.)

[Vidal 1997] Carlos J Vidal and Marc Goetschalckx. *Strategic production-distribution models: A critical review with emphasis on global supply chain models.* European journal of operational research, vol. 98, no. 1, pages 1–18, 1997. (Cited in page 22.)

[Wang 2011] Fan Wang, Xiaofan Lai and Ning Shi. *A multi-objective optimization for green supply chain network design.* Decision Support Systems, vol. 51, no. 2, pages 262–269, 2011. (Cited in page 24.)

[Wieberneit 2008] Nicole Wieberneit. *Service network design for freight transportation: a review.* OR spectrum, vol. 30, no. 1, pages 77–112, 2008. (Cited in pages 2 and 29.)

[Yeh 2005] Wei-Chang Yeh. *A hybrid heuristic algorithm for the multistage supply chain network problem.* The International Journal of Advanced Manufacturing Technology, vol. 26, no. 5-6, pages 675–685, 2005. (Cited in pages 24, 27, and 28.)

[Yimer 2010] Alebachew D Yimer and Kudret Demirli. *A genetic approach to two-phase optimization of dynamic supply chain scheduling.* Computers & Industrial Engineering, vol. 58, no. 3, pages 411–422, 2010. (Cited in pages 25, 27, and 28.)

[Zhu 2014] Endong Zhu, Teodor Gabriel Crainic and Michel Gendreau. *Scheduled service network design for freight rail transportation.* Operations research, vol. 62, no. 2, pages 383–400, 2014. (Cited in page 51.)

Résumé : La problématique que nous étudions est inspirée d'une collaboration industrielle entre un prestataire logistique, DHL Supply Chain, et une grande chaîne de restauration française. Dans le cadre de ce partenariat, DHL Supply Chain coordonne les acteurs d'un réseau logistique national composé de fournisseurs, d'entrepôts et de restaurants. Les restaurants émettent, sur un horizon temporel, des demandes de produits génériques (produits surgelés, boissons, etc.) fabriqués par les divers fournisseurs. La mission de DHL Supply Chain consiste à assurer l'approvisionnement des restaurants. Pour cela, l'entreprise détermine l'origine d'expédition de chaque produit commandé, et conçoit un plan de chargement caractérisant les itinéraires suivis par les marchandises. DHL Supply Chain souhaite développer des solutions innovantes afin d'améliorer sa compétitivité et d'optimiser la rentabilité de ses opérations logistiques. Dans cette thèse, nous présentons le Logistics Service Network Design Problem (LSNDP) qui formalise la problématique de planification des opérations de transport dans une chaîne d'approvisionnement. Nos travaux ont pour but d'apporter des solutions méthodologiques permettant la résolution d'instances industrielles du LSNDP. Or, ces instances industrielles sont trop complexes pour être résolues par des méthodes génériques de recherche opérationnelle. Nous proposons donc plusieurs algorithmes surmontant la mise à l'échelle des différents paramètres. Nous développons notamment une heuristique de réduction de graphe, ainsi qu'une stratégie de Benders dynamique adaptée à l'augmentation du nombre de produits. À travers diverses études expérimentales, nous évaluons la scalabilité de chaque algorithme par rapport au paramètre considéré. Enfin, nous hybridons ces méthodes pour la résolution d'un cas réel.
Mots clés : logistique, transport, recherche opérationnelle, programmation mathématique, optimisation combinatoire

Abstract: The problem we study is inspired by an industrial collaboration between a third-party logistics, DHL Supply Chain, and a large French restaurant chain. As part of this partnership, DHL Supply Chain coordinates the actors of a domestic logistics network composed of suppliers, warehouses and restaurants. Over a certain time horizon, the restaurants issue requests of generic products (frozen products, beverages, etc.) that are manufactured by the suppliers. The mission of DHL Supply Chain is to ensure the supply of the restaurants. For that purpose, the company determines the shipping origin of each product ordered and designs a loading plan that characterizes the routes followed by the goods. DHL Supply Chain wants to develop innovative solutions to improve its competitiveness and optimize the profitability of its logistics operations. In this thesis, we present the Logistics Service Network Design Problem (LSNDP) which formalizes the problematic of planning transportation operations in a supply chain. Our work aims to provide methodological solutions for solving industrial instances of the LSNDP. However, these industrial instances are too complex to be solved by generic operations research methods. We thus propose several algorithms that overcome the scaling of the parameters. In particular, we develop a graph reduction heuristic, as well as a dynamic Benders strategy that adapts to the increasing number of products. Through various computational studies, we evaluate the scalability of each algorithm with respect to the considered parameter. Finally, we combine these methods for the resolution of a real case.

Keywords: logistics, transportation, operations research, mathematical programming, combinatorial optimization