



**Politecnico
di Torino**

Politecnico di Torino

M. Sc. in Engineering and Management

A. y. 2024/2025

Graduation session: July 2025

SNDP with Asset Management

repositioning of empty containers

Supervisors:

Sara Khodaparasti

Aylin Altun (external, TU Darmstadt)

Candidate:

Alberto Romeo

Acknowledgements

The present work has been conducted under the joint supervision of DIGEP from Polytechnic of Turin and the Chair of Management and Logistic of Technische Universität Darmstadt. I wish to thank both my supervisor from Turin, Professor Sara Khodaparasti, and my supervisor from Darmstadt, Aylin Altun, for their guidance and important insights.

Furthermore, I would like to extend my gratitude to my parents Simonetta and Riccardo and to all the people who have had a positive impact on my life during this master's degree program, whom I will seek to list here in any particular order.

Thanks to: Vayda, Alberto, Beatrice, Lorenzo, Francesco, Susanna, Daniele, Diego, Chen, Joe, Louis, Mohammad, Alessandro, Riccardo.

Table of Contents

Acknowledgements	i
Table of Contents	ii
List of Figures.....	iv
List of Tables	vi
List of Abbreviations.....	vii
1 Introduction.....	1
1.1 Problem Statement and Objectives	1
1.2 Transport: basic concepts	3
1.3 Brief history of Operations Research and Optimization	5
2 Literature Review.....	11
3 Methodology.....	16
3.1 Technical problem statement and assumptions	16
3.2 Model	20
3.2.1The time-expanded network	20
3.2.2 Commodities	23
3.2.3 Resource management: acquisition, usage, repositioning and rentals	24
3.2.4 Management of locomotives and train length	28
3.2.5 Objective function	29
4 Application.....	31
4.1 Model implementation	31
4.1.1 The AIMMS software	31
4.1.2 Constants declaration	33
4.1.3 Network Features	34
4.1.4 Commodities Features	35
4.1.5 Declaration	36
4.2 Dataset overview	42
4.2.1 Network generator	42
4.2.1.1Incidence matrix	43
4.2.1.2 Distances matrix	44
4.2.1.3 Fixed costs matrix	45

4.2.1.4	Variable costs matrix	45
4.2.1.5	Travel times matrix	45
4.2.2	Graph printer	46
4.2.3	Commodities generator	47
4.2.4	Selected test instances	48
4.3	KPIs	49
4.4	Results	51
4.4.1	Repositioning with large commodities	53
4.4.2	Repositioning with small commodities	54
4.4.3	Consolidation	55
4.4.4	Rental profits	56
5	Conclusions.....	58
5.1	Managerial insights	58
5.2	Limitations and obstacles	59
5.3	Final remarks and future research	60
References.....		vi

List of Figures

Figure 1: Summary scheme of SWL transport.....	5
Figure 2: Example of a network with a hub-and-spoke topology	16
Figure 3: Scheme describing the relations between nodes and arcs in the model	23
Figure 4: scheme summarizing acquisitions.....	26
Figure 5: scheme summarizing (un-)loading operations	26
Figure 6: scheme summarizing rentals.....	26
Figure 7: AIMMS model explorer interface.....	31
Figure 8: Example of a table view in AIMMS	32
Figure 9: Constants Declaration, from the model explorer view in AIMMS	33
Figure 10: Network Features declaration, from the model explorer view in AIMMS	34
Figure 11: Commodities Features declaration, from the model explorer view in AIMMS.....	35
Figure 12: main Declaration, from the model explorer view in AIMMS	36
Figure 13: CommodityFlow_x variable details from AIMMS interface.....	37
Figure 14: ResourceFlow_y variable details from AIMMS interface	37
Figure 15: Service_z variable details from AIMMS interface	37
Figure 16: RentedResourceSource variable details from AIMMS interface.....	38
Figure 17: RentedResourceSink variable details from AIMMS interface	38
Figure 18: MaxResources constraint details from AIMMS interface	39

Figure 19: ServiceCorrectness constraint details from AIMMS interface	39
Figure 20: FlowConservation constraint details from AIMMS interface.....	40
Figure 21: DesignBalance constraint details from AIMMS interface.....	40
Figure 22: TotalCost variable details from AIMMS interface	41
Figure 23: Active Subsets and MP declaration, from the model explorer view in AIMMS.....	42
Figure 24: Comparison between a not-suitable hub-and-spoke network and a correct one	47

List of Tables

Table 1: Summary of the variables present in the mathematical model.....	23
Table 2: Distances matrix from the German rail network (DB Cargo, 2025)	44
Table 3: Summary of features of the test instances	49
Table 4: Summary of the KPIs gathered from tests	52

List of Abbreviations

CTSNDP	Continuous Time Service Network Design Problem
LP	Linear Programming
MIP	Mixed Integer Programming
OF	Objective Function
OR	Operations Research
SND	Service Network Design
SNDAM	Service Network Design with Asset Management
SNDP	Service Network Design Problem
SWL	Single Wagon Load
TEU	Twenty-foot Equivalent Unit
TSP	Traveling Salesman Problem

1 Introduction

1.1 Problem Statement and Objectives

In the European Union, as of year 2022, a significant disparity between rail and road freight volumes was still observed, with the former covering only the 17% of inland transport, the latter accounting for about 77%, and inland waterways the remaining part (European Commission, 2024). It is observable that rail transport was not the main mode of transport in any of the countries.

Considering flexibility, road transport is the best mode for transporting freight, being the only one that allows door-to-door shipping. Every other mode allows, in fact, to only travel between specific points and through specific paths in which supporting infrastructures are present and require transferring the goods between different vehicles: an operation, named mode shift, that is personnel and time demanding; in other words: every other mode must be part of a multimodal approach to transportation.

Road transportation, however, in addition to increasing pressure on congested roads, offers a significantly lower performance in terms of safety and environmental sustainability (European Court of Auditors, 2023). The transport sector is responsible for almost a quarter of greenhouse gas emissions in Europe, nearly three quarters of this amount (72% in 2019) being due to road transport. Trucks and lorries carrying freight on roads are responsible for around a quarter of road transport emissions (European Commission: Directorate-General for Mobility and Transport, 2021, cited by European Court of Auditors, 2023).

The development of standardized loading units (namely, containers, swap bodies or semi-trailers) lead to a specific form of multimodal transport called intermodal transport; moreover, introducing flat cars allowed to reduce handling costs even more and speed up loading operations. Nonetheless, the road-only alternative is still, on average, 36% less expensive than the intermodal one (European Court of Auditors, 2023). The goal stated by European Commission (2011) is to “[reduce] greenhouse gas emissions from the transport sector for the first time, aiming for a 60% reduction by 2050 compared with 1990 figures” (European

Court of Auditors, 2023, p.9). However, *“[the] CO₂ emissions from the transport sector did not decrease, but increased by 24 % between 1990 and 2019. While [in fact] the efficiency of heavy-duty vehicle transport (vehicles and logistics) improved during this period, increases in demand for freight transport outpaced these efficiency gains”* (ibid.). For this reason, further research on intermodal transport is crucial nowadays, to foster its competitiveness and allow a more widespread adoption, with the ultimate goal of reducing road transportation and, in turn, greenhouse gas emissions.

This work focuses on a specific segment of an intermodal transportation, that is the one between two rail terminals, and considers the Single Wagon Load (SWL) approach, defined as transporting freight from customers to the most suitable marshalling yard, consolidating it in a long distance train that travels to the destination marshalling yard, and finally shipping it to the destination terminal, again by using feeder transportation (Islam et al., 2016).

Starting from the state-of-the-art in Operations Research (OR) applied to rail transport, the objective of this work is to propose a model that allows to compute the optimal quantity and initial allocation of resources (namely: containers) in a hub-and-spoke network in which customers orders are fulfilled with the SWL approach. It can be therefore defined as a Service Network Design Problem (SNDP) with Asset Management and Repositioning. Additionally, as a major element of novelty introduced by this work, the possibility to rent idle containers at terminals as storage space to customers has been considered as a source of income, potentially influencing the overall economic performance of the system and impacting the optimal values of the decision variables the model aims to determine.

The structure of the paper is organized as follows: in chapter 1, a brief introduction about freight transportation and how OR has become a fundamental part of it is presented; in chapter 2 a literature review is proposed, analyzing the state-of-the-art in the field as the starting point for this work; chapter 3 explains in detail the developed model; chapters 4 presents the implementation and the results of the performance test conducted on the

model; finally, chapter 5 draws the conclusions from this work, offering a possible basis for future investigation of the topic.

1.2 Transport: basic concepts

The term “Transport” refers to a movement of goods, from one location to another, using a vehicle of transport (e.g. trains, trucks, ships) and a transportation infrastructure (e.g. railways, roads, canals) and is, in general, included in the broader field of Logistics, which entails coordinating not only the transportation of goods, but also their production and distribution, and therefore implies managing flows of information as well.

Following the subdivision presented by Bektaş (2017), the main actors in this process can be identified in *shippers*, *carriers* and *intermediaries*. *Shippers* generate the demand for freight transportation and may operate their own fleet or outsource transportation to an external party. *Carriers* operate and offer transportation services for shippers; they may provide customized services or operate through consolidation (distinction which is well described in the next Paragraph). Finally, *intermediaries* are third parties that manage the shipment on behalf of shippers by contracting with one or several carriers.

The field of Transport is itself very broad. Without the pretense of being exhaustive, a high-level taxonomy of the field is proposed, to establish the framework within which the subsequent work will be contextualized and introduce some useful terminology. The dimensions presented in the taxonomy are derived from Crainic et al. (2024), and are:

- *extension of the covered area;*
- *transportation mode;*
- *shipper-carrier relationship.*

The first dimension, the *extension of the covered area*, refers to the difference between short haul and long-haul transport. The focus of this work is on long-haul transportation, which concerns “the movement of goods over relatively long distances, between terminals or cities, [...] by rail, truck, ship, etc., or any combination of modes” (Crainic, 1999).

The *transportation mode* is then the object of the second dimension. As a primary and rough distinction, modes can be differentiated by the “natural element” they travel through, that can be land, water, air. Air mainly refers to air cargo transportation, while water refers to both maritime transportation and river (and canal) navigation; land refers, instead, to both road and rail transportation. A second distinction can be made based on whether the shipment is executed with one mode entirely, and it is referred to as “unimodal”, or with the combination of multiple modes, and thus defined as “multimodal”. The uniformization of freight transportation through the usage of standardized containers, which is referred to as “containerization”, has led to the development of a specific type of multimodal transportation referred to as “intermodal”. The main advantage of intermodal transportation is simplifying modal shifts, since standard containers may be moved from one mode to another without unloading the goods they contain. The facilities at which these shifts are performed are called “intermodal terminals”; the most relevant example for the scope of this work is a rail yard, in which containers can be transferred between road (trucks and lorries) and rail.

Finally, the *shipper-carrier relationship* dimension refers on how the carrier capacity is allocated to the shipper request. It can be *dedicated*, when a loading unit, e.g. a vehicle or a convoy, is assigned to a unique shipper demand, which is paying for the entire journey; furthermore, the shipment travels untouched until its destination. *Consolidating*, on the other hand, means merging several shipments, possibly with different origins and destinations and possibly for only a portion of their total travel, in order to fully exploit the capacity of the utilized vehicles. It becomes a useful technique when the volumes of single orders are not big enough to justify paying a dedicated travel, from the shipper side, or to offer a dedicated direct service with a reasonable quality in a profitable manner, from the carrier side.

This work has its focus on rail freight transportation and models a situation in which shippers request services (i.e. orders) departing from an origin terminal and arriving to a destination one on specific dates; the service is performed following the SWL approach

presented in Paragraph 1.1, thus going through a network of interconnected hubs, as summarized in Figure 1.

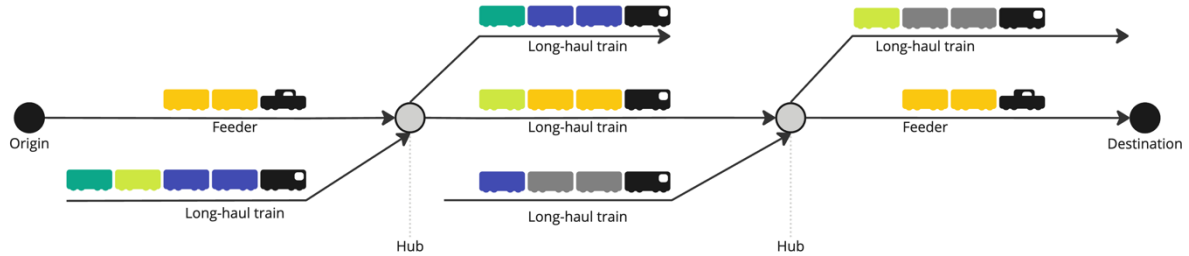


Figure 1: Summary scheme of SWL transport

Since feeders may be, in reality, either trains or trucks and goods travel in container that are explicitly modeled to track their position and movements, this work falls into the category of intermodal transport with consolidation, being the hubs both consolidation facilities and intermodal terminals. It is however assumed, without loss of generality, that all transports in the model are carried via rail. For a complete description of the problem setting and assumptions, reader is referred to Paragraph 3.1.

1.3 Brief history of Operations Research and Optimization

Operations Research, or equivalently Operational Research in British English, is defined as *“A method of mathematically based analysis for providing a quantitative basis for management decisions (originally for military planning)”* (Oxford English Dictionary, 2023).

The origins of OR are be found during the Second World War, during which many brilliant minds coming from the fields not only of Mathematics, Physics, Chemistry and Statistics, but also Psychology, History and Law were assigned to difficult and unfamiliar problem settings. Their research was initially used to improve war operations, but then their foresight *“led to the successful transfer of OR to post-war commerce and industry”* (Gass et al., 2005, p. x).

The aim of this section is to offer a condensed, and definitely not exhaustive, timeline of the development of the OR science. For the sake of brevity, the focus here is solely on the problems, methods, and solutions that have a direct impact on decision making for transports; however, this choice is not intended to diminish the contributions of those scholars who developed the mathematical tools and models upon which these solutions rely. Gass et al. (2005), who defines OR as “*the science of decision making*” (Gass et al., 2005, p. ix), served as the main source for gathering the information that are here proposed.

In 1645, the Italian mathematician Evangelista Torricelli solved a problem whose formulation is attributed (with a certain degree of uncertainty, according to Krarup et al., 1997) to the French mathematician Pierre Fermat in 1643; the problem can be expressed as: “Given three points on a plane, find a fourth point such that the sum of the distances to the three given points is a minimum”. It was only at the beginning of XX century that the Alfred Weber¹ brought this problem to the attention of economists and analysts, as a primal form of a facility location problem.

In the late 1940s, the American mathematician Merrill M. Flood popularized what Gass et al. (2005, p. 48) define as “the most celebrated combinatorial problem”: the Traveling Salesman Problem (TSP). While his paper “The Traveling Salesmen Problem” appeared in the *Journal of Operations Research Society of America* in 1956, Flood, according to Gass et al. (2005), recalls being told about the problem in 1937 by the Albert W. Tucker², who in turn identifies the original source in Hassler Whitney³ around 1931–1932.

The first statement and solution procedure for the classical transportation problem (shipping goods from supply origins to demand destinations at minimum cost) is due to the American mathematician Frank Lauren Hitchcock, in 1941. Since the Dutch American

¹ Alfred Weber (1868–1958), brother of the sociologist Max Weber was German economist and a professor at the University of Heidelberg.

² Albert William Tucker (1905–1995) was a Canadian mathematician who mainly worked on topology, game theory, and non-linear programming,

³ Hassler Whitney (1907–1989) was an American mathematician and one of the founders of singularity theory.

economist and Mathematician Tjalling C. Koopman also worked independently on this problem, it is referred to as Hitchcock-Koopmans transportation problem. However, the formal statement and computational resolution is due to George B. Dantzig.

To the American mathematician George B. Gantzig is attributed, in fact, the first formulation of a linear-programming problem, that is as follows: Minimize (or Maximize) cx , subject to $Ax=b$, $x \geq 0$, where c is a $(1 \times n)$ row vector, x is a $(n \times 1)$ column vector, A is an $(m \times n)$ matrix and b is a $(m \times 1)$ column vector. Dantzig is also the father of the Simplex Method, that he proposed in 1947, revolutionizing decision making. Gass et al. (2005, p. 64) refer to the Simplex Method as "*the workhorse of LP [Linear Programming]*" and reports that it was picked as "*one of the 20th century best algorithms*" (ibid.).

The general statement of Nonlinear programming is instead due to Harold W. Kuhn⁴ and Albert Tucker, it is as follows: Minimize $f(x)$, subject to $g_i(x) = (\text{or } \geq) 0$ (for $i=1, \dots, m$) where all functions are twice continuously differentiable.

In 1951, in the Case Institute of Technology in Cleveland, Colorado, the first OR M.Sc. and Ph.D. degree programs were instituted and in November 1952 the volume 1, number 1 of *The Journal of Operations Research Council of America* was published. From volume 4, number 1, the name was changed to *Operations Research*. It is nowadays published by INFORMS (Institute for Operations Research and the Management Sciences).

In their 1954 seminar paper "*The solution of a large-scale traveling salesman problem*," three American mathematicians (the aforementioned Dantzig, D. Ray Fulkerson and Selmer M. Johnson) demonstrated the efficacy of cutting planes by solving a 49-city TSP by starting with a good solution and adding cuts to the assignment formulation; only 25 cuts sufficed to rule out non-integer solutions to proven optimality.

⁴ Harold William Kuhn (1925–2014) was an American mathematician who mainly worked on game theory.

In the same year, a TSP study by the aforementioned Dantzig and Fulkerson together with Lester Ford was the first work to use a Branch and Bound type of approach, while, as Gass et al. (2005, p. 97) reports, *“the work of Ailsa H. Land⁵ and Alison D. Doig⁶, proposed in 1957 and published in 1960, is considered the [official] origin of Branch and Bound as a general technique for solving linear problems”*.

The concept of *“heuristics”* was first introduced by H. A. Simon⁷ (1955), focusing mainly on bounded rationality, that is the idea that people make decisions with limited time, mental resources and information. Therefore, in contrast with *optimization*, that aims to obtain an optimal solution starting from complete information, *heuristics* refers to the situation in which incomplete information is used to formulate a *“good enough”* solution. In computer science, heuristic algorithms are a class of algorithms that only explore a subset of the possible solutions at each step, in order to, as quickly as possible, provide the aforementioned *“good enough”* solution. Herbert A. Simon, together with Allen Newell⁸ and J. C. Shaw⁹, also implemented Logic Theorist, a program that used heuristic rules to prove algorithms; Gass et al. (2005) reports that the program was able to prove a theorem for the first time in 1956.

It also dates to 1955 the early introduction of stochastic variables in linear programming, by Dantzig and E. M. L. Baele¹⁰.

Different researchers have been working on an efficient algorithm to solve shortest path problems, but the first efficient one is credited to the Dutch computer engineer and mathematician Edsger Wybe Dijkstra. In 1956, Dijkstra proposed an algorithm with complexity

⁵ Ailsa Horton Land (1927–2021) was a professor of Operations Research in the Department of Management at the London School of Economics. She has been the first woman professor of Operations Research in Britain.

⁶ Alison Grant Harcourt (born A. G. Doig in 1929) is an Australian mathematician and statistician who worked at the at the London School of Economics.

⁷ Herbert Alexander Simon (1916–2001) was an American scholar whose work influenced the fields of computer science, economics, and cognitive psychology.

⁸ Allen Newell (1927–1992) was an American researcher in computer science and cognitive psychology.

⁹ John Clifford Shaw (1922–1991) was a systems programmer at RAND Corporation.

¹⁰ Evelyn Martin Lansdowne Beale (1928–1985) was a British applied mathematician and statistician.

$O(n^2)$ for solving shortest path problem with n nodes and non-negative edge costs as well as an algorithm for the shortest spanning tree problem.

Gass et al. (2005) reports that the first international conference in operations research was held at Oxford University, in September 1957, and was attended by 250 delegates from 21 countries. Organized by the OR societies of U.K., U.S. and Canada, the conference theme was *"to unify and extend the science of operational research"*.

George Dantzig is also credited for the first formulation of the Knapsack problem, in 1957: Maximize $c_1x_1 + c_2x_2 + \dots + c_nx_n$ subject to $a_1x_1 + a_2x_2 + \dots + a_nx_n \leq b$, with each x_j equal to 0 or 1, with all (a_j, c_j, b) taken as positive numbers. The Knapsack problem model is often applied in logistics, especially in warehouse management and cargo loading.

In 1962, Lotfi A. Zadeh¹¹ introduced the notion of "fuzzy sets", that can be formulated as follows: given a subset A of X , and a point x in X , the grade of membership of x in A can be expressed as a function $m_A(x)$ that takes values between 0 and 1.

In the 1970s, the American mathematicians Michael Held and Richard M. Karp developed a method to obtain bounds and solutions for integer-programming problems, based on Lagrangian relaxation. The method starts from the possibility to separate the constraints of the original problem into two parts, one of which has a special structure that allows the problem to be solved more easily. The complicating constraints are then relaxed, and the problem is solved retaining only the easy subset of constraints.

In 1971, the American computer scientist and mathematician Stephen Cook demonstrated that every set of strings accepted in polynomial time by a nondeterministic Turing machine can be reduced to a Boolean satisfiability problem (abbreviated SAT) in polynomial time. This result identified SAT as the *"archetypically intractable problem"* (Gass et al., 2005,

¹¹ Lotfi Aliasker Zadeh (1921–2017) was a mathematician and computer scientist, and professor of computer science at the University of California, Berkeley.

p.158). In the following year, Richard Karp identified a class of problem that he called “polynomial complete” (nowadays “NP-complete”) and provided a list that reduced a number of classical combinatorial optimization problems to SAT. Cook’s result, also introducing Karp’s definition, stated that being able to solve a SAT problem in polynomial time would mean, in turn, being able to solve NP-complete problems in polynomial time, that is $P = NP$.

In the same year, 1972, the American mathematicians Victor Klee and George Minty provided an example of a LP problem for which the simplex method would have to evaluate all vertices of the defining polytope. Considering this value could be increased exponentially, this problem proved that the simplex method is not a polynomial-time algorithm.

After more than ten years of research, in 1975 the American scientist John Holland introduced genetic heuristic algorithms, in which the randomized search of optimal solutions mimics natural selection. After representing the “*population*” of possible solutions as binary strings, similar to chromosomes, it will “*evolve*” following the “*survival of the fittest*” principle, producing a high-quality solution for the problem.

In 1986, the American computer scientist introduced the term “*tabu search*”, referring to a *matheuristic* procedure in which specific memory structures are used to exclude certain solutions from the search neighborhood, with the ultimate objective of preventing the underlying heuristic from becoming trapped in local optima.

Coming to the recent years, over the last three decades the field of OR, and of optimization of freight transportation, has witnessed notable progress. The papers that have been the most relevant for this work, serving both as starting point and sources, are analyzed from a technical perspective in the literature review presented in the next chapter.

2 Literature Review

In the recent years, the rapid evolution of technology resulted in an exponential growth of computational power, allowing to solve progressively more complex problems, adding specifications and increasing the size of instances that models and algorithms can solve.

Back to almost twenty years ago, Wieberneit (2008) gathered and compared five different problems that entail SND, stating, however, that realistic instances including resource re-positioning were still difficult to solve. SNDP cover a wide range of situations in which optimization of freight transport is required, starting from express shipment (Barnhart 2002) and coming to multimodal transportation (Jansen 2004).

StadieSeifi et al. (2014), when covering the literature on multimodal freight transport published between 2005 and 2014, adopts a subdivision based on the decision horizon of planning problems: strategic, tactical, operational planning. Reader is referred to the work by StadieSeifi et al. (ibid.) for a review on strategic and operational planning decisions.

SND is a category of problems included in tactical planning, and it is not limited to rail transport only. Andersen et al. (2009a) stated that – at least at the time – not much research effort had been dedicated to asset management, except for the introduction of design-balance constraints, that is the requirement that an equal number of assets should enter and leave each node. The proposed model provides management of resources together with a service schedule and is applicable to multiple transportation modes; the single type of asset taken into account can be assimilated to trucks or locomotives for example, or even to work force, such as crews. Assets are assumed in a limited quantity and follow cyclic paths, which are modelled with a time-expanded network with periodic schedule; the number of services that can be run is, therefore, bounded to the number of assets, which are managed individually and with the assumption that one asset only is required to run a service. The computational study shows that cycle-based formulations are faster than formulations based on arc design variables.

Pedersen et al. (2009) also propose an arc- and cycle-based formulation for a multicommodity SNDP with asset management that makes use of design-balance constraints, furthermore showing the efficiency of a tabu-search heuristic methodology applied to the arc-based formulation.

Zhu et al. (2014) propose a multi-layer model that includes service selection and scheduling, car classification and blocking, train makeup and routing of time-dependent commodities. The three layers are constituted by three time-expanded networks, respectively tracking the movements of cars, blocks, trains; the model is therefore multi-asset, with “vertical arcs” connecting the layers (and, in turn, the assets) to each other, that is (un-)grouping cars into blocks and blocks into trains. Three types of decision variables need to be introduced: a binary service selection variable stating whether a service is selected or not in the first layer; a binary block selection variable stating whether a block is built or not in the second layer; an integer car flow distribution variable, which is multi-indices and states how many cars of every type flow through each arc in the third layer. A metaheuristic solving methodology is introduced, combining slope scaling with long-term-memory-based perturbation strategies, a mechanism for dynamic block generation, a novel neighborhood exploring mechanism called ellipsoidal search. The computational analysis shows that the model, paired with the aforementioned dedicated methods, outperforms commercial solvers.

Coming to more recent days, Crainic et al. (2024) offers an updated and comprehensive overview of the state-of-the-art in consolidation-based freight transportation. The authors distinguish between models that have resource management as their main focus and models that, instead, integrate the aspect on the more general SNDP setting; moreover, the importance of design balance constraints is remarked, underlying that considering multiple resources entails introducing a set of design balance constraints for each of them. The increasing complexity introduced by these numerous sets of constraints, however, necessarily requires heuristic methods to be tackled, but also *“induces a structure to the problem*

and solutions" (Crainic et al., ibidem): the authors refer here to the introduction of cycle-based solutions and meticulously crafted matheuristic methods.

Many studies have been conducted on how to make a cycle-based formulation usable for large instances: the most suitable solution appears to be to split the problem into subproblems, generating *good cycles* first and then finding the best solution possible using only those. To find whether the algorithm is capable of finding *good cycles*, it is necessary to run it on small instances and compare its result with the ones from a general solver generating *all* the possible cycles, as shown by Crainic et al. (2016).

Crainic et al. (2018) introduces a two-stages model that is capable to select and allocate the necessary resources to support the schedule it produces, with the possibility to either buy or rent assets. The key elements for the first stage are an *Acquisition node* and a set of nodes specifically dedicated to reallocation. Other than acquisition and renting and, clearly, travel costs, the OF also includes specific expenses needed to put the new resources into work. The proposed solving method entails, once again, a matheuristic generating *good cycles* first, then using them to produce a solution.


Boland et al. (2017) focuses on the aspect of time discretization in dynamic SNDP, acknowledging that avoiding generating unnecessary nodes for unrequired timepoints can dramatically decrease the size of the time-expanded network and, therefore, the number of variables the model has to enumerate. Thus, the paper provides an algorithm that iteratively refines what it calls a *partially time-expanded* network, that only includes a portion of the correspondent *fully time-expanded* one and is therefore considerably smaller, until it is proven that an optimal solution computed on this smaller network is also an optimal solution to the *full time-expanded* one.

Liu et al. (2020) consider a static network in which nodes are given with supply and demand values for resources in the form of fuzzy numbers. The goal is to minimize repositioning costs on arcs, with the constraints that supply, demand and arc capacity *should* be respected

within the desired confidence level. Authors then propose a two-stage genetic algorithm that first generates supply and demand relationships between nodes, then solves the instance as a multicommodity flow problem; the computational study shows the efficiency of the solving method, which however, authors remark, needs further improvements in order to work in proper time on large instances.

Frisch et al. (2023) proposes a model that integrates a strategic planning step, that is a freight car routing problem, with SND; furthermore, a routing matrix is produced together with the service schedule. After showing that a flexible routing strategy would be more efficient, but stating that the same strategy is subject to human error and therefore usually excluded by companies, authors conclude the best option would be to use a periodically updated routing matrix that reflects the changes in customers requests. The construction of the routing matrix is however, due to the additional constraints it requires, a highly demanding task in terms of computational effort: with a detailed performance study with operational KPIs, authors show that, on a 14 hubs network, an optimal 24-hours hourly schedule can be produced for a maximum of 120 commodities in around one and a half hours, while an optimal 72-hours hourly schedule can be produced for 20 commodities only.

What emerges from the literature review is that asset management and repositioning in SNDP has been extensively investigated, with cycles emerging as the best solution, so far, to model resources that are countable and in a relatively small number, such as locomotives in rail transport. Fewer studies have been conducted on assets held in more substantial volumes, such as cars or containers, for which is reasonably impossible to track the movement of every unit individually, not even mentioning creating a cycle. Furthermore, to the best knowledge of the author, there is no available model that tracks the usage of a resource existing in a large quantity with the objective of knowing its optimal initial allocation to deliver a given list of commodities. The objective of this work is to fill this gap by proposing a model to track empty containers while solving a SNDP; model that can be paired, as a subject for future research, with a cycle-based formulation that tracks



locomotives in order to manage, with appropriately designed solving methods, all the main assets required in rail freight transport, all within a single model.

3 Methodology

3.1 Technical problem statement and assumptions

In SWL transport, commodities¹² that start their journey at a terminal (also referred to as *source node*) are at first routed to the most comfortable marshalling yard through feeders, that can either be smaller trains or trucks. Here, containers from different orders are consolidated in a single train that takes its departure through a network of hubs, in every of which it is decomposed and sorted; then, after the required maintenance and safety procedures, new trains are assembled and can depart again. When the commodity reaches the hub its destination (also referred to as *sink node*) is connected to, the commodity is delivered, again with feeders. It is clearly noticeable that the best topology to describe this process is a hub-and-spoke network. Without lessening precision and to make the notation more concise, from now on marshalling yards will be referred to as *hubs*, while terminal nodes, either sources or sinks, in which the contact with customers (i.e. *shippers*) happens, will be referred to as *terminals*.

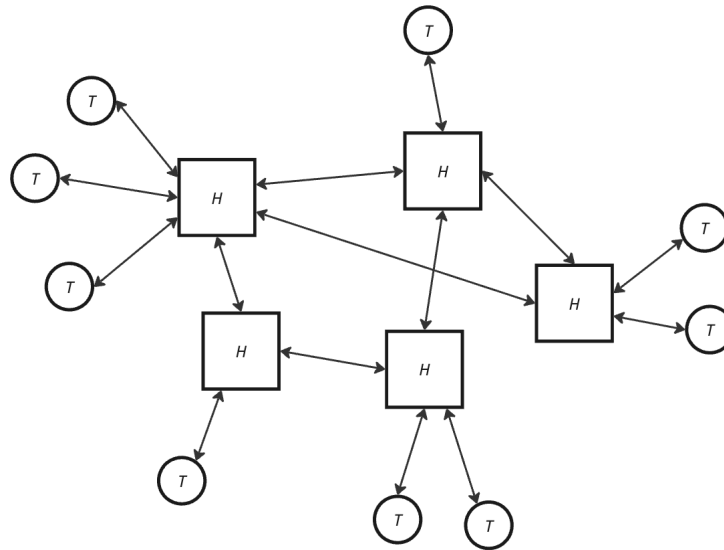


Figure 2: Example of a network with a hub-and-spoke topology

¹² The terms “commodity” and “order” are used interchangeably.

Determining the optimal network structure requires considering additional problem categories, namely hub location and vehicle routing problems, which fall outside the scope of this work; therefore, the hub-and-spoke network is assumed as given. Follows a description of the characteristics of this network.

- Each terminal is connected to one hub only, while a hub can be connected to an arbitrary number of terminals.
- Realistically, the network of hubs is not complete, meaning each hub should not have a direct connection to every other hub.
- The length (in time) of travel between two nodes (both for terminal-hub or hub-hub travels) is given, as well as the fixed cost of running a service, that is the cost to move a single train, and the variable cost, which depends on the train length and is, therefore, provided as *per container*. Both the variable and the fixed cost are reasonably assumed to be proportional to the travel length and symmetrical for every couple of nodes.

Orders constitute a strict requirement to the solution of the problem to be fulfilled by managing resources (i.e. containers) efficiently, that is planning both their usage and repositioning. Ultimately, the objective of the model is to determine the optimal number of containers to acquire and their initial allocation at terminals, in order to fulfil all the orders minimizing the total cost.

Thresholds on train length, stating the minimum and maximum number of containers that can be transported by a single service, are introduced. In order to maintain the complexity of the problem on a level that reasonably allows finding solution to realistically large instances without implementing a dedicated method, however, the thresholds are only used when computing KPIs to analyze the behavior of the model, not to introduce further constraints. The usage and repositioning of cars and locomotives is also outside the scope of this work, and cars and locomotives are assumed available on require.

As an element of novelty, this model introduces the possibility of renting idle containers at terminals as storage space, for a fee that depends on the length of the rental itself. The objective to manage the movement of containers, together with the one of allowing rentals, requires explicitly modeling time and, thus, introducing a time-expanded network, about which the Paragraph 3.2.1 gives a detailed description.

All commodities are included in the set K and are accompanied by comprehensive information about space, time and size, specifically provided in the following format:

Commodity $\coloneqq ((o^k, e^k), (d^k, l^k), v^k, \dot{r}^k, \ddot{r}^k)$, being:

o^k : origin terminal (source) of the order

e^k : ("early") time at which the order departs

d^k : destination terminal (sink) of the order

l^k : ("late") time at which the order is due

v^k : volume of the order, expressed in n. of containers

\dot{r}^k : rental order length at source terminal

\ddot{r}^k : rental order length at sink terminal

Commodities cannot be split, meaning all containers belonging to the same order must be shipped as a single block. This assumption allows to track commodity flows through binary variables, significantly reducing the complexity of the model.

Rentals are modelled according to the following assumptions:

- Rentals can only take place at terminals and are offered as an additional service to customer who place a transport order; consequently, rentals can only take place at o^k or d^k for a k in K .
- Rentals may occur before a commodity departs from its source or after it arrives at its sink, meaning a rental period must either terminate at e^k or start at l^k for a k in K .

- Rentals do not constitute strict requirements as commodities do, it is therefore possible to not fulfil a rental if the resources availability does not allow to do so.
- The size of rentals corresponds to the volume of the related commodity.
- The length of rentals is assumed known and is given by the second to last and by the last element of the commodity tuple, depending on whether the rental takes place at the commodity's source or sink, respectively.
- Rentals may be fulfilled partially in volume, meaning it is possible to offer less storage space (i.e. less containers) than requested.
- Commodities does not necessarily carry one or two rental orders; if a rental is not required, the corresponding length parameter in the commodity tuple will be simply set to zero.

Processing times of containers and vehicles at marshalling yards are taken into account as follows. A precise analysis of the operations that need to be performed at a marshalling yard between the arrival of a service and the departure of the next one is offered by Pollehn et al. (2021), and their optimization requires a model itself, which falls outside the scope of this work. For this reason, those operations, which also include consolidation, can be assumed, to have a constant time length for which a realistic value is also taken from Pollehn et al. (ibid.), provided with a unit of measure in minutes or hours. The model has however a higher granularity (half-days), meaning that a service that starts at a certain node will terminate *at least* half a day later. It is therefore reasonable to consider all the processing times already included in that interval. The timepoint at which the *service arc* terminates at a hub indicates, therefore, the moment when the commodities and resources carried by that service are ready for departure, already having undergone the necessary processing operations in the hub.

3.2 Model

In this Paragraph, a presentation of the model is proposed. It has been chosen, for a clearer exposition, to introduce each component in a separate sub-paragraph, then finally provide a complete overview.

3.2.1 The time-expanded network

Before introducing the time-expanded network, the relative static (also referred to as “flat”) version is presented. As already mentioned, the starting point is a classic hub-and-spoke network, composed of a set of nodes N that can be divided into a set of hubs H and a set of terminals T , connected by arcs that constitute set A (Greek Alpha¹³).

Therefore, the flat network can be denoted as:

$$Z = (N, A) = (H \cup T, A)$$

To define the time-expanded network, for which the approach proposed by Crainic et al. (2018) is followed¹⁴, it is first necessary to introduce the set of the timepoints D through which services are defined:

$$D = \{0, 1, 2, \dots, T_{\text{MAX}}\}$$

The timepoints go from 0 to T_{MAX} , which is the *schedule length*. The actual schedule goes from timepoint 0 to timepoint $(T_{\text{MAX}} - 1)$. In fact, considering the schedule is cyclically repeated, timepoint T_{MAX} of a schedule is nothing but timepoint 0 for the next one. Defining the appropriate value for T_{MAX} sets a difficult challenge in finding a tradeoff between accuracy and efficiency of the model, on one side, and reasonable computational effort and time, on the other. A greater T_{MAX} would mean a higher amortization of the cost of resources, but also

¹³ For a better clarity of exposition, it has been decided to denote node sets with Latin letters and arc sets with Greek ones; to avoid misinterpretation in similar characters, the LaTeX notation has been followed: Latin letters are always italic, Greek letters are not.

¹⁴ Since a software implementation of the model is also provided, it has been chosen to keep the notation consistent between the formal model and the implemented version. For example, Crainic et al. (2018) use the latin letter D to denote the flat network, while here Z is preferred, since D is used to denote the set of timepoints (= half-days).

higher complexity, in terms of the number of variables and constraints to be analyzed. Finally, the time-expanded network Z_T can be defined as $Z \times D$.

Once Z_T is defined, it is possible to differentiate two types of arcs: *service* and *holding arcs*. *Service arcs* (also referred to as *transportation* or *transport arcs*) model a movement, a travel, that means, in the context of this work, a transport between two different nodes (either terminal-hub or hub-hub), in a particular period in time. Thus, there are different possible parallel service arcs running between the same pair of nodes, all with the same length (equal to the travel time between the two nodes) but starting at different timepoints. On the other hand, *holding arcs* model the possibility for a resource or commodity to be held at a node (either a terminal or a hub) until the subsequent period, thus start and end at the same node. It is a common and useful practice, in the literature, to separate these two types of arcs in different subsets of the set A . Naming the service arcs set T (for “transport”) and the holding arc H :

$$i = j \wedge \bar{t} = t + 1 \quad \forall ((i, t), (j, \bar{t})) \in H$$

$$i \neq j \wedge \bar{t} > t \quad \forall ((i, t), (j, \bar{t})) \in T$$

And it holds:

$$A = T \cup H \text{ and } T \cap H = \emptyset.$$

Following the approach presented by Crainic et al. (2018), a single acquisition node Q is introduced, connected to every terminal through so-called acquisition arcs, that form a subset named \emptyset . The main “decision” to be taken by the model is how many containers to initially allocate to each terminal, that is to determine, for each of the arcs included in \emptyset , the best value to give to the variable that regulates the flow of containers. The expense for resource acquisition is computed by summing all these variables and multiply the result by the cost of a single container; this initial expense is then added to the objective function (OF) to be minimized.

Acquisition arcs always start from the *acquisition node* Q and end at a terminal. Acquisition and allocation of resources are carried on before the schedule starts: dynamically acquiring new resource during the schedule length is assumed not possible. Therefore, *acquisition arcs* symbolically start at a timepoint equal to -1 and have length of one period, so that resources are available at timepoint 0 . *Acquisition arcs* are also used to model the cyclical nature of a schedule. To do so, more arcs are introduced, (again symbolically) exiting from terminals at timepoint T_{MAX} and reaching node Q at timepoint $T_{MAX}+1$.

$$(i = Q \wedge j \in T \wedge t = -1 \wedge \bar{t} = 0) \vee (j = Q \wedge i \in T \wedge t = T_{MAX} \wedge \bar{t} = T_{MAX} + 1)$$

$$\forall ((i, t), (j, \bar{t})) \in \Theta$$

Thus, by then introducing a constraint that forces the outgoing flow at T_{MAX} to be equal to the incoming flow at timepoint 0 for every terminal, schedule repeatability is assured.

Rentals are modelled with a dedicated set of arcs, sharing similar characteristics with holding arcs and having a few additional conditions. The set of *rental arcs* is denoted with P and is subject to:

$$\left((i, t) = (d_k, l_k) \wedge \bar{t} - t = \dot{r}^k \right) \vee \left((j, \bar{t}) = (o_k, e_k) \wedge \bar{t} - t = \dot{r}^k \right)$$

$$i = j$$

$$\forall ((i, t), (j, \bar{t})) \in P$$

Finally, (un-)loading processes, not having an additional duration than the interval between one timepoint and the next one, do not require any additional variable nor constraint and are modelled through the design balance one, described in Paragraph 3.2.3.

The complete time-expanded network can thus be denoted as:

$$Z_T = (H \cup T \cup Q, T \cup H \cup P \cup \Theta)$$

Figure 3 and Table 1 provide a visual representation and a summary of the topology introduced so far.

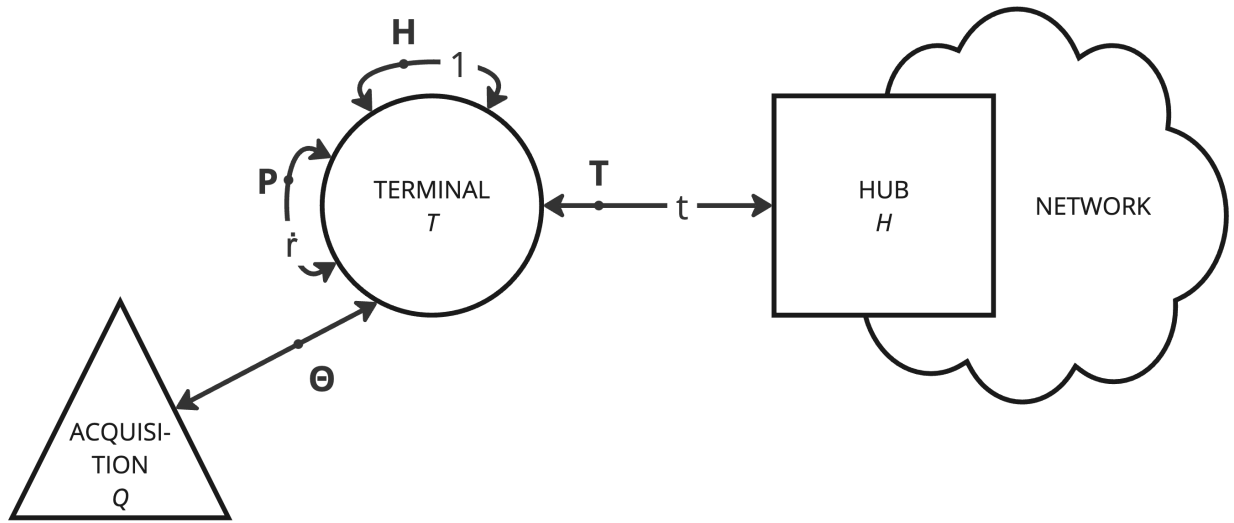


Figure 3: Scheme describing the relations between nodes and arcs in the model

H	Set of hub nodes (marshalling yards)
T	Set of terminal nodes
Q	Acquisition node
T	Set of service (transport) arcs
H	Set of holding arcs
P	Set of rental arcs
Θ	Set of acquisition arcs
t	Travel time of a service arc
\dot{r}^k (or \ddot{r}^k)	Time length of a rental

Table 1: Summary of the variables present in the mathematical model

3.2.2 Commodities

As already mentioned, commodities are given in the format:

$$\text{Commodity} := \left((o^k, e^k), (d^k, l^k), v^k, \dot{r}^k, \ddot{r}^k \right) \forall k \in K,$$

meaning each of them starts its travel from its source terminal node o^k at a timepoint e^k and is required to be at its sink terminal node d^k at a timepoint l^k ; v^k gives the volume of the commodity, measured in the number of containers it is composed of; \hat{r}^k and \bar{r}^k give the length of rental orders related to the commodity, respectively at source and sink nodes (as mentioned, when there is no rental order, the value is simply set to zero).

The flow of commodity is modeled through the multi-indices variable:

$x_{ij}^{kt\bar{t}}$ being:

$k \in K$ a commodity

$((i, t), (j, \bar{t}))$ the starting and ending points of an arc in the time-expanded graph

$$x_{ij}^{kt\bar{t}} \in \{0,1\} \quad \forall k \in K, \forall ((i, t), (j, \bar{t})) \in T \cup H$$

Therefore, basing on the assumption that commodities cannot be split, it is possible to use binary variables $x_{ij}^{kt\bar{t}}$ to model whether the commodity k travels from node i to node j in the time interval going from t to \bar{t} .

$$\sum_{((i,t),(j,\bar{t})) \in T \cup H} x_{ij}^{kt\bar{t}} - \sum_{((j,\bar{t}),(i,t)) \in T \cup H} x_{ji}^{kt\bar{t}} = \begin{cases} 1 & \text{if } (i, t) = (o_k, e_k) \\ -1 & \text{if } (i, t) = (d_k, l_k) \\ 0 & \text{otherwise} \end{cases}$$

$$\forall k \in K, \forall (i, t) \in T \cup H$$

A standard multicommodity flow conservation constraint is introduced, stating each order has to start at its source and end at its sink in the appropriate timepoints, not being lost in between.

3.2.3 Resource management: acquisition, usage, repositioning and rentals

The key resource is, in this work, containers. A multi-indices variable $y_{ij}^{t\bar{t}}$ is introduced to manage the usage and repositioning of containers, having the same indices of the aforementioned x variable. It is important to specify that y is used to only account for empty

containers, since the number of laden containers can be easily derived from x^k variables and v^k constants, for every k in K .

$$\sum_{((i,t),(j,\bar{t})) \in T \cup H} y_{ij}^{t\bar{t}} - \sum_{((j,\bar{t}),(i,t)) \in T \cup H} y_{ji}^{\bar{t}t} = \Delta_Q + \Delta_K + \Delta_P \quad \forall (i,t) \in T \cup H$$

Being:

$$\begin{aligned} \Delta_Q &= \sum_{((j,\bar{t}),(i,t)) \in \Theta} y_{ji}^{\bar{t}t} - \sum_{((i,t),(j,\bar{t})) \in \Theta} y_{ij}^{t\bar{t}} \\ \Delta_K &= \sum_{k \mid (i,t) = (d_k, l_k)} v^k - \sum_{k \mid (i,t) = (o_k, e_k)} v^k \\ \Delta_P &= \sum_{((j,\bar{t}),(i,t)) \in P} y_{ji}^{\bar{t}t} - \sum_{((i,t),(j,\bar{t})) \in P} y_{ij}^{t\bar{t}} \end{aligned}$$

This constraint starts from a standard design balance constraint, that can be observed on the left side of the equation. The right side is, instead, given by the sum of three *deltas*.

Δ_Q (acquisition delta) is a design balance constraint on *acquisition arcs*, therefore it takes a positive value only at $t=0$ and a negative value only at $t=T_{MAX}$, when resources are allocated at node i , also ensuring the repeatability of the schedule.

Δ_K (commodities delta) is used to model (un-)loading processes. Being commodities a strict requirement, it is granted, in feasible solutions, that v^k containers are loaded at (o_k, e_k) and unloaded at (d_k, l_k) for every k in K ; since y variables track the movements of empty containers, it is sufficient to subtract (or add) v^k to the design balance to model loading (or unloading).

Δ_P (rentals delta) is given by the difference between incoming and outgoing flows of resources through rental arcs, which are contained in the set P , and is therefore a standard design balance constraint on *rental arcs*.

The following schemes (Figures 4, 5, 6) summarize acquisitions, (un-)loading processes and rentals as modeled through the aforementioned *deltas*.

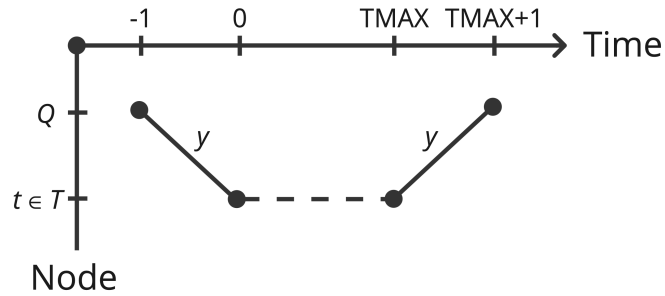


Figure 4: scheme summarizing acquisitions

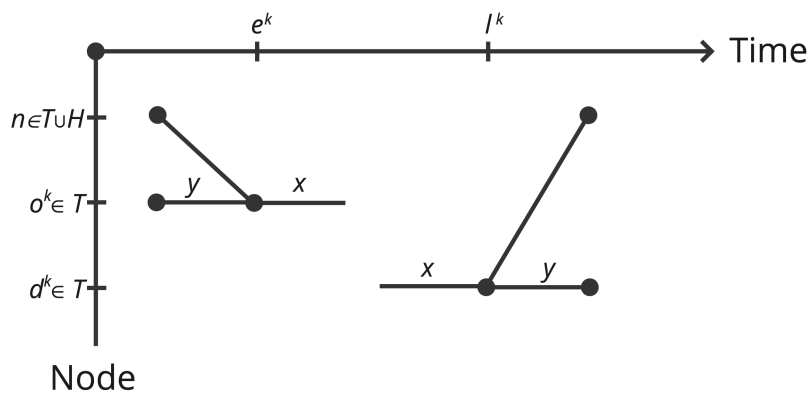


Figure 5: scheme summarizing (un-)loading operations

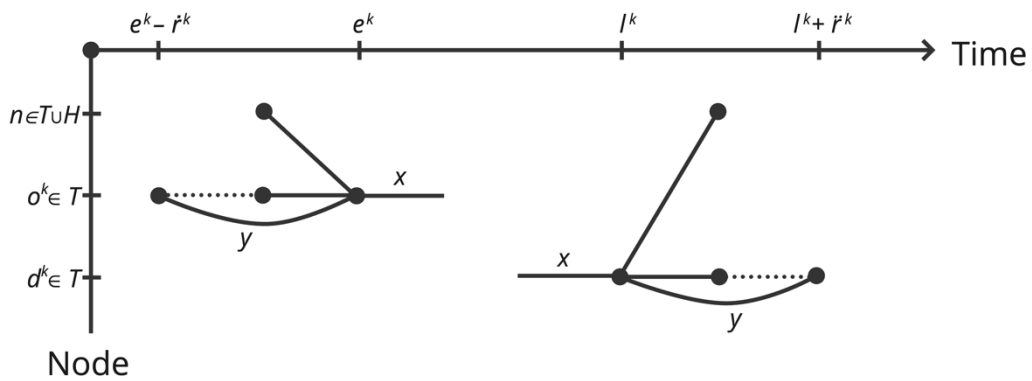


Figure 6: scheme summarizing rentals

$$y_{ij}^{t\bar{t}} \in \mathbb{Z}_{\geq 0} \quad \forall ((i, t), (j, \bar{t})) \in H \cup T \cup P \cup \Theta$$

The domain of y needs to be an integer non-negative number. Note that y is allowed to be non-negative also on arcs included in the set Θ , which is the *acquisition arcs* set. The objective to find the number of resources that could optimally satisfy a previously known demand, without the possibility to dynamically adjust this number during the course of the schedule, is granted by the definition of *acquisition arcs* and, therefore, no further limitation on y variables is needed.

As already mentioned, rentals are modeled with y variables through a dedicated set of arcs that are introduced in a way that fits the features around which rentals themselves are built. Specifically:

- Containers are considered to be at that terminal, but unavailable for a given, precise amount of time, being *outside* the terminal node until the whole length of the *rental arc*, that is the length of the rental order itself, is terminated.
- The flow of resources through *rental arcs* can be less than or equal to the volume of the corresponding rental order, allowing to give for rent less than the required space, without completely neglecting the corresponding order (which can still be done, if it is the most convenient, i.e. optimal, solution).
- The length of *rental arcs* is equal to the one specified in the request; this feature is granted by how set P is defined.

$$y_{ij}^{t\bar{t}} \leq \sum_{k \mid (i, t) = (d_k, l_k) \vee (j, \bar{t}) = (o_k, e_k)} v^k \quad \forall ((i, t), (j, \bar{t})) \in P$$

This constraint ensures that the maximum number of containers traveling through a rental arc should equal the volume of the related commodity, that is the one that has the source at the head of the arc or the sink at its tail (considering elapsing time as the direction).

$$\sum_{((i, T_{\text{MAX}}), (j, \bar{t})) \in \Theta} y_{ij}^{(T_{\text{MAX}})(\bar{t})} = \sum_{((j, \bar{t}), (i, t)) \in \Theta} y_{ji}^{\bar{t}t} \quad \forall (i, t = 0) \in T$$

Finally, since the problem setting considers as the schedule length a time period that periodically repeats, a further constraint is required, to ensure that the final state of the network is equal to the initial one. To achieve this objective using *acquisition arcs*, the outgoing flow at period T_{MAX} is set to be equal to the incoming flow at period 0, for every terminal.

3.2.4 Management of locomotives and train length

The multi-indices Boolean variable $z_{ij}^{t\bar{t}}$ is introduced to keep track of whether a specific service departing from node (i, t) and arriving at node (j, \bar{t}) is activated or not. This purpose could be easily achieved with a Heaviside function of the number of containers traveling through the aforementioned arc, that is:

$$z_{ij}^{t\bar{t}} = H\left(y_{ij}^{t\bar{t}} + \sum_{k \in K} x_{ij}^{kt\bar{t}} v^k\right)$$

To keep the z variable linear, it is sufficient to reshape it with a big M formulation, as follows:

$$M \cdot z_{ij}^{t\bar{t}} \geq y_{ij}^{t\bar{t}} + \sum_{k \in K} x_{ij}^{kt\bar{t}} v^k$$

Then, introducing N_{MIN} and N_{MAX} as the lower and upper thresholds on train length, the big M in the previous formulation can be replaced with N_{MAX} . One more constraint is finally introduced to guarantee the minimum length for every departing service.

$$N_{MAX} \cdot z_{ij}^{t\bar{t}} \geq y_{ij}^{t\bar{t}} + \sum_{k \in K} x_{ij}^{kt\bar{t}} v^k$$

$$N_{MIN} \cdot z_{ij}^{t\bar{t}} \leq y_{ij}^{t\bar{t}} + \sum_{k \in K} x_{ij}^{kt\bar{t}} v^k$$

$$z_{ij}^{t\bar{t}} \in \{0,1\}$$

$$\forall ((i, t), (j, \bar{t})) \in T$$

In this way, when at least one of the x and y variables related to a specific arc is different than zero, the related z variable is forced to 1. Since a complete tracking of the usage of locomotives extends the scope of this work, z variables are associated to *transport arcs* only and no design balance constraint is needed; they will, however, be useful in the objective

function, when computing the fixed costs. As already mentioned, constraints on train length are relaxed in the implementation, in order to be able to obtain feasible solutions for realistically large instances without implementing dedicated methods; the average service length is, however, computed as a KPI to describe the behavior of the model.

3.2.5 Objective function

After introducing the following constant parameters:

γ : acquisition cost for a container

c_{ij} : cost for shipping one unit of commodity, i.e. one container, through arc ij

f_{ij} : cost running a service, i.e. moving one locomotive, through arc ij

π : fixed fee for the rental of one container for one time period

it is finally possible to introduce the objective function. The SNDP with Asset Management SNDAM(Z_T) seeks to:

$$\text{minimize} \left\{ \sum_{((i,t),(j,\bar{t})) \in T} \left(y_{ij}^{t\bar{t}} + \sum_{k \in K} x_{ij}^{kt\bar{t}} v^k \right) c_{ij} + \sum_{((i,t),(j,\bar{t})) \in T} z_{ij}^{t\bar{t}} f_{ij} + \Gamma - \Pi \right\}$$

Being:

$$\Gamma = \gamma \cdot \sum_{((i,t),(j,\bar{t})) \in \Theta | \bar{t}=0} y_{ij}^{t\bar{t}}$$

$$\Pi = \pi \cdot \sum_{((i,t),(j,\bar{t})) \in P} y_{ij}^{t\bar{t}} (\bar{t} - t)$$

Under the following constraints:

$$\sum_{((i,t),(j,\bar{t})) \in T \cup H} x_{ij}^{kt\bar{t}} - \sum_{((j,\bar{t}),(i,t)) \in T \cup H} x_{ji}^{k\bar{t}t} = \begin{cases} 1 & \text{if } (i, t) = (o_k, e_k) \\ -1 & \text{if } (i, t) = (d_k, l_k) \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in K, \forall (i, t) \in T \cup H$$

$$\sum_{((i,t),(j,\bar{t})) \in T \cup H} y_{ij}^{t\bar{t}} - \sum_{((j,\bar{t}),(i,t)) \in T \cup H} y_{ji}^{\bar{t}t} = \Delta_Q + \Delta_K + \Delta_P \quad \forall (i, t) \in T \cup H$$

$$\begin{aligned}
y_{ij}^{t\bar{t}} &\leq \sum_{k \mid (i, t) = (d_k, l_k) \vee (j, \bar{t}) = (o_k, e_k)} v^k \quad \forall ((i, t), (j, \bar{t})) \in P \\
\sum_{((i, T_{\text{MAX}}), (j, \bar{t})) \in \Theta} y_{ij}^{(T_{\text{MAX}})(\bar{t})} &= \sum_{((j, \bar{t}), (i, t)) \in \Theta} y_{ji}^{\bar{t}t} \quad \forall (i, t = 0) \in T \\
* \quad N_{\text{MAX}} \cdot z_{ij}^{t\bar{t}} &\geq y_{ij}^{t\bar{t}} + \sum_{k \in K} x_{ij}^{kt\bar{t}} v^k \quad \forall ((i, t), (j, \bar{t})) \in T \\
* \quad N_{\text{MIN}} \cdot z_{ij}^{t\bar{t}} &\leq y_{ij}^{t\bar{t}} + \sum_{k \in K} x_{ij}^{kt\bar{t}} v^k \quad \forall ((i, t), (j, \bar{t})) \in T
\end{aligned}$$

* Relaxed in the software implementation.

Being:

$$\begin{aligned}
\Delta_Q &= \sum_{((j, \bar{t}), (i, t)) \in \Theta} y_{ji}^{\bar{t}t} - \sum_{((i, t), (j, \bar{t})) \in \Theta} y_{ij}^{t\bar{t}} \\
\Delta_K &= \sum_{k \mid (i, t) = (d_k, l_k)} v^k - \sum_{k \mid (i, t) = (o_k, e_k)} v^k \\
\Delta_P &= \sum_{((j, \bar{t}), (i, t)) \in P} y_{ji}^{\bar{t}t} - \sum_{((i, t), (j, \bar{t})) \in P} y_{ij}^{t\bar{t}} \\
x_{ij}^{kt\bar{t}} &\in \{0, 1\} \quad \forall k \in K, \forall ((i, t), (j, \bar{t})) \in T \cup H \\
y_{ij}^{t\bar{t}} &\in \mathbb{Z}_{\geq 0} \quad \forall ((i, t), (j, \bar{t})) \in H \cup T \cup \Theta \cup P \\
z_{ij}^{t\bar{t}} &\in \{0, 1\} \quad \forall ((i, t), (j, \bar{t})) \in T
\end{aligned}$$

4 Application

4.1 Model implementation

4.1.1 The AIMMS software

AIMMS (acronym for Advanced Interactive Multidimensional Modeling System) is a pre-scriptive analytics software that offers a mathematical programming environment integrating many different solvers to address different mathematical optimization problems, among which Mixed Integer Programming (MIP) problems.

Among the main advantages that lead to the choice of AIMMS for the purpose of this work, the main one is the graphical interface, that allows to seamlessly create, rename and modify sets, parameters, variables and constraints, and to divide them into multiple levels of “*folders*”, named sections and declarations, thank to which even big models can be maintained clean and organized.

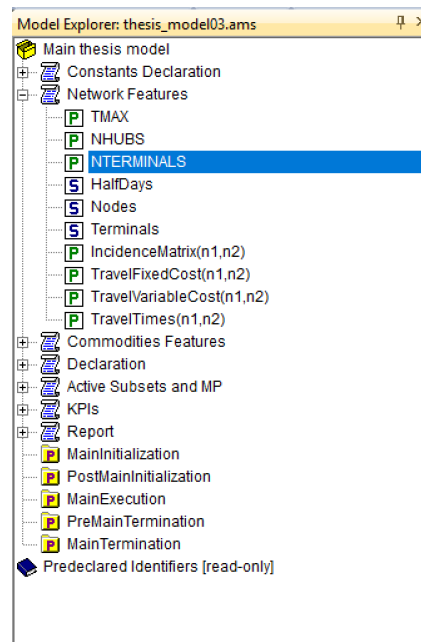


Figure 7: AIMMS model explorer interface

Since the model presented in this work is reasonably small and does not make use of external libraries, no further sections were needed. Instead, it has been chosen to divide its component into declarations for a better clarity of visualization and exposition. Figure 7

shows an extract from the AIMMS interface in which all the declarations in the model are visible and, as an example, the Network Features one has been expanded. A precise description of every element is provided in the following Paragraphs.

AIMMS also offers a clear visualization of multi-indices elements, such as variables and parameters, with the possibility to create as many views as needed by combining the required indices in tables.

		hd1	half-day00	half-day01	half-day02	half-day03	half-day04	half-day05	half-day07	half-day08
k	n1									
1	terminal19									
2	terminal20									
3	terminal20									
4	terminal16									
5	terminal13									
6	terminal19									
7	terminal08									
8	terminal08									
9	terminal05									
10	terminal16									
11	terminal08									
12	terminal07									
13	terminal06									
14	terminal17									
15	terminal03									
16	terminal19									
17	terminal03									
18	terminal04									
19	terminal01									
20	terminal17									
21	terminal11									
22	terminal12									
23	terminal18									
24	terminal06									
25	terminal01									
26	terminal06									
27	terminal14									
28	terminal09									
29	terminal19									
30	terminal03									
31	terminal03									
32	terminal13									
33	terminal19									
34	terminal20									
35	terminal16									
36	terminal12									
37	terminal08									
38	terminal15									
39	terminal01									
40	terminal19									
41	terminal19									
42	terminal09									
43	terminal04									
44	terminal11									
45	terminal15									
46	terminal06									
47	terminal07									
48	terminal09									
49	terminal11									
50	terminal09									

Figure 8: Example of a table view in AIMMS

Figure 8 shows a view of the binary Supply parameter for an instance with fifty commodities: indices k (for Commodities) and n1 (for Nodes) are represented by rows, while the index hd1 (for half-days, the chosen granularity for time periods) is represented by columns. Since it is known that the table is sparse, the *Dense* option has been set to “No” for all three indices, allowing to hide rows and columns containing only zeros.

Another useful feature in AIMMS is the possibility to create sets of variables and constraints. The latter, in particular, comes in handy when troubleshooting in a not-working model. By giving the mathematical program a subset only of the total list of constraints, it is possible to relax the ones that are left out, a useful technique to exactly locate possible issues and *bottlenecks*.

Finally, AIMMS offers a *Math Program Inspector* tool which, other than showing possible non-binding constraints, i.e. constraints whose modification does not affect the optimal solution, can show violated constraints in infeasible programs and tell which variables with which coefficient are present in it, considerably facilitating the process of finding and tackling possible issues in the model.

The solver of reference is IBM ILOG CPLEX Optimization Studio (abbreviated CPLEX), a commercial solver whose main use is to solve MIP problems using primal and dual simplex methods.

A detailed description of the AIMMS implementation of the model proposed in Chapter 3 is presented, divided in Paragraphs following the subdivision in Declarations shown in Figure 7 and highlighting how the elements presented in the mathematical formulation has been translated into AIMMS language.

4.1.2 Constants declaration



Figure 9: Constants Declaration, from the model explorer view in AIMMS

The first two parameters state the cost for a single resource and the fee that a customer has to pay to require a single container as storage space for a single time-period. After research on logistic service providers and marketplace websites and a few tests, their

values have been reasonably set to 1500€ and 5€ per period and never changed, except when running an unrealistic scenario with the purpose of showing a limit behavior of the model (which is described in detail in Paragraph 4.4.4).

The third parameter is automatically computed when loading the instance data, and its value states the total volume of commodities included in the instance, that is:

$$\sum_{k \in K} v^k$$

The fourth and fifth parameters state the minimum and maximum number of containers allowed for a service at standard running cost. The upper threshold has been computed starting from the same assumption used by Frisch et al. (2023) that the maximum train length should be 400 meters and dividing that value by the length, in meters, of a standard 1 TEU container: the result, rounded down, is 65 containers. The lower threshold has been chosen so that the total cost of running a minimum-length service, including fixed and variable costs, would be equal to 1.2 times (i.e. 120% of) its fixed cost: this result is obtained with a value of 13 containers.

Finally, the *isAcquisition* parameter is indexed through timepoints (half-days, *hd*) and takes value of 1 when *hd* – the index – equals zero, of -1 when *hd* equals T_{MAX} and 0 otherwise; its purpose will be explained contextually with the Design Balance constraint in Paragraph 4.1.5.

4.1.3 Network Features

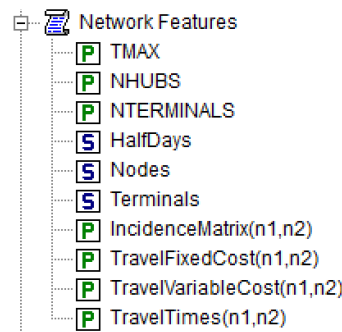


Figure 10: Network Features declaration, from the model explorer view in AIMMS

The first three parameters, that are loaded contextually with the instance, are constants and indicate the schedule length and the number of hubs and terminals in the network, respectively. The schedule is modeled through the set HalfDays, whose cardinality equals TMAX+1: it goes in fact from 0 to TMAX. The network is modelled with the set Nodes, that contains hubs 1 to NHUBS and terminals 1 to NTERMINALS. Set Terminals is a subset of Nodes containing only the terminal ones; a hubs subset is not required, since no constraint applies to nodes only.

The four parameter matrices carry the information about the arcs. AIMMS allows to create as many indices as needed from the same set, but those need to be named all differently from each other; n1 and n2 are two indices related to the Nodes set and are used to create the four matrices. The IncidenceMatrix contains binary values: 1 when n1 and n2 are connected, 0 otherwise; having 1 values in the diagonal, i.e. stating every node is connected to itself, allows to model holding arcs. The other three matrices respectively express the fixed and variable travel costs and the travel length in time-periods. For the scope of this work, that is to show the functioning of a model containing elements of novelty, it is reasonable to assume both costs to be zero for holding arcs. Since no solving method is implemented, holding arcs always have length equal to 1; as proposed by Boland et al. (2017) and already discussed in the Literature Review in Paragraph 2, however, it would be possible to implement an algorithm that optimizes the number of generated arcs, allowing longer holding arcs.

4.1.4 Commodities Features

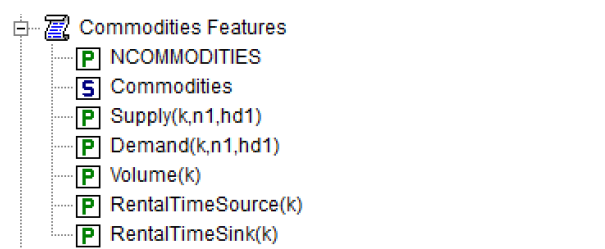


Figure 11: Commodities Features declaration, from the model explorer view in AIMMS

The NCOMMODITIES parameter, loaded contextually with the instance, states the number of commodities and sets the cardinality of the Commodities set, that contains elements 1 to NCOMMODITIES.

The Supply and Demand matrices are three dimensional and use the indices: k from the Commodities set, $n1$ from the Nodes set, $hd1$ from the HalfDays set; they carry binary data and contain 1 in the cell that corresponds to the origin node and availability time (or to the destination node and due time) of commodity k , 0 otherwise. It is clear that storing commodities data this way is extremely inefficient, since the matrix has size $NCOMMODITIES * (NBUHS + NTERMINALS) * (TMAX + 1)$ and only NCOMMODITIES cells filled with ones; it is however very convenient to have the Supply and Demand matrices when writing the constraints for the mathematical program. Once shown the model is working and behaving as expected, a further development for the model would entail storing data more efficiently and reshaping the constraints, allowing to run even larger instances.

Finally, the last three parameters state the volume of each commodity and the length of the related rental orders, at source and at sink respectively.

4.1.5 Declaration

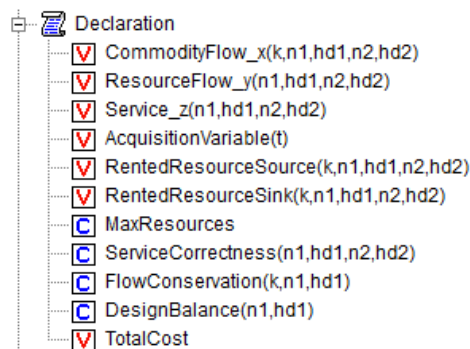


Figure 12: main Declaration, from the model explorer view in AIMMS

The CommodityFlow_x, ResourceFlow_y and Service_z variables exactly translate the x , y and z variables from the mathematical formulation into the AIMMS implementation; they refer to an arc which is expressed by their indices, being $(n1, hd1)$ the node and timepoint

from which the arc starts and $(n2, hd2)$ the node and timepoint at which the arc terminates. Moreover, being `CommodityFlow_x` commodity specific, it also requires index k .

Identifier	CommodityFlow_x
Index domain	$(k, n1, hd1, n2, hd2) \mid hd2 > hd1 \text{ and } hd2 = hd1 + TravelTimes(n1, n2) \text{ and } IncidenceMatrix(n1, n2) = 1$
Text	
Range	binary

Figure 13: `CommodityFlow_x` variable details from AIMMS interface

Identifier	ResourceFlow_y
Index domain	$(n1, hd1, n2, hd2) \mid hd2 > hd1 \text{ and } hd2 = hd1 + TravelTimes(n1, n2) \text{ and } IncidenceMatrix(n1, n2) = 1$
Text	
Range	integer

Figure 14: `ResourceFlow_y` variable details from AIMMS interface

Identifier	Service_z
Index domain	$(n1, hd1, n2, hd2) \mid hd2 > hd1 \text{ and } hd2 = hd1 + TravelTimes(n1, n2) \text{ and } IncidenceMatrix(n1, n2) = 1$
Text	
Range	binary

Figure 15: `Service_z` variable details from AIMMS interface

AIMMS allows to specify the domain of a parameter, variable or constraint in the Index Domain field. `CommodityFlow_x`, `ResourceFlow_y` and `Service_z` have the same domain, specifically only existing on arcs for which the incidence matrix is equal to 1 and the time-length equals the travel time between the two connected nodes (the condition $hd2 > hd1$ is technically redundant, since `TravelTime` for existing connections is always a positive value). `ResourceFlow_y`, representing the flow of empty containers, is an integer variable; `CommodityFlow_x` is binary, being commodities non-splitable by assumption; `Service_z` is also binary, representing whether a service between the two specified nodes in the time-expanded network is run or not.

AcquisitionVariable models the initial acquisition of resources, it is therefore an integer variable and it is only indexed through terminals – t is an index from set Terminals¹⁵. Its functioning is explained in detail contextually with the Design Balance constraint.

Identifier	RentedResourceSource
Index domain	$(k, n1, hd1, n2, hd2) \mid n1=n2 \text{ and } \text{Supply}(k, n2, hd2)=1 \text{ and } hd1<hd2 \text{ and } hd1=hd2-\text{RentalTimeSource}(k)$
Text	
Range	$\{0.. \text{Volume}(k)\}$

Figure 16: RentedResourceSource variable details from AIMMS interface

Identifier	RentedResourceSink
Index domain	$(k, n1, hd1, n2, hd2) \mid n2=n1 \text{ and } \text{Demand}(k, n1, hd1)=1 \text{ and } hd2>hd1 \text{ and } hd2=hd1+\text{RentalTimeSink}(k)$
Text	
Range	$\{0.. \text{Volume}(k)\}$

Figure 17: RentedResourceSink variable details from AIMMS interface

RentedResourceSource and RentedResourceSink model rentals and are the AIMMS implementation of y variables through rental arcs, which form the set P in the mathematical model. The two sets of variables have the exact same purpose; to allow a more comfortable modelling, however, it has been chosen to split them. RentedResourceSource starts and ends at a source node and starts a timepoint equal to the availability time minus the rental length; RentedResourceSink starts and ends at a sink node and ends a timepoint equal to the due time plus the rental length. Since rental orders are strictly related to commodities, the variables need k among their indices; they are integer variables and their maximum allowed value is the volume of the related commodity.

¹⁵ Since an index for the Terminals set in AIMMS was needed, it has been chosen to use the Latin small letter t , consistently with the Latin capital letter T used to name the set in the mathematical model; regarding the indices for AIMMS set of timepoints, named HalfDays, it has been chosen to use the abbreviation hd .

Identifier	MaxResources
Index domain	
Text	
Unit	
Property	
Definition	$\text{sum}[t, \text{AcquisitionVariable}(t)] - \text{sum}[k, \text{Volume}(k)] \leq 0$

Figure 18: MaxResources constraint details from AIMMS interface

The MaxResources constraints states that the maximum total number of acquirable resources equals the total volume of commodities transported through the whole schedule¹⁶. This constraint, although not needed for the correct functioning of the model, has been introduced when studying an unexpected behavior the model adopted with short schedule instances: acquiring a total number of resources that was *higher* than the total commodities volume. A detailed explanation and comment on this behavior can be found in the Observations on Results in Paragraph 4.4).

Identifier	ServiceCorrectness
Index domain	(n1,hd1,n2,hd2) hd2>hd1 and hd2=hd1+TravelTimes(n1,n2) and IncidenceMatrix(n1,n2)=1
Text	
Unit	
Property	
Definition	$\text{Service_z}(n1,hd1,n2,hd2) * \text{Total_Commodities_Volume} - (\text{ResourceFlow_y}(n1,hd1,n2,hd2) + \text{sum}[k, \text{CommodityFlow_x}(k,n1,hd1,n2,hd2)]) \geq 0$

Figure 19: ServiceCorrectness constraint details from AIMMS interface

The ServiceCorrectness constraint assures the correctness of the Service_z variable, forcing its value to 1 when at least one between ResourceFlow_y and the CommodityFlow_x variables of the related arc are positive.

¹⁶ It has been chosen, as a convention, to formulate all the constraints to have a zero right-hand side.

Identifier	FlowConservation
Index domain	(k,n1,hd1)
Text	
Unit	
Property	
Definition	<pre> sum[(n2,hd2), CommodityFlow_x(k,n1,hd1,n2,hd2)] - sum[(n2,hd2), CommodityFlow_x(k,n2,hd2,n1,hd1)] - Supply(k,n1,hd1) + Demand(k,n1,hd1) =0 </pre>

Figure 20: FlowConservation constraint details from AIMMS interface

The FlowConservation AIMMS constraint exactly translate the multicommodity flow conservation constraint formulated in the mathematical model: the index domain includes every commodity in every node in the time-expanded network and the constraint makes sure every commodity starts at its source node, terminates at its sink node and it's not lost or duplicated in between. It is observable that storing commodities information in Supply and Demand matrices format allows a very clean and straight-forward formulation of the constraint.

Identifier	DesignBalance
Index domain	(n1,hd1)
Text	design balance for empty containers only
Unit	
Property	
Definition	<pre> ! outgoing - incoming flow sum[(n2,hd2), ResourceFlow_y(n1,hd1,n2,hd2)] - sum[(n2,hd2), ResourceFlow_y(n2,hd2,n1,hd1)] ! acquisition - AcquisitionVariable(n1)*isAcquisition(hd1) ! commodities - sum[(k), Demand(k,n1,hd1)*Volume(k)] + sum[(k), Supply(k,n1,hd1)*Volume(k)] ! rentals that end at n1,d1 - sum[(k,n2,hd2), RentedResourceSource(k,n2,hd2,n1,hd1) + RentedResourceSink(k,n2,hd2,n1,hd1)] ! rentals that start at n1,d1 + sum[(k,n2,hd2), RentedResourceSource(k,n1,hd1,n2,hd2) + RentedResourceSink(k,n1,hd1,n2,hd2)] =0 </pre>

Figure 21: DesignBalance constraint details from AIMMS interface

The DesignBalance constraint applies to every node in the time-expanded network and regulates the flow of empty containers, serving as a point of contact between all types of

variables. It starts from a standard design balance constraint: the sum of all outgoing flows from a node must equal the sum of all incoming flows, holding arcs included. At timepoint 0, the `isAcquisition` parameter is set to 1: resources allocated in the corresponding node – according to the value of the `AcquisitionVariable` – are “generated”; conversely, at timepoint `TMAX`, `isAcquisition` is set to -1: the same quantity of resources is “destroyed”. The `isAcquisition` parameter is, therefore, not only serving as a source and a sink for resources, allowing a correct functioning of the design balance constraint, but also ensures the repeatability of the schedule; it translates in the AIMMS implementation the characteristics that Acquisition arcs have in the mathematical formulation. The delivery of commodities is guaranteed by the flow conservation constraint; the assumption on (un-)loading times being already included in travel times allows to directly convert commodities into empty containers by simply multiplying the values from the Supply and Demand matrices by the commodities volumes. Finally, all rentals that end at the corresponding node are converted into empty containers, while starting rentals are handled as outgoing container flows.

Identifier	TotalCost
Index domain	
Text	
Range	free
Unit	
Default	
Property	
Priority	
Nonvar status	
Definition	<pre> ! total FC sum[(n1,hd1,n2,hd2), Service_z(n1,hd1,n2,hd2)*TravelFixedCost(n1,n2)] ! total VC + sum[(n1,hd1,n2,hd2), (sum[k, CommodityFlow_x(k,n1,hd1,n2,hd2)*Volume(k)] + ResourceFlow_y(n1,hd1,n2,hd2))*TravelVariableCost(n1,n2)] ! acquisition + sum[t, AcquisitionVariable(t)] * RESOURCE_PRICE ! profit from rentals - sum[{k,n1,hd1,n2,hd2} n1=n2 and [Supply(k,n2,hd2)=1 or Demand(k,n1,hd1)=1], [RentedResourceSource(k,n1,hd1,n2,hd2)+RentedResourceSink(k,n1,hd1,n2,hd2)] * (hd2-hd1) * RENTAL_FEE] </pre>

Figure 22: TotalCost variable details from AIMMS interface

Concludes the main declaration section the `TotalCost` free variable, which is composed of four elements. The fixed costs are computed by multiplying the `Service_z` variable of an arc by the travel fixed cost for the route. The variable costs are computed by multiplying the

train length by the travel variable cost for the route. The acquisition cost is trivially the multiplication between the total number of acquired resources at all terminals and the acquisition price. Finally, the profit from rentals considers the volume of the rental, its length and the constant rental fee (which is expressed per container per period).

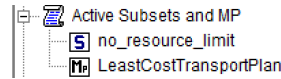


Figure 23: Active Subsets and MP declaration, from the model explorer view in AIMMS

The set `no_resource_limit` has been used to conduct the tests relaxing the `MaxResources` constraint. `LeastCostTransportPlan` is a mixed integer mathematical program whose objective is to minimize the `TotalCost` variable. **H**

4.2 Dataset overview

This Paragraph describes the dataset with which the model has been tested, starting from the python code used to generate the instances. It is important to notice that, for the purpose of testing a model with a high degree of novelty, it is sufficient for the instances to be reasonably realistic, not necessary for them to be real: the objective is, in fact, to be able to observe that the model can produce its solution by following all the rules that it received in form of constraints and that, by tweaking parameters in the test instances, the KPIs show a change in the expected direction.

4.2.1 Network generator

The first step to generate an appropriate instance for the model is the creation of a static hub-and-spoke network, which is done by following a few important steps:

- generate an incidence matrix;
- generate the related distance matrix, i.e. how physically distant, in Km, the nodes are supposed to be;
- generate travel times, fixed cost and variable cost matrices, with their values to be proportional to the distance.

4.2.1.1 Incidence matrix

The complete incidence matrix is created from three smaller incidence matrices: one connecting hubs between each other (named HH), one connecting terminals between each other (named TT), one connecting hubs and terminals (HT).

To create HH, the algorithm iterates through NHUBS¹⁷ rows. For each row, an integer value between 1 and NHUBS/2 is extracted: that is the number of other hubs the present one will be connected to, excluding itself. The right number of connections are randomly generated; then the matrix is mirrored through its diagonal, to make sure it is symmetric. It is important to notice that there is no check that assures that every hub is connected to at least another one after the mirroring step. This occurrence is however very unlikely when the number of nodes is reasonably large (say, higher than 5); moreover, many instances are generated from the same network, meaning the network generation algorithm is rarely run. It has been, therefore, decided to simply print the network graph and let the user manually check its correctness, possibly discarding the incorrect network and generating a different one. More detail on the graph printing python script is provided in Paragraph 4.2.2.

With the assumption that every terminal is connected to one and one hub only, creating TH, which is a NHUBS×NTERMINALS matrix, simply consist of iterating through NTERMINALS rows and extracting a random number between 1 and NHUBS for each row.

Finally, given the assumption that terminals are never connected to each other, TT is an identity matrix.

Then, the total incidence matrix has size NNODES×NNODES and is composed as follows:

$$\text{Incidence matrix} = \begin{pmatrix} \text{HH} & \text{TH}^T \\ \text{TH} & \text{TT} \end{pmatrix}$$

¹⁷ The same nomenclature used in AIMMS is here adopted. Therefore: NHUBS and NTERMINALS are the numbers of hub and terminal nodes in the network, respectively; NNODES equals NHUBS + NTERMINALS; TMAX is the last timepoint (remember the schedule is TMAX+1 timepoints long).

4.2.1.2 Distances matrix

After subtracting an identity matrix from the incidence matrix created as explained above, creating a distances matrix consists of generating a random value for every 1 in one half of the resulting matrix, and then mirroring it to make sure distances are symmetric.

To find a reasonable interval from which to extract distances, the DB Cargo (2025) website has been consulted, using the Germany network as reference. Follows the distances matrix that has been produced after an accurate analysis of the source.

	Maschen	Seelze	Seddin	Hagen-Vorhalle	Oberhausen	Köln Gremberg	Mannheim	Saarbrücken	Nürnberg	München Nord	Basel	Halle (Saale)
Maschen	0	180	327	368	0	0	0	0	0	0	0	0
Seelze	180	0	295	228	0	0	490	0	528	0	0	242
Seddin	327	295	0	0	0	0	0	0	0	0	0	170
Hagen-Vorhalle	368	228	0	0	87	92	0	0	0	0	0	0
Oberhausen	0	0	0	87	0	100	0	0	0	0	0	0
Köln Gremberg	0	0	0	92	100	0	275	0	0	0	0	0
Mannheim	0	490	0	0	0	275	0	151	355	386	274	521
Saarbrücken	0	0	0	0	0	0	151	0	0	0	0	0
Nürnberg	0	528	0	0	0	0	355	0	0	198	0	333
München Nord	0	0	0	0	0	0	386	0	198	0	0	0
Basel	0	0	0	0	0	0	274	0	0	0	0	0
Halle (Saale)	0	242	170	0	0	0	521	0	333	0	0	0

Table 2: Distances matrix from the German rail network (DB Cargo, 2025)

The matrix shows that a network that extends through the whole country has distances in the interval from 87 to 528 Km. However, to allow faster travels and better exploit the full length of the schedule, it has been decided to consider a less spread network, drawing distances in the interval between 50 and 200 Km. In the next Paragraph, a detailed explanation on how the travel times matrix is generated clarifies the reasoning behind this choice.

4.2.1.3 Fixed costs matrix

Following the approach by Boland et al. (2017), fixed costs for services are assumed proportional to the service length in kilometers, with a coefficient of 0.55 dollars per mile. An online calculator that relies on the Consumer Price Index from the United States has been used to roughly estimate inflation from 2017 to 2025, then dollars has been converted to euro (with the exchange rate as of January 2025) and miles have been converted to kilometers. Being the maximum accuracy on financial computations outside the main scope of this work, the final result of the aforementioned conversions can be rounded to a coefficient of 0.43 euros per kilometer, then named FC_MF (for *Fixed Cost Multiplying Factor*).

The fixed costs matrix is obtained by multiplying the distances matrix by FC_MF . Holding operations are assumed costless, and modeled by the diagonal of the fixed costs matrix being all zeros.

4.2.1.4 Variable costs matrix

Finding an accurate variable cost per container constitutes a very difficult and time-consuming challenge, not having official data directly available from service providers. Remembering that in order to obtain an accurate result it is sufficient to feed the model with real data instead of – still realistic but – generated one and that the scope of this work is to show the functioning of a model, not to give real-life advice, it has been chosen to compute the variable cost of a service so that a full length train would have a total cost equal to two times its fixed cost. Since the upper threshold on train length is set to 65 containers, the variable costs matrix is obtained by multiplying the fixed costs matrix by a factor of $1/65$.

4.2.1.5 Travel times matrix

Crainic et al. (2018) uses a coefficient of 50 to convert a travel length in time to a fixed cost: already having the travel fixed costs matrix, it is sufficient to multiply it by this coefficient, which can be renamed $TIME_MF$ (for *Time Multiplying Factor*) and has a value of $1/50$, to obtain the travel times matrix.

It is however necessary to check the granularity of timepoints. From the work by Crainic et al. (ibid.), it is possible to compute a “travel speed” of circa 116 Km per time unit; comparing it with the aforementioned data obtained from DB Cargo (2025), it has been found reasonable to assume that a time unit corresponds to a day, also considering all processing operations, as a model-simplifying assumption, already included in the travel time. Since the time granularity of the present model has been increased to half-days, the result of the multiplication between a travel length in Km and TIME_MF is first rounded to the closest multiple of 0.5, then multiplied by 2. To recap:

$$TravelTimes(n1, n2) = 2 * \text{round_to_multiple}(\text{TravelFixedCosts}(n1, n2) * \text{TIME_MF}, 0.5)$$

Being `round_to_multiple` a function that rounds the value given as the first parameter to the closest multiple of the value given as the second parameter.

Finally, it is important to make sure the diagonal of the travel times matrix is filled with ones, as that is the length of holding arcs.

4.2.2 Graph printer

The `networkx` python library is used to print a graph for each network generated with the aforementioned algorithm. As already mentioned, the printed graph assumes a significant role for the user to check the correctness of the network, before using it to generate a test instance for the model.

The graph printer script gives as output an image of the network, with hub nodes in a bigger size and different color from the terminal ones. Connecting edges are drawn based on the

network incidence matrix, from which an equal size identity matrix is subtracted to avoid drawing meaningless edges connecting every node to itself.

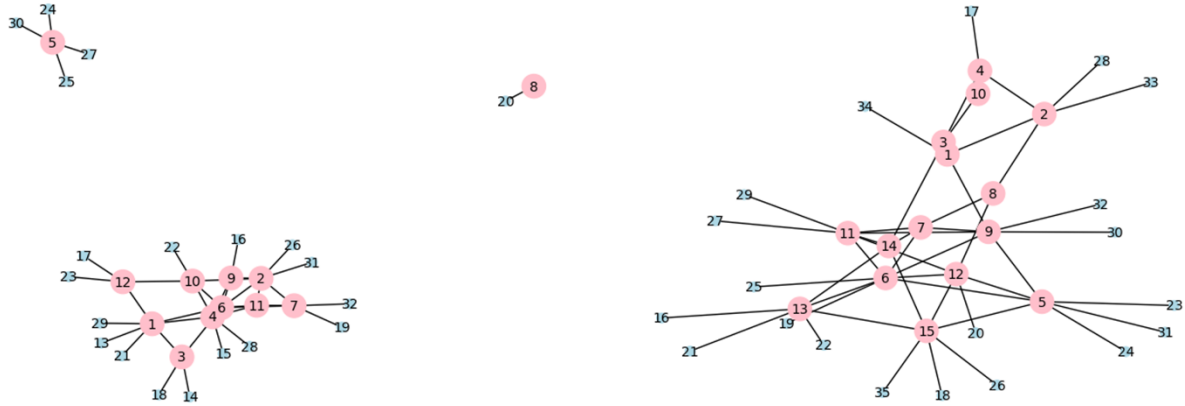


Figure 24: Comparison between a not-suitable hub-and-spoke network and a correct one

Figure 24 above shows the results of giving the graph printer as input the incidence matrices of one network that is not suitable to generate test instances (on the left) and of one that is (on the right).

4.2.3 Commodities generator

The commodities generator python script gets as input the characteristics of a network, namely: size, number of timepoints, incidence matrix and travel times matrix, and gives as output the desired number of commodities, each one with: availability and due timepoints, source and sink terminals, volume (in containers), size of the rental order at source and size of the rental order at sink.

The n desired commodities are generated by running the *generate_commodity* function n times. This function requires as input, other than the parameters mentioned above: a range from which to draw the volume of each commodity, a TL_MF (abbreviation for Time Length Multiplying Factor), a MEAN parameter. Follows a detailed description of this function, in which the usage of each parameter is explained.

First, a couple of terminals is randomly extracted and the shortest path in timepoints between them is computed and multiplied by TL_MF , the result is renamed SP_w_margin : if it is longer than the schedule length, the couple is discarded and another one is drawn. The TL_MF serves to ensure a time margin on the delivery of the commodity, on top of the shortest path length. Suppose a margin of 20% needs to be guaranteed: the TL_MF is set to 1.2 (a non-integer SP_w_margin is always rounded up). Then, the availability timepoint (named *early_time*) is randomly extracted; to make sure the instance is feasible, the interval from which the extraction is conducted goes from 0 to $(TMAX - SP_w_margin)$. The commodity is due at a *late_time* that equals $(early_time + SP_w_margin)$.

The volume of the commodity is drawn from the range which has given as input. Finally, the two rental orders lengths (at source and at sink) are drawn from a Gaussian distribution, whose mean is set equal to the $MEAN$ parameter and the standard deviation is set to 1; drawn values are rounded to the closest integer. The value of the $MEAN$ parameter can be arbitrarily set; in most of the test instances used in the present work, it has been set proportionally to the schedule length, allowing customers to ask for longer rental orders, the longer the considered schedule. In particular, $MEAN$ was set to:

- 1 when $TMAX = 14$;
- 2 when $TMAX = 28$;
- 4 when $TMAX = 56$;
- 6 when $TMAX = 84$.

4.2.4 Selected test instances

Table 3 gathers the characteristics of all the instances used to test the features of the model.

Instance Code	N. Timepo- ints	N. Hubs	N. Termi- nals	Commo- dity Class	N. Commo- dities	Commo- dities Size	TL_MF	Acquisi- tion Price	Rental Fee	MEAN
01-r	14	15	20	R2	50	50-65	1.2	1500	5	1
01-s	14	15	20	R2	50	50-65	1.2	1500	5	1
02	28	15	20	R2	50	50-65	1.2	1500	5	2
03	56	15	20	R2	50	50-65	1.2	1500	5	4
04	84	15	20	R2	50	50-65	1.2	1500	5	6
05	28	15	20	R1	150	20-30	1.2	1500	5	2
06	56	15	20	R1	150	20-30	1.2	1500	5	4
07	84	15	20	R1	150	20-30	1.2	1500	5	6
08	56	15	20	C	150	20-30	1.8	1500	5	4
09	84	15	20	C	150	20-30	1.8	1500	5	6
10	84	15	20	P	150	20-30	1.2	1500	5	8
10-1	84	15	20	P	150	20-30	1.2	800	10	8
10-2	84	15	20	P	150	20-30	1.2	400	20	8
10-3	84	15	20	P	150	20-30	1.2	200	40	8
11	168	15	20	N	300	random	1.2	1500	5	6

Table 3: Summary of features of the test instances

4.3 KPIs

This Paragraph explains which indicators has been chosen to determine whether the model behaves correctly or not, and how are they computed. The KPIs have been introduced in AIMMS as parameters, so that they could be saved on a report .txt file after running every instance.

The main KPI clearly is the TotalCost, which is the variable that the OF aims to minimize. The IniatialAllocation parameter is used to print the values of the AcquisitionVariable variables only when they are non-zero.

TotalAcquiredResources is the sum of the acquisition variables through all the terminals; it will be compared with the Total_Commodities_Volume in order to obtain a containers-to-commodity-volume ratio, which is more accurate when comparing how efficiently the model could solve each instance.

TotalCommodityFlow is the sum of all commodities that flow through travel arcs through the whole schedule; it is therefore computed as the sum of all *CommodityFlow_x* variables, excluding the ones in which the destination equals the origin (that model, instead, holding arcs). It is not of great significance by itself, but it is used when computing the AvgSizeOfServices.

TotalRepositioning is the sum of all empty containers that flow through travel arcs through the whole schedule; it is computed similarly to TotalCommodityFlow, but using *ResourceFlow_y* variables.

NofServices counts the total number of services that are run through the whole schedule; it is the number of arcs for which origin and destination differ from each other and at least one between the *CommodityFlow_x* variables and the *ResourceFlow_y* variable is non-zero and, granted by the ServiceCorrectness constraint, the *Service_z* variable equals 1.

AvgRepositioning is the ratio between TotalRepositioning and NofServices, stating how many empty containers, on average, are repositioned through each service. Clearly it is, together with the TotalRepositioning, an indicator of how efficient the model is in repositioning resources.

AvgSizeOfServices is given by the formula:

$$\frac{\text{TotalCommodityFlow} + \text{TotalRepositioning}}{\text{NofServices}}$$

Together with the NofServices, it is used to determine whether, for the current instance, the model opts for larger services or more frequent smaller ones.

TotalRentalOrders counts how many rental orders are placed in the current instance, that is: the number of commodities multiplied by two (one order at the source and one at the sink), minus the number of rental orders having zero as length.

TotalSatisfiedRentalOrders gives the number of rental orders that the model could fulfill.

PercentageSatisfiedRentalOrders is given by the ratio between TotalSatisfiedRentalOrders and TotalRentalOrders, and constitutes the main KPI when determining how the model behaves with rentals.

TotalProfitRentals gives the total profit from satisfied rentals.

4.4 Results

Table 4 gathers all the results obtained from the test instances, divided in batches. Follows a detailed comment and analysis.

Instance	TotalCost	Total Commodities Volume	Total Acquired Resources	Containers to Commodities		Total Repositioning	N. of Services	Avg Repositioning	Avg Size of Services	Total Satisfied Rental Orders	Total Rental Orders	Satisfied Rental Orders (%)	Total Profit from Rentals
				Volume	Commodities								
01-s	4320181.2	2878	2878	1.00		3871	214	18.1	71.1	54	66	82%	20830
01-r	5151361.0	2878	3428	1.19		3624	205	17.7	73.0	39	66	59%	14230
02	3620203.7	2877	2425	0.84		3623	266	13.6	56.7	78	92	85%	42590
03	1945906.0	2872	1332	0.46		4992	330	15.1	49.6	80	100	80%	79080
04	1451586.1	2895	1045	0.36		5411	343	15.8	46.5	86	100	86%	142945
05	3937931.4	3713	2642	0.71		2815	579	4.9	29.1	214	280	76%	53845
06	2146190.4	3729	1477	0.40		2856	716	4.0	24.2	228	300	76%	98505
07	1336150.3	3704	929	0.25		4586	880	5.2	22.6	149	300	50%	89710
08	2718664.0	3725	1850	0.50		3061	709	4.3	24.8	200	300	67%	85875
09	1725843.0	3719	1200	0.32		4026	807	5.0	22.9	178	300	59%	104660
10	1395028.8	3684	977	0.27		4461	868	5.1	22.2	142	300	47%	103030
10-1	606469.9	3684	1006	0.27		4237	858	4.9	22.2	158	300	53%	230470
10-2	-349940.7	3684	1651	0.45		3894	809	4.8	23.1	268	300	89%	1042020
10-3	-1755214.9	3684	1836	0.50		2872	736	3.9	24.0	274	300	91%	2152360
11	2212770.2	12311	1714	0.14		13558	1727	7.9	35.9	458	600	76%	462265

Table 4: Summary of the KPIs gathered from tests

4.4.1 Repositioning with large commodities

The first four instances are meant to highlight the capability of the model to efficiently reposition empty containers in a schedule with a reasonably low number of commodities, all of them being large in volume. As expected, the model shows that a longer time horizon longer the considered time horizon (that is, the higher the TMAX parameter) allows a better allocation and more efficient usage of owned resources.

Besides the decrease in the number of acquired resources, that diminishes in instances 02 to 04 from 84% to being 36% of the total volume of commodities delivered, are observable an increase in the total and average repositioning, supported by a higher total number of services, and a strong decrease in the average holding per period. The increases in repositioning assume an even more impactful meaning considering that the last instance contains less than half the number of containers of the first one. It is therefore clear that, the longer the schedule, the more the model is inclined to plan smaller – the average size of services also shows a decrease – and more frequent services containing a higher number of empty containers.

The decreasing in the percentage of satisfied rental orders until instance 03 can be justified with the strong decrease in the number of available resources, together with the fact that rentals does not constitute a strong requirement of the problem and that earnings from rental fees are not large enough to give reason to the acquisition of more resources. With a longer schedule, however, since commodities are more widely spread in time, it is possible to satisfy more rental orders, as shown by instance 04.

An isolated observation must be made on instance 01. The characteristics of the network and commodities, that are randomly generated based on realistic parameters, does not allow to satisfy the requirement of a periodical schedule, that is: it is not possible to deliver all the orders at their due destination and time and reposition the resources so that the state of the network at the end of the schedule is the same as the one at the beginning, all within such a short schedule. Two possible relaxations can be made to allow the system to

solve this infeasible instance. The first option is to relax the constraint on the maximum number of acquirable resources, and observe that the only way to produce a cyclical schedule for the given network and commodities is to acquire containers for almost 120% the total volume of delivered goods. The second option is to allow a longer schedule, to see how many more time-periods would the system require to correct the final allocation of an optimal number of resources until it equates the initial one, observing, however, that in this case the optimal number of resources exactly equates the total sum of all the commodities volumes: the model is optimally scheduling services, but again no optimization on the usage of resources is in fact implemented.

4.4.2 Repositioning with small commodities

Instances 05, 06 and 07 present a different situation from the one proposed in the first batch, having a reasonably high number of commodities, all of them being smaller in volume. Comparing these instances with each other, it is again clear that a longer – and therefore more spread – schedule allows a more efficient repositioning of owned resources and consequently a significantly lower total cost, being the resources acquisition expense its main component. The model opts for frequent and small services, as highlighted by the increase in the number of services and in the average and total repositioning and by the decrease in the average size of services. Again, the average holding presents a substantial decrease, but this is also due to the significantly lower number of resources in the system. Instance 07 presents the best result so far, requiring acquisition of containers for as low as the 25% of the total volume of delivered commodities; however, it also shows the lowest percentage of satisfied rental orders so far, a further evidence that earnings from rentals are of minimal significance if compared with the cost of acquiring more resources, at least with the present values that have been given to the ACQUISITION_PRICE and RENTAL_FEE parameters. Specific experiments on these parameters are conducted through the fourth batch of instances and commented in Paragraph 4.4.4.

It is worth of interest comparing instances 07 and 04, that start from the same network and the same schedule length but have several small commodities and few larger commodities, respectively. Having small commodities can exploit the convenience in frequently running small services instead of less frequent, larger ones, as it allows to have the owned resources more widely spread through the network. It is important to notice that this result heavily relies on the research done to quantify travel fixed costs, as an increase in these parameters could favor holding and consolidation and, eventually, the acquisition of more resources.


4.4.3 Consolidation

Instances 08 and 09 have been generated with more time between the availability and the due time, using a multiplying factor of 1.8, against the 1.2 normally used. Comparing instance 08 with 06 and 09 with 07, a reduction in the number of services is observable, together with an increase in the average service size. This is, however, very weak evidence in support of consolidation, as the total number of acquired resources is, in both comparisons, higher when the commodities timespan is longer. The main result of having a longer timespan for orders is presumably, instead, that commodities *need* to be held for longer time periods throughout their journey, introducing a systematic waste of time in the usage of containers. With a higher TMAX, the result is slightly more positive, since a longer schedule attenuates the negative impact of the long commodities time-spans, as shown by the containers-to-commodity-volume ratio, that is around 50% for instance 08, but around 32% for instance 09; moreover, instance 08 shows a decrease in the percentage of satisfied rental orders when compared with instance 06, while an increase is observable when comparing instances 09 and 07. Once again, it is essential to consider that these results heavily rely on the value that is attributed to travel fixed costs; moreover, it is important to notice that holding is assumed costless.

4.4.4 Rental profits

As already mentioned, with realistic resource price and rental fee parameters, the profit from rentals is not high enough to justify the acquisition of more resources; rentals are, however, an opportunity to realize some profit on resources that would be, otherwise, idle. The last batch of instances is meant to fully show the capacity of the model to take into account rental profits, starting with the generation of longer rental orders. Comparing instance 10 and 07, that only differ from each other in the mean value of the distribution used to generate the time length of rental orders, the containers-to-commodity-volume ratio increases from 25% to 27% and the total cost is still lower because, even if the percentage of satisfied rental orders is lower in instance 10, the total profit from rental increases. Finally, instances 10-1, 10-2 and 10-3 constitute variations of instance 10 and show a completely unrealistic scenario, in which the resource price dramatically decreases and the rental fee increases, with the purpose of showing that a significant change in the ratio between those two parameters could potentially justify acquiring more resources with the sole purpose of renting them. These scenarios are clearly utopian, since the ratio is so unbalanced that the total cost even becomes negative, being at all effects a *total profit*, but make the capacity of the model to account for rental orders fully noticeable, a feature that is not as appreciable with realistic parameters.

To conclude, instance 11 is meant to show the behavior of the model with very large instances, in this specific case by using the same network as before, a 168 timepoints schedule (that equals to 12 weeks, 3 months) and 300 commodities, whose volume is randomly chosen between the two aforementioned intervals (20-30 for small commodities and 50-65 for large ones). The KPIs show a positive result, with a containers-to-commodity-volume ratio around 14% and a total cost that is approximately one and a half times the total cost of instances 04 and 07, while delivering approximately 3.5 times their volumes of commodities; moreover, it shows a valuable 76.3% of satisfied rental orders. These results are meant to support with data the intuitive idea that the longer the schedule, the easier and more



efficient the reallocation of resources and the better the amortization of their acquisition expense.

5 Conclusions

5.1 Managerial insights

As mentioned, the intended outcome of the model is the optimal number of resources to be acquired and their initial allocation, which is obtained by routing commodities and supporting the selected services with the necessary resources. Therefore, the optimized schedule of services is also an essential component of the outcome.

The model could be however equally helpful in finding the best schedule of commodities, that is, for a firm operating in logistics, finding how many orders to ship and in which time intervals in order to optimize the usage of owned resources. A good starting point is, for example as suggested by the results collected from the test instances, that delivering frequent small commodities allows a more optimized usage of containers than when delivering fewer large ones, thanks to their more scattered positioning in the network throughout the schedule length, but only if the travel fixed costs are sufficiently low.

This connects to another important remark that must be made when suggesting using not only the present model, but every decision-making tool, that is the importance of giving input parameters an accurate and realistic values. The parameters used in the test instances are realistic, since derived from previously published research papers, when possible, and from accurate research on publicly available data from the industry; however, the model necessarily requires using actual data and creating accurate instances, when used to provide real decision-making advice.

Finally, even though the size and topology of the network and the schedule length can be considered realistic, what instead can be increased is the number of commodities, which has been kept at a reasonably low value when running the tests. A heavy increase in the number of commodities could indeed imply the necessity of a purposely crafted solving algorithm, which is outside the scope of this work, but could constitute a further development. Other suggestions for possible future research based on the result of this work are discussed in Paragraph 5.3.

5.2 Limitations and obstacles

Although starting from existing research, this work introduces significant elements of novelty, namely a variable to account for empty containers and a dedicated set of arcs for rental orders, which limit the possibility to use existing modelling techniques and solving methods. Therefore, being the scope of this work limited to the model only, many assumptions have been introduced in order to be able to solve test instances without requiring the development of dedicated algorithms. It is particularly important to remark that the constraints on train length had to be relaxed; it is however observable that, to avoid unnecessary services and related travel fixed costs, the model is never proposing unreasonably small services.

A first draft of the model would have introduced (un-)loading arcs, serving as a point of contact between the x and the y variables, accounting for laden and empty containers respectively, and modelling the required time interval for these operations. To avoid introducing too many variables, and consequently the risk of the software not being able to provide a solution without requiring heuristic techniques, the granularity of time has been increased up to half-days, a level which is definitely not high, but still reasonably realistic. Meticulous research to quantify the amount of time required for processing operations at shunting yards revealed that this amount is in the order of minutes, a few hours at maximum. It has been decided, therefore, to abandon the idea of (un-)loading arcs and assume the processing time as included in the travel time between nodes, an assumption that, without leading to model an unrealistic scenario, allowed to avoid introducing numerous new variables and significantly simplify computations.

The unavailability of test instances constituted a relevant obstacle, which has been overcome with research about real-life scenarios and the generation of test instances based on the collected data, as explained in detail in Paragraph 4.2. Since the model introduces several elements of novelty, the objective of the computational study was not to show an improved performance, but to present the functioning of the new features. The possibility

to create the test data has therefore become a useful opportunity to craft instances that would highlight the new features at best, with the help of dedicated KPIs: a change of a KPI in the anticipated direction after altering specific parameters in the test instance could lead to the conclusion that the model behaved as expected.

It is however necessary, as a final remark on encountered obstacles, to specify that the issue that prevented the implementation of even bigger test instances has been the difficulty in connecting the python language and the AIMMS software. The difficulty in accessing AIMMS external libraries indeed hindered smoothly proceeding through third party file formats, such as .csv or .xlsx, and led to the necessity to save the generated data on Excel sheets and manually copy-paste them on AIMMS, an operation that became infeasible when the instances would become too large.

5.3 Final remarks and future research

This work proposes a model for SND problems on hub-and-spoke networks, introducing an additional variable to track the movement of empty containers and allowing their efficient usage and repositioning, while delivering multiple commodities, each of them departing from a source node at an availability time and arriving at a sink node at a due time. Other than the services schedule, the model also provides an initial allocation of resources, taking into account their acquisition price. Furthermore, the possibility to rent idle resources to customers at terminals as storage space is introduced as a source of additional profit.

Starting from real data, several test instances are generated and, observing the variation in dedicated KPIs, the behavior of the model is analyzed and the aforementioned features are observed.

Despite behaving as expected and producing satisfying results, a significant number of simplifying assumptions had to be taken for the model to work without implementing *ad hoc* algorithms, a significant number of many simplifying assumptions had to be taken. Future work could entail, therefore, updating the model to relax these assumptions

increasing its capacity to model real cases and implement the aforementioned *ad hoc* algorithms. Specifically, possible future research includes:

- Improving integration with crafted data, possibly generating instances through a dedicated interface directly in AIMMS;
- Reintroducing train length constraints, possibly allowing multiple services on the same arc, that is an integer z variable instead of a binary one;
- Introducing capacitated hubs and, possibly, different processing times and costs depending on the hub size;
- Integrating the present work with a model to optimize shunting yard operations, increase the time schedule granularity and compute (un-)loading times more accurately;
- Exploring the possibilities that a cyclical schedule could offer, e.g. allowing commodities availability time to be after their due time, and ascertain whether they could lead to a better final result;
- Introducing uncertainty, e.g. in availability and due times and in the rental orders time-length, possibly using fuzzy numbers or probability distributions.

The demand for freight transport keeps increasing over years and decades. If it is important to support this growth with the appropriate technology, it is however essential to plan its effective and considerate usage with the objective not only to make transport efficient (and therefore profitable) but also, and more importantly, to avoid wasting important scarce resources, building a network that is environmentally and socially optimized ideally, acceptable at least. For this purpose, Operations Research has a key role nowadays and this work has the goal to add something, even if it is the smallest grain, to the vast amount of knowledge that many brilliant minds have produced from the early days of this Science and will keep producing in the upcoming years.

References

- Andersen, J., Crainic, T.G. and Christiansen, M., 2009. Service network design with asset management: Formulations and comparative analyses. *Transportation Research Part C: Emerging Technologies*, 17(2), pp.197-207.
- Barnhart, C., Krishnan, N., Kim, D. and Ware, K., 2002. Network design for express shipment delivery. *Computational Optimization and Applications*, 21, pp.239-262.
- Bektaş, T., 2017. *Freight transport and distribution: concepts and optimisation models*. CRC Press.
- Boland, N., Hewitt, M., Marshall, L. and Savelsbergh, M., 2017. The continuous-time service network design problem. *Operations research*, 65(5), pp.1303-1321.
- Crainic, T.G., 1999. Long-haul freight transportation. In *Handbook of transportation science* (pp. 433-491). Boston, MA: Springer US.
- Crainic, T.G., Hewitt, M., Toulouse, M. and Vu, D.M., 2016. Service network design with resource constraints. *Transportation science*, 50(4), pp.1380-1393.
- Crainic, T.G., Hewitt, M., Toulouse, M. and Vu, D.M., 2018. Scheduled service network design with resource acquisition and management. *EURO Journal on Transportation and Logistics*, 7, pp.277-309.
- Crainic, T.G. and Rei, W., 2024. *50 Years of Operations Research for Planning Consolidation-based Freight Transportation*. Bureau de Montreal, Université de Montreal.
- DB Cargo, 2025. Available: <https://network.dbcargo.com/network-en/routes/germany> (Accessed 16.02.2025, Created 22.06.2024, Last changes 14.01.2025).
- European Commission, 2011. *Roadmap to a Single European Transport Area – Towards a competitive and resource efficient transport system*. Available: <https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2011:0144:FIN:en:PDF> (Accessed 07.11.2024, Created 06.04.2011, Last changes 04.08.2024).
- European Commission: Directorate-General for Mobility and Transport, 2021. *EU transport in figures – Statistical pocketbook 2021*. Publications Office.
- European Commission, 2024. *Freight transport statistics – modal split*. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Freight_transport_statistics_-_modal_split (Accessed 04.11.2024, Created 15.04.2018, Last changes 18.10.2024).

European Court of Auditors, 2023. Special report 08/2023: Intermodal freight transport: EU still far from getting freight off the road. Available: https://www.eca.europa.eu/Lists/ECA-Documents/SR-2023-08/SR-2023-08_EN.pdf

(Accessed 07.11.2024, Created 27.03.2023, Last changes 16.05.2024)

Frisch, S., Hungerländer, P., Jellen, A., Lackenbacher, M., Primas, B. and Steininger, S., 2023. Integrated freight car routing and train scheduling. *Central European Journal of Operations Research*, 31(2), pp.417–443.

Gass, S.I. and Assad, A.A., 2005. *An annotated timeline of operations research: An informal history* (Vol. 75). Springer Science & Business Media.

Gallier, J. and Quaintance, J., 2019. *Fundamentals of optimization theory with applications to machine learning*. University of Pennsylvania Philadelphia, PA, 19104.

Islam, D.M.Z., Ricci, S. and Nelldal, B.L., 2016. How to make modal shift from road to rail possible in the European transport market, as aspired to in the EU Transport White Paper 2011. *European transport research review*, 8, pp.1–14.

Jansen, B., Swinkels, P.C., Teeuwen, G.J., de Fluiter, B.V.A. and Fleuren, H.A., 2004. Operational planning of a large-scale multi-modal transportation system. *European Journal of Operational Research*, 156(1), pp.41–53.

Krarup, J. and Vajda, S., 1997. On Torricelli's geometrical solution to a problem of Fermat. *IMA Journal of Management Mathematics*, 8(3), pp.215–224.

Liu, L. and Yang, X., 2020. A Model and an Algorithm for Empty Car Distribution in Railway Transportation. In *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery: Volume 1* (pp. 123–131). Springer International Publishing.

Oxford English Dictionary, s.v. “operational research (n.)”. Available: <https://doi.org/10.1093/OED/8237798379>

(Accessed 06.11.2024, Created 31.07.2023, Last changes 31.07.2023.

Pedersen, M.B., Crainic, T.G. and Madsen, O.B., 2009. Models and tabu search metaheuristics for service network design with asset-balance requirements. *Transportation Science*, 43(2), pp.158–177.

Simon, H.A., 1955. A behavioral model of rational choice. *The quarterly journal of economics*, pp.99–118.

Stahlbock, R. and Voß, S., 2008. Operations research at container terminals: a literature update. *OR spectrum*, 30, pp.1–52.

StadieSeifi, M., Dellaert, N.P., Nuijten, W., Van Woensel, T. and Raoufi, R., 2014. Multimodal freight transportation planning: A literature review. European journal of operational research, 233(1), pp.1-15.

Wieberneit, N., 2008. Service network design for freight transportation: a review. OR spectrum, 30(1), pp.77-112.

Zhu, E., Crainic, T.G. and Gendreau, M., 2014. Scheduled service network design for freight rail transportation. Operations research, 62(2), pp.383-400.