Multi-warehouse stock allocation in a large-scale, multi-product e-commerce environment

Eijsvogels, G.A.

Award date:
2021

Eindhoven University of Technology

Master Thesis

Multi-warehouse stock allocation in a large-scale, multi-product e-commerce environment

Author:

G.A. Eijsvogels (Guus) 0956264

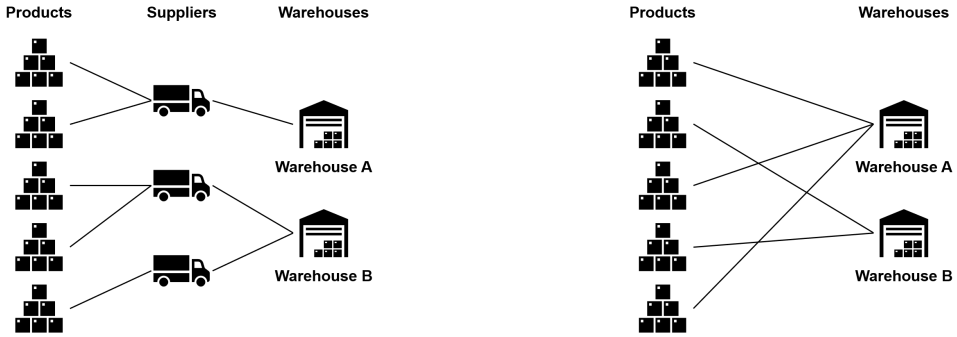
Supervisors:

Dr. S. Dabadghao (Shaunak)	TU/e
Dr. A. Marandi (Ahmadreza)	TU/e
C. Wisse (Christianne)	Company X
P. van de Ven (Peter)	Company X

October 12, 2021

Management Summary

Introduction
This research considers the stock allocation problem, defined as the decision on how to allocate inventory of products over the different warehouses such that capacity is optimally utilized, and costs are minimized (Marklund & Rosling, 2012). The effects of decreasing the aggregation level of a multi-warehouse stock allocation model from supplier-level to product-level in a large-scale, multi-product e-commerce environment are analyzed (Figure 1). To do so, the use-case of Company X is considered, one of the largest e-commerce providers in The Netherlands.



(a) Supplier-level allocation (b) Product-level allocation

Figure 1: Difference in decreasing the aggregation level of the stock allocation.

While there is literature available looking at the effect of changing the aggregation level of optimization on scalability, it is mentioned that in the domain of e-commerce stock allocation such research is scarce. Furthermore, X. Li, Zheng, Zhou, and Zheng (2019) stated that "*the optimal allocation of multiple products at multiple warehouses, though important, is not fully studied in the literature*".

Methods
The allocation models both solve a MILP, which has the objective of total cost minimization, subjected to four sets of constraints:

1. Outbound line capacity constraints
2. Full demand assignment constraints
3. Warehouse-exclusivity constraints
4. Unique constraints imposing a minimum ratio of distribution of allocated demand over the available warehouses (pand-split).

Using the input data from Company X, computational experiments are conducted and results are gathered. The models and methods used are presented in Table 1.

Table 1: Overview of models used in this research.

Model	Description
SAM	The supplier-level allocation model SAM. It uses the open-source CBC (CoinOR Branch and Cut) solver. Since PLS is not using certain details of the model environment, SAM is also simplified to mirror the PLS model complexity to be able to fairly compare the models.
PLS-GUR	The product-level allocation model PLS using the commercial Gurobi solver.
PLS-CBC	The product-level allocation model PLS using the open-source CBC solver.

Models are evaluated on **scalability** and **optimality**. Besides, a use-case analysis is conducted to retrieve operational insights on the new aggregation level, focusing on capacity utilization, supplier splits,

the pand-split constraints and warehouse characteristics. Different configurations and thus different model scales are evaluated. The parameters used for this are H and T. H is defined as the amount of historical data included in the computation of the product-level forecast, influencing the amount of products to be allocated in the model. T is the optimization period.

Results
Scalability:

Table 2: Results scalability and optimality runs.

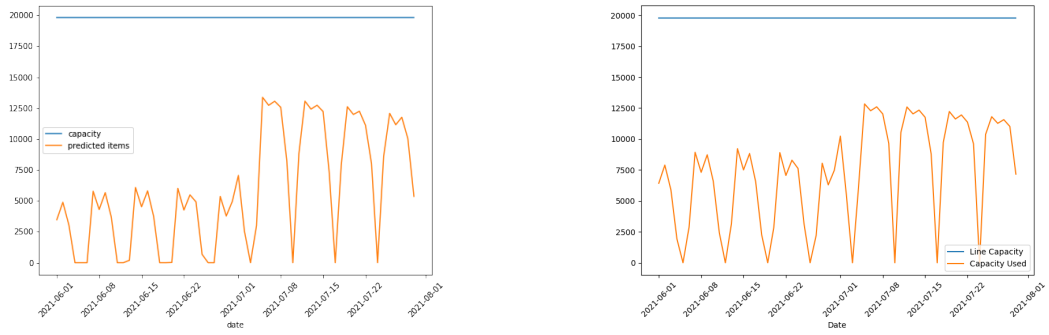
	$H=2, T=2$			$H=7, T=7$			$H=14, T=14$			$H=31, T=31$			$H=60, T=60$		
Model	**SAM**	**PLS-GUR**	**PLS-CBC**	**SAM**	**PLS-GUR**	**PLS-CBC**	**SAM**	**PLS-GUR**	**PLS-CBC**	**SAM**	**PLS-GUR**	**PLS-CBC**	**SAM**	**PLS-GUR**	**PLS-CBC**
Variables	3353	119704	119704	3778	223760	223760	4373	314492	314492	5818	439200	439200	8283	588082	588082
Constraints	1803	59878	59878	2283	111961	111961	2955	157404	157404	4587	219945	219945	7371	294705	294705
Run-time	0.22	1.08	9.54	0.75	9.47	92.75	0.93	30.25	422.93	2.62	144.40	1722.12	25.16	658.21	6067.32

Optimality:

Table 3: Cost reduction PLS over SAM different configurations.

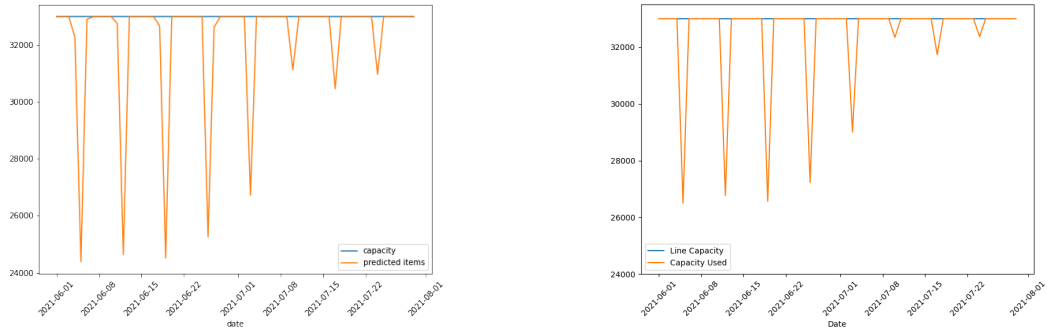
Configuration	$H=2, T=2$	$H=7, T=7$	$H=14, T=14$	$H=31, T=31$	$H=60, T=60$
Total cost reduction	€4582.72058	€14220.3336	€28043.2427	€57281.4922	€86739.8213
Cost reduction in % of SAM objective	1.48903%	1.42645%	1.37520%	1.23252%	0.86600%
Cost reduction per day	€2291.36029	€2031.47623	€2003.08877	€1847.79007	€1445.6637
Supplier splits	92,44%	93,83%	94,74%	95,34%	93,56%

Capacity Utilization:



(a) SAM (b) PLS

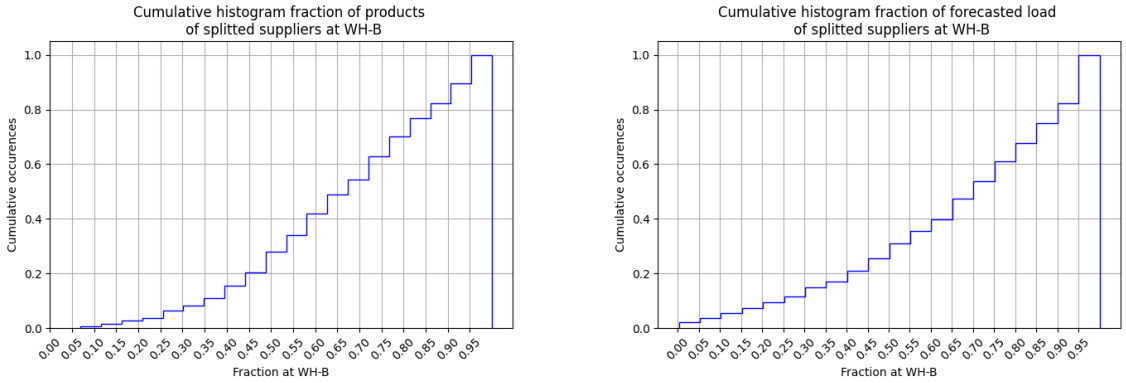
Figure 2: Capacity utilization results for a single outbound line (1).



(a) SAM (b) PLS

Figure 3: Capacity utilization results for a single outbound line (2).

Supplier splits:



(a) Products

(b) Forecasted demand

Figure 4: Supplier split results for amount of products and forecasted demand level of a supplier's assortment.

Conclusion & Discussion

Multiple conclusions are formulated from this research:

- Despite the large scale increase, it is feasible to solve the product-level allocation model using mathematical optimization software.
- A cost reduction of $\pm 1\%$ is realized decreasing the aggregation level from supplier-level to product-level, caused by better capacity utilization.
- Supplier splits are present for a substantial part of the suppliers, but it might be operationally favourable to not split suppliers where the majority of the assortment is allocated to a single warehouse. Therefore, a hybrid allocation level using multiple aggregation levels is recommended.
- Product-level allocation is a solid solution for the as-is situation, and will become even better in the future given the expansion plans of Company X.

While this research has contributed to the domain of multi-product, multi-warehouse stock allocation in an e-commerce environment, future research is possible. The model can be extended by e.g. introducing the complexity of split shipments, demand stochasticity, including other aggregation levels, or adding more warehouses.

Abstract

This research considers the stock allocation problem, defined as the decision on how to allocate stock of products over the different warehouses such that capacity is optimally utilized and costs are minimized. The effects of decreasing the aggregation level of a multi-warehouse stock allocation model from supplier-level to product-level in a large-scale, multi-product e-commerce environment are analyzed with respect to scalability and optimality. To do so, the use-case of Company X is considered, one of the largest e-commerce providers in The Netherlands. The stock allocation problems we consider are modelled as mixed-integer linear programming problems (MILP) considering capacity constraints, demand assignment constraints, warehouse exclusivity constraints and the unique constraints imposing a ratio of distribution of total forecasted demand over the available warehouses. These models are solved using the Gurobi and CBC mathematical optimization software. Computational experiments indicate that the decrease of the aggregation level is feasible from both an operational and technical perspective.

Keywords: Stock allocation, e-commerce, inventory management, combinatorial optimization, large-scale, multi-warehouse, multi-product

1 Introduction

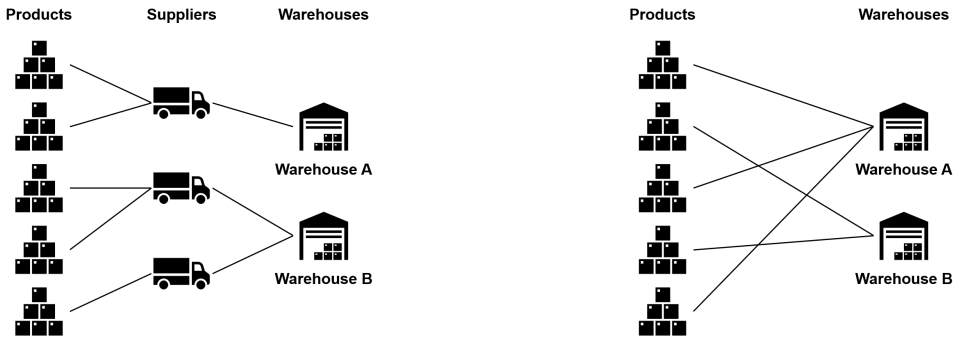
E-commerce has thrived on the technical developments of the internet since the 1990's and has evolved into an on-demand retail service that can be accessed at every moment of the day (Santos, Sabino, Macedo Morais, & Gonçalves, 2017). These innovations in online retailing or e-tailing generated opportunities of developments in the associated logistical processes. Physical stores became less necessary and the shipment process of sold goods increased in importance (de Koster, 2002). It became evident that e-commerce created opportunities to operate on larger scales compared to offline retailing, which induced a focus on quantitative modeling and analyses of the ongoing processes (Agatz, Fleischmann, & van Nunen, 2008). According to de Koster (2002), one of the biggest challenges these developments bring is the organisation of the e-fulfillment process. Inventory management is a domain in the e-fulfillment process that has been researched thoroughly over the years. Complex models have been developed to determine optimal inventory levels, replenishment strategies, stock allocation over warehouses and inside warehouses, facility locations, etc. (Jolayemi & Olorunniwo, 2004). As the years passed, developing technologies increased the availability and accessibility of processing data (Bertsimas, Kallus, & Hussain, 2016). This created opportunities for further optimization of inventory management, for example through stock allocation, which is the domain we focus on.

Stock allocation is defined as the decision on how to allocate stock of products over storage locations such that demand is fulfilled, capacity is optimally utilized and an objective is optimized (Marklund & Rosling, 2012). Different industries have to deal with the decision of stock allocation, of which the retailing industry is one. As scales are increasing, offline retailers have to deal with the allocation of product assortment of a central warehouse to the retail stores (J. Li, Toriello, Wang, Borin, & Gallarno, 2021). E-commerce providers, however, have to make the decision what to stock at the available warehouses, as retail stores are eliminated from the supply chain. There is a significant difference in scale between those two allocation decisions, as the storage capacities of warehouses are substantially larger (Agatz et al., 2008). The goal of stock allocation for an e-commerce provider is to manage inventory such that all online orders can be fulfilled optimally from the available warehouses.

The stock allocation decision can be made at multiple aggregation levels of products, influencing the scalability. While there is literature available looking at the effect of changing the aggregation level of optimization on scalability (Hane et al., 1995; Jélvez, Morales, Nancel-Penard, Peypouquet, & Reyes, 2016; Maltese, Ombuki-Berman, & Engelbrecht, 2018), it is mentioned that in the domain of e-commerce stock allocation such research is scarce. Furthermore, X. Li et al. (2019) stated in their research that *"the optimal allocation of multiple products at multiple warehouses, though important, is not fully studied in the literature"*. Therefore, we focus on analyzing the scalability of the stock allocation decision in a large-scale, multi-warehouse, multi-product e-commerce environment.

To be able to analyze this, a real-world example is considered. This research is conducted using the processes, data, and cooperation of one of the largest e-commerce providers in The Netherlands, hereafter referred to as 'Company X'. Numbers presented have been anonymized and are therefore fictional. Company X provides a very broad assortment, ranging from USB-sticks and fridges to (e-)books, clothing, and travel gear. This is a strategic choice, as the goal is to fulfill as much customer demand as possible in different domains, instead of focusing on a niche market. This 'long-tail' assortment indicates some products stored are only sold occasionally, creating a more complex stock allocation problem. A stock allocation model is used to distribute the assortment over multiple warehouses with the objective of costs minimization and optimal capacity utilization. This current optimization model is referred to as SAM, short for Stock Allocation Model. SAM solves a Mixed-Integer Linear Programming problem (MILP) and allocates assortment over two warehouses: WH-A and WH-B.

SAM allocates on supplier-level, indicating that the full assortment delivered by a supplier is allocated altogether to a single warehouse. This allocation decision is aimed to be exclusive, meaning that a supplier can either be fully allocated to WH-A or to WH-B. SAM allocates all suppliers to the warehouses such that outbound capacities are not exceeded, minimum distributions of processed products over the warehouses are present, there is exclusivity of allocation and all forecasted demand is processed. The resulting allocation decision minimizes the total costs associated. Despite the fact that SAM works properly, it is expected that decreasing the aggregation level of the allocation to product-level is operationally advantageous and increases capacity utilization, which leads to cost reduction. Product-level allocation is the concept of not grouping the products together based on the assortment of a supplier, but to allocate stock of individual products to the warehouses. The difference between those aggregation levels is presented in Figure 5.



(a) Supplier-level allocation (b) Product-level allocation

Figure 5: Difference in decreasing the aggregation level of the stock allocation.

From a technical point of view, the main challenge is the scalability of the allocation problem that needs to be solved. As Company X is a market leader in e-commerce, the scale at which it operates is large. SAM allocates thousands of suppliers, which are responsible for the delivery of hundreds of thousands of products to the warehouses. This increases the problem scale exponentially. If no constraints would be present, allocating 1000 suppliers to two warehouses induces 2^{1000} possible solutions. Allocating $100,000$ products already increases the solution space to $2^{100,000}$ possible solutions. However, the constraints in the allocation problem reduce the size of the solution space as many solutions become infeasible. The feasibility of solving this large-scale allocation problem is not known. The open-source CBC solver is used to tackle the optimization problem of SAM. We are interested whether the new problem scale at product-level would become too large for this solution method. Besides, scales are increasing from multiple sides: the assortment of products is expanding, new warehouses are being added and other processing capacities are being added to the model environment. On the other hand, product-level allocation allows the assortment of a supplier to be split over multiple warehouses, referred to as a 'supplier split', which might be unfavorable. We test multiple problem scales to evaluate problem scale and the effects. Before introducing the allocation problem considered in detail, the theoretical background and academic contribution are elaborated on.

1.1 Literature review

Analyzing literature, it is found that different streams of research focus on different objectives in stock allocation, and use different solution methods to solve such problems. The allocation problem is an instance of the more general assignment problem or matching problem, a combinatorial optimization problem studied in the field of operations research (Krokhmal & Pardalos, 2009). The assignment problem is used to define storage assignment problems which consist of finding the correct location to store a product in a warehouse layout, minimizing costs associated with in-warehouse transportation times (Tabatabaei, Valilai, Abedian, & Khalilzadeh, 2021). We consider a capacitated resource, as warehouses have maximum production capacities. The stock allocation problem under capacitated supply chain restrictions is recurring in literature (Gupta & Selvaraju, 2006; de Véricourt, Karaesmen, & Dallery, 2002). In the domain of e-commerce, multi-product allocation is repeatedly researched, as e-tailers offer broad assortments (X. Li et al., 2019; Lim & Liu, 2018; Sathyanarayana & Patro, 2020; Liu, Zhou, & Zhang, 2010). The formulation of the stock allocation problem however differs per situation.

Sathyanarayana and Patro (2020) optimize the allocation decision of a fashion e-tailer's assortment to warehouses by minimizing delivery costs and delivery times, leading to maximum regional utilization. X. Li et al. (2019) optimally allocate the assortment of a large Chinese e-commerce provider by minimizing total shipping costs given allocation density. Lim and Liu (2018) allocate assortment of a major online Asian fashion e-tailer subjected to capacity constraints while also optimizing order sourcing financially. In some cases of stock allocation, it is possible to let the model determine the optimal inventory levels (e.g. base-stock levels) (de Véricourt et al., 2002; Rappold & Muckstadt, 2000; Akçay, 2002). Rappold and Muckstadt (2000) and de Véricourt et al. (2002) minimize total costs, including holding and backorder costs, by determining optimal stock levels in make-to-stock production environments.

Another stream of research analyzes stock allocation with regards to split shipments, indicating products in a single customer order are sourced from multiple warehouses. Catalán and Fisher (2012) studied the assortment allocation problem in a multi-product, multi-warehouse e-commerce environment to minimize costs associated with splitting shipments. This integer programming problem includes capacity constraints and allows for double-stocking of a product. Ardjmand, Young, Weckman, and Sanei Bajgiran (2018), Acimovic and Graves (2015) consider the problem of order allocation minimizing split shipments in e-commerce environments, connected to the transportation problem. While order allocation is a different perspective than stock allocation, the assignment problem is similar as exclusivity, capacity and demand assignment constraints are present and costs are minimized.

The stock allocation problem considered can be identified in other problem definitions, for example the knapsack problem (KP) formulation (Yang, Chen, Wang, Chang, & Sun, 2010; Akçay, 2002). Yang et al. (2010) described the similarities of a KP to a multi-warehouse, multi-product order allocation problem considering capacity constraints, minimizing total costs. Akçay (2002) determine the optimal base-stock levels and optimal component allocation maximizing revenue for a multi-component, multi-product assemble-to-order problem using the MKP formulation. Moreover, the Capacitated Facility Location Problem (CFLP) is connected to the stock allocation problem, as demand allocation to capacitated facilities is included. Liu et al. (2010) defined the assignment problem of online demand to capacitated regional warehouses as a CFLP, which resulted in a non-linear integer linear programming problem where total associated costs are minimized. Capacity constraints and exclusivity constraints are present, the latter assuring exclusive allocation of demand to a warehouse. Silva and de la Figuera (2007) consider a CFLP with demand assignment. The solved allocation problem is an MILP including capacity constraints and demand assignment constraints. Demand is stochastic and backlogging is possible.

To solve such large-scale stock allocation problems, multiple solution methods have been implemented. Problem-specific algorithms have been designed, providing exact solutions (Tabatabaei et al., 2021; de Véricourt et al., 2002; X. Li et al., 2019). Next to this, problem-specific approximation heuristics have proved to be able to solve the problems (Lim & Liu, 2018; Rappold & Muckstadt, 2000; Catalán & Fisher, 2012; Acimovic & Graves, 2015; Liu et al., 2010; Silva & de la Figuera, 2007). Another popular method is to implement mathematical solvers that utilize branch-and-cut methods to solve an MILP (Yang et al., 2010; Sathyanarayana & Patro, 2020; Akçay, 2002). Solvers are easy to implement and

can be a good candidate for solving the allocation problem if model scales allow. However, there is a possibility that solvers are not able or less favorable to tackle the allocation problem given the increased scale at product-level (Yang et al., 2010; Mohammadi & Musa, 2020).

Additionally, literature has shown that meta-heuristics provide close-to-optimal solutions to a variety of operational problems in a reasonable amount of time (Griffis, Bell, & Closs, 2012). Different instances of meta-heuristics have been implemented to comparable problems. In the category of local search meta-heuristics, tabu search seems to be suitable (Sun, 2005; McKendall, 2008; Wu, Yeh, & Syau, 2004), followed by simulated annealing (Griffis et al., 2012; McKendall, 2008; Malikia, Souierb, Dahanec, & Sarib, 2017). In the category of population search methods, genetic algorithms are used frequently to tackle comparable problems (Yang et al., 2010; Ardjmand et al., 2018; Mohammadi & Musa, 2020; Malikia et al., 2017; Griffis et al., 2012), followed by particle swarm optimization (Soni, Jain, Chan, Niu, & Prakash, 2019; Mohammadi & Musa, 2020; Mousavi, Bahreininejad, Musa, & Yusof, 2017). While these methods seem promising, the starting point in this research are mathematical solvers due to flexibility in implementation. If scalability becomes problematic, extensions to other methods can be considered.

It can be concluded that there are multiple directions in literature that tackle comparable problems to the considered stock allocation problem. Developments of e-commerce and the growing scales at which these organisations operate bring light to new problems that can be optimized to gain an operational advantage. To our knowledge, the literature is not complete in the region of stock allocation in e-commerce that analyzes the effect of decreasing the aggregation level in terms of scalability, along with the operational insights this brings. This research takes a step in filling this research gap by analyzing the scalability of the stock allocation decision in a large-scale, multi-warehouse, multi-product e-commerce setting.

1.2 Contribution

The considered stock allocation problem is different from existing literature. The constraints assuring a minimum distribution of total demand being allocated over the available warehouses were not identified in connected literature. Furthermore, the allocation decision is constrained by rather simple structures, but the large scale makes the decision complex. Demand is deterministic and should be fully and exclusively allocated to a single warehouse, inventory levels per product can not be determined by the model and double-stocking is not allowed. Distribution of demand over the capacitated resources at the warehouses is realized using a unique higher-level aggregation, referred to as 'segments', reducing model scale significantly.

We focus on decreasing aggregation in stock allocation from the supplier side, which is a point of view not well known in literature. Besides, multiple scales are tested to evaluate problem scale and the effects on the model scalability. This contributes to the literature of finding the optimal allocation of multiple products at multiple warehouses at different aggregation levels (X. Li et al., 2019). As mentioned, it is a challenge to find which solution method is required to solve the problem resulting from the new aggregation level, which is investigated. Next to this, interesting operational insights are gathered. We provide a basis for future research, as the model environment is able to be extended in various ways. For example, the complexity of minimizing split shipments, double-stocking, stochastic demand levels, other aggregation levels of allocation, or the addition of more warehouses could be introduced.

To realize those contributions, the remainder of this paper is organized as follows. Section 2 describes the tackled allocation problem in detail and formulates the MILP models at supplier-level and product-level. Section 3 elaborates on the input data used for implementation. The results to the conducted computational experiments are presented in Section 4. In Section 5, those results are discussed. Finally, Section 6 concludes the paper.

2 Problem description and model definition

The allocation problem considered has the goal to distribute the forecasted demand of products over the warehouses such that capacity is utilized optimally while minimizing costs. The allocation models

implemented are slightly simplified compared to reality. These simplifications still allow for a thorough analysis of scalability. Before elaborating in detail on the allocation problem and the models, those simplifications are briefly discussed.

Company X uses a platform strategy, it allows other e-tailers or 'platform partners' to sell and store their products using their channels. Next to this, a considerable part of the sold goods is Company X's own assortment. The parties delivering the own assortment are referred to as 'suppliers'. The delivering parties of assortment of platform partners are referred to as 'retailers'. For simplicity, we only allocate the assortment of suppliers.

Products being sold on the webshop are categorized in multiple ways. Every product is categorized in a size-group based on its characteristics such as dimensions and weight. Size-groups being processed at WH-A and WH-B are 3XS, XXS, XS, S, M, and L. Furthermore, every product is categorized in a product-group, which is an aggregation to be able to refer to similar products. Examples of product-groups being processed at WH-A and WH-B are 'Electronics', 'Sports', and 'Travel'. An overview of all product-groups is presented in Appendix H. In the current SAM, constraints assuring a minimum distribution of size-groups to be processed over the different warehouses are present, such that both warehouses process a partition of all size-groups. However, we do not consider these constraints, as this complexity is not required for the scalability analysis.

Products to process can be subjected to restrictions, indicating that a product has to be processed by a specific warehouse. This can either be related to product characteristics such as high-value or fragility, or induced by other limitations such as the fact that new-product releases can only be handled by one of the two warehouses. Examples of restricted products in the assortment are presented in Appendix A.2. For simplicity, we do not take into account product restrictions in the allocation decision.

2.1 Problem description

Taking those simplifications into account, the problem complexity included in the stock allocation models is elaborated on. The allocation of products on different aggregation levels to WH-A and WH-B is being optimized, restricted to four main sets of constraints. Process information is introduced to explain those constraints, after which the existing and proposed model formulations are presented.

The allocation decision is considering outbound processing capacities, indicating that e.g. stocking and inbound capacities are not in the model environment. Outbound processing regards the processing of a product i on a capacitated outbound line j. An outbound line is defined as a station to pack orders after products have been retrieved from their stocking location. Processing a product on such an outbound line incurs costs c_j. A distinction is made between the order types 'mono-orders' and 'multi-orders'. Mono-orders are orders of individual products, while multi-orders are orders containing multiple products. Outbound lines are order type and warehouse specific. Different products can be processed by different outbound lines. Therefore the concept of 'segments' is introduced, which are defined as sets of outbound lines. A segment S is also order type and warehouse specific, as only outbound lines of the same order type at the same warehouse are grouped in a segment. All segments S available at warehouse w are grouped in the set G_w.

Every product is assigned to segments at both warehouses, indicating that this product can be processed only by the outbound lines in those segments. A product i can at most be assigned to **one** segment per order type at a warehouse, so one mono-order segment and one multi-order segment. A product is always assigned to the most general possible segment S of an order type, which is the segment that contains all the outbound lines j at which product i can be processed. The assignment of a product to a segment is based on product characteristics. Since there are different outbound lines j for mono- and multi-orders, a product can only be assigned to mutually exclusive segments S at a warehouse w. It is possible that a product is fully processed on either mono- or multi-order outbound lines. This structure indicates that a product i is at least assigned to two segments S (one in both warehouses, 100% allocated to a single segment) and at most assigned to four segments S (two in both warehouses, spread out over a mono- and multi-order segment).

The relationship between segments S, outbound lines j and products i is used to distribute daily forecasted demand per product $d_{i,t}$ over the outbound lines. The main parameter facilitating the relationship is $f_{i,S}$, the fraction of forecasted demand $d_{i,t}$ of product i that should be processed by outbound lines j in segment S. Given a warehouse w, the segment fractions of a product always sum up to 1: $\sum_{S \in G_w} f_{i,S} = 1, \forall i \in I, \forall w \in W$. It is known beforehand which fraction of a product's forecasted demand needs to be allocated to what segment at a warehouse. This is a deterministic input parameter and is based on historic data. To be able to distribute the forecasted demand of products over outbound lines through segments, continuous variables $x_{S,j,t}$ are introduced. These variables represent the absolute forecasted demand processed in segment S by outbound line j at day t, where line j is in the set S ($j \in S$). $x_{S,j,t}$ is used to sum up all the allocated forecasted demand levels in different ways to make it comply with the constraints in the model environment.

Line capacity constraints assure that the allocated demand to the outbound lines remains within daily capacity. A constraint is added for every day t the model is optimizing over and for every outbound line j. The left-hand-side of the constraint is summing up the allocated demand to outbound line j on day t, indicating to sum $x_{S,j,t}$ over all segments S that contain line j: $S|j \in S$. This sum should be less than or equal to daily line capacity in products n_j.

Demand assignment constraints, also referred to as *segment constraints*, assure that all forecasted demand on a daily basis is allocated to be processed by the outbound lines. It does so by constraining that the sum of allocated demand to outbound lines j in segment S, thus summing $x_{S,j,t}$ for $j \in S$, is equal to the total demand allocated to that segment S on day t. The calculation of the latter is dependent on the allocation decision and aggregation level. Constraints are added for all days t to optimize over, for all warehouses w and for all segments in that warehouse $S \in G_w$.

Both warehouses WH-A and WH-B can process similar kind of products in the e-fulfillment process. The geographical location of the warehouse is irrelevant, as both warehouses are located close to each other and can provide every product to the same locations in The Netherlands. The stock allocation therefore does not incorporate any distance or delivery-time based constraints. It is the case that WH-B is more automated. Therefore, it is generally favorable to allocate products to WH-B, due to lower processing costs per product. To balance this, **pand-split constraints** assure that the ratio of distribution of total allocated demand D over the available warehouses in the optimization period is within a predetermined range. The total demand D in the optimization period T can be calculated using $D = \sum_{i,t} d_{i,t}$. This range, referred to as the 'pand-split range', is a combination of the target ratio of allocation at that warehouse p_w and the flexibility given to the model δ. To enforce this in the allocation decision, all allocated demand $x_{S,j,t}$ is summed up over all days t for segments exclusive to the warehouse the constraint is calculated over, thus summing over $S \in G_w$ and over the lines in inside the segments $j \in S$. The resulting total processed demand at a warehouse should be within the pand-split range.

Warehouse-exclusivity constraints assure that an allocated element, in our case a supplier k or product i, only gets allocated to a single warehouse. The allocation decision variables are binaries, indicating that an element is either fully allocated (1) or not at all allocated (0) to warehouse w. The constraints enforce that the sum of all binary variables associated should sum up to 1.

Using these constraints, the forecasted demand levels being processed on outbound lines on a daily basis are calculated. This is multiplied with the costs per processed product c_j. This multiplication is summed over every day t over every outbound line j, which can be translated to summing $x_{S,j,t}$ over all available segments S at the available warehouses w ($\sum_{w \in W} \sum_{S \in G_w}$), and all outbound lines in these segments $j \in S$. This sum is the total cost of the allocation problem being minimized. This capacitated allocation problem is a typical MILP and is an instance of combinatorial optimization.

2.2 Existing model: supplier-level allocation (SAM)

As introduced in Section 1, SAM is the current model solving the allocation problem at supplier-level. To realize supplier-level allocation, binary decision variables $z_{k,w}$ are considered accounting for the allocation of the assortment of supplier k to a single warehouse w. The assortment of supplier k is a set of products i, defined as the set A_k. Since the demand forecast $d_{i,t}$ is on product-level, a parameter $r_{i,k}$ is defined as the fraction of forecasted demand of a product i that needs to be fulfilled by supplier k. For product i, historic data is gathered which suppliers k have delivered what fraction of product i. This is used to calculate $r_{i,k}$, which are summing up to 1 summing over all suppliers: $\sum_k r_{i,k} = 1, \forall i$. This historic fraction is assumed to be present in future demand and therefore it is used as an deterministic input to the model.

To utilize the relationship between segments, outbound lines and products, the total allocated demand to a segment S on day t needs to be calculated. The total demand induced on a segment S by a single supplier k on day t for a single product i can be calculated using $f_{i,S} \cdot d_{i,t} \cdot r_{i,k}$. These results are summed up over all products i in the assortment of supplier A_k to calculate total demand induced by this supplier k. Since $f_{i,S}$ is non-zero at segments at both warehouses and the forecasted demand only needs to be distributed over outbound lines in the warehouse the products are being allocated to, the sum is multiplied with binary variables $z_{k,w}$. This results in $z_{k,w} \cdot \sum_{i \in A_k} f_{i,S} \cdot d_{i,t} \cdot r_{i,k}$. This indicates that the allocated forecasted demand of all products i delivered by supplier k is only non-zero for segments S at the warehouse w that supplier is being allocated to. To calculate total daily demand levels processed on segment S, the results are summed up over all available suppliers $k \in K$. This relation is used in the demand assignment constraints of SAM (Equation 3, right-hand-side). This complexity enforces the entire assortment A_k of a supplier to be exclusively allocated to a single warehouse, generating the supplier-level aggregation level in the allocation decision. Together with line capacity constraints (Equation 2), pand-split constraints (Equation 4) and warehouse-exclusivity constraints (Equation 5), the optimal allocation can be computed which minimizes total costs (Equation 1). The resulting MILP of SAM is formulated as follows.

$$\min_{x,z} \sum_{t \in T} \sum_{w \in W} \sum_{S \in G_w} \sum_{j \in S} x_{S,j,t} \cdot c_j \tag{1}$$

s.t.:

$$\sum_{S | j \in S} x_{S,j,t} \leq n_j, \quad \forall t \in T, \forall j \in J, \quad \text{(Line capacity)} \tag{2}$$

$$\sum_{j \in S} x_{S,j,t} = \sum_{k \in K} z_{k,w} \cdot \sum_{i \in A_k} f_{i,S} \cdot d_{i,t} \cdot r_{i,k}, \quad \forall t \in T, \forall w \in W, \forall S \in G_w \quad \text{(Demand assignment)} \tag{3}$$

$$\sum_{t \in T} \sum_{S \in G_w} \sum_{j \in S} x_{S,j,t} - p_w \cdot D \quad \in \quad [-\delta \cdot D, +\delta \cdot D], \quad \forall w \in W \quad \text{(Pand-split)} \tag{4}$$

$$\sum_{w \in W} z_{k,w} \quad = \quad 1, \quad \forall k \in K \quad \text{(Warehouse-exclusivity)} \tag{5}$$

$$x_{S,j,t} \in [0, \infty] \tag{6}$$

$$z_{k,w} \in \{0, 1\} \tag{7}$$

Variables:

- $x_{S,j,t}$: A continuous variable representing the absolute forecasted demand processed from segment S by outbound line j at day t, where line j is in the set S or $j \in S$.
- $z_{k,w}$: Binary variable used to force that the assortment of products i of supplier k ($i \in A_k$) can only be allocated to one of the warehouses w.

Parameters and indices:

i:	A product of which the model needs to determine the allocation.
j:	An outbound line in the warehouse that can produce products i. There are specific outbound lines j that can process respectively mono- and multi-orders.
w:	A warehouse to which products i can be allocated. In this case only two options are available: WH-A and WH-B.
k:	A supplier delivering the products i to the warehouses w.
t:	A day over which the model has to optimize.
$d_{i,t}$:	Forecasted demand of product i on day t.
D:	The total demand produced in the optimization period: $\sum_{i,t} d_{i,t}$
c_j:	Costs of processing a product on outbound line j.
n_j:	Daily production capacity in number of products of outbound line j.
$f_{i,S}$:	The fraction of demand of product i that needs to be assigned to segment S.
$r_{i,k}$	The fraction of demand of product i that needs to be fulfilled by supplier k.
p_w:	The target percentage of warehouse w of total forecasted demand division over the warehouses.
δ:	The delta that provides flexibility in the pand-split constraints.

Sets:

I:	The set of all products i the model needs to allocate.
J:	The set of all lines j the model can allocate products to.
W:	The set of available warehouses w.
K:	The set of suppliers k to allocate.
A_k	A set of products i delivered to the warehouses w by supplier k, referred to as the supplier's assortment.
T:	A set of all days t the model has to optimize over.
S:	A segment S is defined as a set of outbound lines j.
G_w:	A group of segments S available at warehouse w.

The aggregation of products on supplier-level in the allocation decision however has operational disadvantages:

- Allocating stock on supplier-level induces sub-optimal decisions on product-level, as the optimal decision for the aggregated group induces sub-optimal allocations of single products. Suppliers can have a divergent assortment, containing products that have very different characteristics (Appendix A.1). This reduces the flexibility of the allocation substantially. The level of sub-optimality supplier-level allocation induces is not known.
- Company X is experiencing growth and considers to open new warehouses. The sub-optimality experienced due to the supplier-level decision will increase when extra warehouses are added to the model environment. Next to this, there is an idea to open a special warehouse. A lot of demand is experienced during the holiday season, referred to as the 'peak-period'. It is considered to open a warehouse specifically during this peak-period to store products with specific characteristics such as high demand. SAM is not able to allocate such products individually, as those products will not be delivered by a single supplier.
- Restrictions on products require it to be processed on a specific warehouse (Appendix A.2). In the case of supplier-level allocation, this induces the allocation of the entire assortment of a supplier can be influenced by a single restricted product. An exception can be made for this individual product, meaning that the stock of this single product is stored at another warehouse than the rest of the assortment. Both situations are unfavorable.
- Company X is implementing double-stocking. Double-stocking allows a product to be stocked at multiple warehouses. Supplier-level allocation complicates this process as it is not possible to allocate individual products to be double-stocked.

To overcome these disadvantages and analyze the scalability of the allocation decision, the product-level allocation model is proposed.

2.3 Proposed model: product-level steering (PLS)

To be able to realize product-level allocation, the PLS model is formulated. The binary decision variables $z_{k,w}$ are replaced by binary decision variables per product $z_{i,w}$. This greatly increases the number of integer variables in the model environment, which has an evident influence on the computation time and scalability to solve the optimization problem (Cao & Sun, 2016).

Another important difference in formulation is the calculation of the total allocated demand on a segment S on day t. At supplier-level, the grouping of products in A_k is realized in this calculation. Since this is not required anymore at product-level, the computation is slightly less complex. The demand a single product i induces on a segment S on day t can be calculated using $f_{i,S} \cdot d_{i,t}$. Including the binary allocation variable $z_{i,w}$ in the calculation assures that the forecasted demand to process is only non-zero for the warehouse to which the product is being allocated to, similarly to including $z_{k,w}$ as described in Section 2.2. Summing over all products, the total allocated demand levels to segment S on day t are calculated, resulting in the right-hand-side of the demand assignment constraints in Equation 10. These constraints assure all forecasted demand is being allocated, complying with line capacity constraints (Equation 9), pand-split constraints (Equation 11) and warehouse-exclusivity constraints (Equation 12). Again, the objective is total cost minimization (Equation 8). The resulting MILP of PLS is formulated as follows.

$$\min_{x,z} \sum_{t\in T} \sum_{w\in W} \sum_{S\in G_w} \sum_{j\in S} x_{S,j,t} \cdot c_j \tag{8}$$

s.t.:

$$\sum_{S|j\in S} x_{S,j,t} \leq n_j, \qquad \forall t \in T, \forall j \in J \quad \text{(Line capacity)} \tag{9}$$

$$\sum_{j\in S} x_{S,j,t} = \sum_{i\in I} f_{i,S} \cdot d_{i,t} \cdot z_{i,w}, \qquad \forall t \in T, \forall w \in W, \forall S \in G_w \quad \text{(Demand assignment)} \tag{10}$$

$$\sum_{t\in T} \sum_{S\in G_w} \sum_{j\in S} x_{S,j,t} - p_w \cdot D \in [-\delta \cdot D, +\delta \cdot D], \qquad \forall w \in W \quad \text{(Pand-split)} \tag{11}$$

$$\sum_{w\in W} z_{i,w} = 1, \qquad \forall i \in I \quad \text{(Warehouse-exclusivity)} \tag{12}$$

$$x_{S,j,t} \in [0, \infty] \tag{13}$$

$$z_{i,w} \in \{0, 1\} \tag{14}$$

Variables:

- $x_{S,j,t}$: A continuous variable representing the absolute forecasted demand processed in segment S by outbound line j at day t, where line j is in the set S or $j \in S$.
- $z_{i,w}$: Binary variable used to force that a product i can only be allocated to one of the warehouses w.

It is possible to create a model that spreads out the fraction of a product's demand over the outbound lines, not using the relation between products, segments and outbound lines. However, this increases the model size significantly by introducing a lot of constraints and continuous variables. Furthermore, outbound line sourcing information per product is not required for this research. Such a model has been defined in Appendix B and was implemented, but ran into memory issues for larger problem scales and therefore is not elaborated on any further. Using the relationship between segments, outbound lines and products, the amount of continuous variables required in the model environment is reduced significantly.

Since PLS allows product-level allocation, the sketched disadvantages of supplier-level allocation in Section 2.2 are reduced. Sub-optimality is reduced, allowing for better capacity utilization. Besides, it allows for the implementation of considered developments. Despite the fact exclusivity in allocation is enforced, product-level allocation allows the model complexity required to implement double-stocking. However, PLS also generates challenges:

- The main technical challenge is the scalability of PLS given the new, larger problem scale. It is a challenge to find which method is required to tackle this problem effectively and efficiently.
- The effect of PLS on supplier splits is unknown. This can be unfavorable for both the supplier and Company X, as this complicates the inbound process.
- It is possible that products with common characteristics are mainly allocated to a single warehouse. From operational point of view this might be unfavorable. Company X therefore aims to distribute its assortment over the warehouses using e.g. size-group constraints. There is an interest in the effect of product-level allocation on this the robustness of the allocation decision of PLS.

It can be concluded that the considered stock allocation problem is unique and complex. We are interested in to what extent the sketched challenges are problematic for the implementation of the proposed model using the new aggregation level, and to what extent it can be used to overcome the disadvantages of the existing model. To be able to evaluate this, the models are implemented and computational experiments are conducted.

3 Input data description

Data is gathered for the implementation of the MILP models. This section elaborates on the data structures used as input, such that all formulated parameters are present.

3.1 Product-level sales forecast

To realize the product-level forecast $d_{i,t}$, a product-group forecast is transformed to product-level using historic order data. Historic order data provides information on which product in a product-group accounted for what fraction of previous sales in that product-group. It is assumed that this ratio will be present in the future sales. Using $f_{i,S}$, a forecast per segment can be calculated. This transformation is illustrated in Appendix C.

The assortment that will be present in the product-level forecast, and therefore the assortment I to be allocated, is dependent on the used historic data. The larger the time-frame of historic data included, the more products sold and thus the more products to be allocated. This is connected to the long-tailed assortment of Company X. The effect is illustrated in Figure 6. The amount of historical order data H used to calculate the forecast is controllable, creating the possibility to allocate a different sized assortment of products to check the effects on scalability of the solutions. For a realistic run, 60 days of historic data are included.

As seen in Figure 6, the increase of assortment to allocate decreases as H increases, which can be described as logarithmic behavior. The scale is also influenced by the time of the year of included historic data, as the number of products sold during the peak-period is larger (Section 1). Despite the relatively small differences in number of products to allocate, demand levels are larger in the peak-period. For example, the average daily demand forecast in the peak-period $[2021/11/01, 2021/12/31]$ is $\pm 380,000$ products, while for a 'normal' period $[2021/06/01, 2021/07/31]$, this is $\pm 200,000$ products.

On the other hand, the number of days to optimize the allocation over T is controllable. The more days included in the forecast, the larger the scale of the optimization problem. This relation is used to analyze the scalability of the solutions. The scale of the product-level allocation model can be computed using Equations 15 and 16, where H influences the number of products to allocate $|I|$.

$$\text{Number of variables} = |T| \cdot \sum_{w} |G_w| \cdot \frac{\sum_S |S|}{|J|} + |W| \cdot |I| \tag{15}$$

$$\text{Number of constraints} = |T| \cdot |J| + |T| \cdot |W| \cdot \frac{\sum_w |G_w|}{|W|} + 2 \cdot |W| + |I| \tag{16}$$

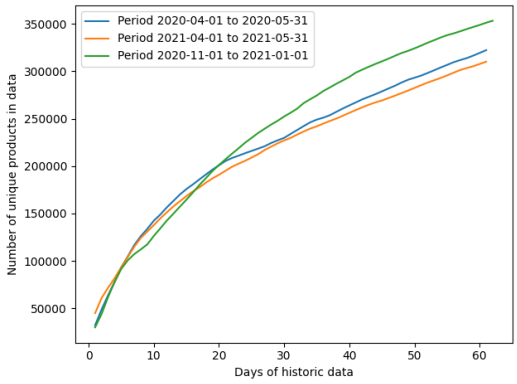


Figure 6: Increase of products included in product-level demand forecast as the historic data taken into account increases.

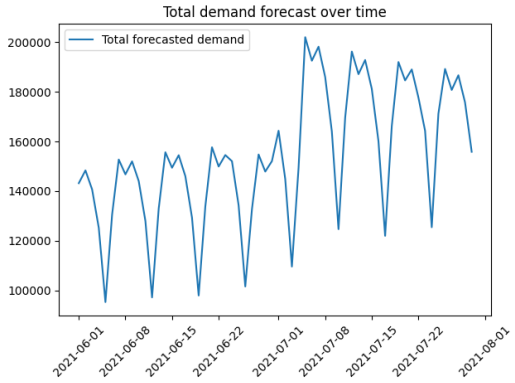


Figure 7: Total demand forecast over time in period $[2021/06/01, 2021/07/31]$.

The number of variables and number of constraints, and thus the model scale, increase linearly as H and T increase. As the number of products being allocated is large, scales are increasing quickly. If a larger optimization period is used, it makes sense to include more historic data and therefore a larger assortment to allocate, indicating H and T are connected. For a realistic run, an optimization period T of 60 days is used. This is connected to the fact that approximately six stock allocation runs are performed each year, therefore making sense to run the model every $365/6 \approx 60$ days. The assortment forecasted to be sold during this optimization period is believed to be related to the assortment sold in the past 60 days ($H = 60$).

To illustrate demand behavior, the total demand forecast over time for this configuration is presented in Figure 7. A weekly demand pattern is recognized, with negative peaks on Saturdays and a mid-week dip on Tuesdays. In the case of this configuration, an increase in demand in the second half of the optimization period is present.

As an input to the allocation models, the forecast is retrieved per product $d_{i,t}$, which is summed up to find the total demand forecast D. This provides information on products $i \in I$ to allocate, as the forecast is only provided for products in historic data H. The forecast is not constrained to be integer, as it is based on the historic fractions and those are not rounded. In reality, an integer number of products will be processed at the warehouses. However, the model needs to output the optimal allocation decision, which is possible using non-integer forecasts.

3.2 Available resources at the warehouses

The available outbound lines $j \in J$ that can process products i are presented in Table 4. The total costs of processing a product on an outbound line c_j includes the costs of other steps performed at the warehouse (inbound, etc.). The costs c_j are equal for all products i, such that different products incur the same costs on outbound line j. For both mono- and multi-orders, the cheapest lines are at WH-B. However, it is not the case that all lines for an order type (mono/multi) are strictly cheaper at WH-B than at WH-A. For example, for mono-orders the sequence from lowest costs to highest costs per product is: CMC1 (WH-B), SMARTMAILER_WH-B (WH-B), SMARTMAILER_WH-A (WH-A), POS-MONO (WH-A), NEOPOST (WH-A), MONO-MANUAL (WH-B), BIG-ITEMS-MONO (WH-A). For multi-orders a similar alternating behavior is present. Daily line capacities n_j are also presented.

Table 4: Outbound lines j available and their characteristics.

Name outbound line j	**Total cost per item** c_j	**Daily capacity** n_j **in products**	**Warehouse** w	**Order type**
SMARTMAILER_WH-A	0.84019156	16500	WH-A	Mono
POS-MONO	0.94317373	73920	WH-A	Mono
NEOPOST	0.98413690	5775	WH-A	Mono
BIG-ITEMS-MONO	1.14735151	36960	WH-A	Mono
POS-MULTI	1.05164317	79200	WH-A	Multi
BIG-ITEMS-MULTI	1.33238762	57750	WH-A	Multi
SMARTMAILER_WH-B	0.82245362	19800	WH-B	Mono
CMC1	0.80331195	33000	WH-B	Mono
MONO-MANUAL1	1.10638834	166650	WH-B	Mono
SORTER1	1.02025084	90750	WH-B	Multi
MULTI-MANUAL1	1.26271195	66000	WH-B	Multi

The available segments S are presented in Table 5. The name of a segment is a simple combination of outbound lines j, therefore providing the information of $j \in S$. Segments S are created in terms of generality. For example, if a product is ordered as mono-order at WH-A, it gets assigned to the segment that contains *all* the mono-order outbound lines it can be processed on at WH-A. This can be a single line, all mono-order lines, or a segment in between. The most general segment for an order type at a warehouse is the segment containing a single outbound line.

Table 5: Segments S available and their characteristics, including an example of $f_{i,S}$.

Name segment S	**Warehouse** w	**Order type**	**Fraction** $f_{i,S}$
BIG-ITEMS-MONO	WH-A	Mono	0.7
BIG-ITEMS-MONO|POS-MONO	WH-A	Mono	0
BIG-ITEMS-MONO|NEOPOST|POS-MONO	WH-A	Mono	0
BIG-ITEMS-MONO|NEOPOST|POS-MONO|SMARTMAILER_WH-A	WH-A	Mono	0
BIG-ITEMS-MULTI	WH-A	Multi	0.3
BIG-ITEMS-MULTI|POS-MULTI	WH-A	Multi	0
MONO-MANUAL1	WH-B	Mono	0.7
CMC1|MONO-MANUAL1	WH-B	Mono	0
CMC1|MONO-MANUAL1|SMARTMAILER_WH-B	WH-B	Mono	0
MULTI-MANUAL1	WH-B	Multi	0.3
SORTER1|MULTI-MANUAL1	WH-B	Multi	0

3.3 Other parameters

To facilitate the relation between outbound lines, products and segments, the parameter $f_{i,S}$ is required. An example of $f_{i,S}$ for a random product i is presented in the last column of Table 5. $f_{i,S}$ facilitates the distinction between mono-orders and multi-orders in the model environment. It is a rather simplistic implementation, as the definition of a multi-order indicates a product is ordered together with another product. This relation between products i is not captured in this research. However, it is assumed that the current method is sufficient to check capacity utilization and make the allocation decision. These fractions $f_{i,S}$ are available for all products $i \in I$.

The available suppliers $k \in K$, the assortment of a supplier A_k, and the fraction of a product delivered by a supplier $r_{i,k}$ are extracted from historic purchase order data, which is different from order history H. The number of suppliers to allocate is lower than the amount of total suppliers delivering to Company X, as it is based on this historical purchase order data. This data is also used to determine if a supplier is split in the product-level allocation.

The parameters p_w and δ are treated as deterministic, external input as they are determined by another department. Therefore these parameters are constant over all runs. Values are presented in Table 6.

Table 6: Values p_w and δ.

Warehouse w	p_w	δ	Lower bound	Upper bound
WH-A	0.42	0.10	0.32	0.52
WH-B	0.58	0.10	0.48	0.68

4 Results

Using the input data, computational experiments are conducted and results are gathered. The models and methods used are presented in Table 7.

Table 7: Overview of models used in Section 4.

Model	Description
SAM	The supplier-level allocation model SAM. It uses the open-source CBC (CoinOR Branch and Cut) solver. Since PLS is not using certain details of the model environment such as product restrictions and size-group constraints (Section 2), SAM is also simplified to mirror the PLS model complexity to be able to fairly compare the models.
PLS-GUR	The product-level allocation model PLS using the commercial Gurobi solver.
PLS-CBC	The product-level allocation model PLS using the open-source CBC solver.

Python is used as the programming language and the package PuLP is used for the implementation of the optimization model[1]. The optimization models are ran on a local machine (2.6 GHz 6-Core Intel Core i7, 16 GB RAM, AMD Radeaon Pro). In this section, the key performance indicators (KPI's) are defined to evaluate the implemented models, a comparison is presented between the implemented models based on these KPI's, and detailed use-case analysis of supplier-level allocation (SAM) and product-level allocation (PLS) is presented.

4.1 KPI's

The main goal is to investigate whether the increase in problem scale of product-level allocation is feasible from both a technical and operational perspective. The two KPI's used for evaluation are **scalability** and **optimality** (Brouer, Karsten, & Pisinger, 2016). The implementation of solvers rather than other methods such as meta-heuristics proved to be feasible given the problem definition. The CBC solver is more basic and open-source, whereas the GUR solver is a high-quality and competitive commercial solver (Anand, Aggarwal, & Kumar, 2017). The results presented in this section validate this.

Scalability
Scalability is defined as the influence of an increase in scale on the performance of a solution to a problem. Scalability evaluation has the goal to get information which scales of problem sizes yield what results. Different combinations of H and T are used to gather results on scalability. To quantify and evaluate scalability, the following model output is extracted:

- The number of variables and number of constraints included in the model;
- The number of suppliers and products being allocated by the model;
- The computation time in seconds per run of the optimization, excluding pre- and post-processing of information. This is the main performance indicator for scalability.

Optimality
Apart from the scale the model is operating on, there is an interest in the optimality of the provided solution by the model. On one hand, it can be evaluated how the solution to the allocation problem is performing operationally, for example to what extent warehouse capacities are being utilized. This provides information on how optimal the allocation is from a business perspective and it is expected that product-level allocation is favorable in this case. To quantify and evaluate optimality, the following model output is extracted:

- The resulting objective value of the optimization;
- The average costs induced per processed product;
- The number of suppliers that are split in resulting allocation;
- More operational output is stored for the use-case analysis in Section 4.3.

[1] `https://coin-or.github.io/pulp/`

On the other hand, it can be evaluated to what extent the optimization problem is solved to optimality. As discussed by Yu, Wang, Hua, and Lau (2013), there are multiple methods for solving an optimization problem exactly. This indicates that the guaranteed best solution in the solution space is retrieved. However, finding the exact optimal solution in integer programming is very time consuming and may therefore be infeasible as scale increases (Yu et al., 2013). Approximation methods can be a solution in this case.

The implemented solvers are used as an approximation method, thus providing a close-to-optimal solution to the integer problem. We measure optimality using the optimality gap, defined as the difference between the best-found integer solution by the optimization method and the best-bound solution (Cao & Sun, 2016; Bayraksan & Morton, 2006). Best-bound solutions are found solving the linearly relaxed optimization problem and are part of the branch-and-cut algorithms used by CBC and GUR (Mitchell, 2002). The resulting integer solution satisfies all constraints and minimizes the objective. The objective value of the solution to the relaxed problem can be used as the aimed optimal solution. Setting a smaller optimality gap can possibly result in a better solution to the allocation problem. This increases the computation time required, and thus decreases the scalability (Glover & Kochenberher, 2003). It is also possible to set a time-limit to the optimization problem, which simply yields the best-found solution in the given amount of time.

No time-limits are set as the solvers proved to find high quality solutions in reasonable times. A relative optimality gap of 10^{-7} is used in this research. An analysis on the influence of decreasing this optimality gap and on the linear relaxation of the integer optimization problem is now presented.

4.1.1 Analysis linear relaxation

Both allocation models are solving an MILP. The implemented solvers use a branch-and-cut algorithm as the optimization method. Therefore, the problem is relaxed after which the resulting non-integer variables are forced to become integer by cutting and branching (Mitchell, 2002). Solving a linear programming problem is less computationally intensive than solving a (mixed-)integer linear programming problem (Wolsey & Nemhauser, 1999; Vielma, 2015).

Using the logging of the GUR solver, it was discovered that for smaller scales of implementation of the PLS model, only a single node was explored after the initial linear relaxation of the MILP is solved to come to the final solution. This could indicate that there exists an integer solution to the allocation problem that is equal to the relaxed solution. If the model holds the property of total unimodularity, the optimal solution to the relaxed linear problem obtained using a simplex algorithm yields a solution consisting of integer variables only and therefore is a solution to the integer problem (Wolsey & Nemhauser, 1999; Jünger et al., 2009). Relaxation can therefore provide information of the underlying problem structure. To get a better understanding what is happening when relaxing the allocation problem, LP-relaxation is implemented manually for the PLS model. In terms of the model formulation, this means that the binary allocation variables are relaxed to become continuous between 0 and 1 (Equation 17).

$$z_{i,w} \in \{0,1\} \longrightarrow z_{i,w} \in [0,1] \tag{17}$$

It is found that the best-bound of the integer solution to the problem becomes close-to-equal to the solution of the relaxed problem (Appendix J.1). It is possible that the value of the relative optimality gap of 10^{-7} is such that the solution to the integer model can be found by only exploring 1 node. Therefore as a sensitivity analysis this gap is decreased to 10^{-13}. This resulted in a higher best-bound in the solution of the integer problem, which is unequal to the solution to the relaxed problem (Appendix J.1). This also resulted in a slightly lower objective value, indicating a better solution has been found to the allocation problem. Next to this, more nodes are explored when solving the integer model. As multiple nodes are explored, and a new best-bound and better solution are found, the solution to the relaxed main problem is not a solution to the integer problem (Jünger et al., 2009). This indicates total unimodularity is not the case. The difference in objective value by decreasing the relative optimality gap is only marginal, therefore it did not increase the quality of the solution significantly. Since decreasing the gap increases computational times significantly while the solution quality increases only marginally,

for the remainder of the research the gap of 10^{-7} is used. More details on the analysis on manually relaxing the optimization problem are presented in Appendix J.

4.2 Results scalability and optimality

Results are presented on scalability and optimality for different combinations of H and T. As mentioned in Section 3.1, $H = 60$, $T = 60$ is a realistic configuration on which SAM operates. SAM, PLS-GUR and PLS-CBC are compared using the configurations presented in Table 8.

Table 8: Configurations used for scalability analyses.

Days of historic data H	**Days to optimize over T**	**Number of products I to allocate**	**Number of suppliers K to allocate**
2	2	59830	714
7	7	111803	746
14	14	157092	741
31	31	219259	730
60	60	293381	761

These configurations all use the same start date, 01/06/2021. The period of historic data on which the forecast is based therefore ranges between $[31/05/2021-H, 31/05/2021]$. The latter influences the number of suppliers included in the model, thus the differences between configurations. The period to optimize over ranges between $[01/06/2021, 01/06/2021 + T]$. This is an off-peak period, chosen deliberately since this is the case the most of the time in the year and the peak-period could induce capacity shortages, which cannot be tackled by the current model. The obtained results are presented in Table 9.

Table 9: Results scalability and optimality runs.

Configuration	**Model**	**Objective**	**Average costs per item**	**Number of variables**	**Number of constraints**	**Run-time (seconds)**	**Supplier splits**
$H = 2$, $T = 2$	**SAM**	307766.01	1.048985	3353	1803	0.22	0
	PLS-GUR	303183.29	1.033365	119704	59878	1.08	660
	PLS-CBC	303183.28	1.033365	119704	59878	9.54	660
$H = 7$, $T = 7$	**SAM**	996901.22	1.071811	3778	2283	0.75	0
	PLS-GUR	982680.89	1.056522	223760	111961	9.47	700
	PLS-CBC	982680.83	1.056522	223760	111961	92.75	700
$H = 14$, $T = 14$	**SAM**	2039208.84	1.077164	4373	2955	0.93	0
	PLS-GUR	2011165.59	1.062350	314492	157404	30.25	702
	PLS-CBC	2011165.44	1.062350	314492	157404	422.93	702
$H = 31$, $T = 31$	**SAM**	4647499.14	1.078248	5818	4587	2.62	0
	PLS-GUR	4590217.64	1.064959	439200	219945	144.40	696
	PLS-CBC	4590217.34	1.064959	439200	219945	1722.12	696
$H = 60$, $T = 60$	**SAM**	10016159.58	1.083758	8283	7371	25.16	0
	PLS-GUR	9929419.76	1.074373	588082	294705	658.21	712
	PLS-CBC	9929419.82	1.074373	588082	294705	6067.32	712

The PLS models outperform SAM for every configuration in terms of objective and costs per processed product. SAM has a higher average processing costs per product for all configurations. This originates from the more efficient use of cheaper outbound lines. Details on the cost reduction are presented in Table 10.

Table 10: Cost reduction PLS over SAM different configurations.

Configuration	$H = 2$, $T = 2$	$H = 7$, $T = 7$	$H = 14$, $T = 14$	$H = 31$, $T = 31$	$H = 60$, $T = 60$
Total cost reduction	€4582.72058	€14220.3336	€28043.2427	€57281.4922	€86739.8213
Cost reduction in % of SAM objective	1.48903%	1.42645%	1.37520%	1.23252%	0.86600%
Cost reduction per day	€2291.36029	€2031.47623	€2003.08877	€1847.79007	€1445.6637

Absolute cost reductions are increasing, while cost reductions per day are decreasing. Table 9 shows that the computation time increases exponentially for all different models, but the model scale and absolute computation time increase is larger for PLS. As expected, the model scale is significantly larger at product-level due to the large influence of the number of products (Equations 15 and 16). Comparing PLS-GUR to PLS-CBC, the absolute increase in computation time is larger for PLS-CBC. Supplier splits are only present at PLS and do not show a specific trend.

Since both models provide high quality solutions given the optimality gap of 10^{-7}, the comparison of the optimality is indifferent between product-level and supplier-level. In terms of optimal warehouse utilization, PLS performs better. For realistic scales, computation time for the open-source solver becomes a few hours.

Results are also presented using a constant $H = 60$ and variable period to optimize over T (Table 11). All of these configurations are allocating the same amount of products (293381) and suppliers (761), as the historic data is constant. Model size increases as T increases (Equations 15 and 16). A larger optimization period and the associated increased model scale affect the computation time required significantly.

Table 11: Results scalability and optimality $H = 60$, T variable.

Configuration	Model	Objective	Average costs per item	Number of variables	Number of constraints	Run-time (seconds)	Supplier splits
T=60	**SAM**	10016159.58	1.083758	8283	7371	25.16	0
	PLS-GUR	9929419.76	1.074373	588082	294705	658.21	712
	PLS-CBC	9929419.82	1.074373	588082	294705	6067.32	712
T=90	**SAM**	15142696.80	1.083543	10835	10252	70.59	0
	PLS-GUR	15012811.30	1.074247	588890	295439	1480.18	712
	PLS-CBC	15012812.14	1.074247	588890	295439	8572.70	712
T=120	**SAM**	20344768.17	1.086084	13383	13131	131.46	0
	PLS-GUR	20172695.64	1.076898	589434	296041	974.39	714
	PLS-CBC	-	-	-	-	-	-

PLS is still dominant over SAM in terms of objective value and associated costs per processed item. Cost reductions are presented in Table 12. The cost reduction per day decreases slightly as T increases. The results for PLS-CBC for the configuration $H = 60$, $T = 120$ are not present. The scale at this configuration becomes too large for the CBC solver to tackle and thus results in errors.

Table 12: Cost reduction PLS over SAM $H = 60$, T variable.

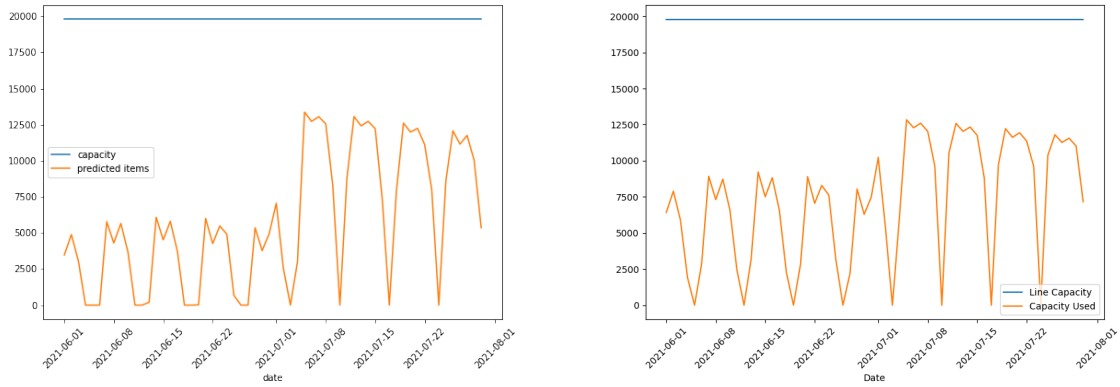
Configuration	$T = 60$	$T = 90$	$T = 120$
Total cost reduction	€86739.8213	€129885.504	€172072.531
Cost reduction in % of SAM objective	0.86600%	0.85774%	0.84578%
Cost reduction per day	€1445.66369	€1443.17227	€1433.93775

4.3 Use-case analysis

In this section, the results using configuration of $H = 60$, $T = 60$ are analyzed in more detail. PLS-CBC and PLS-GUR provide similar results, thus product-level allocation results in this section are referred to as 'PLS'.

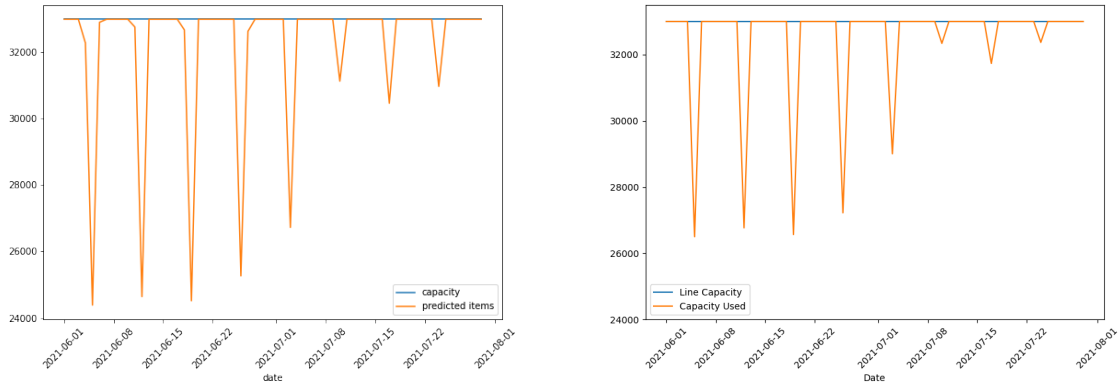
4.3.1 Line capacity utilization

In Section 2 it was stated that only demand forecasts for suppliers are being allocated, not for retailers. Therefore, in reality the total allocated demand and the capacity utilization of the outbound lines are higher. This influences results, but still allows for the analysis of capacity utilization and the comparison between supplier-level and product-level allocation. The utilization results of the SMARTMAILER_WH-B and CMC1 mono-order outbound lines at WH-B are presented in Figures 8 and 9. The results of the other outbound lines are presented in Appendix D.



(a) SAM (b) PLS

Figure 8: SMARTMAILER WH-B outbound line utilization SAM vs PLS



(a) CMC1 WH-B SAM (b) CMC1 WH-B PLS

Figure 9: CMC1 WH-B outbound line utilization SAM vs PLS

4.3.2 Supplier split details

Several results are presented to provide more insights in how the product assortment is allocated using PLS and the connected effect on supplier splits. Table 13 presents the number of suppliers of which the assortment is being split when allocating on product-level. The assortment of 93.6% of the suppliers is split over the two available warehouses. Only a small amount is either fully allocated to WH-A (1.7%) or WH-B (4.7%). Supplier-level allocation is also subjected to exclusivity constraints, indicating a supplier cannot be split and SAM results in 0 supplier splits.

Table 13: Details on allocation assortment of suppliers.

	Split	**100% WH-A**	**100% WH-B**	**Total**
Suppliers	712	13	36	761

Figure 10a presents information on the fraction of products of a supplier's assortment being allocated to WH-B in the form of a cumulative histogram. These fractions are calculated using Equation 18.

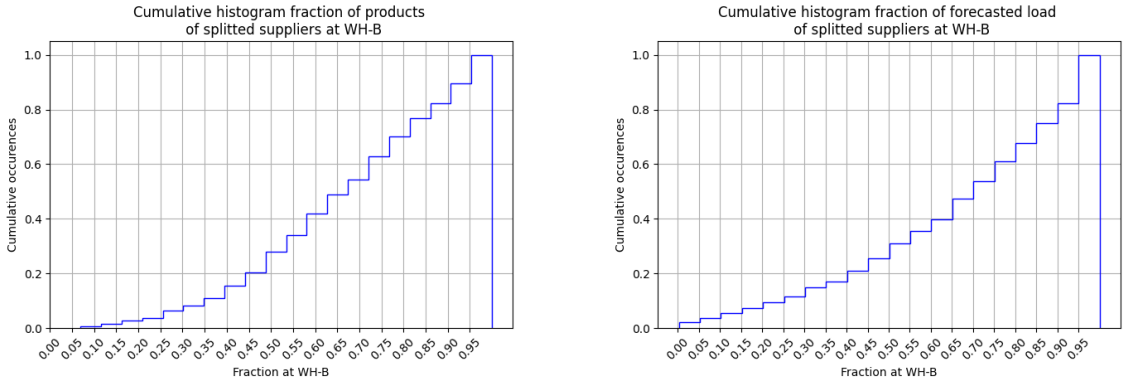
$$\text{fraction}_{k,w} = \frac{\text{count}(i \in A_k | z_{i,w} = 1)}{\text{count}(i \in A_k)} \tag{18}$$

Apart from what fraction of **products** is allocated to a warehouse, results are gathered on what fraction of a supplier's total **forecasted demand** is allocated to a warehouse. These fractions are calculated

using Equation 19 and results are presented in a cumulative histogram in Figure 10b. The fractions of WH-A can be simply calculated using $1 - \text{fraction}_{k,WH-B}$. Figures on WH-A are presented in Appendix F.

$$\text{fraction}_{k,w} = \frac{\sum_t \sum_{i \in A_k} d_{i,t} \cdot z_{i,w}}{\sum_t \sum_{i \in A_k} d_{i,t}} \tag{19}$$

If the resulting fractions (x-axis) are close to 0 or 1, the majority of products or forecasted demand of the assortment A_k of supplier k is allocated to a single warehouse, while only a minority is allocated to the other warehouse.



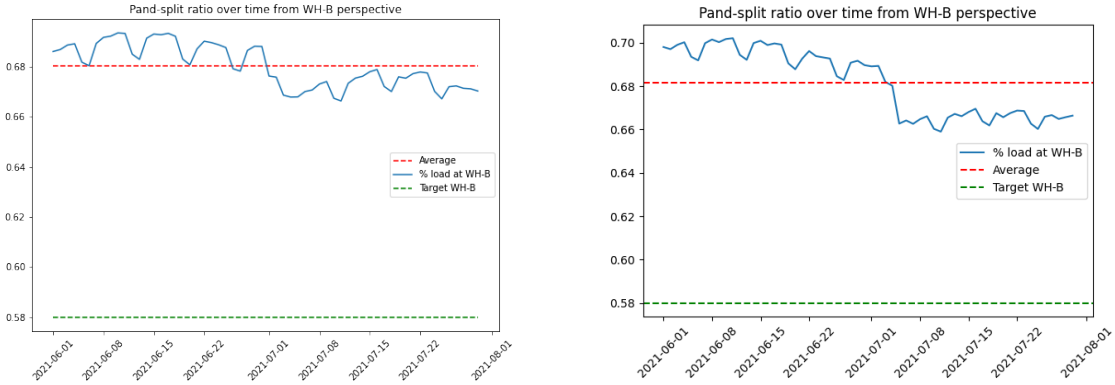
(a) Products

(b) Forecasted demand

Figure 10: Cumulative histograms fraction of **products** and **forecasted demand** of a supplier's assortment at a warehouse for PLS at WH-B, $H = 60$, $T = 60$

4.3.3 Pand-split analyses

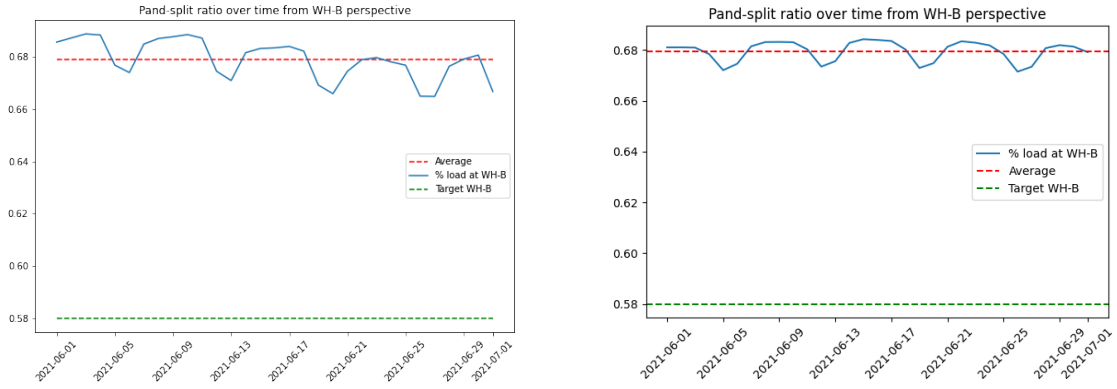
To provide more details on the functionality of the pand-split constraints, the development of the ratio over time is presented for SAM and PLS (Figure 11). The resulting global pand-split ratio is at its upper bound of 68% in both models. In the development of the ratio of both SAM and PLS, a drop is present around 2021-07-01. This is due to the fact that the demand over the optimization period in the use-case is increasing over time, with a significant increase around 2021-07-01 as presented in Figure 7 (Section 3). To reduce the effect the demand forecast variation has on the ratio over time, results for the optimization period $[01/06/2021, 31/06/2021]$ ($H = 31$, $T = 31$) are presented in Figure 12.



(a) SAM

(b) PLS

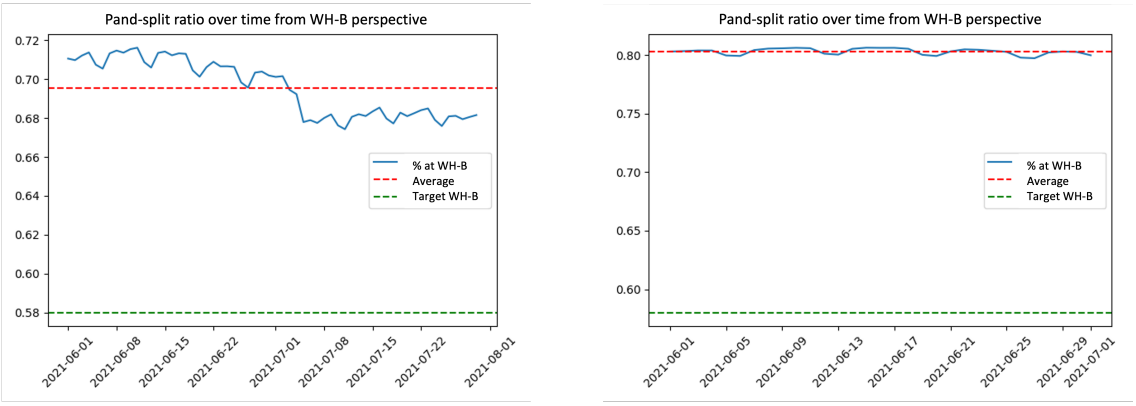
Figure 11: Pand-split ratio over time of optimization period from WH-B perspective for $H = 60$, $T = 60$.



(a) SAM (b) PLS

Figure 12: Pand-split ratio over time of optimization period from WH-B perspective for $H = 31$, $T = 31$.

Again, the constraint is at its upper bound of 68%. However, the pattern is more constant as expected. The influence of the weekly demand pattern is still present, indicating the model is allocating the products such that the upper bound of the pand-split is exceeded on days with higher demands, balanced by days with lower demand. If demand would be fully constant, the average pand-split ratio would be a straight line at 68%. The PLS ratio over time is also presented removing all pand-split constraints in Figure 13. The fraction of total capacity at WH-B is ±58% (Table 4, Section 3.2), which is also the target percentage set for WH-B in the allocation models. Therefore the pand-split ratio would be 58% if the outbound lines would operate at full capacity.

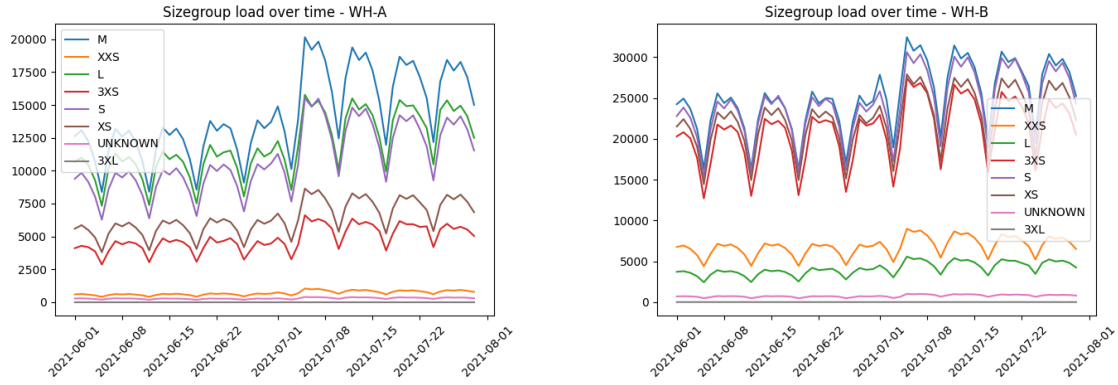


(a) $H = 60$, $T = 60$. (b) $H = 31$, $T = 31$.

Figure 13: PLS pand-split ratio over time without any pand-split constraints in the model environment.

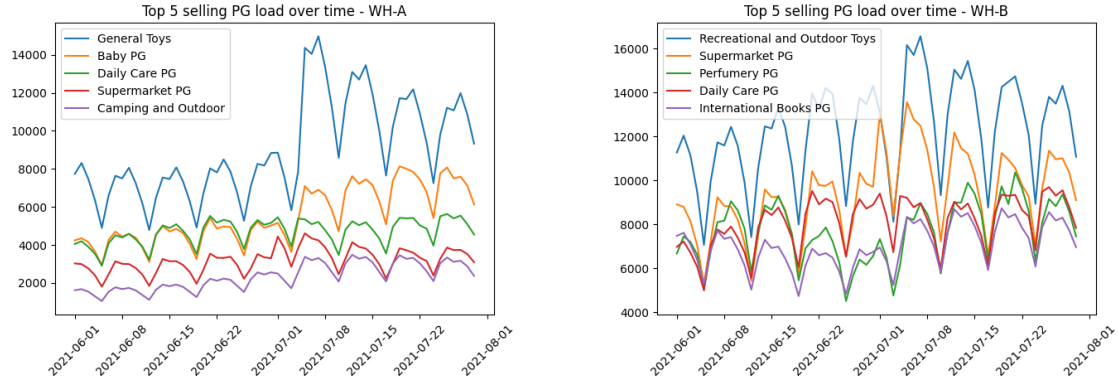
4.3.4 Utilizing warehouse characteristics

To provide more insights in how product-level allocation is utilizing the capacity at WH-A and WH-B, several results are gathered. As elaborated on in Section 1, every product i is assigned to a size-group and product-group. Given the allocation decision, forecasted demand levels can be aggregated on these levels to get information which size-group or product-group is produced most at what warehouse over time. The demand levels over time aggregated on size-group level are presented in Figure 14. The demand levels over time aggregated on product-group level are presented in Figure 15. Note that only the five product-groups that are processed the most at a warehouse are presented. Total processed demand of all product-groups per warehouse are presented in Appendix H.



(a) WH-A (b) WH-B

Figure 14: Sizegroups handled over the time-span of the optimization period for PLS



(a) WH-A (b) WH-B

Figure 15: Product-groups handled over the time-span of the optimization period for PLS

5 Discussion

The obtained results through the modeling and computational experiments of the allocation problem generate multiple insights. The scalability and optimality of product-level allocation is discussed in Section 5.1. Furthermore, insights on the comparison of supplier-level and product-level allocation are discussed in Section 5.2, focusing on cost reduction, capacity utilization, supplier splits, the pand-split constraints and allocation flexibility. Using those insights, recommendations are formulated and discussed in Section 5.3.

5.1 Scalability and optimality product-level allocation

Results have proven solvers are able to solve the product-level allocation problem in acceptable time. The structure of the allocation model utilizes the relationship between products, segments and outbound lines such that the number of variables in the model environment can be restricted. As the model is currently ran in operation every ± 2 months, computation times of several hours are not limiting. The largest run-time of 8572.70 seconds or ± 2.36 hours is obtained for PLS-CBC using $H = 60$, $T = 90$, which is feasible given the running interval in operation. There is an evident difference in scalability performance between the CBC and GUR solver. CBC requires significantly more computation time to get to the same results as the GUR. Next to this, CBC fails when the optimization period and the connected model size become too large. The reason for the difference between the two solvers originates mainly from their internal functionality of tackling the problem. Pre-solving seemed to be eliminating more

solutions using GUR and GUR is also made capable of running on multiple processing cores, while CBC is not. Besides, the heuristics, branching and cutting methods used by the solvers influence computation times.

Company X is considering to decrease the interval between allocation runs, which decreases the model scale required as the days to optimize over T can decrease. Results in Table 11 have shown that computation times can be decreased significantly when decreasing T, while model size only decreases slightly. For example, for PLS-GUR the difference in model size increasing T from 60 to 120 is $\frac{589434-588082}{588082} * 100\% = 0.2299\%$. The difference in computation time is $\frac{974.39-658.21}{658.21} * 100\% = 48.0463\%$, significantly higher. The computation time of solving an integer optimization problem is not only influenced by the model scale, the model structure may also influence the speed of finding high quality solutions, e.g. an outlier in computation time is recognized for PLS-GUR at $H = 60$, $T = 90$ in Table 11. The assortment to allocate related to H can also be decreased. If the period to optimize over becomes smaller, the assortment that needs to be available in the warehouses can be reduced. Table 8 shows that the computation time increases exponentially as the problem scale increases, influenced by H. Therefore it might also be wishful from a scalability perspective to decrease the interval of running. However, this could lead to disadvantages business-wise. More frequent and different allocations influences other processes such as the splitting supplier assortments, inventory layout inside the warehouses, inbound planning, etc. The current situation also has to deal with these problems, but less frequently.

The model simplifications (Section 2) both increased and decreased the scale of the optimization problem to be tackled. On one hand, scale increased. If product restrictions would be taken into account, products and suppliers would have predetermined allocations leading to a decrease in decision variables, decreasing model scale. In the non-simplified SAM, a constraint setting a maximum number of suppliers to switch is imposed. This decreases the possible combinations of solutions significantly and therefore the scale of the problem. On the other hand, the simplifications decreased the scale. Only the allocation of suppliers is considered, while also the assortment of retailers needs to be allocated in reality, increasing the scale. In the near future the amount of warehouses to allocate assortment to increases, increasing model scale significantly. Adding a warehouse would create $|I|$ new binary variables and $|T| \cdot \sum_w |G_w| + 2$ new constraints (Equations 15 and 16). Next to this, there are plans to include capacities of other process steps to the model environment, influencing scalability.

Scalability is connected to the optimality of the solutions to the allocation problem. Decreasing this gap from 10^{-7} to 10^{-13} led to minimal improvements of the solution quality, while decreasing the scalability of the model. However, the allowed gap could also be increased. This allows the model to settle for a 'worse' solution, but could increase the scalability of the implementation.

Taking all this into account, it is expected that the implementation of product-level allocation is feasible to be implemented in operation from a scalability perspective. The use of mathematical solvers was expected to be infeasible, but this research proved otherwise. It can be favorable to use such solvers, as they are flexible in implementation and more model complexity can easily be added, contrary to problem-tailored methods. Model scale does increase significantly compared to supplier-level allocation, which leads to higher computation times. Nevertheless, this increase is not blocking in the operational implementation. The change in aggregation level of allocation also led to other insights, which are now discussed by comparing supplier-level allocation and product-level allocation in detail.

5.2 Product-level allocation versus supplier-level allocation

Cost reduction
Results have shown that there exists a cost reduction of ±1% in favor of product-level allocation (Tables 9, 10, 11 and 12). For $H = 60$, $T = 60$, a daily cost reduction of €1445.66 was found. Naively extrapolating this results in a yearly cost reduction of €1445.66 · 365 = €527665.90. This is a rough estimation as the cost reduction is related to the demand levels in a period. Including the allocation of retailers assortment or other demand patterns such as in the peak-period lead will lead to different results.

Table 10 shows that the cost reduction per day decreases as H and T increase. A smaller H indicates

that the assortment of products to allocate is smaller (Section 3.1), which indicates that the number of products aggregated on the supplier-level is also smaller. As the product-group forecast is spread out over the products to allocate, the absolute forecast per product is higher when H is smaller. Lower absolute demand forecasts per product allow for more efficient capacity utilization. This means a larger H increases the flexibility of the allocation, decreasing total costs. The effect of the higher absolute forecasts per product for a smaller H on this flexibility is more evident for SAM, as it is less flexible due to the grouping of products. Using PLS, it is possible to change the allocation of a single product that can efficiently utilize a small portion of capacity. However, the cost reduction per day of PLS over SAM decreases as H increases, indicating that SAM benefits more from the increase in flexibility. The demand forecast calculation structure therefore has a notable influence on the outcomes, meaning that the quality of the allocation decision is dependant on the quality of the forecast. Next to increasing H, T is increased. The models allow just **one** allocation decision per element over the total optimization period. Daily trade-offs are made to find the best overall solution over T. The more days included, the larger the effect of this trade-off, decreasing daily cost reduction. This effect is shown in Table 12, as H is kept constant. The effect is not large, as cost reduction decreases only slightly as T increases.

The cost reduction of product-level allocation comes at the price of scalability. The computation times of SAM are only a fraction of PLS. A similar exponential increase in computation time is present at both supplier-level and product-level, but the absolute computation time is significantly higher due to the larger model scale. This trade-off between cost reduction and scalability has to be taken into account when implementing product-level allocation.

Capacity utilization
The cost reductions originate from better capacity utilization of the outbound lines. PLS is utilizing more capacity of the cheaper outbound lines compared to SAM (Section 4.3.1). The CMC1 line at WH-B is the cheapest line in the model environment (Table 4). It makes sense that the model tries to utilize this capacity as much as possible, which is confirmed by Figures 9a and 9b. CMC1 processes products close to capacity in both cases, but it is shown that PLS processes closer to line capacity. Similar results are present for SMARTMAILER_WH-B, the second cheapest outbound line 8a and 8b. Comparable conclusions can be drawn for the outbound lines presented in Appendix D, where expensive lines are utilized less and cheaper lines are utilized more. Some outbound lines are not being utilized in these results, which is due to the fact that the total forecasted demand is below total capacity. The differences of utilization between PLS and SAM are not large, as the relative cost reduction is small.

Supplier splits
Results in Section 4.2 show that the majority of suppliers is split to deliver both at WH-A and WH-B. Section 4.3.2 provided more details on how these suppliers are split. There are suppliers where the majority of the assortment is allocated to WH-B and the minority to WH-A (Figure 10). 30% of all suppliers should deliver $0-20\%$ of their **products** to WH-A and the remainder to WH-B (Figure 10a). On the other hand, 65% of the suppliers should deliver between $20-80\%$ of their products to WH-A and the remainder to WH-B, which is a more balanced distribution. In the case of **forecasted demand**, the skewness of the fraction towards WH-B is more evidently present (Figure 10b). For 19% of all suppliers, $95-100\%$ of the total forecasted demand is allocated to WH-B, while only $0-5\%$ at WH-A. Next to this, 40% of the suppliers deliver between $80-100\%$ of its forecasted demand to WH-B. The forecasted demand levels of the considerable amount of 50% of suppliers is not in the extremes (fraction between $20-80\%$). The assortment of the remaining 10% of the suppliers is mainly allocated to WH-A. These results show that splitting the assortment of a supplier over multiple warehouses contributes to obtaining lower objectives and better capacity utilization. While supplier splits may be unfavorable from a business perspective, it is possible to think of alternatives. Transshipments could be allowed to stock products on a different location than delivered. Besides, given the fact that WH-A and WH-B and possibly future warehouses are located close to each other, an inbound warehouse could be considered. All incoming goods can be received at this warehouse, after which products can be distributed as preferred.

Results indicate it might be considerable to force the model to not split a supplier if the majority of the assortment is allocated to a single warehouse. This can be realized by including a minimum fraction of distribution over multiple warehouses before splitting of the assortment of a supplier. The cost of this is

accepting the sub-optimality in allocation. Including such a threshold would generate a hybrid version of product-level and supplier-level allocation, as it imposes product-level allocation is only used for suppliers meeting the threshold. As adding such restrictions decreases flexibility, the resulting objective is believed to be between full supplier-level allocation and full product-level allocation.

Pand-split analyses
Pand-split results are comparable for SAM and PLS, but generate other insights. Results in Section 4.3.3 validate that in the use-case it is favorable to process more at WH-B, as the resulting average pand-split ratio is at the upper bound. The reason for this is the alternating processing cost structure of the outbound lines (Section 3.2), making a different warehouse favorable to process at for different demand levels. If the daily forecasted demand levels can be fulfilled using the capacity of low-cost WH-B outbound lines, the ratio tends towards WH-B. If daily demand levels exceed the capacity of those lines, WH-A outbound lines become financially favorable and therefore the ratio tends towards WH-A. If these capacities are met, WH-B is in favor again to eventually end up filling the most expensive lines at WH-A. In Section 4.3.3, the daily demand forecast levels are such that WH-B is in favor. This is closely related to $f_{i,S}$ and the hierarchical structure of the segments, as these fractions contain the information at which outbound line a product can be processed.

Removing pand-spit constraints (Figure 13), the optimal ratio at $H = 31$, $T = 31$ (Figure 13b) would be to allocate $\pm 80\%$ of the total forecasted demand at WH-B. Demand levels increase as T is increased for the used dataset (Figure 7). Therefore the optimal ratio without constraints decreases using $H = 60$, $T = 60$ (Figure 13a), resulting in the average of 69.5% at WH-B. Including the constraints, it makes sense that the optimal average ratio is at its upper bound of 68% in both cases.

The pand-split analyses show that the development of demand levels over time has an effect on the development of the pand-split ratio over time. Constant demand levels indicate less fluctuations above and below the resulting average ratio for both SAM and PLS, in our case the upper bound (Figures 11 and 12). Upper and lower bounds can be exceeded due to the fact that the ratio is computed globally over the optimization period. To avoid the pand-split ratio to exceed the bounds, a daily constraint could be imposed, analyzed in Appendix E.

Flexibility
The flexibility product-level allocation brings allows warehouse characteristics to be utilized better, of which examples are presented in Section 4.3.4. Figure 14 shows that in the results, both warehouses are processing a lot of products in the size-groups M and S. However, WH-A is relatively processing a lot of size-group L, while WH-B is processing more 3XS and XS. This indicates that a warehouse has favorable size-groups to process, connected to the processing capabilities of the outbound lines at those warehouses. This is the case for multiple configurations (Appendix G). Since a supplier's assortment often contains a variety of size-groups (Appendix A.1), product-level allocation is advantageous. Similar implications can be derived from analyzing product-groups being processed at a warehouse (Figure 15). PLS allocates such that e.g. the product-group 'General Toys' is mainly being processed at WH-A, while 'Recreational and Outdoor Toys' is favorable at WH-B. At supplier-level, the grouping of products decreases the ability to make use of the warehouse characteristics this way. Note that again absolute levels of processed products are higher at WH-B than WH-A in both cases.

Company X does not prefer to allocate the entire assortment of similar products to a single warehouse (Section 1), as this influences day-to-day flexibility. Unexpected demand levels, malfunctions or other problems could also influence the processing capabilities of a warehouse. If a product-group or size-group is only represented at one warehouse and this warehouse is restricted in processing capabilities, demand satisfaction could be harmed. This creates a trade-off between optimal utilization through stock allocation and day-to-day processing flexibility. Company X tries to limit this by using size-group constraints, assuring for a balanced distribution of a size-group over the warehouses. However, it is possible that using another characteristic as a constraint to create this balance is favorable.

The flexibility gain from product-level allocation is operationally wishful from multiple perspectives. Product restrictions are a lot easier to incorporate using product-level allocation, it is possible to handle

exceptional cases. This will also be convenient for the implementation of double-stocking, increasing day-to-day flexibility. Double-stocking also allows better processing of multi-orders, which influences processing costs significantly. This requires the model to be able to incorporate inter-product relations, which it does not at the moment. The plans to open a warehouse in the peak-period stocking the high demand level products also benefits from the possibility of allocating individual products.

The advantages of product-level allocation over supplier-level allocation are expected to increase in the future given the planned developments. Adding more warehouses to the model environment does not just increase the problem scale, it also influences the optimal allocation of products. If more warehouses are available, the negative effect of restricting all products of a supplier to be allocated at a single warehouse will become larger. It is expected that the cost reduction of product-level allocation increases as the number of warehouses increases. Also the incorporation of other warehousing process steps in the allocation decision such as stocking capacities and inbound capacities will benefit from the flexibility of product-level allocation, as also for these steps sub-optimal decisions are taken when grouping products. Product-level allocation is a solid solution for the as-is situation, and will become even better in the future.

5.3 Recommendations

The research has proven that both from a scalability and operational perspective, product-level allocation is feasible for implementation. It is recommended to start implementation using open-source optimization software (CBC), as it is proven to be able to tackle the problem at large scales. If the resulting scalability of CBC is insufficient given the development of the model, commercial solvers (GUR) can be evaluated, as results have proven the dominance of this method. Besides, more computation power could be a solution. Solvers are easy to implement and extensions can be plugged in rather easy. Another option is the implementation of another solution method. However, this would require a translation of the problem into the format of this solution method. Given the pace at which the model is being developed, this might be less favorable due to this tailoring process. If decided to implement such methods, it is recommended to start using a GA, as literature proved them to be effective, flexible and rather easy to implement. Besides, it is possible to reduce the optimality of the allocation model. Increasing the optimality gap will lead to a lower quality solution, but increases scalability.

It is recommended to analyze the possibilities of suppliers to be split, as the research has shown that the assortment of a considerable number of suppliers gets split using product-level allocation. Are there any costs connected to splitting suppliers for Company X? These costs could be integrated in the optimization model. Are there cases where suppliers simply deny to be split? What about retailers? Using this analysis, the implementation of product-level allocation can be tailored best to the needs of the organisation.

Another recommendation is to explore the implementation of hybrid stock allocation model in terms of aggregation, which could be a solution for unfavorable supplier splits. Allocating every product individually is not required to realize the sketched operational advantages. However, this requires a definition of which products to allocate individually, for which an analysis is required. Using this, products can still be aggregated in the allocation decision when meeting certain requirements, having a positive effect on scalability. This higher aggregation level can be supplier-level, but also other levels such as product-groups or multi-ordered products could be explored. However, using such a hybrid model is less flexible than using full product-level allocation. As model complexity is increasing, it is recommended to run a full product-level model occasionally to assess the optimality of the used aggregation of products in the hybrid model. The aggregation levels can then be evaluated and changed accordingly.

The last recommendation is to re-evaluate the input data processing steps used in the allocation models. The historic data included H, the associated assortment to allocate I and the optimization period T have a significant influence on the model scale and outcomes. The influence of H is larger on product-level. At supplier-level, products not included in the model are still being allocated if that product is in the assortment of an allocated supplier, which is not the case at product-level. The correct H should be used to allocate the correct assortment. Next to this, a solution needs to be formulated that is able to

allocate products not included in the model (e.g. new products), which requires research. Moreover, there might be better alternatives to the naive method of translating the product-group forecast to product-level using historic sales. A similar naive method is used in calculating the fraction of a product ordered as respectively a mono- or multi-order. The current model does not incorporate any inter-product relationships, while this is a very important characteristic of a multi-order. These relations should be introduced to some extent. Finally, it is recommended to take a look at the cost structure used in the model environment. Different products incur the same costs on an outbound lines, while in reality it is probable that this is variable due to different product characteristics (size, shape, etc.). This can also be extended to the processing costs of other steps in the warehousing process.

6 Conclusion

This research analyzed the scalability of the decrease of the aggregation level of the stock allocation model at Company X from supplier-level to product-level. Despite the substantial increase in scale of the new problem formulation, product-level allocation is feasible in terms of scalability and optimality. An use-case analysis provided more insights in the operational effects of product-level allocation. Absolute cost reductions proved to be small, but the dominance over supplier-allocation is evident. Results have shown that it is feasible to tackle the new problem using solvers. The advantages of product-level allocation are believed to increase given the developments Company X is working on. The research provides a good base for future research and interesting recommendations for Company X have been formulated. Operational advantages can be realized and the solution has great potential for the future.

Apart from this, the research has contributed to the domain of analyzing the scalability of the stock allocation decision that has to be made in a large-scale, multi-warehouse, multi-product e-commerce environment. The world of e-commerce introduced new logistical challenges to be tackled, of which large-scale stock allocation is one. It is validated that choosing the correct aggregation level for the allocation together with the correct formulation of the model, such that all required information is present and unnecessary information is not, is not a simple decision and can be used to gain a competitive advantage. While a contribution to the analysis of optimal allocation of multiple products to multiple warehouses is realized, the research domain can still be extended.

This research had its limitations. Model simplifications influenced the scalability of the allocation models. The exclusion of retailers in the allocation lead to under-utilization of capacity due to not allocating all forecasted demand levels. Another limitation is the fact that the model has been tested in off-peak optimization periods. To analyze the robustness of the scalability of product-level allocation, the analyses could be extended by looking at performance using different demand structures, e.g. the peak-period. Furthermore, all models were ran on a local machine. Having more computation power could lead to lower computation times, therefore increasing scalability. It is not known to what extent this would be advantageous. Lastly, the quality of the used demand forecast influences the quality of the allocation solution. The solution can be of high quality using a very low optimality gap, but if in reality demand levels are different, the resulting allocations are likely to be sub-optimal.

This research also allows for future research. Results in a non-simplified model environment could be analyzed to validate the dominance of product-level allocation. Besides, additional complexity could be added to the model environment. Adding other process steps, adding more warehouses, allowing double-stocking and other sketched developments will influence the model outcomes and performance. Adding more complexity to the calculation of the product-level forecast or adding stochasticity to the model environment will increase the overall model complexity, but might increase the quality of the decision taken with respect to real demand levels. Complexity could also be added in terms of the allocation decision, e.g. the model could be allowed to switch an allocation for an X number of times in the optimization period. It might also be possible to increase or decrease the aggregation level in the allocation, to be able to compare multiple levels of aggregation on scalability and optimality. For example, clusters can be allocated using a clustering analysis or supplier-product combinations can be allocated. Connected to this, the proposed hybrid version of supplier-level and product-level allocation allows for future research. Such a hybrid can be realized by imposing a minimum number of products

(Minimum Order Quantity), forecasted demand (Minimum Order Value) or gain (Minimum Splitting Gain) to be present before splitting a supplier. Another option is to introduce a penalty on the objective function in case of splitting a supplier. Furthermore, the effect of constraint relaxation can be analyzed, thus allowing to exceed e.g. outbound line capacity while inducing penalty costs.

It can be concluded that there are a lot of possibilities of extending the model. Next to this, future research on solution methods is possible. A Genetic Algorithm Linear Programming approach (GA-LP, Appendix I) was implemented to solve the problem, but resulted to be computationally disadvantageous due to the Python programming structure used. While solvers are able to tackle the problem, it might be interesting to evaluate other solution methods. Meta-heuristics such as GA's, PSO, tabu search or simulated annealing might be able to tackle the stock allocation problem (Section 1.1). To conclude, another direction of future research is the analysis of the effect of extra computation power on the scalability of product-level allocation.

Acknowledgements

The author thanks Shaunak Dabadghao and Ahmadreza Marandi from TU/e for their supervision and support from the academic perspective, providing useful input and feedback during the process. Furthermore, Christianne Wisse and Peter van de Ven from Company X are thanked for their time, guidance in the organisation and operational input.

References

Acimovic, J., & Graves, S. (2015, 02). Making better fulfillment decisions on the fly in an online retail environment. *Manufacturing & Service Operations Management*, *17*, 34-51.

Agatz, N. A., Fleischmann, M., & van Nunen, J. A. (2008). E-fulfillment and multi-channel distribution – a review. *European Journal of Operational Research*, *187*(2), 339-356.

Akçay, Y. (2002). Three essays on resource allocation problems: Inventory management in assemble-to-order systems and online assignment of flexible resources..

Anand, R., Aggarwal, D., & Kumar, V. (2017). A comparative analysis of optimization solvers. *Journal of Statistics and Management Systems*, *20*(4), 623-635.

Ardjmand, E., Young, W., Weckman, G., & Sanei Bajgiran, O. (2018, 05). A multi-objective model for order cartonization and fulfillment center assignment in the e-tail/retail industry. *Transportation Research Part E: Logistics and Transportation Review*, *115*.

Bayraksan, G., & Morton, D. P. (2006, 09). Assessing solution quality in stochastic programs. *Mathematical Programming*, *108*(2-3), 495.

Bertsimas, D., Kallus, N., & Hussain, A. (2016). Inventory management in the era of big data. *Production and Operations Management*, *25*(12), 2006-2009.

Brouer, B. D., Karsten, C. V., & Pisinger, D. (2016). Big data optimization in maritime logistics. In A. Emrouznejad (Ed.), *Big data optimization: Recent developments and challenges* (pp. 319–344). Springer International Publishing.

Cao, Y., & Sun, D. (2016). Large-scale and big optimization based on hadoop. In A. Emrouznejad (Ed.), *Big data optimization: Recent developments and challenges* (pp. 375–389). Springer International Publishing.

Catalán, A., & Fisher, M. (2012, 10). Assortment allocation to distribution centers to minimize split customer orders.

de Koster, R. B. (2002). The logistics behind the enter click. In A. Klose, M. G. Speranza, & L. N. Van Wassenhove (Eds.), *Quantitative approaches to distribution logistics and supply chain management* (pp. 131–148). Springer Berlin Heidelberg.

de Véricourt, F., Karaesmen, F., & Dallery, Y. (2002). Optimal stock allocation for a capacitated supply system. *Management Science*, *48*(11), 1486-1501.

Glover, F., & Kochenberher, G. (2003). *Handbook of meta-heuristics*.

Griffis, S. E., Bell, J. E., & Closs, D. J. (2012). Metaheuristics in logistics and supply chain management. *Journal of Business Logistics*, *33*(2), 90-106.

Gupta, D., & Selvaraju, N. (2006). Performance evaluation and stock allocation in capacitated serial supply systems. *Manufacturing & Service Operations Management*, *8*(2), 169-191.

Hane, C. A., Barnhart, C., Johnson, E. L., Marsten, R. E., Nemhauser, G. L., & Sigismondi, G. (1995). The fleet assignment problem: Solving a large-scale integer program. *Mathematical Programming*, *70*(1-3), 211–232.

Jolayemi, J. K., & Olorunniwo, F. O. (2004). A deterministic model for planning production quantities in a multi-plant, multi-warehouse environment with extensible capacities. *International Journal of Production Economics*, *87*(2), 99-113.

Jünger, M., Liebling, T. M., Naddef, D., Nemhauser, G. L., Pulleyblank, W. R., Reinelt, G., ... Wolsey, L. A. (2009). *50 years of integer programming 1958-2008: From the early years to the state-of-the-art*. Springer Science & Business Media.

Jélvez, E., Morales, N., Nancel-Penard, P., Peypouquet, J., & Reyes, P. (2016). Aggregation heuristic for the open-pit block scheduling problem. *European Journal of Operational Research*, *249*(3), 1169-1177.

Krokhmal, P. A., & Pardalos, P. M. (2009). Random assignment problems. *European Journal of Operational Research*, *194*(1), 1-17.

Li, J., Toriello, A., Wang, H., Borin, S., & Gallarno, C. (2021). Dynamic inventory allocation for seasonal merchandise at dillard's. *INFORMS Journal on Applied Analytics*.

Li, X., Zheng, Y., Zhou, Z., & Zheng, Z. (2019). Demand prediction, predictive shipping, and product allocation for large-scale e-commerce.

Lim, Y. F., & Liu, F. (2018, 01). Optimal policies and heuristics to match supply with demand for online seasonal sales. *SSRN Electronic Journal*.

Liu, K., Zhou, Y., & Zhang, Z. (2010, 11). Capacitated location model with online demand pooling in a multi-channel supply chain. *European Journal of Operational Research*, *207*, 218-231.

Malikia, F., Souierb, M., Dahanec, M., & Sarib, Z. (2017, Summer). The use of metaheuristics as the resolution for stochastic supply chain design problem: A comparison study. *International Journal of Supply and Operations Management*, *4*(3), 193-201.

Maltese, J., Ombuki-Berman, B. M., & Engelbrecht, A. P. (2018). A scalability study of many-objective optimization algorithms. *IEEE Transactions on Evolutionary Computation*, *22*(1), 79-96.

Marklund, J., & Rosling, K. (2012). Lower bounds and heuristics for supply chain stock allocation. *Operations Research*, *60*(1), 92–105.

McKendall, J., Alan R. (2008, 10). Improved tabu search heuristics for the dynamic space allocation problem. *Computers & Operations Research*, *35*(10), 3347.

Mitchell, J. E. (2002). Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, *1*, 65–77.

Mohammadi, M., & Musa, N. (2020, 05). Optimization of multi-plant capacitated lot-sizing problems in an integrated supply chain network using calibrated metaheuristic algorithms. *International Journal of Operational Research*, *39*, 325-363.

Mousavi, S. M., Bahreininejad, A., Musa, S. N., & Yusof, F. (2017, 01). A modified particle swarm optimization for solving the integrated location and inventory control problems in a two-echelon supply chain network. *Journal of Intelligent Manufacturing*, *28*(1), 191-206.

Raidl, G. R., & Puchinger, J. (2008). Combining (integer) linear programming techniques and metaheuristics for combinatorial optimization. In C. Blum, M. J. B. Aguilera, A. Roli, & M. Sampels (Eds.), *Hybrid metaheuristics: An emerging approach to optimization* (pp. 31–62). Berlin, Heidelberg: Springer Berlin Heidelberg.

Rappold, J. A., & Muckstadt, J. A. (2000). A computationally efficient approach for determining inventory levels in a capacitated multiechelon production-distribution system. *Naval Research Logistics (NRL)*, *47*(5), 377-398.

Santos, V., Sabino, L., Macedo Morais, G., & Gonçalves, C. (2017, 10). E-commerce: A short history follow-up on possible trends. *International Journal of Business Administration*, *8*, 130.

Sathyanarayana, G., & Patro, A. (2020). Intelligent warehouse allocator for optimal regional utilization. *CoRR*, *abs/2007.05081*.

Silva, F., & de la Figuera, D. (2007). A capacitated facility location problem with constrained backlogging probabilities. *International Journal of Production Research*, *45*(21), 5117-5134.

Soni, G., Jain, V., Chan, F. T. S., Niu, B., & Prakash, S. (2019). Swarm intelligence approaches in supply chain management: potentials, challenges and future research directions. *Supply Chain Management*, *24*(1), 107-123.

Sun, M. (2005, 01). A tabu search heuristic for the uncapacitated facility location problem. In (Vol. 30, p. 191-211).

Tabatabaei, S.-K., Valilai, O. F., Abedian, A., & Khalilzadeh, M. (2021, Feb 16). A novel framework

for storage assignment optimization inspired by finite element method. *PeerJ Computer Science*.

Vielma, J. (2015, 02). Mixed integer linear programming formulation techniques. *SIAM Review*, *57*.

Wolsey, L. A., & Nemhauser, G. L. (1999). *Integer and combinatorial optimization* (Vol. 55). John Wiley & Sons.

Wu, T.-H., Yeh, J.-Y., & Syau, Y.-R. (2004, 05). A tabu search approach to the generalized assignment problem. *Journal of the Chinese Institute of Industrial Engineers*, *21*, 301-311.

Yang, F.-C., Chen, K., Wang, M.-T., Chang, P., & Sun, K.-C. (2010, 12). Mathematical modeling of multi-plant order allocation problem and solving by genetic algorithm with matrix representation. *The International Journal of Advanced Manufacturing Technology*, *51*, 1251-1259.

Yu, D., Wang, A., Hua, Q.-S., & Lau, F. (2013, 01). Faster and space efficient exact exponential algorithms: Combinatorial and algebraic approaches..

Appendices

A Detailed information products to allocate

A.1 Size-groups

In Table 14 an example is given of the assortment of suppliers, including the distribution of size-groups of this assortment. A single supplier can deliver thousands of products, therefore the model scale increases significantly when decreasing the aggregation level of the allocation decision. It can also be mentioned that a single supplier delivers different size-groups, which might demand different warehouse characteristics to be processed efficiently.

Table 14: Assortment size (total and distribution of size-groups) of top 50 suppliers that delivered the most distinct products in [01/04/2021, 31/05/2021]

SupplierID	Size assortment	3XS	XXS	XS	S	M	L	XL	XXL	3XL	Unknown
1	47166	12349	7710	5258	1045	16973	2790	1	1	4	1035
2	36640	9550	5623	2527	776	13360	3297	11	1	2	1493
3	13776	5568	2718	2661	456	1959	341	-	-	-	73
4	11893	5152	4311	2026	347	45	7	-	-	-	5
5	10472	2619	1632	1445	308	3834	426	-	-	1	207
6	9517	1677	1614	997	267	3739	512	1	-	-	710
7	6098	7	344	1035	2526	1656	412	1	-	-	117
8	6029	4697	23	55	25	1211	8	-	-	-	10
9	4622	965	1814	578	147	528	122	-	-	-	468
10	3761	1591	1206	782	112	32	8	-	-	-	30
11	3137	2374	14	78	47	203	-	-	-	-	421
12	2613	1286	328	228	33	66	5	-	-	-	667
13	2443	830	23	13	8	268	3	-	-	-	1298
14	2115	1634	10	24	2	441	-	-	-	-	4
15	1734	369	376	267	354	277	84	3	-	-	4
16	1693	988	5	342	191	159	2	-	-	-	6
17	1663	394	554	236	80	319	46	-	-	-	34
18	1553	70	136	634	504	172	7	-	-	-	30
19	1517	327	277	419	371	96	3	-	-	-	24
20	1455	114	14	176	385	456	275	2	-	-	33
21	1396	950	18	3	3	180	-	-	-	-	242
22	1338	149	6	594	442	136	4	-	-	-	7
23	1198	1014	7	75	54	39	-	-	-	-	9
24	1173	690	2	10	6	330	1	-	-	-	134
25	1118	240	393	340	55	83	7	-	-	-	
26	1074	28	26	111	371	428	103	5	-	-	2
27	1074	820	5	13	1	233	-	-	-	-	2
28	979	1	26	97	159	667	5	-	-	-	24
29	973	17	31	579	254	62	22	-	-	-	8
30	967	-	-	212	511	235	7	-	-	-	2
31	918	-	-	1	371	544	2	-	-	-	
32	917	6	56	256	292	234	2	-	-	-	71
33	864	345	2	7	2	62	-	-	-	-	446
34	852	788	4	26	20	5	-	-	-	-	9
35	848	285	117	114	124	158	33	1	-	-	16
36	848	1	-	17	42	756	18	1	-	-	13
37	823	-	-	-	4	815	2	-	-	-	2
38	822	7	18	356	303	113	18	-	1	-	6
39	800	36	134	265	249	84	32	-	-	-	
40	771	412	256	33	16	12	2	-	-	-	40
41	750	363	19	145	124	93	3	-	-	-	3
42	747	547	8	4	1	106	-	-	-	-	81
43	736	559	21	155	1	-	-	-	-	-	
44	730	32	14	46	94	249	185	57	35	2	16
45	716	118	1	353	200	40	1	-	-	-	3
46	712	674	6	15	7	10	-	-	-	-	
47	698	2	-	22	60	399	214	1	-	-	
48	672	2	-	47	134	451	37	-	-	-	1
49	670	24	19	106	222	181	115	-	-	-	3
50	648	130	60	132	160	118	37	-	-	1	10
...	...		...	...	...	...	...	...	...	...	...

A.2 Restrictions

In Table 15, an example is given on the restricted assortment sold at WH-A and WH-B. There are different categories of restrictions, which have different stocking demands. There are relatively not a lot of restricted products, but since a single restricted product in the assortment of a supplier can influence the allocation decision of all products in the assortment, the effect is significant as the assortment of a supplier can be large (Appendix A.1).

Table 15: Snapshot total numbers of distinct products ordered and the amount of restricted products in $[01/04/2021, 31/05/2021]$. Note: model simplifications are not taken into account.

Total number of products	**PGS15**	**XL**	**LP**	**Medicine**	**High-theft**
460411	2289	1659	5286	185	2433
100%	0.4972%	0.3603%	1.1481%	0.0402%	0.5284%

B Alternative model definition: fraction of a product assigned to outbound line

Note: This model spreads out the demand of a single product over the different outbound lines. This information is not necessarily required for Company X, since another department focuses specifically on the order sourcing from the outbound lines. The allocation model just needs to be able to tell whether the allocated forecasted demand can be fulfilled.

Objective:

$$\min_{x,z} \sum_{t\in T}\sum_{w\in W}\sum_{j\in J_w}\sum_{i\in I} x_{i,j,t}\cdot d_{i,t}\cdot c_j \tag{20}$$

s.t.:

$$\sum_{i\in I} x_{i,j,t}\cdot d_{i,t} \leq n_j \qquad \forall t\in T, \forall w\in W \forall j\in J_w \quad \text{(Line capacity)} \tag{21}$$

$$\sum_{j\in S} x_{i,j,t} - f_{i,S}\cdot z_{i,\sigma(S)} = 0 \qquad \forall t\in T, \forall i\in I, \forall S\in \mathcal{S}_i \quad \text{(Demand assignment)} \tag{22}$$

$$\sum_{t\in T}\sum_{j\in J_w}\sum_{i\in I} x_{i,j,t}\cdot d_{i,t} - p_w\cdot D \in [-\delta\cdot D, +\delta\cdot D] \qquad \forall w\in W \quad \text{(Pand-split)} \tag{23}$$

$$\sum_{w\in W} z_{i,w} = 1 \qquad \forall i\in I \quad \text{(Warehouse-exclusivity set 1)} \tag{24}$$

$$\sum_{t\in T}\sum_{j\in J_w} x_{i,j,t} - T\cdot z_{i,w} = 0 \qquad \forall w\in W, \forall i\in I \quad \text{(Warehouse-exclusivity set 2)} \tag{25}$$

$$x_{i,j,t}\in[0,1] \tag{26}$$

$$z_{i,w}\in\{0,1\} \tag{27}$$

Definition variables and parameters:

Variables:

$x_{i,j,t}$: Fraction of forecasted demand of product i to be processed by outbound line j on day t, ranging between 0 and 1.

$z_{i,w}$: Binary variable used to force that a product i can only be assigned to one of the warehouses w.

Parameters:

i:	A product of which the model needs to determine the allocation.
j:	An outbound line in the warehouse that can produce products i.
w:	A warehouse to which products i can be allocated. In this case only two options are available: WH-B and WH-A.
W:	The set of available warehouses w.
J_w:	A set of available outbound lines j at warehouse w.
t:	A day over which the model has to optimize.
T:	A set of all days t the model has to optimize over.
$d_{i,t}$:	Forecasted demand of product i on day t.
D:	The total demand produced in the optimization period: $\sum_{i,t} d_{i,t}$
c_j:	Costs of processing a product on outbound line j.
n_j:	Daily production capacity of outbound line j.
S:	A segment S is defined as a set of outbound lines j, where a segment is a subset of all the available outbound lines at a warehouse J_w.
$\mathcal{S}_i$:	A set of segments S assigned to product i.
$f_{i,S}$:	The fraction of demand of product i that needs to be assigned to segment S.
$\sigma(S)$:	$= \{w : S \subseteq J_w\}$, returns the unique warehouse w, where segment S is a subset of J_w
p_w:	The target percentage of warehouse w of total demand division over the warehouses.
δ:	The delta that provides flexibility in the pand-split constraints.

C Product-group forecast to product-level forecast

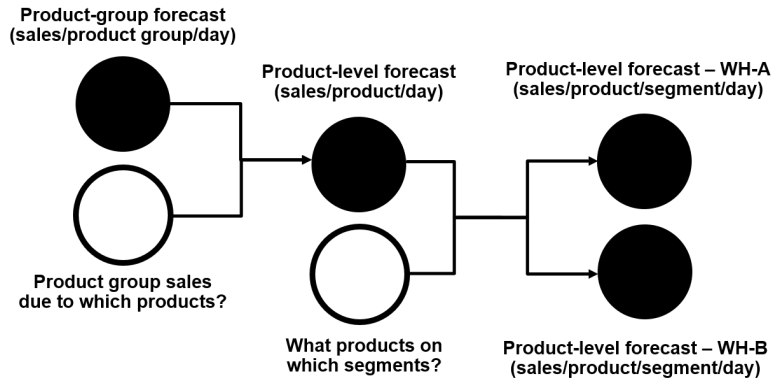
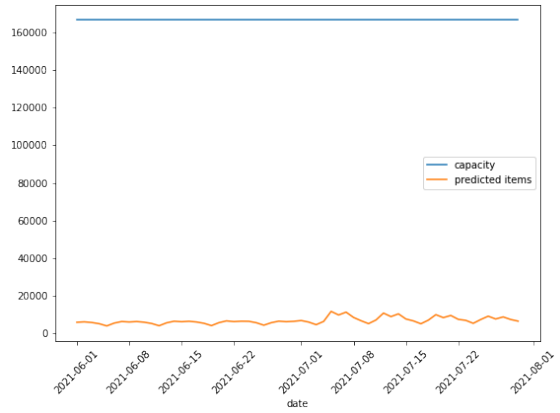


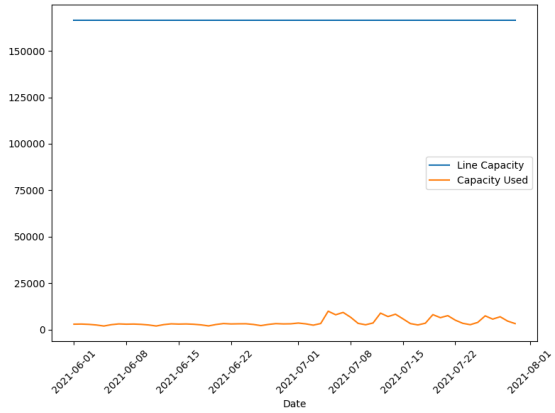
Figure 16: Overview translation product-group forecast to product-level forecast.

D Figures line capacity utilization

D.1 Mono-lines WH-B



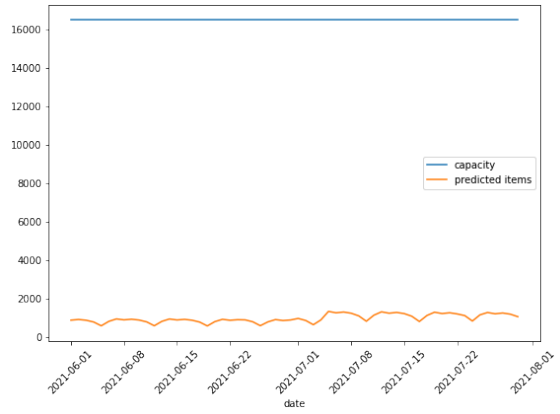
(a) MONO-MANUAL WH-B SAM



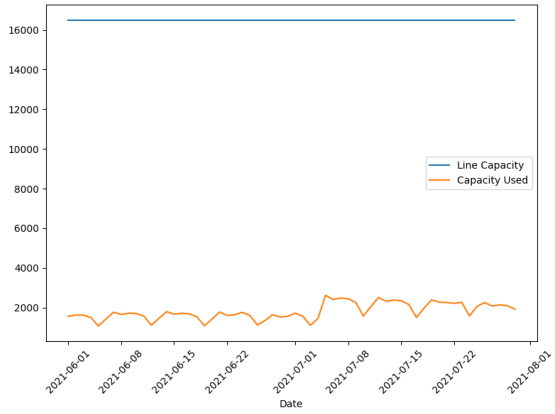
(b) MONO-MANUAL WH-B PLS

Figure 17: MONO-MANUAL WH-B outbound line utilization SAM vs PLS

D.2 Mono-lines WH-A

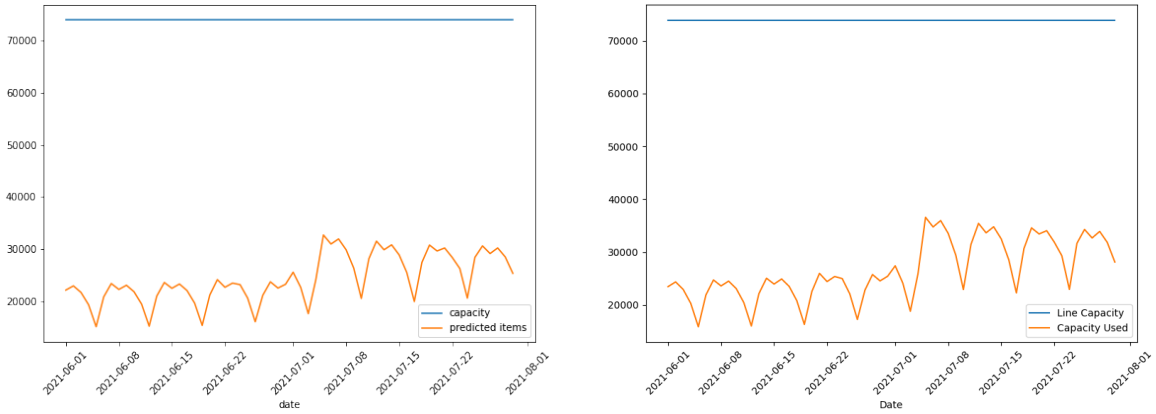


(a) Smartmailer WH-A SAM



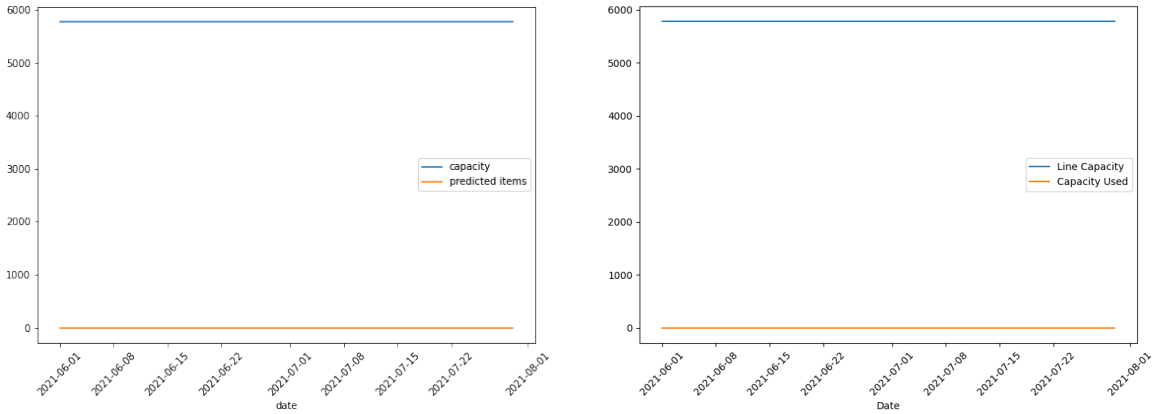
(b) Smartmailer WH-A PLS

Figure 18: SMARTMAILER WH-A outbound line utilization SAM vs PLS



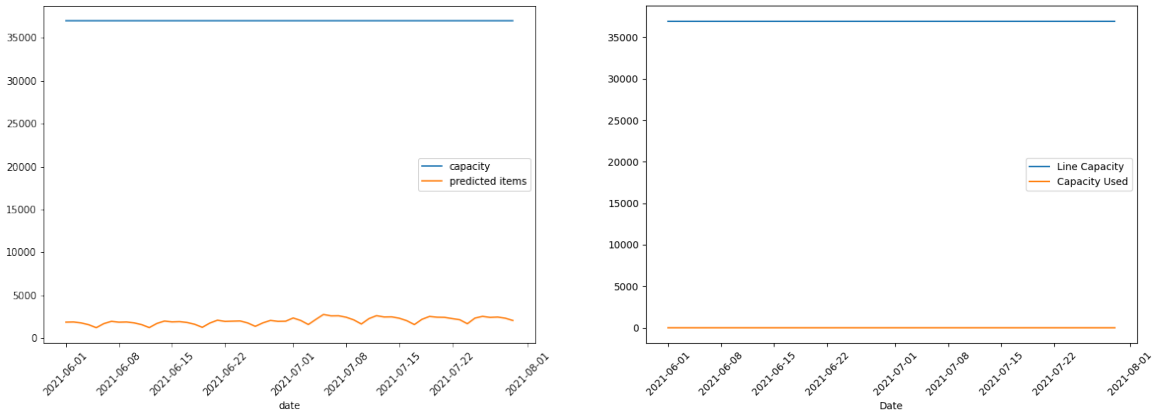
(a) POS-MONO WH-A SAM (b) POS-MONO WH-A PLS

Figure 19: POS-MONO WH-A outbound line utilization SAM vs PLS



(a) Neopost WH-A SAM (b) Neopost WH-A PLS

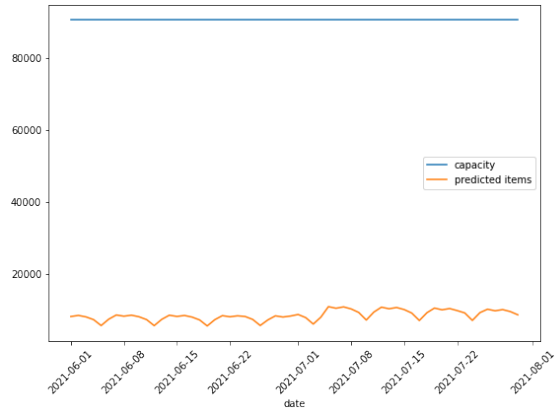
Figure 20: NEOPOST WH-A outbound line utilization SAM vs PLS



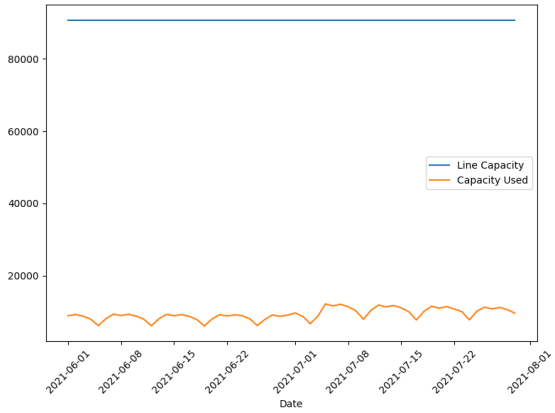
(a) BIG-ITEMS MONO WH-A SAM (b) BIG-ITEMS MONO WH-A PLS

Figure 21: BIG-ITEMS MONO WH-A outbound line utilization SAM vs PLS

D.3 Multi-lines WH-B

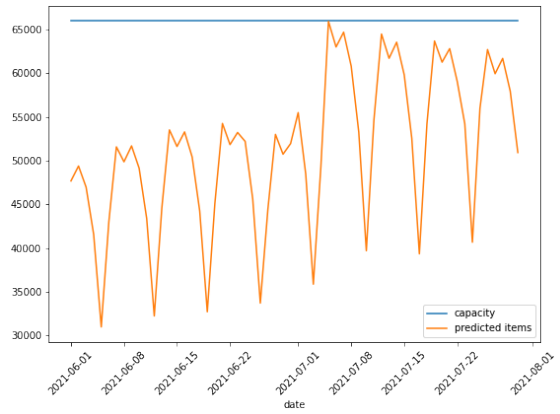


(a) SORTER1 WH-B SAM

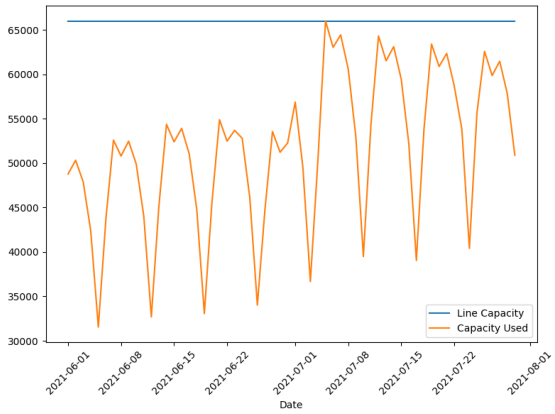


(b) SORTER1 WH-B PLS

Figure 22: SORTER1 WH-B outbound line utilization SAM vs PLS



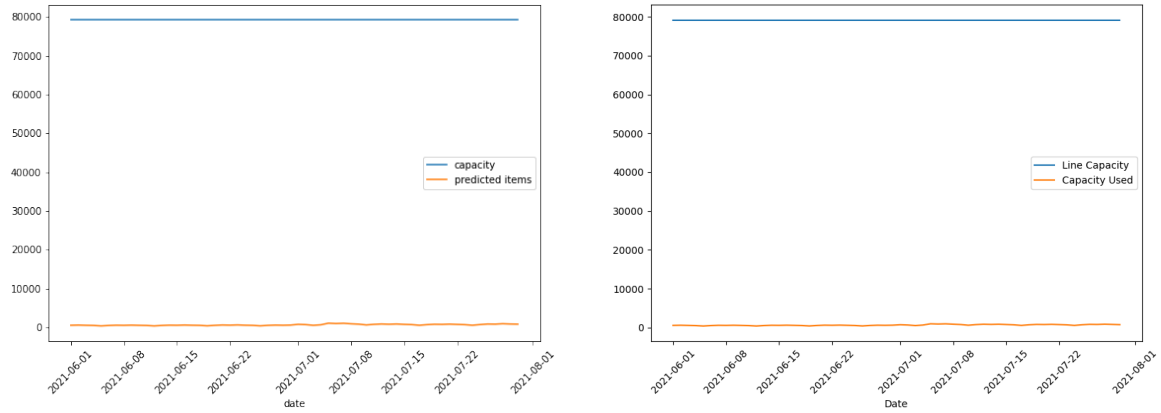
(a) MULTI-MANUAL WH-B SAM



(b) MULTI-MANUAL WH-B PLS

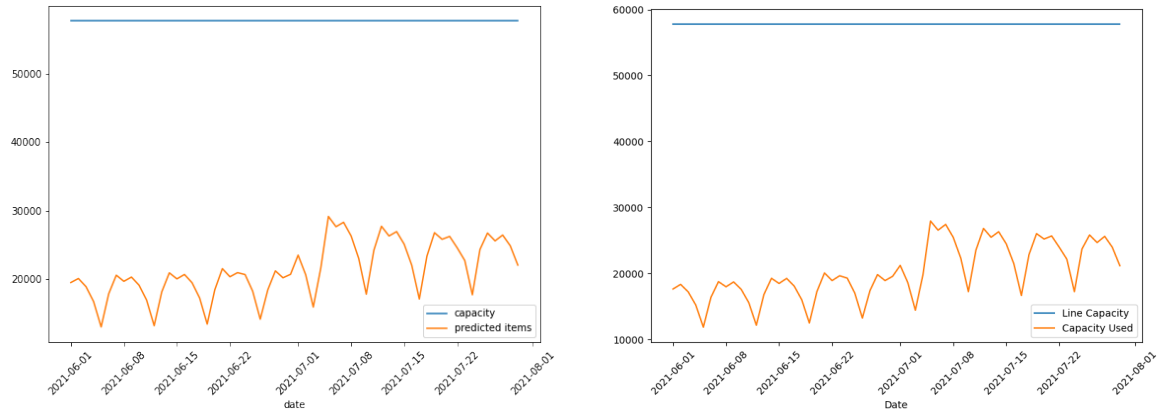
Figure 23: MULTI-MANUAL WH-B outbound line utilization SAM vs PLS

D.4 Multi-lines WH-A



(a) POS-MULTI WH-A SAM (b) POS-MULTI WH-A PLS

Figure 24: POS-MULTI WH-A outbound line utilization SAM vs PLS



(a) BIG-ITEMS MULTI WH-A SAM (b) BIG-ITEMS MULTI WH-A PLS

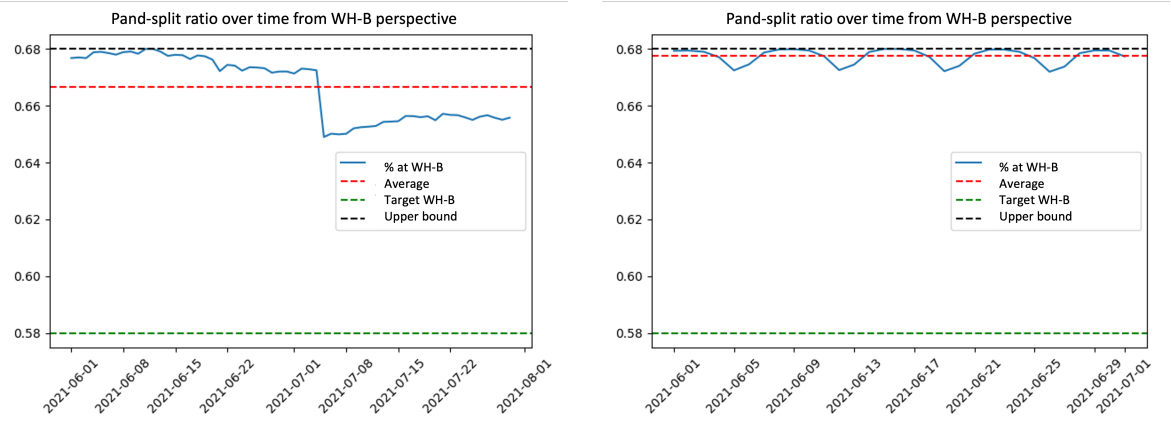
Figure 25: BIG-ITEMS MULTI WH-A outbound line utilization SAM vs PLS

E Analysis daily pand-split constraints

The models as defined in Section 2 allow to exceed the upper bound of the constraint on a daily basis and balance it out on other days to still comply with the global constraint. If this is undesirable on a daily basis, it is possible to add a pand-split constraint on a daily level. The pand-split constraint in PLS (Equation 11, Section 2) then becomes Equation 28, while the rest of the model remains. The effects of using daily pand-split constraints on the ratio over time are presented for PLS in Figure 26.

Daily pand-split constraint:

$$\sum_{S \in G_w} \sum_{j \in S} x_{S,j,t} - p_w \cdot D_t \quad \in \quad [-\delta \cdot D_t, +\delta \cdot D_t] \qquad \forall w \in W, \forall t \in T \tag{28}$$

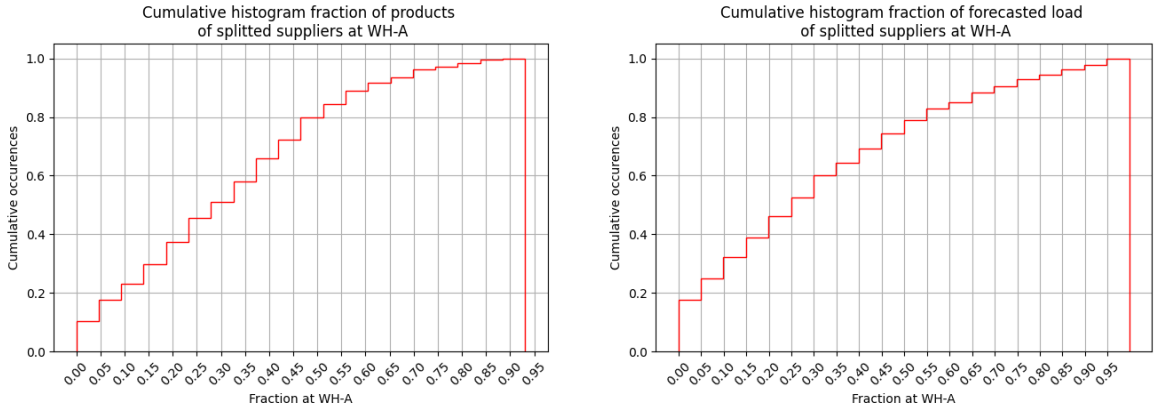


(a) $H = 60$, $T = 60$. (b) $H = 31$, $T = 31$.

Figure 26: PLS pand-split ratio over time with daily pand-split constraints.

For both configurations the influence of the demand behavior is still present to some extent, but never exceeds the upper bound. Because of the demand patterns, the resulting average pand-split ratio over the total optimization period is below the upper bound. The larger the influence of the demand pattern, the lower the average pand-split ratio using daily pand-split constraints. In Figure 26b, it is shown that because of the low variation in demand over time, the resulting average ratio still is very close to the upper bound of 68%. However, adding daily pand-split constraints also influences the scalability of the model, resulting in a higher computation time. For $H = 31$, $T = 31$ the model using daily constraints took 278.80 seconds to solve, while the model with the global constraint only took 144.40 seconds (Table 9), thus decreasing scalability. Next to this, daily constraints decrease flexibility. Spikes in demand due to discount promotions should be within daily pand-split, while in reality such spikes can be tackled by e.g. postponing orders. Other solutions as constraining a rolling average to be under an upper bound or imposing a weekly constraint could be an alternative if wishful.

F Histograms fraction of products and forecasted demand at WH-A

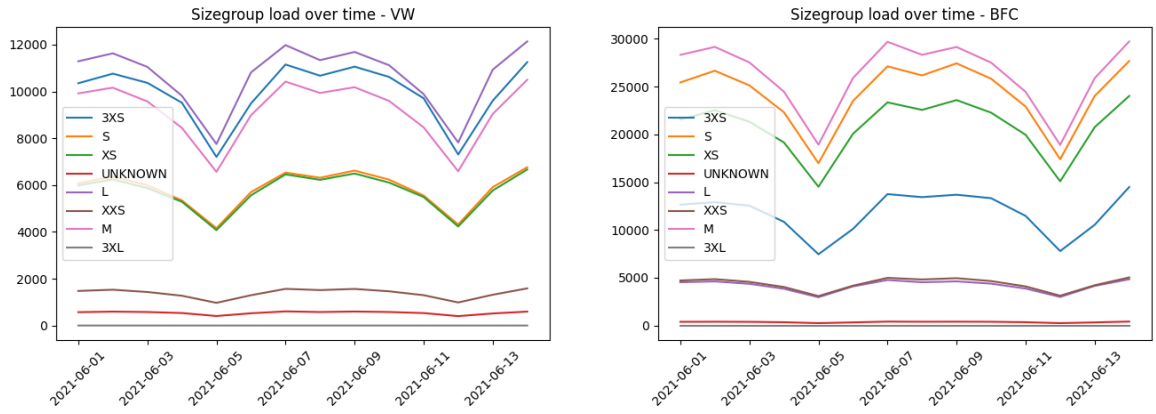


(a) Products (b) Forecasted demand

Figure 27: Cumulative histograms fraction of **products** and **forecasted demand** of a supplier's assortment at a warehouse for PLS at WH-A, $H = 60$, $T = 60$

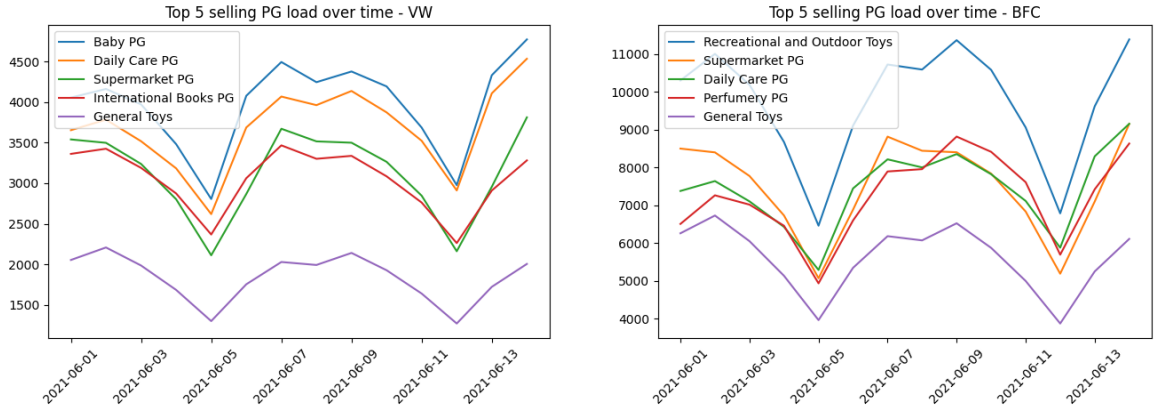
G Utilization warehouse characteristics other configurations

G.1 Configuration $H = 14$, $T = 14$



(a) WH-A (b) WH-B

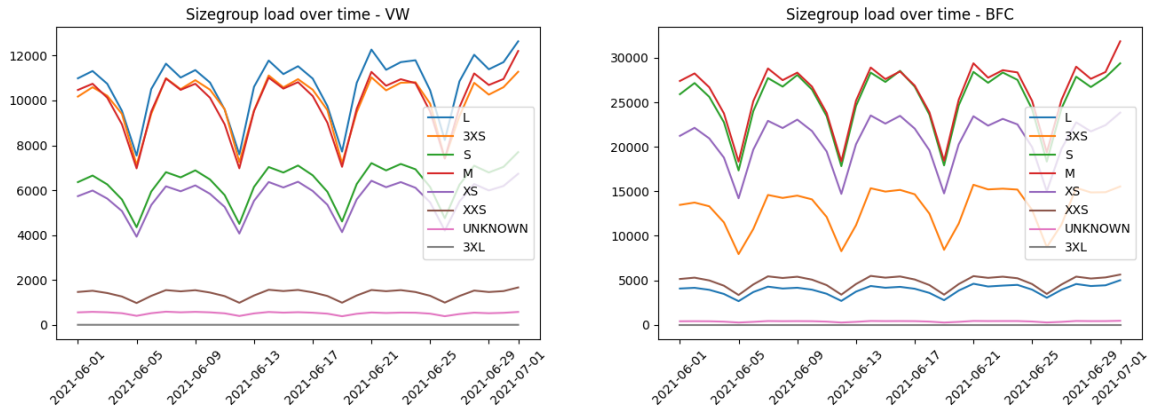
Figure 28: Size-groups handled over the time-span of the optimization period for PLS.



(a) WH-A (b) WH-B

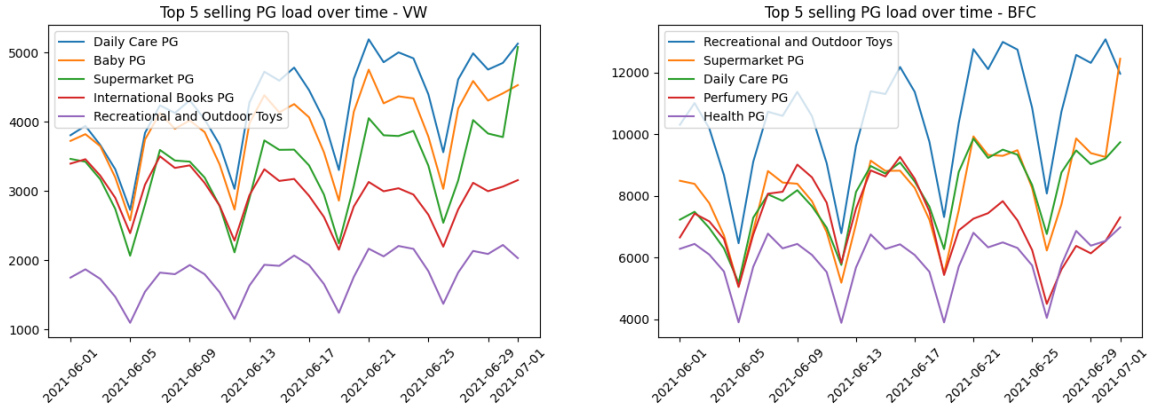
Figure 29: Product-groups handled over the time-span of the optimization period for PLS

G.2 Configuration $H = 31$, $T = 31$



(a) WH-A (b) WH-B

Figure 30: Sizegroups handled over the time-span of the optimization period for PLS.



(a) WH-A (b) WH-B

Figure 31: Product-groups handled over the time-span of the optimization period for PLS

H Total processed demand demand all product-groups $H = 60$, $T = 60$

Table 16: Processed demand all product-groups WH-A.

Product-group	Total demand over optimization period
General Toys	538471
Baby PG	331524.4
Daily Care PG	282253.5
Supermarket PG	191438.5
Camping and Outdoor	139695.2
Household	133797.3
Kitchen	113485.8
Kitchen Machines	104295.6
Games Software Physical	94468.52
Health PG	90211.64
Perfumery PG	87768.94
Sound and Vision Accessories	73449.95
Plumbing and Safety	64236.84
Telephones and Tablets	58266.18
Baby and Kids Fashion	54468.25
Recreational and Outdoor Toys	50239.85
Pet PG	48683.99
Cycling	46287.18
Travel Bags and Accessories	37106.41
Mens and Womens Fashion	30239.54
Household Appliances	29103.85
Sporting Equipment	27129.11
Bodyfashion and Beachwear	26561.5
Tools and Paint	25803.47
Lighting	23302.71
Heating and Air	23033.54
Garden	21043.02
International Books PG	18151.27
Sportswear	17924.65
Educational International	15816.9
Camera	15305.97
PC Accessories	13928.44
Dutch Books PG	13699.38
Personal Audio	12451.61
Home Entertainment	11579.92
Car and Motorcycle	11252.13
Major Domestic Appliances PG	10567.98
Laptop Computers	9089.077
Personal Care	8543.792
Home Decoration	7073.937
Music	6632.744
Educational Dutch	5407.932
Games Consoles	4574.565
Printing and Ink	4558.093
Textiles	4521.459
Telephone and Tablet Accessories	4211.356
Movies	3646.066
Games Accessories	3622.487
Desktop Monitor and Beamer	2670.489
Storage and Network	1853.362
Jewelry and Watches	1483.437
Footwear	1433.506
Wearables	954.0858
Furniture	424.3677
Ereaders and Accessories	83.18525
Gift Cards Physical	0
Ebooks and Audiobooks	0

Table 17: Processed demand all product-groups WH-B.

Product-group	Total demand over optimization period
Recreational and Outdoor Toys	736789.9
Supermarket PG	563259.9
Daily Care PG	486656.3
Perfumery PG	461261.9
International Books PG	422649.8
Health PG	409105.7
Dutch Books PG	241762.1
Baby PG	193258.5
Personal Care	188108.7
Kitchen	139238.6
Gift Cards Physical	134861.6
Educational International	124903.7
Camping and Outdoor	122145.8
Personal Audio	120302.6
Movies	112104.7
Lighting	109584.2
Storage and Network	108288.1
Music	104553.8
PC Accessories	100476.6
Cycling	86874.58
Kitchen Machines	86760.68
Sound and Vision Accessories	76421.68
Pet PG	73783.57
Telephone and Tablet Accessories	72631.45
Sportswear	70466.23
Games Accessories	70114.96
Bodyfashion and Beachwear	68763.59
Sporting Equipment	66152.41
Footwear	63386.47
Educational Dutch	55584.45
Household Appliances	52133.27
Plumbing and Safety	50028.24
Wearables	49406.82
Travel Bags and Accessories	47078.39
General Toys	40760.23
Jewelry and Watches	34382.27
Mens and Womens Fashion	33682.06
Baby and Kids Fashion	28700.31
Telephones and Tablets	27449.04
Ereaders and Accessories	27109.48
Garden	24486.86
Games Software Physical	23277.47
Household	22248.4
Textiles	20470.07
Laptop Computers	18563.21
Printing and Ink	17795.38
Desktop Monitor and Beamer	15925.78
Home Entertainment	15193.46
Games Consoles	14891.14
Car and Motorcycle	14525.96
Tools and Paint	9837.309
Heating and Air	9211.759
Home Decoration	8875.481
Major Domestic Appliances PG	5682.271
Camera	1414.452
Furniture	853.1907
Ebooks and Audiobooks	0

I GA-LP combination

A hybrid form of a meta-heuristic and a LP optimization was implemented. As mentioned in literature recurringly, supply chain oriented problems such as the allocation problem in this research could be solved efficiently by using a combination of meta-heuristics and Linear Programming solutions (Raidl & Puchinger, 2008). In the PLS model definition in Section 2.3, a decomposition of two separate components in the optimization could be recognized:

1. The allocation decision to be made for the products. This is the main goal of the optimization, finding the correct (warehouse exclusive) allocation of products.

2. Given this allocation, optimal capacity utilization in terms of outbound lines needs to be calculated within the line capacity constraints and pand-split constraints while conforming with the segment constraints.

This decomposition allows for the combination of a meta-heuristic, in this case a Genetic Algorithm (GA) seemed to fit the problem the best, and a Linear Programming solution. The GA smartly explores the allocation decisions, while the evaluation step inside the GA is performed by calculating optimal capacity utilization for that allocation, which is a LP problem since the integer variables have been eliminated from the problem. The GA is presented in Appendix I.1. The LP problem in the evaluation step is presented in Appendix I.2. The implementation of this GA-LP combination led to computational challenges apart from the optimization which were too time intensive to solve in the scope of this research. This led to a computational disadvantage compared to the solver implementations. Therefore this GA-LP method is not elaborated on further.

I.1 Genetic Algorithm

Algorithm 1: Pseudo-code Genetic Algorithm in GA-LP combination.

Data: Population size N, Number of generations to perform G, Crossover probability C, Mutation probability M.
Result: Best-found solution and objective to allocation problem using GA-LP combination.

```
Initialization
Initialize population of N chromosomes, where a chromosome consists of |I| binaries.
Initialize optimization model for random chromosome in population (Appendix I.2).
Initialize best solution found and best objective using random chromosome in population.
End initialization

for generations G do
    Perform evaluation by computing fitness of all chromosomes in population using LP model in
     Appendix I.2 → here the GA is combined with the LP.
    if new best solution found then
        Update best solution.
        Update best objective.
    end
    Perform tournament selection to retrieve selected chromosomes for evolution.
    while new population size < N do
        Select two parents from pool of selected chromosomes.
        Perform uniform crossover using C on parents to create two children.
        for resulting children do
            Perform random mutation using M.
            Add child to new population.
        end
    end
    Update current population = new population.
end
```

I.2 LP model in case f(i,S) is already based on an allocation, which is used in the GA-LP implementation.

Note: This model assumes a given product-warehouse allocation, which it can use to determine which fraction of product demand has to be fulfilled by which segment on which warehouse. This is used to generate $f_{i,S}(A)$, the fractions to distribute based on allocation A. The segment fractions of a product are multiplied by 1 if that product is allocated to the warehouse to which that segment belongs. The segment fractions are multiplied by 0 if this is not the case (thus setting the fractions at the warehouse the product is not allocated to, to 0). This removes the binary variables $z_{i,w}$, making it a LP problem, which is defined as follows.

$$\min_{x} \sum_{t \in T} \sum_{w \in W} \sum_{S \in G_w} \sum_{j \in S} x_{S,j,t} \cdot c_j \tag{29}$$

s.t.:

$$\sum_{S|j \in S} x_{S,j,t} \leq n_j, \qquad \forall t \in T, \forall j \in J, \quad \text{(Line capacity)} \tag{30}$$

$$\sum_{j \in S} x_{S,j,t} = \sum_{i \in I} f_{i,S}(A) \cdot d_{i,t} \qquad \forall t \in T, \forall w \in W, \forall S \in G_w \quad \text{(Demand assignment)} \tag{31}$$

$$\sum_{t \in T} \sum_{S \in G_w} \sum_{j \in S} x_{S,j,t} - p_w \cdot D \in [-\delta \cdot D, +\delta \cdot D], \qquad \forall w \in W \quad \text{(Pand-split)} \tag{32}$$

$$x_{S,j,t} \in [0, \infty] \tag{33}$$

Variables:

$x_{S,j,t}$: A continuous variable representing the absolute forecasted demand processed from segment S by outbound line j at day t, where line j is in the set S or $j \in S$.

J Relaxation analysis PLS

To provide more insights on the comparison of the solution to the relaxed problem and the integer problem, results are presented for 60 days of historical data ($H = 60$) and 60 days to optimize over ($T = 60$) using the GUR solver and a relative optimality gap of 10^{-7}. Logging for both is presented in Appendix J.1. In both of the logs, pre-solving is present. Here, the solver tries to find solutions in the solution space that are not feasible or are certainly not optimal. These solutions are omitted from the solution space to decrease the scale and make the problem easier to solve. For example, the pand-split constraints force the allocated demand levels to be within a certain ratio. A lot of combinations of variables are not complying with this constraint and therefore these solutions do not need to be evaluated in the optimization. The specific methods used for pre-solving are solver-specific and not elaborated on any further.

After pre-solving, the solver starts exploring the solution space. In the case of the integer problem, relaxation is implemented after which branching and cutting is performed to find an integer solution. As a result, the best objective and the best-bound are presented. In the case of the relaxed problem, relaxation by the solver is not required as it already is a non-integer problem. The solver analyzes the problem structure and executes one of the available methods to solve non-integer problems, such as the simplex method. The results of the models are presented in Table 18.

Table 18: Results PLS integer and manual relaxation for $H = 60$, $T = 60$.

Model	Objective	Costs per item (Total)	Costs per item (WH-A)	Costs per item (WH-B)	Nr variables	Nr constraints	Runtime (seconds)	Supplier splits
Integer	9929419.756	1.074373	1.100543	1.06206	588082	294705	2148.10	712
Relaxed	9929419.718	1.074373	1.100534	1.06206	588082	294705	159.05	712

Results are very close to each other, but the relaxed problem is solved a lot quicker. However, there is an important difference in the two solutions. The relaxed problem is not constrained to integers and therefore results in non-integer product allocations, as presented in Table 19.

Table 19: Resulting non-integer variables $z_{i,w}$ from solution to relaxed problem.

Product	Variable $z_{i,w}$	Value
1	$z_{1,\text{WH-A}}$	0.138188701
1	$z_{1,\text{WH-B}}$	0.861811299
2	$z_{2,\text{WH-A}}$	0.953065113
2	$z_{2,\text{WH-B}}$	0.046934887
3	$z_{3,\text{WH-A}}$	0.824216896
3	$z_{3,\text{WH-B}}$	0.175783104

Only three products are resulting in non-integer variables, while all other variables are integer. The fact that there are non-integer variables indicates that the found solution by the relaxation is not feasible for the integer problem. Product demand is split to fill up capacity gaps, resulting in minimized total costs. The solvers tackle this by adding new constraints to the problem such that an integer solution is eventually found. These new constraints are found using branching and cutting, which are designed to force the model to find integer variables as a solution to the problem. This is not simple, as setting a variable to a specific value can influence a lot of other variables, which increases the computation time significantly as presented in Table 18.

J.1 Logging

```
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (mac64)
Copyright (c) 2021, Gurobi Optimization, LLC

Read LP format model from file /var/folders/cn/m_gh4h4j50l0sfz5sqj66cqr0000gp/T/6543c201bc3b4a2abc34a614ea3ebc7a-pulp.lp
Reading time = 59.74 seconds
OBJ: 294705 rows, 588082 columns, 51743182 nonzeros
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads
Optimize a model with 294705 rows, 588082 columns and 51743182 nonzeros
Model fingerprint: 0x37521719
Variable types: 1320 continuous, 586762 integer (586762 binary)
Coefficient statistics:
  Matrix range     [2e-04, 1e+03]
  Objective range  [8e-01, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 6e+06]
Presolve removed 605 rows and 730 columns (presolve time = 8s) ...
Presolve removed 293621 rows and 293746 columns (presolve time = 13s) ...
Presolve removed 293978 rows and 533905 columns (presolve time = 428s) ...
Presolve removed 293466 rows and 533393 columns
Presolve time: 428.45s
Presolved: 1239 rows, 54689 columns, 11055278 nonzeros
Variable types: 783 continuous, 53906 integer (43521 binary)

Root simplex log...

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    7.2014920e+06   1.201336e+06   0.000000e+00    446s
     662    7.7867744e+06   3.007316e+05   0.000000e+00    450s
     948    8.8882877e+06   2.693684e+05   0.000000e+00    455s
    1574    9.9294197e+06   0.000000e+00   0.000000e+00    457s

Root relaxation: objective 9.929420e+06, 1574 iterations, 11.84 seconds

 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0 9929419.72    0    5          - 9929419.72      -     -  457s
     0     0 9929419.73    0    6          - 9929419.73      -     -  540s
     0     0 9929419.73    0    5          - 9929419.73      -     -  561s
     0     0 9929419.73    0    8          - 9929419.73      -     -  580s
     0     0 9929419.73    0    8          - 9929419.73      -     -  597s
     0     0 9929419.73    0   10          - 9929419.73      -     -  603s
     0     0 9929419.73    0   10          - 9929419.73      -     -  658s
H    0     0                    9929445.5900 9929419.73  0.00%     -  991s
     0     2 9929419.73    0   10 9929445.59 9929419.73  0.00%     - 1102s
     3     8 9929419.73    2   10 9929445.59 9929419.73  0.00%   2.0 1107s
     7    14 9929419.73    3   10 9929445.59 9929419.73  0.00%   2.7 1110s
    31    38 9929419.73    6   10 9929445.59 9929419.73  0.00%   1.8 1273s
    37    44 9929419.73    7   10 9929445.59 9929419.73  0.00%   1.8 1461s
    43    86 9929419.73    8   10 9929445.59 9929419.73  0.00%   1.8 1467s
    85   244 9929419.73   14   10 9929445.59 9929419.73  0.00%   1.5 1484s
   243   605 9929419.73   43    5 9929445.59 9929419.73  0.00%   1.5 1569s
   604  1099 9929419.73   92    6 9929445.59 9929419.73  0.00%   1.3 1637s
  1098  1498 9929419.73  192    5 9929445.59 9929419.73  0.00%   1.2 1670s
  1497  1847 9929419.73  267    5 9929445.59 9929419.73  0.00%   1.2 1699s
  1846  2148 9929419.73  325    5 9929445.59 9929419.73  0.00%   1.2 1726s
  2147  2488 9929419.73  363    5 9929445.59 9929419.73  0.00%   1.2 1756s
  2487  2911 9929419.73  404    6 9929445.59 9929419.73  0.00%   1.2 1790s
  2910  3326 9929419.73  450    5 9929445.59 9929419.73  0.00%   1.1 1832s
  3325  3756 9929419.73  498    5 9929445.59 9929419.73  0.00%   1.1 1872s
  3755  4186 9929419.73  544    5 9929445.59 9929419.73  0.00%   1.1 1907s
  4186  4703 9929419.73  600    5 9929445.59 9929419.73  0.00%   1.1 1944s
  4703  5263 9929419.73  665    5 9929445.59 9929419.73  0.00%   1.1 1983s
  5263  5763 9929419.73  727    5 9929445.59 9929419.73  0.00%   1.1 2022s
  5763  6236 9929419.73  792    5 9929445.59 9929419.73  0.00%   1.1 2068s
  6236  6709 9929419.73  856    5 9929445.59 9929419.73  0.00%   1.1 2115s
  6710  6710 9929420.65  591   10 9929445.59 9929419.73  0.00%   1.1 2663s
  6712  6711 9929419.73 1192    5 9929445.59 9929419.73  0.00%   1.1 2796s
  6713  6712 9929419.73  229    6 9929445.59 9929419.73  0.00%   1.1 2950s
  6714  6713 9929419.76  755    5 9929445.59 9929419.73  0.00%   1.1 3021s
  6715  6713 9929420.66  174    5 9929445.59 9929419.73  0.00%   1.1 3039s
H 6715  6377                    9929421.0717 9929419.73  0.00%   1.1 3114s
H 6715  6058                    9929419.7560 9929419.73  0.00%   1.1 3181s

Cutting planes:
  Flow cover: 2

Explored 6715 nodes (10396 simplex iterations) in 3185.31 seconds
Thread count was 12 (of 12 available processors)

Solution count 3: 9.92942e+06 9.92942e+06 9.92945e+06

Optimal solution found (tolerance 1.00e-07)
Best objective 9.929419755986e+06, best bound 9.929419725870e+06, gap 0.0000%

Best objective 9.929419755986e+06, best bound 9.929419725870e+06, gap 0.0000%
```

Figure 32: Logging integer PLS problem optimality gap 10^{-7}. Note: omitted logging to decrease size.

```
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (mac64)
Copyright (c) 2021, Gurobi Optimization, LLC

Read LP format model from file /var/folders/cn/m_gh4h4j50l0sfz5sqj66cqr0000gp/T/5812ef8f7e114433ba0527de120d22fa-pulp.lp
Reading time = 46.54 seconds
OBJ: 294705 rows, 588082 columns, 51743182 nonzeros
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads
Optimize a model with 294705 rows, 588082 columns and 51743182 nonzeros
Model fingerprint: 0x6cc0ec35
Coefficient statistics:
  Matrix range     [2e-04, 1e+03]
  Objective range  [8e-01, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 6e+06]

Concurrent LP optimizer: primal simplex, dual simplex, and barrier
Showing barrier log only...

Presolve removed 0 rows and 0 columns (presolve time = 5s) ...
Presolve removed 721 rows and 849 columns (presolve time = 17s) ...
Presolve removed 9872 rows and 10000 columns (presolve time = 25s) ...
Presolve removed 19872 rows and 20000 columns (presolve time = 25s) ...
Presolve removed 169872 rows and 170000 columns (presolve time = 31s) ...
Presolve removed 219872 rows and 220000 columns (presolve time = 35s) ...
Presolve removed 269872 rows and 270000 columns (presolve time = 40s) ...
Presolve removed 293737 rows and 293865 columns (presolve time = 47s) ...
Presolve removed 293737 rows and 293865 columns (presolve time = 111s) ...
Presolve removed 293737 rows and 293865 columns (presolve time = 117s) ...
Presolve removed 293737 rows and 538470 columns (presolve time = 122s) ...
Presolve removed 293739 rows and 538470 columns (presolve time = 125s) ...
Presolve removed 293739 rows and 538470 columns
Presolve time: 129.17s
Presolved: 966 rows, 49614 columns, 11562931 nonzeros

Ordering time: 0.01s

Barrier statistics:
 AA' NZ     : 1.760e+05
 Factor NZ  : 2.204e+05 (roughly 20 MBytes of memory)
 Factor Ops : 9.658e+07 (less than 1 second per iteration)
 Threads    : 4
                 Objective                Residual
Iter       Primal          Dual         Primal    Dual     Compl     Time
   0   3.10666003e+08 -2.97883210e+05  1.14e+09 0.00e+00  4.60e+04   138s
   1   1.92885579e+07 -9.84981780e+05  3.83e+07 1.09e-12  1.64e+03   139s
   2   1.15783203e+07  6.60783553e+06  9.64e+06 9.73e-12  4.02e+02   139s
   3   1.04561567e+07  7.20363343e+06  6.69e+06 9.45e-12  2.81e+02   140s
   4   9.99345786e+06  7.53125430e+06  5.47e+06 8.46e-12  2.30e+02   141s
   5   9.71479512e+06  8.39408278e+06  4.20e+06 9.89e-12  1.78e+02   141s
   6   8.79008948e+06  8.41475036e+06  6.08e+05 1.03e-11  2.83e+01   142s
   7   8.68081535e+06  8.55192624e+06  2.42e+05 6.12e-11  1.12e+01   142s
   8   8.62202592e+06  8.58385687e+06  7.45e+04 1.62e-10  3.46e+00   143s
   9   8.61174836e+06  8.59435062e+06  4.22e+04 1.27e-10  1.95e+00   143s
  10   8.60982685e+06  8.60281479e+06  3.55e+04 1.17e-10  1.63e+00   144s
  11   8.60910145e+06  8.60781817e+06  3.08e+04 1.12e-10  1.41e+00   145s
  12   8.60908050e+06  8.61168537e+06  2.88e+04 1.14e-10  1.32e+00   145s
  13   8.60927866e+06  8.61887912e+06  2.60e+04 1.04e-10  1.19e+00   145s
  14   8.60962497e+06  8.63051913e+06  2.49e+04 1.06e-10  1.15e+00   146s
  15   8.61012121e+06  8.64216931e+06  2.32e+04 9.98e-11  1.07e+00   146s
  16   8.61099968e+06  8.65340184e+06  2.23e+04 9.73e-11  1.02e+00   147s
  17   8.61209464e+06  8.65771530e+06  2.16e+04 9.57e-11  9.89e-01   147s
  18   8.61501620e+06  8.65127728e+06  2.01e+04 9.49e-11  9.25e-01   147s
  19   8.61646545e+06  8.66425190e+06  1.96e+04 9.23e-11  9.01e-01   148s
  20   8.61679473e+06  8.66784574e+06  1.94e+04 9.04e-11  8.95e-01   148s
  21   8.61866787e+06  8.67599757e+06  1.87e+04 9.03e-11  8.62e-01   149s
  22   3.10666003e+08 -4.66444949e+08  1.14e+09 0.00e+00  6.79e+07   150s
  23   1.43797711e+07 -1.63566071e+10  1.62e+07 3.42e-09  2.07e+06   150s
  24   1.09140517e+07 -1.52320052e+09  3.26e+06 2.74e-09  4.13e+05   151s
  25   1.06200934e+07 -4.35965341e+08  2.08e+06 1.17e-09  2.72e+05   152s
  26   1.02550216e+07 -7.06337042e+07  3.05e+05 2.42e-10  3.99e+04   152s
  27   1.01870061e+07 -5.32278381e+06  4.44e+04 6.00e-11  5.92e+03   153s
  28   1.04735052e+07  1.36944164e+05  1.12e+04 2.23e-11  1.56e+03   153s
  29   1.03419992e+07  3.65139094e+06  9.64e+02 1.73e-11  1.92e+02   154s
  30   1.01282802e+07  7.08838629e+06  2.59e+01 1.46e-11  3.33e+01   154s
  31   1.00846047e+07  7.98375378e+06  1.51e+01 1.84e-11  2.23e+01   155s
  32   1.00719740e+07  8.16928902e+06  1.23e+01 1.68e-11  2.01e+01   155s
  33   1.00517013e+07  8.32824325e+06  8.19e+00 1.21e-11  1.80e+01   156s
  34   1.00233141e+07  8.51290178e+06  2.90e+00 1.30e-11  1.55e+01   156s
  35   1.00021543e+07  8.83346121e+06  1.02e+00 1.68e-11  1.19e+01   156s
  36   9.99589676e+06  9.08831966e+06  7.91e-01 1.23e-11  9.23e+00   157s
  37   9.98654695e+06  9.17252046e+06  4.11e-01 1.31e-11  8.27e+00   157s
  38   9.97917337e+06  9.29955290e+06  2.82e-01 1.43e-11  6.90e+00   158s
  39   9.97655213e+06  9.30792471e+06  2.40e-01 1.03e-11  6.79e+00   158s
  40   9.96778006e+06  9.45118411e+06  9.52e-02 1.19e-11  5.24e+00   159s

Barrier performed 40 iterations in 158.98 seconds
Barrier solve interrupted - model solved by another algorithm


Solved with dual simplex
Solved in 965 iterations and 159.05 seconds
Optimal objective  9.929419718e+06
```

Figure 33: Logging relaxed PLS problem optimality gap 10^{-7}.

```
Set parameter MIPGap to value 1e-13
Set parameter LogFile to value gurobi.log

Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (mac64)
Copyright (c) 2021, Gurobi Optimization, LLC

Read LP format model from file /var/folders/cn/m_gh4h4j50l0sfz5sqj66cqr0000gp/T/f8e1c8feb81846a8a8de3e15b44f438d-pulp.lp
Reading time = 37.38 seconds
OBJ: 294705 rows, 588082 columns, 51743182 nonzeros
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads
Optimize a model with 294705 rows, 588082 columns and 51743182 nonzeros
Model fingerprint: 0x37521719
Variable types: 1320 continuous, 586762 integer (586762 binary)
Coefficient statistics:
  Matrix range     [2e-04, 1e+03]
  Objective range  [8e-01, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 6e+06]
Presolve removed 79281 rows and 730 columns (presolve time = 5s) ...
Presolve removed 293978 rows and 533905 columns (presolve time = 346s) ...
Presolve removed 293466 rows and 533393 columns
Presolve time: 345.76s
Presolved: 1239 rows, 54689 columns, 11055278 nonzeros
Variable types: 783 continuous, 53906 integer (43521 binary)

Root simplex log...

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    7.2014920e+06   1.201336e+06   0.000000e+00    358s
     559    7.6856767e+06   3.198947e+05   0.000000e+00    360s
     948    8.8882877e+06   2.693684e+05   0.000000e+00    365s
    1574    9.9294197e+06   0.000000e+00   0.000000e+00    367s

Root relaxation: objective 9.929420e+06, 1574 iterations, 9.61 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Tim

     0     0 9929419.72    0    5          - 9929419.72      -     -  367s
 11379648 1284839 9929419.73  179    1 9929419.73 9929419.73  0.00%   1.3 3005s
H11405183  2409                    9929419.7264 9929419.73  0.00%   1.3 3010s

Cutting planes:
  Gomory: 3
  Cover: 1884
  MIR: 17
  StrongCG: 5
  Flow cover: 12
  Inf proof: 13
  RLT: 2

Explored 11405830 nodes (15301601 simplex iterations) in 3019.22 seconds
Thread count was 12 (of 12 available processors)

Solution count 8: 9.92942e+06 9.92942e+06 9.92942e+06 ... 9.92945e+06

Optimal solution found (tolerance 1.00e-13)
Best objective 9.929419726695e+06, best bound 9.929419726429e+06, gap 0.0000%

Wrote result file '/var/folders/cn/m_gh4h4j50l0sfz5sqj66cqr0000gp/T/f8e1c8feb81846a8a8de3e15b44f438d-pulp.sol'

Model run completed after 5603.595191001892 seconds, will now start producing output
```

Figure 34: Logging integer PLS problem optimality gap 10^{-13}. Note: omitted logging to decrease size.